DOCTOR OF PHILOSOPHY

Developing a PPM based named entity recognition system for geo-located searching on the Web

Bold, Kieran

Award date:
2023

Awarding institution:
Bangor University

Link to publication

PRIFYSGOL
BANGOR
UNIVERSITY
School of Computer Science and Electronic Engineering
College of Environmental Sciences and Engineering

Developing a PPM based named entity recognition system for geo-located searching on the Web

Kieran David Bold

Submitted in partial satisfaction of the requirements for the
Degree of Doctor of Philosophy
in Computer Science

Supervisor Dr. William Teahan

March 2023

Acknowledgements

> *Sometimes life is going to hit you in the head with a brick. Don't lose faith.*
>
> — **Steve Jobs**

Acknowledgements are difficult for me to write, as over the years it would be impossible to mention everyone that has helped me. I would like to extend my special thanks to the following people for their support during my PhD.

- This thesis and project were possible with thanks to the KESS 2 Knowledge Economy Skills Scholarship. My sincere gratitude for this funding, as without it, it would not have made my PhD possible. I would like to extend special thanks to the company partners of this project, Mr. Edwin Smith and Mr. Warren Greveson for their guidance and the contribution that they made to make this project possible.

- Dr. William Teahan, my supervisor, I am forever thankful for all his guidance, expertise and support throughout my PhD. Thank you especially for being there throughout my family situations and for providing the best pastoral care any student could ask for.

- To my Mum, Sister and Stepdad, thank you for supporting me thick and thin throughout my entire PhD. I would like to especially mention my Nan, who poured her heart and soul into my education and life, who got me to where I am today. I'm grateful that you saw me graduate my undergraduate degree and saw the start of my PhD. I severely miss you and I'm sorry that you didn't get to see the end of my PhD journey, but I know you are looking down at me proud of your 'Marmite' getting his PhD completed.

- Carrie, my girlfriend, for being my rock through the ups and downs I've had throughout the past few years, her patience, tolerance and understanding when I had to work late. Thank you for always keeping me going.

- The administrative team and academics at the Bangor University School of Computer Science & Electronic Engineering who have supported me throughout my PhD. I would like to extend special thanks to Yvonne, David, Julie and Siân who without your kindness, chats, laughter, being a shoulder to cry on and camaraderie, I wouldn't know where I would have gone without you.

- Finally my dogs Buster and Tilly. For encouraging me to take the necessary breaks away from the computer and reminding me that walks are the best medicine for clearing my mind.

Statement of Originality

The work presented in this thesis/dissertation is entirely from the studies of the individual student, except where otherwise stated. Where derivations are presented and the origin of the work is either wholly or in part from other sources, then full reference is given to the original author. This work has not been presented previously for any degree, nor is it at present under consideration by any other degree awarding body.

Student:

Kieran David Bold

Statement of Availability

I hereby acknowledge the availability of any part of this thesis/dissertation for viewing, photocopying or incorporation into future studies, providing that full reference is given to the origins of any information contained herein. I further give permission for a copy of this work to be deposited with the Bangor University Institutional Digital Repository, the British Library ETHOS system, and/or in any other repository authorised for use by Bangor University and where necessary have gained the required permissions for the use of third party material. I acknowledge that Bangor University may make the title and a summary of this thesis/dissertation freely available.

Student:

Kieran David Bold

Abstract

This thesis describes the development of a web-based interface to showcase a service for searching for housing and jobs. The thesis explores different web scraping techniques and directions taken during the development of the housing and job website using a map-based interface without the reliance on external APIs.

This thesis also developed a novel approach to named entity recognition using Prediction by Partial Matching (PPM). This involved creating a system for web mining of job and housing information using PPM called Merlin. After evaluating the Merlin system against the ground truth data, the best precision came to 91.0%, recall to 78.7%, and F-measure to 84.4%. For job websites using the Merlin system, the best precision came to 89.0%, recall to 75.0%, and F-measure to 86.9%. Notably, combining both the housing and job information resulted in an precision of 77.0%, recall 65.0%, and F-measure 70.5%.

The thesis compared the Merlin system to an existing state of the art named entity recognition software called spaCy. Evaluating spaCy using the same data and ground truth used for the Merlin system show that for data from housing websites, the best precision for spaCy came to 50.0%, recall to 1.82% and F-measure to 3.5%. For job websites using spaCy, the best precision came to 40.48%, recall to 28.33% and F-measure to 33.33%. When combining both the housing and job information resulted in an precision of 47.15%, recall 26.36% and F-measure 33.82%.

The experiments show that the new PPM-based method for named entity recognition is significantly better both in terms of classifier performance and training time.

The final prototype showcases web scraped listings on a map-based interface using tagged data from the Merlin system, stored in a database and searched by a query. The prototype's advantages are that the information displayed is simple to navigate, compare and explore the different housing and job listings. This prototype's limitations are that it uses static information, and the location for jobs is broad and not specific, unlike housing results. The improvements needed for this prototype are a further crawl to find the specific job search listings location to improve the placement of jobs and to make the data dynamically updated when a user enters a query.

This thesis gathered feedback on the web service compared to an alternative interface that relied on information provided from external vendors APIs. This website evaluation study was conducted using the System Usability Scale to measure the usability of each website. Our results indicate that a higher preference was with novel information techniques developed in this thesis rather than the information provided by the external vendor APIs.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

There is no search service available at the moment that allows searching for both jobs and housing at the same time. The overall aim for this project is to investigate whether such a service is feasible. This project is in collaboration with a startup company called CloodUp, and funded by the Knowledge Economy Skills Scholarships (KESS 2) project [1]. The inspiration behind the project came about by a personal experience of one of the company directors Mr. Edwin Smith. The start-up company director's son decided to spend a year in Australia and his son wanted to move to Australia for work but found it challenging to find both a job and a house that met the requirements of what he was looking for. This resulted in the idea of having a single online service that had the ability to extract housing and job information from multiple sources that would produce the information on a map-based interface instead of having to go to multiple websites to find and compare jobs and housing listings.

Therefore the aim of this project is to explore the development of a novel web service that facilitates the searching of both jobs and accommodation at the same time. The purpose of the web service is to find both job and accommodation information that matches with the user's needs without having to go to multiple websites to find the same information. The purpose is to make it easier for the user to find suitable housing and jobs in their own field of work, along with nearby housing that would not be a long distance away. The project additionally provides the opportunity to fuse data and to use named entity recognition for text on jobs and housing.

[1] Knowledge Economy Skills Scholarships (KESS 2) is a major pan-Wales operation supported by European Social Funds (ESF) through the Welsh Government.

1.2 Research Questions

This project conducts an investigation into an effective way of merging web-based job and accommodation based information. The specific research questions are as follows:

- What is the best way of extracting and then fusing information obtained from housing and job sites such as sites like Rightmove and Zoopla into a singular web-based platform?

- What is an effective interface design for such a platform? For example, would a geographical based interface be more appealing to potential clients for such a service?

The rationale is that no existing web service makes it easy to search for this information all at the same time, and making decisions concerning future employment can require substantial and tedious research which may overlook important local information especially if one is moving to an unfamiliar region or abroad.

1.3 Aim & Objectives

The overall aim of the project is to create a web-based service that will allow users to search within a given geographical region both for accommodation and employment opportunities.

The specific objectives of the project are as follows:

- Produce a literature review of web services and technologies that provide capabilities related to the proposed web service.

- Develop and evaluate a named entity tagger relating to jobs and accommodation that is able to identify the unstructured text of HTML source code and is able to identify and tag the unstructured data appropriately.

- Compare the performance of the created named entity tagger to the performance of an existing state of the art named entity recognition software.

- Design and implement prototypes using a web-based geographical search interface that enables filtering of search criteria relating to jobs and accommodation using dynamic filters.

- Evaluate the usability and effectiveness of the different web-based interfaces.

1.4 Methodology

The methodology adopted for this research was agile software development [21]. This involves the evaluation and improvement of the prototypes that were created according to the requirements of the project.

Three prototypes were developed, which were then evaluated by the company partners involved with the project as well as potential users, and then improvements were made based on their feedback.

Agile software development was used for this project as this approach finds solutions through a collaborative effort between the supervisor, company partners and student. Agile software development was used for the Beta I and Beta II prototype. The agile process fundamentally incorporates the iteration and continuous feedback, also allowing for adaptive planning, development, early delivery and continual improvements. It also encourages rapid and flexible responses to change [39]. Rapid prototyping development [110] [47] is used in this project as it provides many advantages such as:

- It has the ability to explore and realise concepts more quickly. This efficiency in time and cost allows people to move beyond the mere visualisation of a product and make it easier to grasp such a product's properties and design.

- It involves repeated designs and incorporation of changes that involves for the evaluation and testing of a product. This iterative process provides a roadmap to developing and refining the final product.

- It results in concepts being communicated more concisely and effectively. Rapid prototyping takes ideas, images, concepts from paper to a visual product.

- It allows concepts to be thoroughly tested and refined as the process continues. Minimising the design flaws with a rapid prototype allows eliminating costly design flaws that might not be shown at early assessment.

- It allows for frequent interaction with company collaborators and provides for regular feedback from the users.

The disadvantages of using this software development method are that it focuses on working with the software and could lack on the documentation. As there are many available software development models, this project's suitability acknowledged by people involved with this project believes it is best to work with agile software development using rapid prototyping development.

The initial prototype (Alpha) involved investigation of different techniques for scraping data from the web. To ensure this was relevant to housing and accommodation related information, the investigation involved looking at local points of interest such as cafe's, local shops, cinemas for example within an area, extracting different elements and placing the data onto a webpage. Evaluation of this prototype involved users conducting various searches and the quality of the results shown by the system were evaluated based on their relevance.

The second prototype (Beta I) involved investigation of the use of existing APIs for housing and job websites. This involved designing, implementing and evaluating a website that used these APIs. Evaluation of this prototype involved users conducting searches and providing feedback with a questionnaire.

The third prototype (Beta II) involved investigating the use of static data and moving away from the dependency of needing to use APIs. This involved designing, implementing and evaluating a website with the static data gathered.

Evaluation methods such as accuracy, precision and recall are used on the named entity tagger relating to jobs and accommodation. Ground truth is used to compare how well the named entity tagger performs at extracting the individual components of accommodation, jobs and combined housing and jobs.

The evaluation of the two prototypes, Beta I and Beta II, involved performing a usability study to evaluate the user interface. Feedback was gathered from 50 end-users, using the System Usability Scale methodology.

Insights gained from the usability study evaluation for this prototype allowed further improvements to be made to the final prototype. The final prototype used methods of extracting housing and job website data and using features from earlier prototypes with changes that were suggested by the end-users from the usability study. Another usability study was performed for the final prototype as a final evaluation for the project and to give the company partners scope to potentially take the project further.

1.5 Contributions

As this thesis is concerned with the design and development of an innovative way that a web service can show information extracted from housing and job websites, the main contribution of this thesis is the design, implementation and evaluation of three prototypes and the use of innovative techniques for extracting such elements from websites.

The significant results concerning the thesis are listed below:

- The development of a novel approach to named entity tagging that produces structured tagged text for housing and jobs. This work is discussed within Chapter 4: A System for Web Mining of Job and Housing information using PPM.

- The comparison of using the Merlin PPM-based system as opposed to using an existing state of the art named entity recognition tools has shown significantly better performance for PPM. This work is discussed in Chapter 4: A System for Web Mining of Jobs and Housing information using PPM.

- The development of effective methods for scraping web content concerning accommodation and jobs, by exploring a new approach to extracting website information with a step-by-step process. This work is discussed within Chapter 6: Developing and Evaluating Prototype Beta I.

- The design and implementation of an innovative accommodation and job website using vendor APIs. This involved evaluating the APIs with a questionnaire to gather quantified results of usability and performance. This work is discussed within Chapter 6: Developing and Evaluating Prototype Beta I

- The design and implementation of a novel web service that provides accommodation and jobs using methods from Chapter 7 along with other resources. This work is discussed in Chapter 7: Developing and Evaluating Prototype Beta II

1.6 Thesis Outline

Chapter 2 examines the background. This includes an introduction to the methods used within the thesis. Chapter 3 discusses existing web services, and their features and services along with their quality in terms of accuracy of the information they provide. Chapter 4 presents a system that has been created for extracting information from websites using PPM. Chapter 5 covers development and evaluation of prototype Alpha. Chapter 6 discusses the development and evaluation of prototype Beta I. Chapter 7 discusses the development and evaluation of prototype Beta II. Chapter 8 discusses the lessons learnt from the project and chapter 9 concludes the thesis and sets out the direction for future work.

Chapters	**Title**
Ch. 1	Introduction
Ch. 2	Background
Ch. 3	A review of existing web services
Ch. 4	A System for Web Mining of Job and Housing information using PPM
Ch. 5	Developing and Evaluating Prototype Alpha
Ch. 6	Developing and Evaluating Prototype Beta I
Ch. 7	Developing and Evaluating Prototype Beta II
Ch. 9	Conclusions and Future Work

Table 1.1: Overview of Chapters

Chapter 2

Background

The purpose of this chapter is to give some contextual explanation of the background. The background includes the following topics that are related to the research questions and aim and objectives: information retrieval, web scraping, named entity recognition, text mining & information extraction and performance metrics for evaluating IR Systems.

Web search services that extract relevant information based from a user query have become one of the most important computer-based applications. As of April 2020, a survey performed of internet users aged 16 to 64 have reported performing each activity from the last month [1]:

- 81% of people search online for a product or service.

- 74% have purchased online via any device and 66% use mobile apps.

- 90% have visited an online retail site or store.

The survey indicates a strong indication of how online web services are used in different regards each day.

The way we search for information has changed as well, in figure 2.1 shows over how the past decade the way desktop vs. mobile vs. tablet searching has changed.

[1] https://www.globalwebindex.com/reports/trends-19

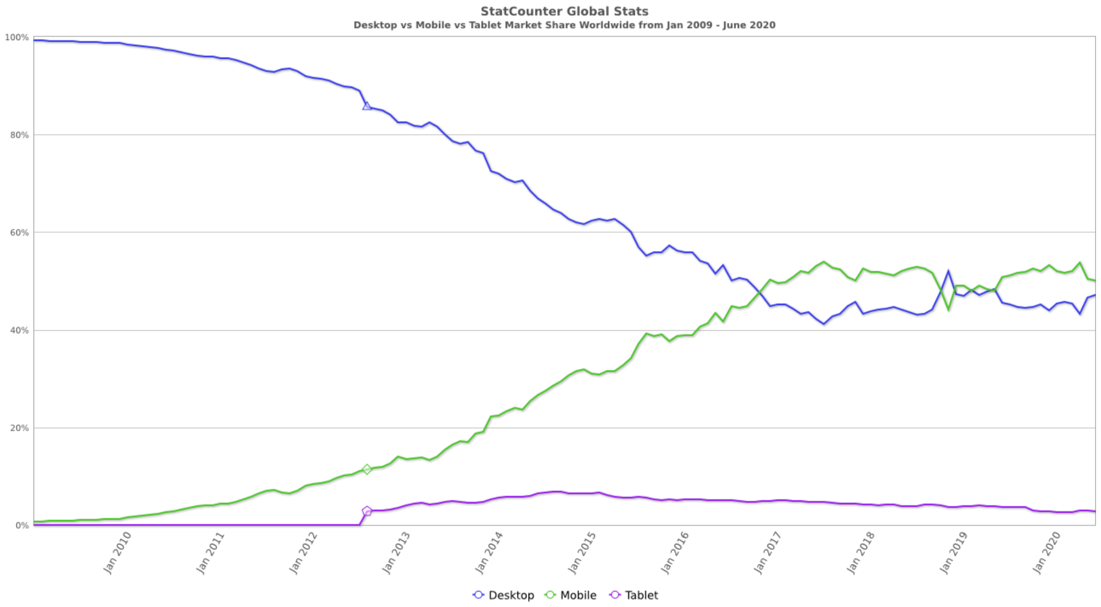


Figure 2.1: Desktop vs Mobile vs Tablet Market Share Worldwide Jan 2009 — June 2020

In figure 2.1, the marketshare of desktop vs. mobile vs. tablet searches have changed dramatically from a decade ago. January 2009 saw a 99.33% share of people using a desktop computer to search, such as home PC's and laptops. As of June 2020, 50.13% of searches are now performed on a mobile device, 47.06% on a desktop device and 2.81% on a tablet device [2]. The way we search has changed, the devices we use are now more on the go, not requiring a laptop or being at a desk with a desktop instead of more using smartphones to do our searching.

Web search engines are used each day to provide search results to queries sent. Google search engine as of June 2020 has 91.75% of the market share world-wide [3]. The development of a robust and efficient search is required to handle daily tasks that we perform each day online. The internet continues to grow exponentially with different type of data and infiltrates into every element of our daily lives [122].

[2] https://gs.statcounter.com/search-engine-market-share
[3] https://gs.statcounter.com/search-engine-market-share

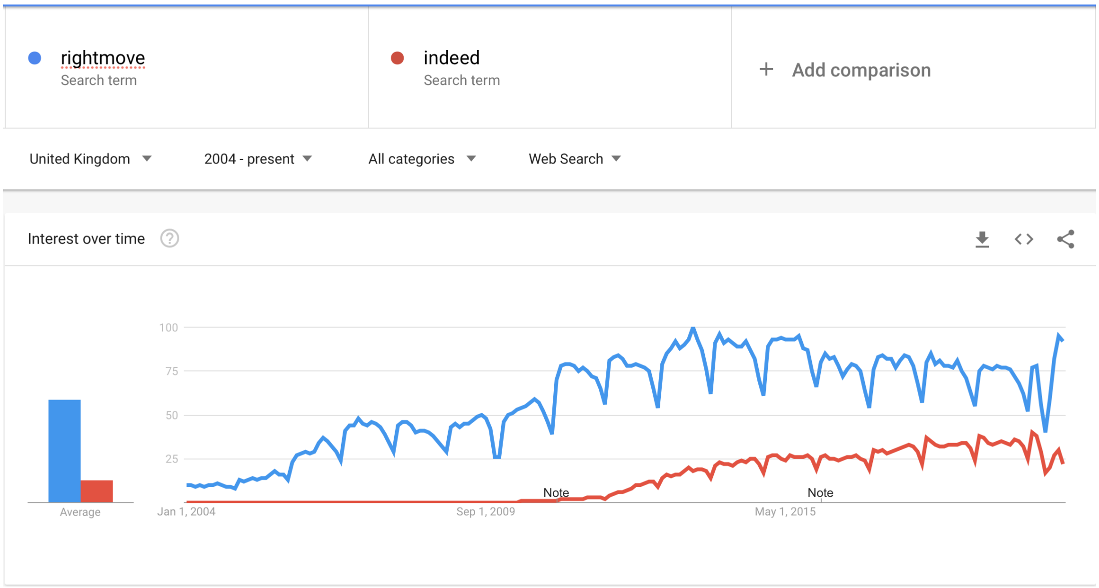


Figure 2.2: Interest over time of Rightmove and Indeed from 2004 — present in the United Kingdom

Be-spoke services such as e-commerce sales are projected to increase from 1.3 trillion in 2014 to 4.5 trillion in 2021. By the end of 2021, 73% of e-commerce sales will take place through a mobile platform [4]. Be-spoke services such as job websites and housing websites that provide listings to the user has also increased in popularity. Figure 2.2 shows the interest over time of Rightmove, a housing website and Indeed a job website from 2004 to the present in the United Kingdom. Rightmove is the United Kingdom's biggest property website that has over 800,000 properties for sale at any given point along with rental homes [5]. Indeed is the most popular of all UK job websites and adds ten jobs to the site every second [6].

Despite the rise of mobile devices being used in the market, the direction of this thesis is to focus on the desktop development rather than to develop a mobile application. The reason behind this is so that the company partners found the ability to incorporate further functionality and features during this phase of research which could then be expanded to the mobile device sector with investment.

[4] https://www.oberlo.co.uk/blog/ecommerce-trends
[5] https://www.bystored.com/blog/best-property-websites
[6] https://www.wikijob.co.uk/content/features/useful-resources/10-best-uk-job-boards-2020

2.1 Information Retrieval

This section reviews important terminology, concepts and background to Information Retrieval (IR).

2.1.1 Ad hoc Information Retrieval

Information Retrieval (IR) is concerned with the structure, analysis, organization, storage, searching, and dissemination of information. IR is found to be useful in office automation and software engineering [57][34]. An IR system is designed to give stored collection items available to users and has been in development since the 1940s. Early IR systems consisted of stored bibliographic items, such as online catalogues and scientific articles although in today's world, the information is full-length documents that are stored in a single location e.g., newspaper archives.

In *ad hoc* information retrieval, the user interacts with the retrieval system one or more times in order to locate results for one information need [132]. In modern times for retrieval systems such as in e-commerce, ad hoc what is used to return a ranking over documents to increase the probability of relevance. An IR model's effectiveness refers to its ability to discriminate between relevant and non-relevant documents.

2.1.2 Keyword and Verbose Queries

Keyword queries are rather short compared to verbose queries which can contain multiple pieces of information [63]. Keyword queries contain only a small selection of keywords from a more verbose description of the actual information that is underlying to the query.

Verbose queries are a significant part of the query in web search, and are common in other applications such as collaborative question answering (CQA) [70]. Verbose queries can be put into long keyword queries by the removal of 'stopwords' or 'stop structure'. Stopwords are high-frequency words that appear in many documents, e.g., 'she', 'if', 'do' 'with' and 'the'. An example of verbose queries is where description topics often begin with 'Find data about' or 'Find documents that describe'. In terms of stopwords, such as stop structures are phrases that do not provide information about the topic of a text. [70].

Complete documents can be considered a form of verbose query. Retrieval using long text as a query is known as query-by-document. An example is where several legal search tasks, such as patent retrieval and invalidity search [58]. In search engines, patent retrieval poses challenges [71]. Using extensive use of highly generic language, acronyms, novel words, and technical terminology muddles documents semantics, while there is a demand for high recall comes an expensive in terms of precision.

In regards to the ad hoc retrieval tracks, a particular type of query might provide static sets of queries, including title queries. The title queries are keyword queries in which they are sequences of content-bearing words with no syntax. Ad hoc queries are associated with a description and narrative. The description is a natural language description of information. The narrative is a short paragraph of text that details information need [95].

2.1.3 Web Based Search Engines

There has been a growth in information on the World Wide Web over the years, which has posed a challenge to traditional information research. There are no consistent indexing or classification principles to order materials on the web. Additionally, there are no filtering practices of web search situations to ensure credibility and quality being provided to the user [144]. This has shown that web searchers provide little input [73] and the users are sensitive to time and effort they put into a search [135]. The majority of web users are sensitive to time and effort to find the information they require. This means that the ability to optimise search order becomes important to the search engines performance [40] [1] [141] [108] [32].

Cooper [40] stated that retrieval systems' primary function is to save the user as much time as possible in search for relevant information by pursuing and discarding documents.

A popular web based search engine, Google, is one of the most heard terms on the Internet. Google has become recognisable and now use it as a verb [80]. Using Google as a verb has made users say, "Hey, what is the solution to this query?" and people reply with "I don't know, Google it". Google Inc. is an American public corporation that specialises in products and internet searching. Larry Page and Sergey Brin founded the Google Company in 1998 [80].

The World-Wide Web has a wide range of content. Users may browse the web through entry points such as Google, but many information seekers use a web search engine to begin their web activity. Users submit a query, which is usually a list of keywords and then they receive a list of web pages that may be relevant. These web pages typically contain the keywords [15]. The World Wide Web has revolutionised the way that people access information and has opened to many new possibilities for information dissemination and retrieval, education, commerce, health care and entertainment services. There has been rapid development in the world wide web since the incorporation [81].

Search engines use well-known information retrieval (IR) algorithms and techniques [124] [50]. IR algorithms were developed for relatively coherent collections such as newspaper articles or book catalogs. The Web is massive, less coherent, and changes rapidly. This requires new techniques, or extensions to the old ones, to deal with the gathering of the information, to make index structures scalable and efficiently updateable.

There are ways in which a person can satisfy with a information need, such as visiting a library, calling on the phone or searching the web. The Internet has become a key information source to finding information online by browsing web pages, posting a question to a question and answering site or using IM or email to contact someone. Search engine use is the most popular approach to online information seeking [49].

There has been a rise in popularity of social networking sites, such as Facebook, Twitter and Linked in as new options to finding information online. Related work has been done by Morris on a comparison of information seeking using search engines and social networks showing that subjects generally preferred searching as it provided more personalised answers and increased confidence of search engine results [102].

One of the first studies on Web user behaviour for searching on the web mainly investigated aspects of browsing the World Wide Web [37] [29] [146]. Choo, Detlor and Turnbull [33] investigated the information seeking behaviour of knowledge workers over a period of two weeks. They combined surveys, interviews and client-side logging and were able to charactise a number of information seeking behaviours of web users.

Navarro-Prieto, Scaife and Rogers [105] identified cognitive strategies related to Web searching. They compared Web searchers with high and low experience and concluded that expert searchers plan ahead in their searching behavior based on their knowledge about the Web, while novice searchers hardly plan at all and are rather driven by external representations (what they see on the screen) [68].

2.1.4 Challenges with Web Based Search Engines

Web search engines have difficult problems in maintaining or enhancing the quality of their performance. This section will discuss the different elements Search Engines encounter.

Spam: As discussed, users tend to look at the first page of search results [134]. Silverstein [135] shows that 85% of the queries only the first result screen is looked at, meaning this only shows the top 10 results. This results in increased traffic to a websites on that first page, whilst exclusion on the rest. Commercially-oriented websites on those who income depends on the traffic will have their best interest to be ranked in that top 10 for the queried result [66]. It is known that users will try and manipulate their placement of this order rank on these search engines to achieve this goal, which is known as search engine spam. Luckily as time has progressed search engines are now tackling search engine spam with search engine optimisation, which identifies and remove these spam listed websites.

Content Quality: There are also issues with ensuring that the quality of the articles on the web are low-quality or full of noise. Web search engines try and tackle this for instance with the PageRank [25] and other similar approaches to rate the web structure and estimate the pages quality.

Vaguely Structured Data: The structure present in data has a significant influence on how search and retrieval is performed. The IR community focuses more on unstructured text documents whilst for example a database community focuses on highly-structured data. JSON and XML methods provide structured data although web pages, which are in HTML come into the fold of not close to free text or well structured meaning this would not allow for a corpora to be used for the web as a whole [66].

2.1.5 Web Crawlers

Crawling is the process of exploring the web automatically. It aims to discover the web pages of a web application by navigating through the system. Web users increasingly rely on search engines to find the data that they are requiring. In order to have an effective search engine, as new data appears, the web crawler has to constantly update the search engines database.

There are motivations to crawling which are:

- Content indexing for search engines meaning that each search engine must require a web crawler to fetch data.
- Automated testing and model checking for the web application.
- Automated security and vulnerability assessment, allowing to detect issues and security vulnerabilities and usability problems in an automated manner [20] [45] [96].

When the users enter a query into a search engine, the search engine examines its index and provides a listing of the best-matched web pages according to the criteria. This is usually the title and a short description of the website text.

The usefulness of a search engine depends on the relevance when the results come back. There may be millions of web pages for a specific word although some may be more popular than others. Search Engines usually employ methods to rank the results to provide good results first. How a search engine decides which pages should be in what order varies depending on different search engine platforms [131].

2.1.6 Web Services and Search Engines

A web service is "a software system designed to support interoperable machine-to-machine interaction over a network" as defined by the World Wide Web Consortium [7]. Having this ability means that web services can achieve automatic and dynamic interoperability between systems and tasks [152]. Web services are self-contained, self-describing, modular applications that can be used across the web [152]. Web services perform a range of functionalities, such as requests for information e.g., web documents, JSON, XML and images, to creating and executing processes. Deployed web services can invoke other applications or web services, allowing it to be discovered by others. Web services provide the ability to create applications at ease, using reusable software components. Asynchronous JavaScript And XML (AJAX) is a dominant technology in Web services, that is used within this project for calling and the structuring data being requested from APIs and other web services [52].

In contrast, search engines are a web service that allows user to search for documents by specified keywords on the World Wide Web [64]. Additionally a search engine allows users to find other web pages through this web service. Examples of search engines are Google, Yahoo and Bing. Different web search engines can differ from one another, the differences being how far crawling is reached, the frequency of the updates and the relevancy of the analysis. Most users view only the first couple of page results from search query results according to Spink and Jenson [138].

2.1.7 Legal Requirements for search engines and web crawlers

The legal liability of search engines requires an understanding of the framework that search engines operate with. The technology that search engines use is an important component to determine the potential liability from legal problems. The methods for storing, collecting and disseminating Internet-based content determine the scope of potential responsibility for a search engine [65]. Search engines use tools to catalog websites using a process called "crawling" [98]. Search engines gather information from the web pages and retain this information in a "cache". The software extracts information from these different websites and places them into an index. The search results would then be returned to the user based on the submitted query [98].

[7] http://www.w3.org/TR/ws-gloss

Search engines are Internet-based operations. There are Internet-based laws for the content providers, host providers and online business operators, although this can be challenging to apply to the search engines. Search engines are distinct and would require legislatures and courts to address legal issues related to search engines from a different perspective.

There was a case law in the United States that emerged after search engines came to the Internet that primarily addressed the definition of search engine as distinct from other Internet identities. ACLU v. Reno [43] was the first ruling where the functionality and importance of web search engines were discussed. The ACLU v. Reno defined the functionality of web search engines services that "allow users to search for Web sites that contain certain categories of information" and provide a list of links to relevant websites.

Courts have been dealing with legal issues regarding the further development of search engine technology and as it expands, search engine law continues to grow. Search engine operators face ever increasing legal action to related intellectual property and data protection including third-party trademark infringement from advertising, copyright violation from search result displays from the aggregated crawling [65]. Interventions have occurred in courts, legislators and regulators alike have been issue-specific ranging from meta tagging, spiders to caching and paid inclusion [59].

There have been different types of concerns and conflicts that have evolved over time and made their way into the legal system. In the early years of web searching up to the year 2000, meta tagging was the most frequent subject of litigation involving search engine operators [59]. Following meta tagging lawsuits, post-2000 lawsuits now are more diverse against engine operators, primarily focusing on intellectual property issues. The intellectual property issues ranging from trademark to more issues regarding copyright issues. The number of claims based on defamation, privacy and other areas show the ever increasing growth of search engines broadening what they do [59].

Web crawling is where a computer program technique is used to scrape a large amount of data from websites, where information can be extracted and put into formats easily read in structured formats, such as JSON or XML. An example of web crawling is where Search Engine Optimisation needs to create sitemaps and give their permissions to let Google crawl their website, which helps Google to rank their websites.

Web crawling can have a negative connotation as it can be used for malicious purposes, examples could be [60]:

- Scraping classified or private information.
- Disregarding the website's terms and services, scraping without the permissions owner.
- Requesting a large amount of data requests could lead to web server crashes, especially with heavy load.

A data service provider could refuse a request if:

- The data is private.
- The Terms of Service prohibits the action of web scraping
- The data is copyrighted.

In 1999, the first web crawling case of a commercial purpose between eBay and Bidder's Edge was heard in US court [30]. eBay sued its competitor, Bidder's Edge, which used spiders to compile listings for specific items from several online auction websites, including eBay and displayed them in an aggregated form on its own website. Bidder's Edge accessed eBay approximately 100,000 times without authorization [79] [8]. Technological measures aimed at blocking the competitor's spiders failed, resulting in eBay filing the suit and claimed that the defendant was committing trespass to chattels. The court stated that the spiders used was likely to qualify as trespassing in eBays servers, thereby consuming eBay's bandwidth and server capability, depriving eBay of the ability to use that portion for its own personal purposes [59].

[8] https://roboticsandautomationnews.com/2020/04/06/essential-legal-issues-associated-with-web-scraping/31501/

In 2019, the US Court of Appeals denied LinkedIn's request to prevent an analytics company (HiQ) from scraping its data. This verdict is a historic decision as it demonstrated that any data publicly available that it is not copyrighted is legally valid for the use of web crawlers. The decision did not grant HiQ or any other web crawlers the freedom to use data obtained for unlimited commercial purposes [137] [9].

The legal requirements towards this project's future development and investment is important, as there will need to be further investigation into the legalities of how data is obtained, processed and used in a commercial regard to ensure the issues as discussed in this section do not arise.

2.2 Web Scraping

The term web scraping means the aggregation of data from multiple websites [103] [136]. Web scraping occurs typically in a piece of software that is known as a web crawler. A web crawler searches the world wide web and retrieves information from the user's set parameters from the query. Several algorithms can be used to scrape information on the internet.

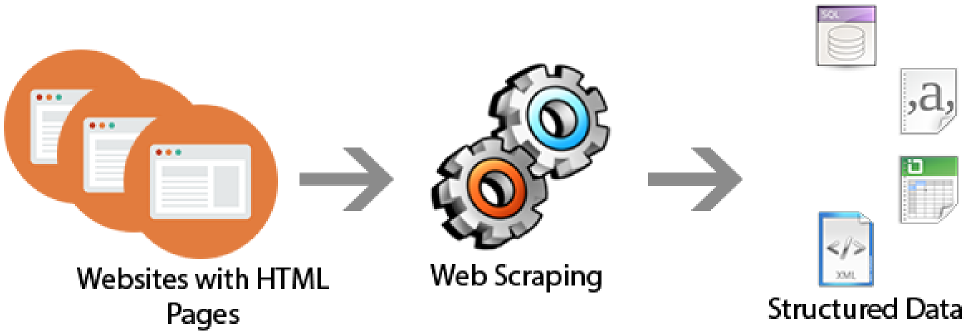


Figure 2.3: Illustrating the process of web scraping

The process of web scraping from Figure 2.3 shows the steps that are used when web scraping occurs with websites:

- Multiple web pages are grouped in a file structure.
- The pages are then parsed using a selected web scraping technique.

[9] https://roboticsandautomationnews.com/2020/04/06/essential-legal-issues-associated-with-web-scraping/31501/

- The request sent from the parser then searches for information that meets the criteria set by the user.

- The information found is then stored in a database.

The query parameters are essential when they are provided. They can give the web crawlers precise information that can be found. The web crawler will be able gather the information which may be at different areas and place it together.

Web scraping is very convenient when the user wants to search and gather data from the internet Search engines would search the index for anything related to the user's criteria. This would result in the user seeing the results that have been found.

2.3 Named Entity Recognition

Named-entity recognition is a subtask of information extraction that seeks to locate and classify named entities mentioned in unstructured text into pre-defined categories that are set, e.g., locations, organisations [104] [99] [82] [119].

Named Entity Recognition (NER) is a field that has been around for more than twenty years; it aims of extracting and classifying mentions of rigid designators from text, such as proper names. There is a diversity of languages, domains, textual genres and entity types in literature. Techniques employed to develop NER systems, from handcrafted rules to machine learning approaches, are used. Handcrafted systems provide good performance at a high system engineering cost. Whilst supervised learning is used, there must be a prerequisite of an extensive collection of annotated data. NER can significantly impact our society, such as NER systems monitoring trends in textual media produced each day by organisations. NER comes from a general class of problems in Natural Language Processing (NLP) called sequence tagging [48].

The use of "Named Entity", which is now used in Natural Language processing, was coined for the Sixth Message Understanding Conference (MUC-6) [62]. MUC was focused on Information Extraction tasks where the structured information of company-related activities and defence-related activities were extracted from unstructured texts. In defining the task, people had noticed it is essential to recognise information units such as names, person, location and organisations, money, date, and times. Identifying references to these entities in the text was recognised as being essential sub-tasks of Information Extraction and was called “Named Entity Recognition and Classification”.

Early systems made use of handcrafted rule-based methods, whilst modern systems use machine learning techniques. Research into automatically identifying named entities in texts forms a vast pool of strategies, representations and methods. In the early stages, one of the first research papers was presented by Rau [113] at the IEEE Conference on Artificial Intelligence Applications. Rau’s paper explores the usage of a system that will “extract and recognise [company] names”, and it relies on handcrafted rules. There is additional previous research in Information Extraction that includes Named Entity Recognition in a constrained manner [22][44][46][61][67][85].

FASTUS, a finite-state Processor for Information Extraction from Real-world Text [14] system was created as approaches to text processing that rely on parsing the text with a context-free grammar tend to be slow and error-prone due to the ambiguity of sentences. FASTUS employs a nondeterministic finite-state language model that produces sentences into noun groups, verb groups and particles.

There is early work within the NER problem of recognising “proper names” in general [104]. The most studied types of “proper names” are names of “persons”, “locations”, and organisations”. These are known as enamex since MUC-6 competition.

The “location” type can be put into “fine-grained locations” [54], such as city, state etc. As well “fine-grained person” subtypes of “entertainer”, “lawyer”, “politican” for example, appear in the work of M. Fleischman and Hovy [54].

The type “miscellaneous” is used in the CONLL conferences and includes proper names falling outside the norm of “enamex”. The class is sometimes with the type of “product,” e.g., Bick[23]. There is also “Timex”, which is a term within MUC that does types of “date” and “time”, and the “numex” types “percent” and “money” are in the literature.

Marginal types are sometimes handled for specific needs for example “email address” and “phone number” [156], “research area” and “project name” [158] and “job title” [38].

The ability to recognise unknown entities is an essential part of NER systems. This ability relies upon recognition and classification rules triggered by distinctive features associated with positive and negative examples. It can be done by handcrafted rules or by supervised machine learning, which will automatically induce rule-based sequence labelling algorithms or systems starting from a collection of training examples. When there are no training examples available, handcrafted rules are applied. A noteworthy example of this is in S. Sekine and Nobata [130] who developed a NER system with 200 entity types to acknowledge the unknown entities present on information.

The point of supervised learning for named entity recognition is to study the features of positive and negative examples of a named entity over an extensive collection of annotated documents and design rules that capture the type's instances. The main issue with this is that there must be a large amount of annotated corpora.

Tackling NER problems is accomplished using supervised learning. Supervised learning techniques include the Hidden Markov Models (HMM) [53], Decision Trees [129], and Support Vector Machines (SVM) [16] to name a few. These variants of the supervised learning approach consist of a system that reads a large amount of annotated corpora, memorises lists of entities, and then creates disambiguation rules based on different features. A supervised learning method consists of tagging words of a test corpus when annotated as entities in the training corpus. A baseline system's performance will depend on the vocabulary transfer, which is the proportion of words, without repetitions, appearing in both the training and test corpus.

There has been some successful semi-supervised systems Ando and Zhang [12]; Suzuki and Iszaki [142] have illustrated that unlabeled text can be used to improve NER systems performance [119]. Qi et al [112] proposed an iterative Word Class Distribution Learning (WCDL) framework and applied it to a set of Wikipedia webpages. WCDL does not self-assign labels that may be subject to learning bias if the model introduces incorrectly labelled examples to the corpus [119].

2.4 Text Mining & Information Extraction

Text Mining obtains information resources that are relevant to the information need from a collection of information resources. The tasks involved with text mining, for example, are document classification and document clustering. Text Mining involves the use of natural-language information extraction. In this instance, text mining's relevance is that housing and jobs listing are needed for the information need.

Text mining has received more importance as of late, according to Sebastiani [128] due to the increasing number of electronic documents out there. The objective of text mining is to extract information from text resources. It is pertinent to Natural Language Processing (NLP), data mining, information retrieval, and machine learning techniques that classify the patterns.

An early approach for text mining adopted in the 1980s involved knowledge engineering by defining a set of rules based on expert knowledge about how to classify documents. In the 1990s, many approaches to text mining started using machine learning techniques. These machine learning methods involved composing an automatic classifier by learning from a set of pre-labeled documents, which belong to specific categories. This approach has achieved better accuracy than human experts [127]. Text categorization shares tasks such as text mining, and information extraction [78] and can be considered an instance of text mining [127].

Information Extraction (IE) is then an approach to text mining. Information extraction takes knowledge from unstructured text by identifying references to the named entities which are relationships between the entities [101]. Information Extraction is the given name to processes that selectively structures and combines data that is found, explicitly stated or implied, in one or more texts. The output of the extraction varies in every situation. Still, it also involves the creation of structured representation to populate some database. Examples could be the retrieval of documents from collections and tagging of particular terms in the text. Information Extraction is relevant for this research as the project requires extracting listings of information from both housing and job websites.

Information extraction to reduce information is used frequently. An early implementation for medical texts was devised by Sager [121]. Information extraction dates back to the late 1970s [11]. An early commercial system from the mid-1980s was JASPER which was built for Reuters by Carnegie Group Inc. with the aim of providing real-time financial news to traders [154].

In 1987, Information Extraction (IE) was spurred by a series of Message Understanding Conferences (MUC). MUC being a competition-based conference [41] that focused on the following:

- MUC-1 (1987), MUC-2 (1989): Naval operations messages.
- MUC-3 (1991), MUC-4 (1992): Terrorism in Latin American countries.
- MUC-5 (1993): Joint ventures and microelectronics domain.
- MUC-6 (1995): News articles on management changes.
- MUC-7 (1998): Satellite launch reports.

At present, the significance of Information Extraction comes in the growing amount of information in it unstructured form. The world wide web largely consists of unstructured documents that lack any semantic metadata [125]. The knowledge contained within these documents could be made more accessible for a machine by marking it up for example by using XML tags. Typically, IE's task is to scan a set of documents written in a natural language and populate a database with information extracted [139].

2.4.1 Tasks and Subtasks for Information Extraction

Applying information extraction to text is linked to the problem of text simplification in order to create a structured view of the information present in free text. The goal is to create machine-readable text to process the sentences. The Information Extraction tasks and subtasks include the following [42]:

- **Event extraction:** Given an input document, output zero or more event templates. An example is a newspaper article might describe multiple terrorist attacks.

- **Template filling:** Extracting a fixed set of fields from a document e.g., victims, perpetrators, time. from a newspaper article about a terrorist attack. A newspaper article may describe multiple terrorist attacks.

- **Knowledge Base Population:** Fill a database given a set of documents. A database is usually in the form of triples e.g., (entity 1, relation, entity 2. e.g., Barack Obama, Spouse, Michelle Obama).

- **Named entity recognition:** recognition of known entity names e.g., people, and organisations. Employing existing knowledge of the information extracted from other sentences.

- **Relationship extraction:** identification of relations between entities, such as: PERSON works for ORGANISATION (extracted from sentence "Bill works IBM.") PERSON located in LOCATION (extracted from the sentence "Bill is in France.").

2.5 Performance Metrics for Evaluating IR Systems

A confusion matrix, known as an error matrix, is used to describe the performance of the classification algorithm [140]. Many evaluation measures have been proposed and are currently employed in Information Retrieval (IR) [91]. At a basic level, an IR system or model's effectiveness is measured in terms of precision and recall [151].

Each row displayed in the confusion matrix represents the instances of the predicated class while each column represents the actual instances of the class [111].

		Class 1	Class 2
Predicted class	Class 1	(True Positives)	(False Positives)
	Class 2	(False Negatives)	(True Negatives)

Table 2.1: Example of confusion matrix for two classes.

Here *TP* represents for the number of True Positives, *FN* the number of False Negatives and *TN* the number of True Negatives.

Calculating the Accuracy for each classification [107] [9], macro-averaging of the class C accuracies occur:

$$Accuracy = \frac{1}{N} \sum_{C \in Classes} \frac{TP_C + TN_C}{TP_C + TN_C + FP_C + FN_C}. \tag{2.1}$$

$Classes$ are the set of classes, and N is the number of classes. In order to calculate the Precision and Recall [9], macro-averaging is used:

$$Precision = \frac{1}{N} \sum_{C \in Classes} \frac{TP_C}{TP_C + FP_C}. \tag{2.2}$$

$$Recall = \frac{1}{N} \sum_{C \in Classes} \frac{TP_C}{TP_C + FN_C}. \tag{2.3}$$

In order to evaluate the performance further, F-measure is calculated as follows:

$$F - measure = 2 \times \left(\frac{Precision \times Recall}{Precision + Recall} \right). \tag{2.4}$$

2.6 Agile Software Development

Agile software development is the approach to finding solutions through a collaborative effort. It allows for adaptive planning, development, early delivery and continual improvement. It also encourages rapid and flexible responses to change [39].

The term agile was made famous in the Manifesto for Agile Software Development [21]. The agile software development values are based on their combined experience of developing software. There were seventeen signatories to the manifesto that proclaimed that they value the following [21]:

- It would focus on individuals and the interactions instead of the process and tools.

- It would work on ensuring the software is working instead of extensive documentation.

- It would do customer collaboration over being a contract.

- It would respond to change instead of following a plan.

The Manifesto for Agile software development is based on principles developed in Utah in 2001 [56]. The principles that are behind this are the following:

- The highest priority is to satisfy the customer with continuous delivery of software. This is rather important towards this projects company partners.

- To welcome changing requirements, even if they are in the late of development. The company partners want to be able to easily change and adapt elements at any stage of development, making this suitable.

- To deliver working software frequently could be from a couple of weeks to a couple of months. The company partners are relying on frequent edits to software made.

- All people associated with the project must work together throughout the entire project. This is suited towards this project as it keeps everyone involved from the student, supervisor and company partners.

- To provide the individual with support and the environment they need and to trust them. Support being provided by both the company partners and supervisor meets the relevance of this.

- The most efficient and effective method of conveying information is to have a face-to-face conversation.

- Working software is the primary measure of progress. The company partners requires in this criteria for the project.

- Agile processes promote sustainable development.

- The project team should be able to maintain a constant pace. The project follows regular scheduled weekly and monthly meetings to ensure pace is being kept.

- There should be continuous attention to technical excellence and design.

- Simplicity is important. Maximizing the amount of work not done is important. The company partners prefer to use 'quick and dirty' prototyping to allow for effective but simple prototyping.

- Having regular intervals, so the team can reflect on how it can be more effective and adjusts its behaviour accordingly [21].

It is essential to acknowledge the advantages and disadvantages of the different software development approaches and why the decision has been taken to use agile software development with rapid prototyping development.

The methodology meant for software development is considered a structure used for planning and controlling the procedure of creating a specialised information system.

In relation to the current project, the tendency of these procedures is to offer customised software development as per the project's requirements. This means all software developments must be examined before selecting which one to use [10].

2.7 Agile Development

Agile development provides the ability to create and respond to change. It is a way of dealing with an uncertain and turbulent environment that can occur in a project such as this. This process is best suited to this project as the user can encounter different paths and routes, which ultimately could lead to a dead end. It would mean adaption would need to occur. The user can think about how the user can understand what is going on at the present moment, identify what uncertainty the user would face, and then adapt to that as the user goes along.

[10] https://acodez.in/12-best-software-development-methodologies-pros-cons/#Agile_Software_Development_Methodology

The authors of the Agile Manifesto chose “Agile” as the label for this whole idea because that word is represented as the adaptiveness and response to change, which was essential to the approach [21].

The agile approach is suitable for designing and then implementing and evaluating a piece of software carried out in this project. Alongside designing, implementing and evaluating the software, there needs to communicate and produce rapid prototyping on development. Using agile methodology alongside rapid prototype development is a benefit of the process. It encourages feedback from the customers who in terms of this project are the company partners.

Agile development also is beneficial as it allows the software to be released in iterations. Iterative releases improve efficiency by allowing teams to find and fix defects and then align expectations early. It also allows users to realise software benefits earlier on with frequent incremental improvements.

A disadvantage of this method is that it relies on real-time communication, so new users often lack documentation. It requires a big commitment in time and means that the developer will have much labour-intensive work. Each feature will need to be fully implemented within each iteration for company approval.

Agile software development refers to a group of software development methodologies based on iterative development. It is where the requirements and solutions evolve by collaboration between the supervisor, company partners and student. The agile process fundamentally incorporates iteration and continuous feedback, refining the software and delivering a software system.

Rapid prototyping development in which is to be considered for this project, provides many advantages such as [11]:

- It has the ability to explore and realise concepts more quickly. This efficiency in time and cost allows people to move beyond the mere visualisation of a product and make it easier to grasp such a product’s properties and design.

[11] https://www.3erp.com/blog/rapid-prototyping-advantages-applications/

- It involves repeated designs and incorporation of changes that involves for the evaluation and testing of a product. This iterative process provides a roadmap to developing and refining the final product.

- It results in concepts being communicated more concisely and effectively. Rapid prototyping takes ideas, images, concepts from paper to a visual product.

- It allows concepts to be thoroughly tested and refined as the process continues. Minimising the design flaws with a rapid prototype allows eliminating costly design flaws that might not be shown at early assessment.

- It allows for frequent interaction with company collaborators and provides for regular feedback from the users.

The disadvantages of using this software development method are that it focuses on working with the software and could lack on the documentation and get off-track as the outcomes are not clear.

As there are many available software development models, the people involved with this project believes it is best to work with agile software development using rapid prototyping development as it suites the style and development processes required for this project.

2.8 Software Development Plan

The research project conducted for this dissertation involved working in collaboration with the KESS 2 initiative. This gave the opportunity to have an industrial Ph.D. to research the development of several prototypes in order to investigate the feasibility of its use for a future business for the company partners.

The plan after the prototypes were developed and fully evaluated was for the company partners to approach investors for further investment to take the project forward in order to fully exploit a potential financial opportunity.

Throughout the time of the Ph.D., the company partners had meetings on a monthly basis, where they would visit Bangor University for a few hours to discuss progression of work and help to develop ideas. Each meeting had positive engagement between both academic and company partners.

Weekly reports were provided to the company partners and a broad dialogue to both the company partners and supervisor was given. The company partners had researched heavily into the current market into what was available, and they took a keen interest the research that was undertaken for the project, including investigations into existing web services. Their feedback was taken into consideration for future amendments to the prototypes.

The company partners engaged with the design process of the prototypes using the Five Design Sheets and also took a keen interest into agile design.

The next steps from this industrial Ph.D. for the company partners are to potentially reach out to investors and to take this research and further develop the prototypes in order to create a product for public use.

2.9 Summary and Discussion

This chapter has examined the background to the topics that are presented in the following chapters. We have reviewed Information Retrieval and relevant methods that have been used in the subsequent chapters. We have also reviewed the subject areas of Information Extraction, Text Mining and Web Scraping. This chapter also reviewed existing Named Entity Recognition software since it is related to the web mining approach adopted for this project.

Chapter 3

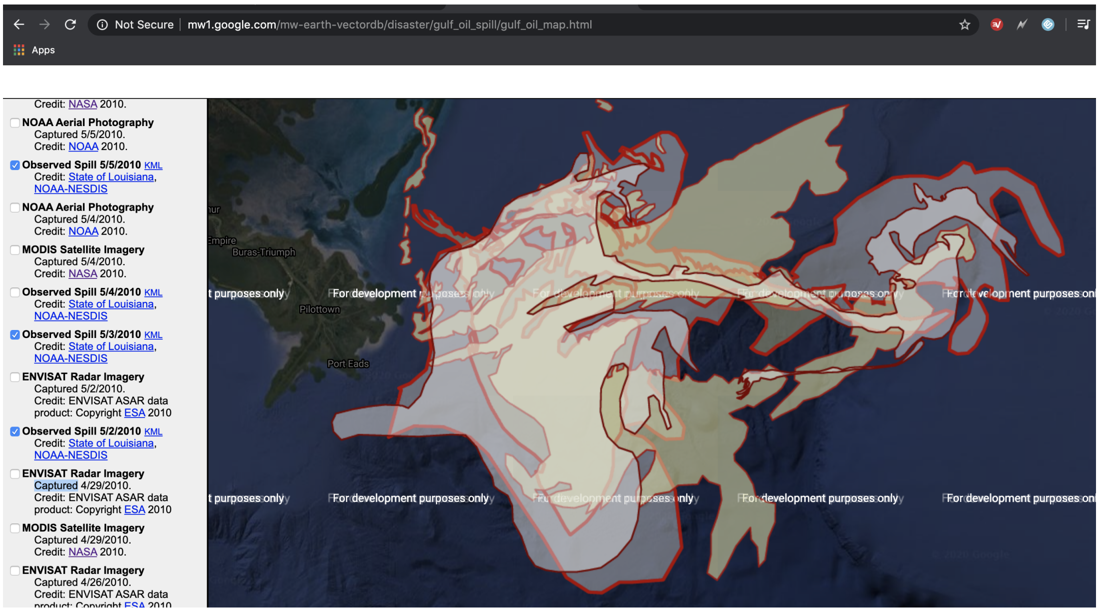
A review of existing web services

The purpose of this chapter is to give a review of existing web services. This chapter investigates various existing web services that are relevant to the objectives (stated in section 1.3) of producing a review of existing web services.

3.1 Existing Web Services

Mapping solutions on the web are a growing people are searching for places of interest, addresses, and exploring information related to geographical locations. Google Maps API [1] allows the user to integrate data in a manner that is usable to the user. A good example of how Google Maps API has been used is in the Deepwater Horizon oil spill in the Gulf of Mexico back in 2010. Google Maps used to show the development of spread over time visualised with different colours, and the shapes indicate the flow as shown in figure 3.1 [143].

[1] https://pypi.python.org/pypi/googlemaps/


Not Secure
mw1.google.com/mw-earth-vectordb/disaster/gulf_oil_spill/gulf_oil_map.html
Apps
Empire
Buras-Triumph
Pilottown
Port Eads
For development purposes only

Credit: NASA 2010.

□ **NOAA Aerial Photography**
Captured 5/5/2010.
Credit: NOAA 2010.

☑ **Observed Spill 5/5/2010** KML
Credit: State of Louisiana, NOAA-NESDIS

□ **NOAA Aerial Photography**
Captured 5/4/2010.
Credit: NOAA 2010.

□ **MODIS Satellite Imagery**
Captured 5/4/2010.
Credit: NASA 2010.

□ **Observed Spill 5/4/2010** KML
Credit: State of Louisiana, NOAA-NESDIS

☑ **Observed Spill 5/3/2010** KML
Credit: State of Louisiana, NOAA-NESDIS

□ **ENVISAT Radar Imagery**
Captured 5/2/2010.
Credit: ENVISAT ASAR data product: Copyright ESA 2010

☑ **Observed Spill 5/2/2010** KML
Credit: State of Louisiana, NOAA-NESDIS

□ **ENVISAT Radar Imagery**
Captured 4/29/2010.
Credit: ENVISAT ASAR data product: Copyright ESA 2010

□ **MODIS Satellite Imagery**
Captured 4/29/2010.
Credit: NASA 2010.

□ **ENVISAT Radar Imagery**
Captured 4/26/2010.
Credit: ENVISAT ASAR data product: Copyright ESA 2010

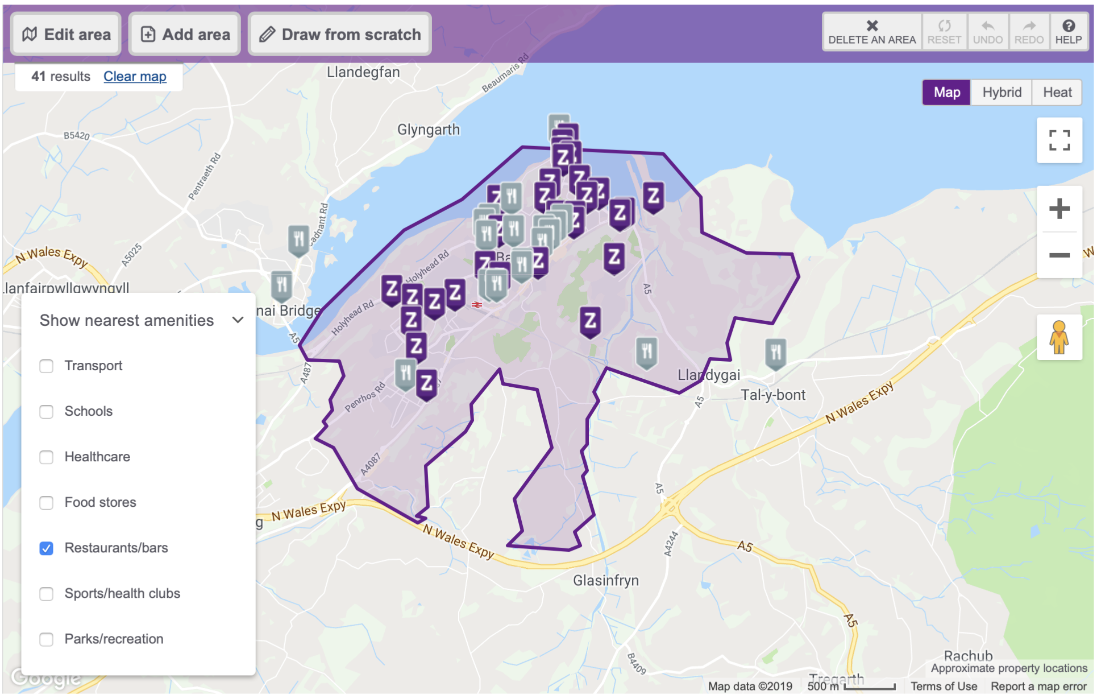
Figure 3.1: The impact of the Deepwater Horizon oil spill visualized in Google Maps [143]

Due to the ever-evolving development of Google Maps API (Application Programming Interface), [2], there are many third-party applications and custom maps that have annotation. Examples of how good applications and custom annotated maps are, is the Health QOF (Quality and Outcomes Framework) database and the London July 2005 Terrorist Attacks map. As Google Maps has an extensive acknowledgement of XML (eXtensible Markup Language), users are able to produce their own custom annotated Google maps, e.g., based on their own GPS (Global Positioning System) location data, the user could even tie in images and video to create interactive multimedia maps.

Zoopla [3], a housing website in the United Kingdom, uses SmartMaps that allows you to view houses and flats for sale in the United Kingdom. The website allows the user to search an area of interest by editing their area boundaries or drawing directly on the map. Zoopla allows the user to search in a region that does not show the user housing, but it also shows the user points of interest that are to the houses the user is viewing.

[2] https://console.developers.google.com

[3] https://www.zoopla.co.uk/for-sale/map/property/gwynedd/bangor/



Show nearest amenities

- □ Transport
- □ Schools
- □ Healthcare
- □ Food stores
- ☑ Restaurants/bars
- □ Sports/health clubs
- □ Parks/recreation

Figure 3.2: Zoopla's Smart Maps

The example shown in Figure 3.2 allows you to be able to visually see on the map the housing locations using pins, similarly to how this project does. It also allows any nearby points of interest such as shopping locations, health and education locations to be visualised. Zoopla has done this by using Google Maps API, to overlay data that is in a structured manner which provides data of the nearest amenities.

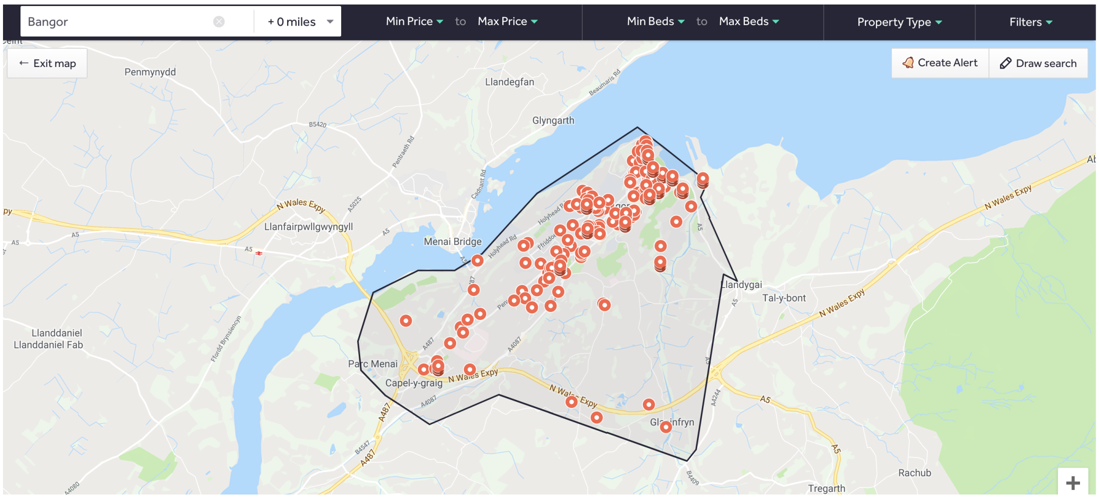


Figure 3.3: Rightmove's Smart Maps

As shown in figure 3.3, Rightmove has a similar style to Zoopla although the main difference is that it's only showing the housing within the area and not the points of interest. Similarly to Zoopla, Rightmove, a housing website in the United Kingdom uses a map version of results, with a similar style to Zoopla.

Primelocation, a housing website in the United Kingdom, uses SmartMaps that allows you to view houses and flats for sale in the United Kingdom. Notably, the company also uses Zoopla's map functionality although it is coloured different. This allows the user to search for exactly where the user is interested in drawing an area boundary. Primelocation allows the user to search in a region that does not show the user housing. However, it also shows the user the local points of interest to the houses the user is viewing.

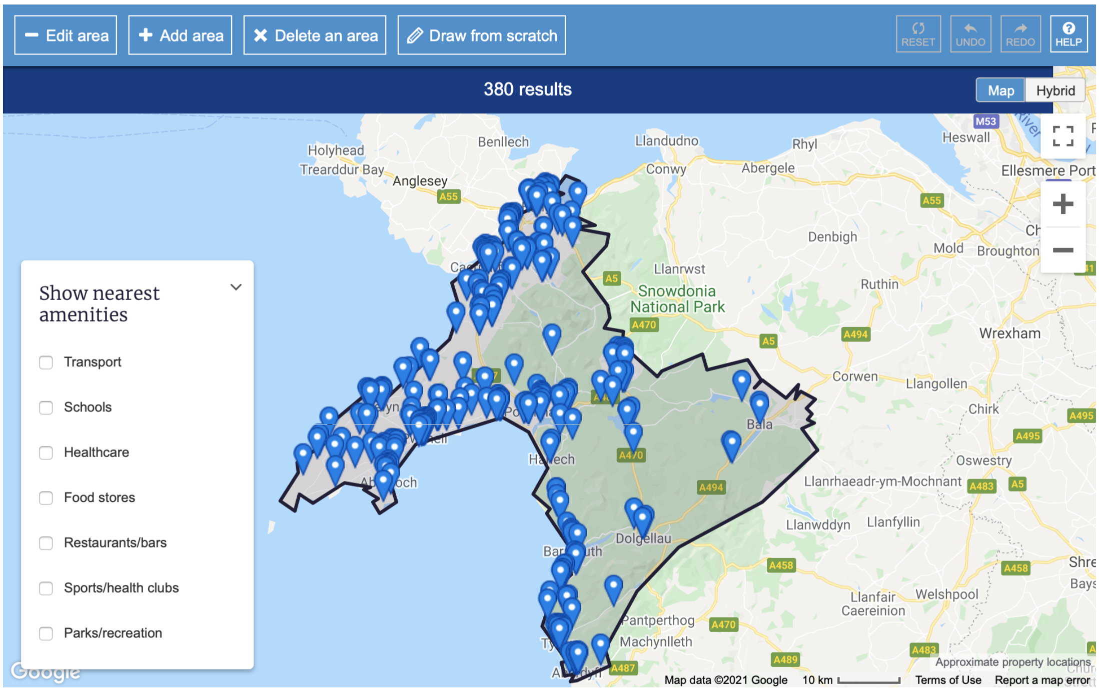


Figure 3.4: Primelocation's Smart Maps

3.2 Features and Services

An investigation has been conducted into the different features and services that have been used for jobs and housing websites. The purpose is to gain a better understanding of what is currently out there in the marketplace, what features are being used and why they have been selected to help promote listings to potential customers.

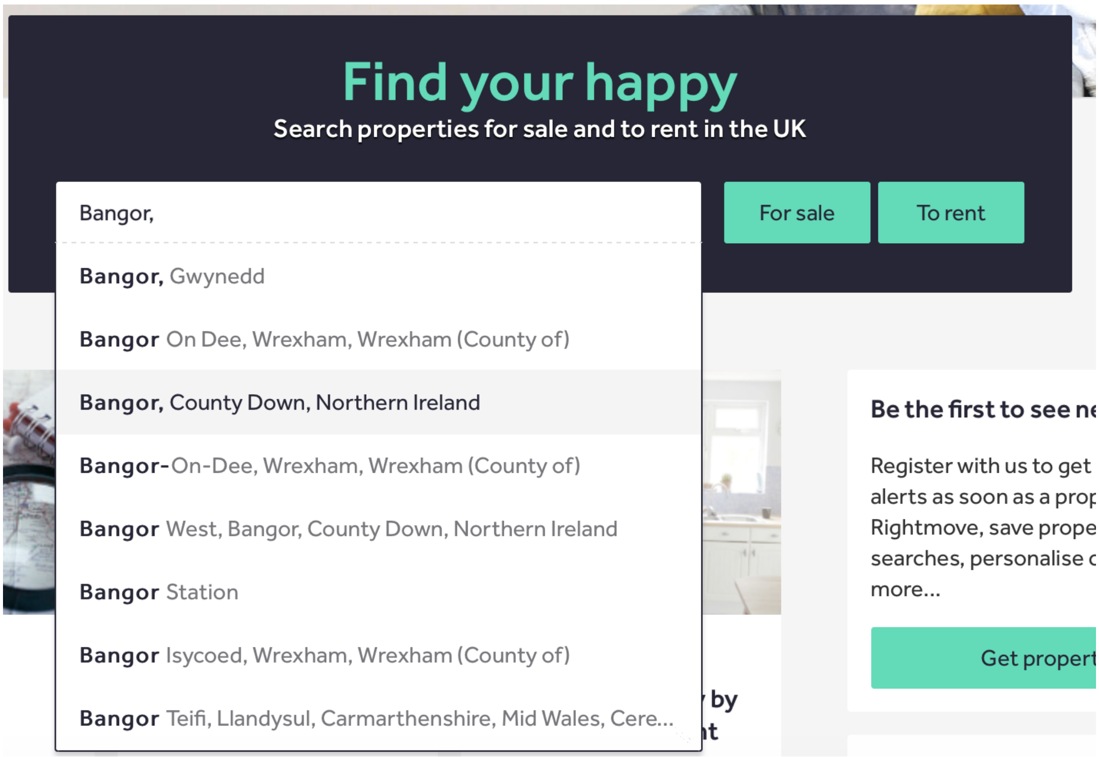
This investigation has shown the following features are being used:

- keyword search;
- advanced search options;
- distance filters;
- saved searches'
- blog articles / information'
- property alerts;
- iOS/Android app;
- property guides / resources;
- map based interface;
- place search;
- job alerts via email.

These will now be discussed in more detail in the sections below.

3.2.1 Keyword Search

In traditional database applications, queries are fully specified by structured queries. In this regard for the housing and job websites, the first task in our investigation was to perform a keyword search as the information is not fully specified by a structured query. A keywords search produces query results that gather relevant information that is usually fragmented and scattered across multiple areas [31]. This feature was selected as it is a known element of searching of stored information, and also provides the additional ability to do incremental searching as an option as shown in figure 3.5. This is where the user has the ability to search for a location of a job or housing and a keyword of a job occupation.


Find your happy

Search properties for sale and to rent in the UK

Bangor,

Bangor, Gwynedd

Bangor On Dee, Wrexham, Wrexham (County of)

Bangor, County Down, Northern Ireland

Bangor-On-Dee, Wrexham, Wrexham (County of)

Bangor West, Bangor, County Down, Northern Ireland

Bangor Station

Bangor Isycoed, Wrexham, Wrexham (County of)

Bangor Teifi, Llandysul, Carmarthenshire, Mid Wales, Cere...

Be the first to see ne

Register with us to get alerts as soon as a prop Rightmove, save prope searches, personalise c more...

Figure 3.5: An example of a keyword Search — Rightmove website

Incremental search is where the user typing text is provided with further real-time suggestions in real-time via the user interface as in the example shown in figure 3.5. As the user types text, one or more matches for the text are found and immediately presented to the user. This immediate feedback allows the user to stop short of typing an entire word they were looking for. This also allows the user to choose a closely related option from the list presented.

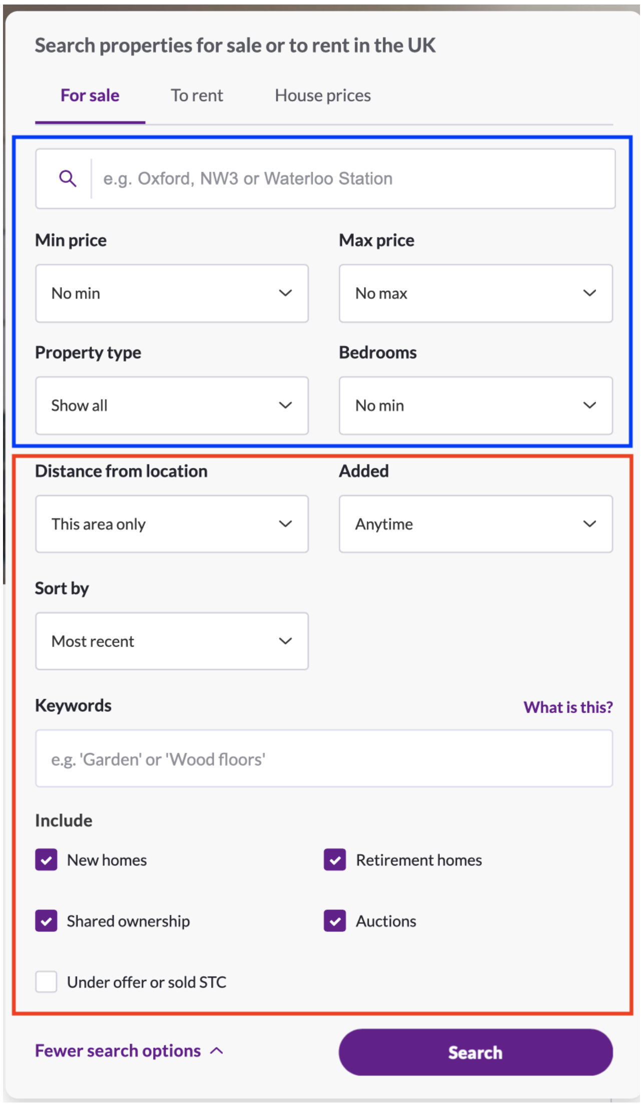
	Rightmove	Zoopla	Onthemarket	Primelocation	Trulia	Monster	JobSite	Reed	Cv-library	Total Jobs	Fish4Jobs
Incremental Searching	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗

Table 3.1: Incremental Searching on Job and Housing websites

Table 3.1 indicates which of the housing and job websites that were investigated provides Incremental Searching. Only one website, Fish4Jobs, does not provide this feature. This shows that incremental searching is being used broadly across housing and job websites.

3.2.2 Advanced Search Options

Keyword search is the default standard for modern applications of Information Retrieval (IR) such as Web search. Advanced search is the ability to take further search criteria options from the user. Allowing advanced search options not only is helpful for the query but it can produce more enhanced results as more specified information is being asked from the outset. This was the reason why this feature was included in our investigation. This approach is powerful although it does not support users well when they have complex questions or they may have insufficient pre-search knowledge. Additionally, the system which is being used maybe poorly-defined or has unpredictable indexing [155].



Search properties for sale or to rent in the UK

e.g. Oxford, NW3 or Waterloo Station

Min price: No min

Max price: No max

Property type: Show all

Bedrooms: No min

Distance from location: This area only

Added: Anytime

Sort by: Most recent

Keywords: e.g. 'Garden' or 'Wood floors'

Include

☑ New homes

☑ Retirement homes

☑ Shared ownership

☑ Auctions

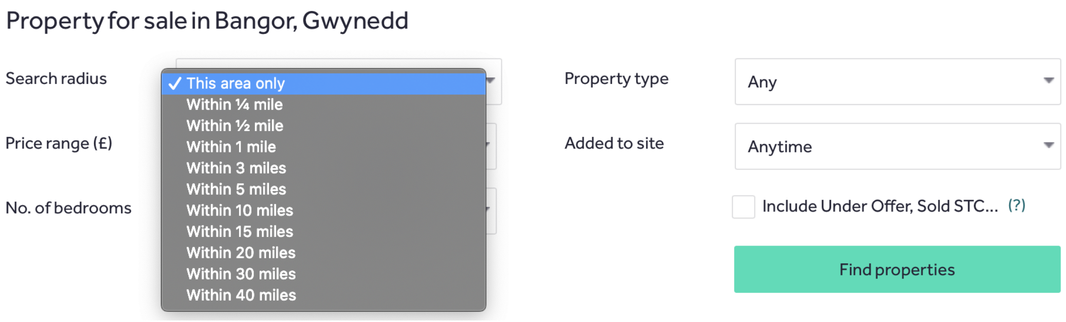
□ Under offer or sold STC

Figure 3.6: An example of standard search and advanced search options — Zoopla website

In figure 3.6, the options in the blue outlined square are the keyword search options available to the user upon visiting the website. The options in the red outlined square are the advanced search options. This is an example of where the user can enter further information of what was initially loaded to the user upcoming arriving to the website.

3.2.3 Distance Filters

In housing and job websites, the ability for the user to expand the distance from the original set query is a useful way for them to search for more relevant results. If the user were to take the keyword search of ‘Bangor, Gwynedd’, a useful feature many websites provides is to allow the the user to expand the search location zone as shown in the example in figure 3.7.



Property for sale in Bangor, Gwynedd

Search radius: ✓ This area only

Property type: Any

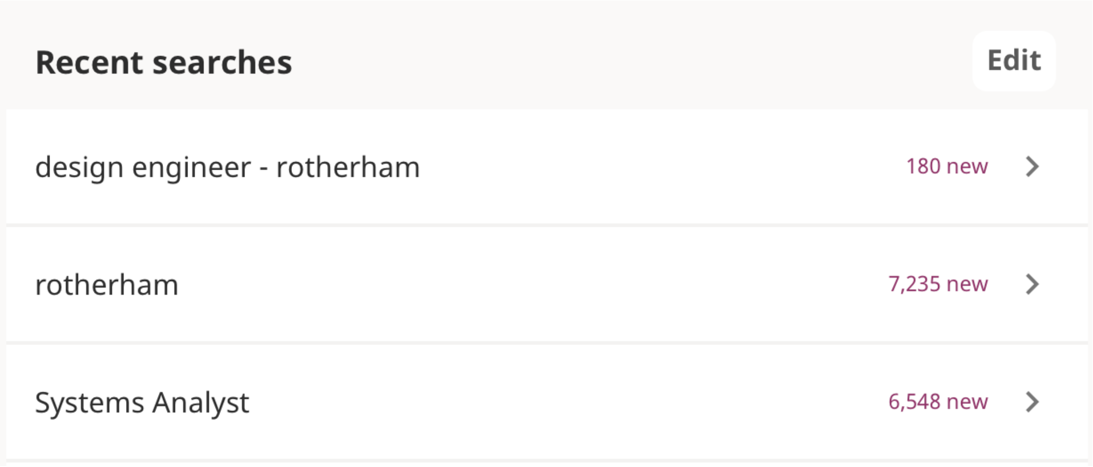
Added to site: Anytime

□ Include Under Offer, Sold STC... (?)

Figure 3.7: An example of the use of Distance feature — Rightmove website

3.2.4 Saved Searches

Saved searching is the ability of users for the website to keep a record of what they have been searching for so that they can refer back to past searches. An example of this is shown in figure 3.8. This feature also gives an indication whether there have been any new listings since they have last visited.


Recent searches

Edit

design engineer - rotherham 180 new ›

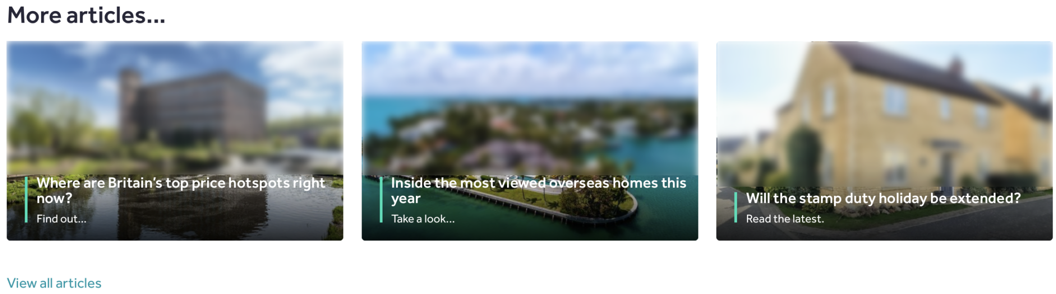
rotherham 7,235 new ›

Systems Analyst 6,548 new ›

Figure 3.8: Saved Searches — Indeed website

3.2.5 Blog Articles / Information

Blog Articles and information about the company is a feature that potential customers could use to find further information about the company, trends or interesting property news. An example of this is shown in figure 3.9 [4]


More articles...

Where are Britain's top price hotspots right now?
Find out...

Inside the most viewed overseas homes this year
Take a look...

Will the stamp duty holiday be extended?
Read the latest.

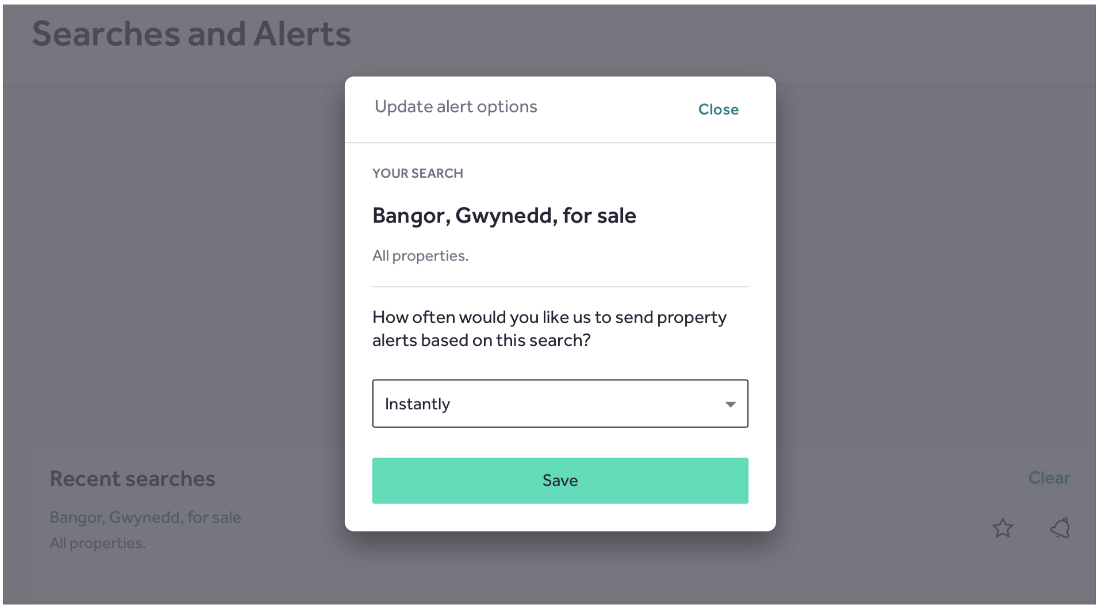
View all articles

Figure 3.9: Blog Articles — Rightmove website

3.2.6 Property Alerts

Property Alerts is the ability to be informed of the latest housing that gets listed on the website. This can be helpful for potential customers seeking out new properties on the market as soon as they become available. This is shown in figure 3.10.

[4] https://www.rightmove.co.uk



Searches and Alerts

Update alert options

YOUR SEARCH

Bangor, Gwynedd, for sale

All properties.

How often would you like us to send property alerts based on this search?

Instantly

Recent searches

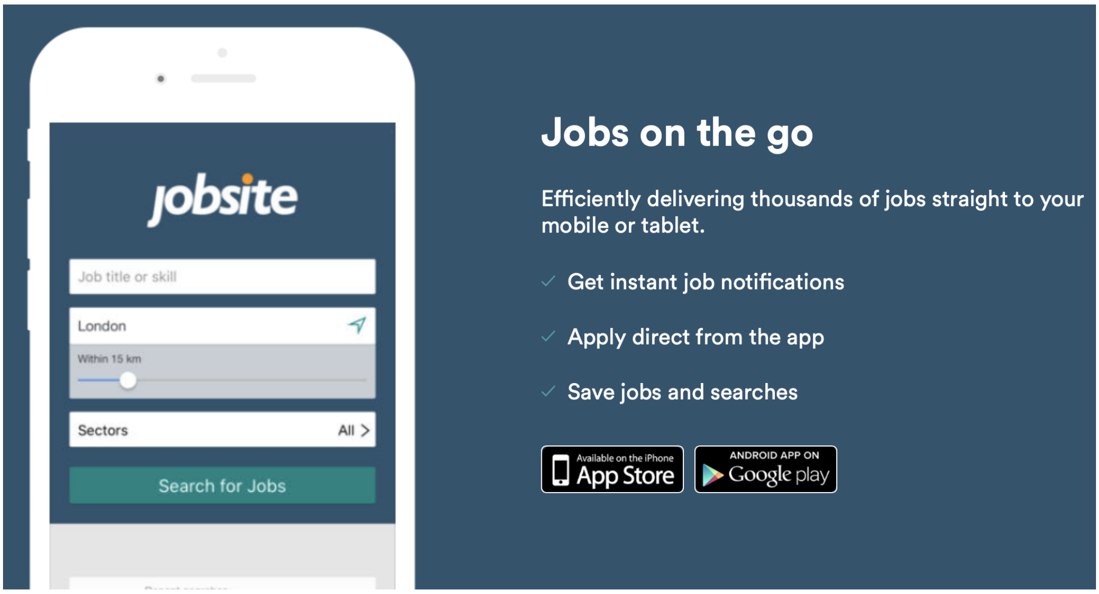
Bangor, Gwynedd, for sale

All properties.

Figure 3.10: Property Alerts — Rightmove website

3.2.7 iOS/Android App

To expand getting more customers, it is important to have a mobile application to view websites listings. This could benefit users that do not have a laptop or a desktop machine. This feature was decided by the company partners as without the use of a bespoke mobile application it could limit the potential customers who can view the listings. Jobsite advertise on the homepage there iOS and Android application. This is shown in figure 3.11 [5].



Jobs on the go

Efficiently delivering thousands of jobs straight to your mobile or tablet.

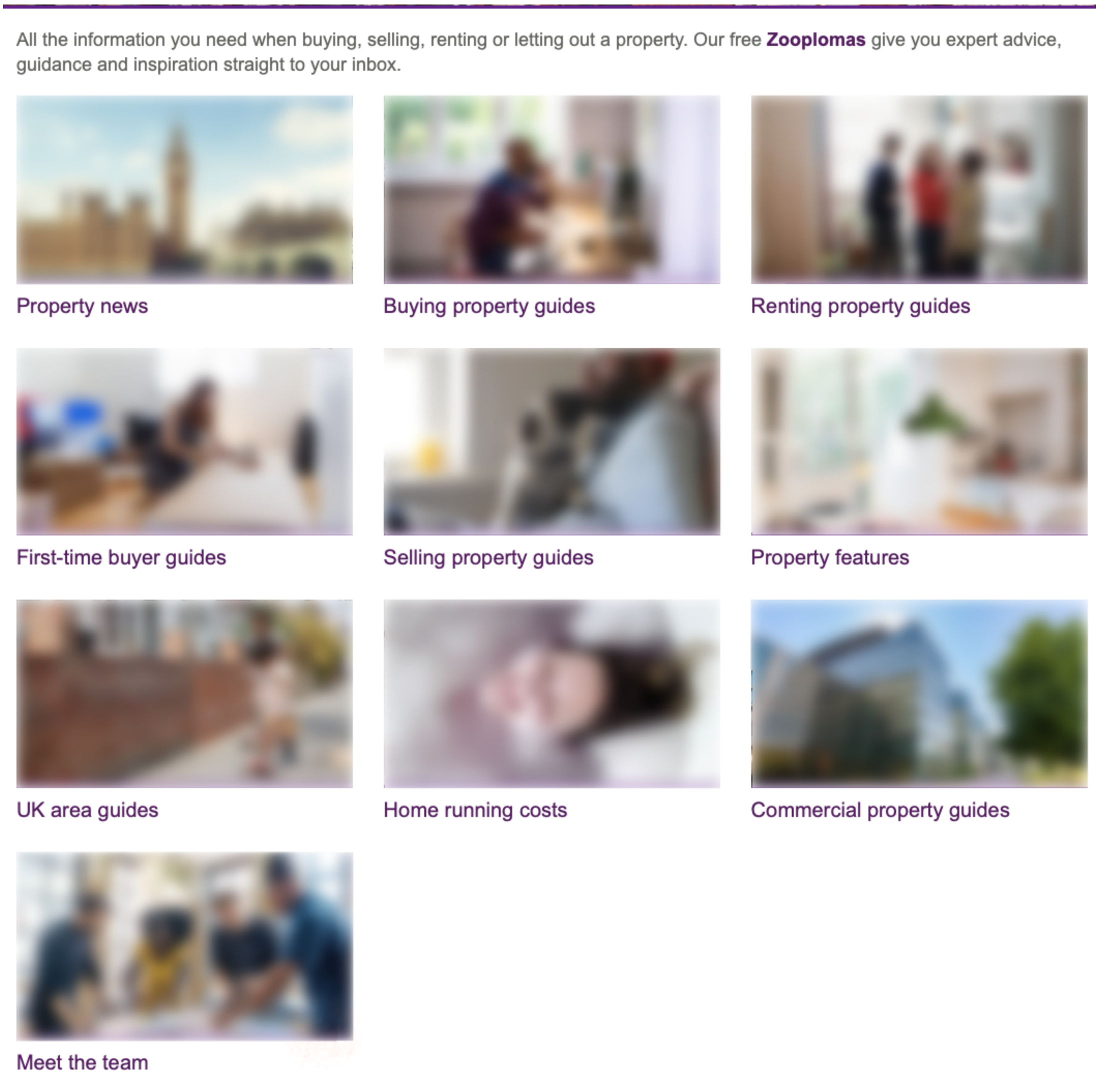
- ✓ Get instant job notifications
- ✓ Apply direct from the app
- ✓ Save jobs and searches

Figure 3.11: iOS App — Jobsite website

[5] https://www.jobsite.co.uk

3.2.8 Property Guides / Resources

A housing website providing guides to potential buyers can be helpful. In this regard, new home buyers could be needing a helping hand from the company in question, so having resources available can show credibility and trust from the company to the customer. An example from Zoopla is shown in the figure 3.12.



All the information you need when buying, selling, renting or letting out a property. Our free **Zooplomas** give you expert advice, guidance and inspiration straight to your inbox.

Figure 3.12: Property Guides / Resources — Zoopla website

3.2.9 Map Based Interface

This project investigates the use of a Map Based Interface for jobs and housing listings. It is important to explore housing and job websites on if they have this built in already. It is noted that housing websites provide a Smart Map, this being the ability to see housing on a map where as jobs do not. An example of a Smart Map that Zoopla uses can be seen in figure 3.13. The figure uses pins to show the locations of the housing and has the ability of showing points of interest nearby [6]

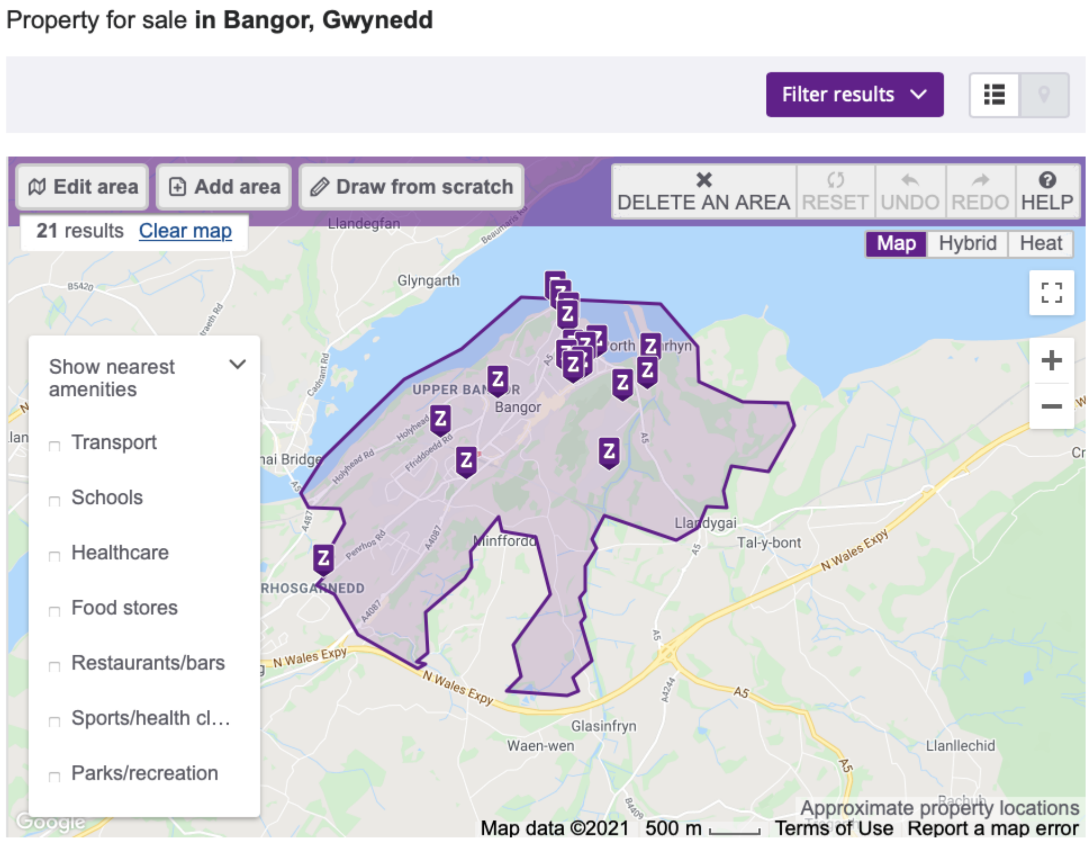


Figure 3.13: Smart Map — Zoopla website

3.2.10 Place Search

Instead of having the ability to search a keyword, various websites have the ability for the user to search by a place. An example can be seen in figure 3.14. This is place searching on the homepage for Reed [7].

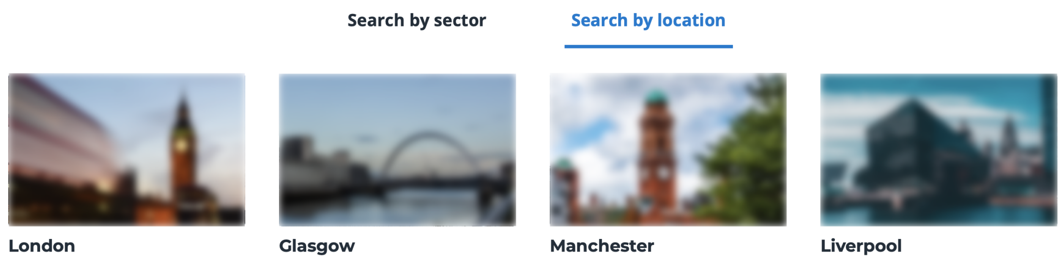


Figure 3.14: Place Search — Reed website

[6] https://www.zoopla.co.uk
[7] https://www.reed.co.uk

3.2.11 Quick Searches / Featured Jobs

The ability to quickly search for example a sector of industry in a job can be beneficial to the user. A user could be in a situation where they just want to browse in different areas and not think of specific keywords to use. Reed has put quick searching available on the homepage. Reed's example can be seen in figure 3.15. Reed has placed trending jobs in the form of a button, which will take you to a listing of the criteria pressed.

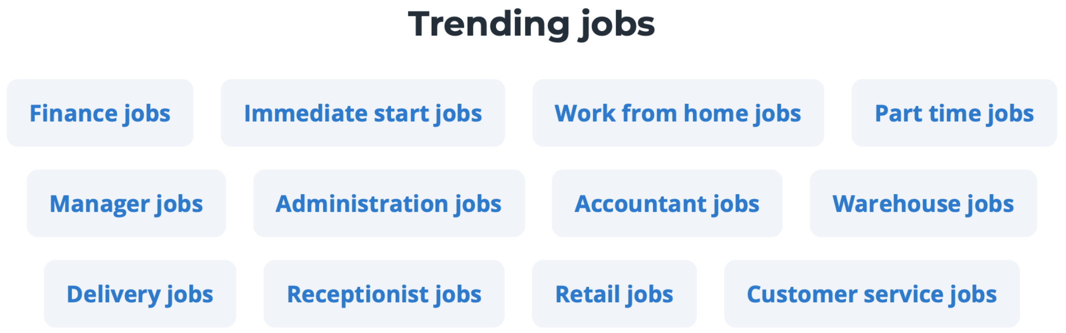
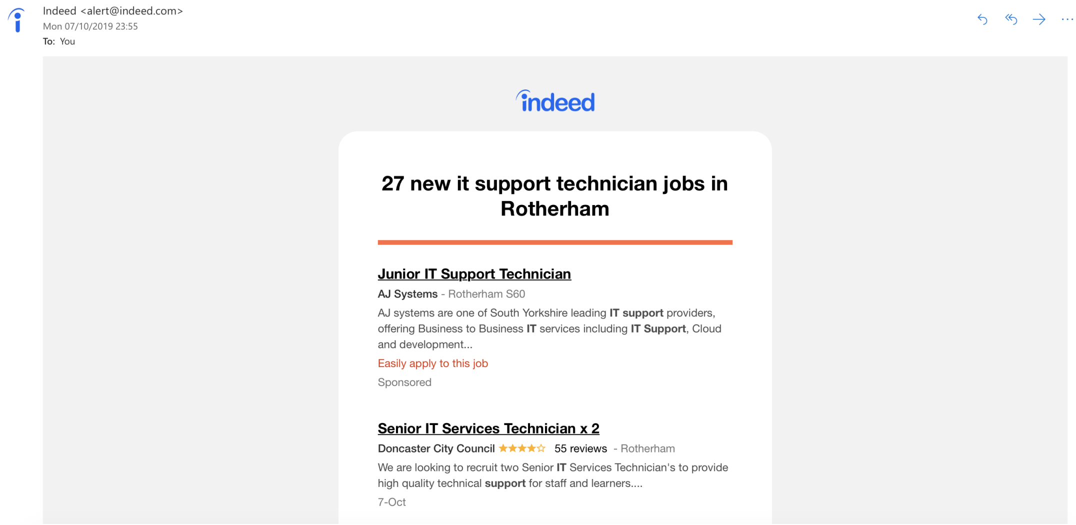


Figure 3.15: Quick Searches — Reed website

3.2.12 Job Alerts via Email

The ability to get job alerts via email is another area of how potential customers could be alerted of new jobs. The item was decided by the company partners as the ability to show listings can be performed outside the usual way of a website. Figure 3.16 shows from Indeed the email listings of new jobs that a user has signed up for a specific area and keyword.



Indeed <alert@indeed.com>
Mon 07/10/2019 23:55
To: You

indeed

27 new it support technician jobs in Rotherham

Junior IT Support Technician

AJ Systems - Rotherham S60

AJ systems are one of South Yorkshire leading **IT support** providers, offering Business to Business **IT** services including **IT Support**, Cloud and development...

Easily apply to this job

Sponsored

Senior IT Services Technician x 2

Doncaster City Council ★★★★☆ 55 reviews - Rotherham

We are looking to recruit two Senior **IT** Services Technician's to provide high quality technical **support** for staff and learners....

7-Oct

Figure 3.16: Job Alerts via Email — Indeed website

3.3 Features and Services of Housing Websites

Website	URL	Date Service Started	Amount Of Users
Rightmove	https://www.rightmove.co.uk	January 2000	250 million unique visitors each month
Zoopla	https://www.zoopla.co.uk	January 2008	Approx 39 million unique visitors each month
Onthemarket	https://www.onthemarket.com	January 2015	Approx 22 million unique visitors each month
Primelocation	https://www.primelocation.com	January 2001	Approx 10 million unique visitors each month
Trulia	https://www.trulia.com	January 2004	Approx 59 million unique visitors each month

Table 3.2: Key Information of Housing Websites

The criteria for deciding the features on housing websites is to review each website and make a note of all features that have been used on the website. This is needed to understand what is being provided by these websites and what is not. Listed in table 3.2 are some key information about the different housing websites being investigated.

Features	Rightmove	Zoopla	Onthemarket	Primelocation	Trulia
Keyword Search	✓	✓	✓	✓	✓
Advanced Searches	✓	✓	✓	✓	✓
Distance	✓	✗	✗	✗	✓
Saved Searches	✓	✓	✗	✗	✓
Blog Articles / Information	✓	✓	✓	✓	✓
Property Alerts	✓	✓	✓	✓	✓
iOS/Android App	✓	✓	✓	✓	✓
Property Guides / Tools / Resources	✓	✓	✗	✓	✓
Map Based Interface / Property Maps / Smart Maps	✓	✓	✓	✓	✓

Table 3.3: Features and Services with Housing Websites

The results from the table 3.3, indicate that the majority of every housing website checked have many features from keyword search to mobile apps. Rightmove and Trulia were the only two websites that had all the features and services mentioned. Zoopla, Onthemarket and Primelocation do not allow the ability to search with distance filters.

Each housing website notably has a map-based interface, which gives the ability to see listings on a map. Each website is similar in regards to having pins. The pins on each website are the logo of the business. The map-based interfaces also provide the ability for users to set different way points to find results in a region and only in that region. The map-based interfaces also provide the opportunity at looking at areas of interest.

3.4 Features and Services of Job Websites

Websites	URL	Date Service Started	Amount Of Users
Monster	https://www.monster.co.uk	January 1999	Approx 10 million unique visitors each month
Jobsite	https://www.jobsite.co.uk	January 1995	Approx 750,000 unique visitors each month
Reed	https://www.reed.co.uk	January 1995	Approx 8 million unique visitors each month
CV-library	https://www.cv-library.co.uk	January 2000	Approx 8 million unique visitors each month
Total Jobs	https://www.totaljobs.com	January 1999	Approx 14 million unique visitors each month
Fish4Jobs	https://www.fish4.co.uk	January 1999	Approx 350,000 unique visitors each month
Indeed	https://www.indeed.co.uk	November 2004	250 million unique visitors each month

Table 3.4: Key Information of Job Websites

The criteria for deciding the features on jobs websites is to review each website and make a note of all features that have been used on the website. This is needed to understand what is being provided by these websites and what is not. Listed in table 3.4 are some key information about the different job websites being investigated.

Features	Monster	Jobsite	Reed	Cv-library	Total Jobs	Fish4Jobs	Indeed
Keyword Search	✓	✓	✓	✓	✓	✓	✓
Place Search	✓	✓	✓	✓	✓	✓	✓
Quick searches	✗	✗	✓	✓	✓	✗	✓
Distance	✗	✓	✗	✓	✓	✓	✓
Upload/Build CV	✓	✓	✓	✓	✓	✓	✓
Career Advice and Resources	✓	✗	✓	✓	✓	✓	✓
Saved Searches/Jobs	✓	✓	✓	✓	✓	✗	✓
Job Alerts via Email	✓	✓	✓	✓	✓	✓	✓
Blog Articles / Information	✓	✓	✓	✓	✓	✗	✓
Featured Jobs	✓	✓	✓	✓	✓	✓	✓
iOS/Android App	✓	✓	✓	✓	✓	✗	✓
Map Based Interface	✗	✗	✗	✗	✗	✗	✗

Table 3.5: Features & Services with Job Websites

The results from table 3.5 indicate that the majority of every job website checked have many features from the keyword search to job alerts and mobile apps. Notably, the searched job websites don't have Map Based Interfaces, which allows users to use a map to set a radius to see different jobs within a region.

The potential reason why these job websites do not have a Map Based Interface is because they do not need to be as restrictive as housing. For example, searching with a query such as ‘java programmer’, results indicate multiple unrelated search queries which can frustrate the user as it’s not what they have queried for.

3.4.1 Job Websites Results

Queries	**Further Detail**	**Fish4Jobs (%)**	**Jobsite (%)**	**Indeed (%)**	**Monster (%)**
Fashion Jobs	Contract Type: Permanent. Hours: Full Time	25.00	94.00	56.00	46.00
computer science	Contract Type: Permanent. Hours: Full Time	82.50	97.00	87.50	24.00
accountant	Wales	60.00	62.50	60.00	62.00
accountant	Within 25 miles of North Wales. Sectors: Accountancy	100.00	59.80	78.00	68.00
academic librarian	Within 15 miles of North Wales. Sectors: Accountancy	42.85	0.00	43.00	45.00
animator jobs	Full time, UK	0.00	64.00	100.00	70.00
estate agents	35 miles of North Wales	50.00	76.00	31.00	80.00
web designer	Part time, 15 miles within LL57	38.00	92.00	60.00	87.00
engineering manager	Full time, £30,000, Wales	75.00	64.00	48.00	61.00
finance advisor	Contract Type: Part Time, within 20 miles of Cardiff	95.00	53.00	31.00	41.00
Average (%)		**57.00**	**66.00**	**59.00**	**58.00**

Table 3.6: Precision Results for Job Websites

In the table 3.6, these are the queries that were randomly selected to be searched by the company partners for the the job websites. The first search results page of each listing were noted and then checked to see if they were relevant to the query in question. Precision was used to calculate the result.

In table 3.6, the queries column is the keyword that has been entered on each of the job websites. The further detail column is any additional options on the search, known as advanced searching. Precision is used and calculated as described in section 2.8 to evaluate the listing results. The following will now discuss each of these websites.

Overview of Fish4: Fish4 allows you to enter keywords to find jobs within a region, salary amount, contract type and sectors. Although after applying several filters to narrow down a result, it either returns with very limited results or the website is unable to find results. Notably, if the user were to remove the filters, the listings show results of what the filter should pick up on when applied. A significant amount of results do not have any relevance to what has been searched for. The ability to provide multiple criteria in advanced searching provides a limited amount of search results. Despite the listings having the criteria required by the filter, the results do not show the relevant results.

Overview of JobSite: Jobsite tackles the accuracy better compared to Fish4. Keywords that are provided by the user are highlighted in bold to emphasis the relevance to what the user has requested. A quarter of the results from the results page show unrelated query results. This is where results are showing outside the scope of what the user entered.

Overview of Indeed: Indeed is one of the most popular job-hunting websites [8]. Notably, the results were found to be more focused on basic keyword searches. When adding further options to the query, the results changed considerably. This led to fewer quality results being returned.

Overview of Monster: The results indicated that the results were accurate without advanced searching. Using advanced searching such as applying a region, this provided results outside of the region which highly affected the precision results.

3.4.2 Housing Websites Results

In the table 3.7, these are the queries that were randomly selected to be searched by the company partners for the the housing websites. The first search results page of each listing were noted and then checked to see if they were relevant to the query in question. Precision was used to calculate the result.

Query details					**Precision (%)**		
Location	**Price Range**	**Property Type**	**Number of bedrooms**	**Number of bathrooms**	**Rightmove**	**Zoopla**	**Primelocation**
LL57	£80,000	Semi-detached	3	1	100	100	100
LL29	£120,000	Detached	4	2	100	100	100
Conwy	£35,000 — £50,000	Any	1	1	100	100	100
Colwyn Bay	£50,000	Any	2	1	100	100	100
Rhos-on-Sea	£120,000 — £140,000	Detached	4	2	100	100	100
Bangor	£60,000 — £100,000	Terraced	2	1	100	100	100
Llanberis	£50,000	Semi-detached	2	1	100	100	100
Abersoch	£40,000 — £160,000	Detached	4	2	100	100	100
Average					100	100	100

Table 3.7: Precision Results for Housing Websites

[8] http://www.alexa.com/siteinfo/indeed.co.uk

The following will now discuss each of these websites.

Overview of Rightmove, Zoopla & Primelocation: Integration of a map based interface has significantly improved the results provided. The interface of results provides several views of either a list, grid or a map. On the right hand side of the website, an outline is provided of a Google Map where results are within the radius circle and outside of that area results aren't provided. This is more effective for the user as the query of what they are requesting is getting fulfilled without clutter.

3.4.3 Precision of existing web-based services of adhoc queries

Further investigation was undertaken to determine the precision of multiple job websites. The purpose for this was to determine the effectiveness of existing websites. Precision is calculated as discussed in section 2.8.

An investigation was conducted into various job websites to find out the precision of the jobs that were returned for specific search queries. The criteria that set out to identify the job websites were as follows:

- Was the postcode found in the location field of the job on the website? Y/N.
- Was the postcode explicitly found in the text? Y/N.
- Was the name of business/organisation explicitly in the text? Y/N.
- Was the postcode needed to be found manually by searching elsewhere? Y/N.
- Is the postcode Unknown due to the listing being an agency? Y/N.
- Is location Unknown when doing manual searching of the web?

The three job websites that were investigated were Indeed, Reed and Jobsite. A total of 100 adhoc queries were made per website. The full detailed results can be found below [9].

[9] https://www.dropbox.com/s/n0s01jl1v5oa3wi/QueriesIndeed3.xlsx?dl=0

3.4.4 Indeed Results

In table 3.8, the investigation into Indeed's website job data indicates how out of 100 queries taken to find the location data, just below half of results were agency listings. These listings are vague on information and do not provide the user with specific location information. It also adds complications to as due to the vague information provided, it's hard to figure out where the job is located.

	Postcode found in location	Postcode explicitly found in text	Name explicitly in text	Found postcode manually	Unknown, agency	Unknown
Yes (%)	1	6	52	49	45	2
No (%)	99	94	48	51	55	98

Table 3.8: Accuracy of identifying location data from Indeed website data

3.4.5 Reed Results

In the table 3.9, the investigation into Reed's website job data indicates how out of 100 queries taken to find the location data, no postcode could be found in the location field of the 100 listings. 66 of the listings came from agencies for which the specifc job location could not be found.

	Postcode found in location	Postcode explicitly found in text	Name explicitly in text	Found postcode manually	Unknown, agency	Unknown
Yes (%)	0	7	31	31	66	2
No (%)	100	93	69	69	34	98

Table 3.9: Accuracy of identifying location data from Reed website data

3.4.6 Jobsite Results

In table 3.10, the investigation into Jobsite's website job data indicates how out of 100 queries taken to find the location taken, the ability to find the postcode in the listing itself found 28 results where the postcode could be found in the location field. 23 results could find the postcode explicitly in the text and 30 results could be found in the name of the listing itself.

	Postcode found in location	**Postcode explicitly found in text**	**Name explicitly in text**	**Found postcode manually**	**Unknown, agency**	**Unknown**
Yes (%)	28	23	30	4	67	2
No (%)	72	77	70	96	33	98

Table 3.10: Accuracy of identifying location data from Jobsite website data

3.4.7 Conclusion

Overall the results from searching indicate for job websites to find a location of a job, the specifics of the whereabouts of a listing is difficult. Each of the job websites investigated an average of 60% of listings were unknown due to them being agency classed jobs. In this case, it would require the user to enquire about the job and wait for a response back from the agency. For an average of 88% of the listings overall, the user cannot find the postcode explicitly in the text. In order to find the postcode and location details further, a manual search outside of the host website has to be done. An average of 28% found listings location details from outside of the website.

The results strongly motivate the need for an integrated approach to job and housing searches as advocated for this project. This is because the quality of job information currently being provided is poor and also that there is no platform that we are aware out there that can provide jobs and housing together. The results show that accommodation web-based search does well, as the incentive of providing full accurate details to the user upon the search is wanted. In contrast, job sites provide minimal information of where the location of the jobs are and require you to get in contact for further information.

Chapter 4

A System for Web Mining of Job and Housing information using PPM

4.1 Summary

This chapter describes an investigation into a new approach of web mining using the PPM character-based text compression scheme. It describes the design, implementation and evaluation of a web mining system called Merlin that uses the new approach. For the evaluation, the focus is on specifically two types of websites: jobsites; and sites that provide housing information. The effectiveness of the character based approach at identifying unstructured text from the web is analysed using standard measures (Recall, Precision, Accuracy and F-measure) by comparing it to ground truth data. The Merlin system's standard measures are then compared to another state of the art toolkit.

4.2 Introduction

This chapter investigates a new approach to web mining using the PPM Prediction by text compression scheme. PPM has been used as it effective at many natural language processing applications and therefore an investigation to see how it performs at web mining with a specific focus on housing websites and job websites is of interest as it relates to the objective of developing and evaluating a named entity tagger related to jobs and accommodation.

The challenge with information extraction from websites is that they are noisy and unstructured web based sources. Websites also change over time meaning standard approaches (such as regular expressions where a specific pattern is used to extract specific data from the web page) often do not work if the layout or identification of the elements change [118].

The purpose of this chapter is to evaluate the effectiveness of a novel approach to web mining by applying Prediction by Partial Matching (PPM) on website data, in particular looking at accommodation websites and job websites. Specifically, the approach adopts PPM techniques at marking up unstructured data using the Tawa toolkit [150].

The rest of this chapter is organised as follows. The design of the Merlin system is described in the next section. This is followed by the implementation of the Merlin system and the evaluation of the Merlin system.

The Merlin system's standard measures are compared to spaCy. spaCy is an open-source software library for advanced natural language processing.

4.3 Prediction by Partial Matching (PPM)

Prediction of Partial Matching (PPM) was introduced by Cleary and Witten [35]. It is an adaptive lossless text compression method that processes characters in the text in a sequential manner.

The PPM compression algorithm applies a statistical model that uses the number of previous symbols to determine the model's maximum order to predict the next symbol. For example, if the PPM model's maximum order is 4, the probability of the upcoming symbol will be estimated based on the four previous symbols. There are several variations of PPM, such as PPMA and PPMB [35], PPMC [97], PPM [36] and PPMO [157].

Predictions are reduced to symbol rankings. Each symbol could be a letter or a character ranked before it is compressed and the ranking system determines the corresponding codeword. The ranking system determines the corresponding codeword (and therefore, the compression rate). In compression algorithms, it is known that the ranking is equivalent to probability mass function estimation. Given the previous letters (or given a context), each symbol is assigned with a probability.

For PPM, a variable order Markov-based model is updated dynamically as the text is processed with both the encoder and decoder maintaining the same model at each stage of the encoding and decoding processes. In order to predict the upcoming character in the text, there is finite context used to predict the upcoming character [88].

PPM uses a fixed full context to make its initial prediction. (This defines the "order" of the model).

It is known that text compression experiments with English and other natural language texts have shown that a fixed maximum context length of 5 (an order of 5 model) works the best. The method estimates probabilities for the upcoming character.

The model uses an "escape" mechanism according to the terminology defined in the PPM literature, that ultimately smoothes the probability estimates by backing-off to a shorter context when novel characters are experienced (e.g., those with zero probabilities).

The backing-off process may need to be undertaken multiple times in order to be able to find the context where the character can be predicted. For characters that may not have been seen anywhere previously in the text, the default order -1 context is used to predict every character with equal probability.

Many escape methods have been devised in the past and have been used to define how the escape probability is estimated. These escape methods are described in the literature as PPMA and PPMB, for example.

The PPMC variant was developed by Moffat [97] and has now become the benchmark. The probability of PPMC is based on using the number of characters that have occured before, called the number of types:

$$e(X) = \frac{t(X)}{f(X) + t(X)} \text{ and } p(x_i|X) = \frac{c(x_i|X)}{n(X) + t(X)} \tag{4.1}$$

where $e(X)$ represents the probability of the escape symbol for context X, $p(x_i|X)$ denotes the probability for character x_i given context X, $c(x_i|X)$ is the number of times the context X was followed by the character x_i, $f(X)$ is the total number of times that the context X has occurred and the $t(X)$ denotes the total number of types of the predictions in that context [88].

There is another variant called PPMD variant by Howard in 1993 [69]. This variant is similar to the PPMC variant with the exception that each count is incremented by a 1/2:

$$e(X) = \frac{t(X)}{2f(X) + t(X)} \text{ and } p(x_i|X) = \frac{2c(x_i|X) - 1}{2f(X)}. \qquad (4.2)$$

Usually text compression algorithms are two types: statistical or dictionary-based [123]. The dictionary based methods use a dictionary data structure to save fragments of text found in the original text. Whilst reading through the fragmented text, if there is a match found in the dictionary, then a pointer to the dictionary entry is made in the compressed stream, otherwise a new entry is added to the dictionary.

Statistical methods create statistical models of the text, where probabilities are allocated to the input symbols. When the models are created, the coding is performed which uses the model to encode the symbols. There are two different ways of creating the models: context or frequency. Using a frequency model, the text symbols are allocated probabilities which are dependent upon how often the symbol has appeared in the text. Text symbols that appear more often are given shorter codes in most cases.

Context-based models, where the text symbol appears following a usually fixed-length prior sequence of symbols (called the "context"), are also used to estimate symbol probabilities. The encoder relies on the symbol's past appearance earlier in the text to calculate the probability. Context based models usually use an "order-*N*" Markov model, where *N* is the number of symbols considered prior to the examined text symbol.

An example of a context-based compression system is the Prediction by Partial Matching (PPM) text compression algorithm. PPM is regarded as one of the most effective text compression algorithms available [89] [94]. PPM has the ability of streaming compression, meaning the statistical models are adaptively updated as the compression occurs. PPM is an example of an adaptive statistical based compression system, that has the characteristics of carrying out the two main processes such as coding and modeling.

The model builds a table of probabilities of all the symbols encountered in the context so far, then uses this to predict what the next symbol might be. The compressor encodes the actual symbol using the probability distribution that is used by the model. This uses a Markov based-approach where the last few characters in the input stream, known as the context, are used to predict what the next character is. The number of characters used in the context is then used to define the order of the model. For example, a context of length 1 is used for an order 1 model.

As each symbol is encoded, the probability distribution models for each order are then created. The models are then combined into a single one using the escape mechanism, which predicts the upcoming character using the highest fixed order first. Although this may then "backs off" to a lower order model if the upcoming character has not been seen.

The compression code length h for encoding a symbol s_i in bits using an order 5 PPM model can be represented by equation 4.3:

$$h(s_i) = -log_2 P(s_i|s_{i-5}s_{i-4}\dots s_{i-1}). \tag{4.3}$$

PPM usually starts with the highest order, k, that is requested by the user. Whilst observing the symbols, if a new symbol is observed that has not been seen before, an escape symbol is issued. The escape sequence then tells the PPM to reduce the order by 1 until the symbol is no longer novel. This escaping process may have to repeat if the shorter contexts still aren't able to predict the symbol.

If the order k reaches -1, then the same probability of $\frac{1}{|A|}$ is assigned to all of the characters. The A represents the size of the alphabet. The effect of the escape mechanism is to "smooth" the probability estimates.

PPM also uses the technique of "full exclusion" where, when the escape mechanism is used, all the symbols that were already predicted by higher orders are excluded.

There are two prominent variants to PPM that have been made that use different methods for calculating the escape probabilities, method C and method D. (These are called PPMC and PPMD). Two equations are used for calculating the prediction probabilities, one for calculating the escape probability, e, and one to calculate the probability of a symbol occurring, $p(s)$.

The equations for PPMC are shown in equations 4.4 and 4.5. The equations for PPMD are shown in 4.6 and 4.7 [4]:

$$e_{PPMC} = \frac{t}{n+t} \tag{4.4}$$

$$p(s)_{PPMC} = \frac{c(s)}{n+t} \tag{4.5}$$

$$e_{PPMD} = \frac{t}{2n} \tag{4.6}$$

$$p(s)_{PPMD} = \frac{2c(s)-1}{2n} \tag{4.7}$$

In the context of the above equations:

- t is the number of types that follows the context.
- n is the number of times a context has occurred.
- $c(s)$ is the number of times a context was followed by the symbol s.

PPMD increments the symbol count by 2 when a previously seen symbol is encountered, but increments the escape count by 1 and then assigns the initial symbol count of 1 for symbols that have not been seen before in the context. PPMC on the other hand estimates the probability for each symbol that uses the raw frequency and assigns the number of types t to the escape count for estimating the probability of an escape occurring when an upcoming symbol is previously unseen in the context.

In order to illustrate our PPMD works, it has been shown that PPMD outperforms PPMC in most compression experiments. Table 4.1 shows what the PPMD model looks like after it has encountered the string *llanfairfechan*.

Order $k = 2$					Order $k = 1$					Order $k = 0$				Order $k = -1$			
Predictions			c	p	Predictions			c	p	Predictions		c	p	Predictions		c	p
ll	→	a	1	$\frac{1}{2}$	l	→	l	1	$\frac{1}{4}$	→	l	3	$\frac{3}{28}$	→	A	1	$\frac{1}{\lvert A\rvert}$
	→	Esc	1	$\frac{1}{2}$		→	a	1	$\frac{1}{4}$	→	a	5	$\frac{5}{28}$				
						→	Esc	2	$\frac{2}{4}$	→	n	3	$\frac{3}{28}$				
la	→	n	1	$\frac{1}{2}$						→	f	3	$\frac{3}{28}$				
	→	Esc	1	$\frac{1}{2}$	a	→	n	3	$\frac{2}{6}$	→	i	1	$\frac{1}{28}$				
						→	i	1	$\frac{1}{6}$	→	r	1	$\frac{1}{28}$				
an	→	f	1	$\frac{1}{2}$		→	Esc	2	$\frac{2}{6}$	→	e	1	$\frac{1}{28}$				
	→	Esc	1	$\frac{1}{2}$						→	c	1	$\frac{1}{28}$				
					n	→	f	1	$\frac{1}{2}$	→	h	1	$\frac{1}{28}$				
nf	→	a	1	$\frac{1}{2}$		→	Esc	1	$\frac{1}{2}$	→	Esc	9	$\frac{9}{28}$				
	→	Esc	1	$\frac{1}{2}$													
					f	→	a	1	$\frac{1}{4}$								
fa	→	i	1	$\frac{1}{2}$		→	e	1	$\frac{1}{4}$								
	→	Esc	1	$\frac{1}{2}$		→	Esc	2	$\frac{2}{4}$								
ai	→	r	1	$\frac{1}{2}$	i	→	r	1	$\frac{1}{2}$								
	→	Esc	1	$\frac{1}{2}$		→	Esc	1	$\frac{1}{2}$								
ir	→	f	1	$\frac{1}{2}$	r	→	f	1	$\frac{1}{2}$								
	→	Esc	1	$\frac{1}{2}$		→	Esc	1	$\frac{1}{2}$								
rf	→	e	1	$\frac{1}{2}$	e	→	c	1	$\frac{1}{2}$								
	→	Esc	1	$\frac{1}{2}$		→	Esc	1	$\frac{1}{2}$								
fe	→	c	1	$\frac{1}{2}$	c	→	h	1	$\frac{1}{2}$								
	→	Esc	1	$\frac{1}{2}$		→	Esc	1	$\frac{1}{2}$								
ec	→	h	1	$\frac{1}{2}$	h	→	a	1	$\frac{1}{2}$								
	→	Esc	1	$\frac{1}{2}$		→	Esc	1	$\frac{1}{2}$								
ch	→	f	1	$\frac{1}{2}$													
	→	Esc	1	$\frac{1}{2}$													
ha	→	n	1	$\frac{1}{2}$													
	→	Esc	1	$\frac{1}{2}$													

Table 4.1: PPMD model after processing the string *llanfairfechan* with maximum order of 2.

As an illustration of the operation of PPM, Table 4.1 shows the state of the four models with order k = 2, 1, 0 and -1 after the input string "*llanfairfechan*" has been processed. For each model, all previously occurring contexts are shown with their associated predictions, along with occurrence counts c and the probabilities p that are calculated from them. By convention, order $k = -1$ designates the bottom-level model that predicts all characters equally; it assigns them each probability $\frac{1}{|A|}$ where A is the alphabet used.

If the next character in the string to be is a, we must take the prediction $ll \rightarrow a$ using the order 2 context. Since the character a has been seen once before in the context ll, then a probability of $\frac{1}{2}$ will be assigned by using equation 4.7 as $c = 1$.

However, if the subsequent character has not been seen previously in the order 2 context (i.e. presuming the next letter would be n instead of a, say), it will be necessary to conduct an escape procedure or back off to a lower order. In this case, the escape probability will be $\frac{1}{2}$ (calculated by equation 4.7), and a lower order of 1 will then be used to make the prediction.

Prediction by Partial Mapping (PPM) is used in various natural language processing and text mining applications. Teahan and Harper [149] in 2003 used PPM to recognise the most relevant author of the text. Khmelev and Teahan [76] and Al-Kazaz and Teahan [75] in 2016 used PPM to perform the automatic cryptanalysis of cyphers and word segmentation in order to make the decoded text more readable. Altamimi and Teahan [10] in 2017 used PPM to classify gender. Alamri and Teahan [5] used PPM for the automatic correction of Arabic dyslexic text. As far as we know, no research before this study has used PPM for web mining purposes and information extraction.

4.4 Successful use of PPM in other tasks

Prediction by Partial Mapping has had many successful uses in other tasks. Altamimi [10] produced a paper on gender and authorship categorisation of Arabic text from Twitter using PPM and achieved 90% and 96% accuracy for gender and authorship respectively.

Tarmom [145] produced a comparison of PPM with a traditional machine learning classifier Sequential Minimal Optimisation (SMO), implemented in Weka, working on Arabic text taken from Facebook. The paper detailed research on ways to detect code-switching in Arabic text automatically. Three experiments were conducted in order to: detect code-switching among Egyptian dialect and English, detect code-switching among the Egyptian dialect, the Saudi dialect and English and detect code-switching among the Egyptian dialect, the Saudi dialect, Modern Standard Arabic and English. The experiments showed that PPM achieved a higher accuracy rate than SMO with 99.8% vs 97.5% in the first experiment and 97.8% vs 80.7% in the second. The third experiment did receive a lower accuracy rate than SMO with 53.2% vs 60.%.

Classifying and Segmenting Classical and Modern Standard Arabic using Minimum Cross-Entropy using PPM was a paper by Alkhazi [7]. Text classification was used to process and assign the text to various predefined classes or categories to reflect their contents. The paper adopts a PPM character-based compression scheme to classify and segment Classical Arabic (CA) and Modern Standard Arabic (MSA) texts. The samples produced in the paper resulted in an accuracy of 99.5%, an average precision of 0.958, an average recall of 0.955 and a F-Measure of 0.954.

Classifying dyslexia text using PPM was a paper by Alamri and Teahan [150] and achieved 100% accuracy on English and Arabic texts. Al-Kazaz [75] produced cryptanalysis of transposition ciphers using PPM training and codelength. The paper introduced a compression-based method adapted for the automatic cryptanalysis of Arabic transposition ciphers. The paper presents how PPM performs when applied to the different natural language processing tasks. The samples produced in the paper resulted in 100% decryption of 90 cryptograms; 0.963 recall and precision after space insertion.

Alkahtani [6] produced English-Arabic bilingual sentence alignment using PPM. The paper discusses a new metric that has been applied to verify the quantity in translation between sentence pairs in parallel corpora of Arabic-English. The results produced in the paper show a 100% at identification of satisfactory and unsatisfactory English-Arabic translations.

Emotion recognition in text using PPM was a paper produced by Al-Mahdawi and Teahan [8]. In the paper an investigation were conducted into automatic recognition of emotion in text. A proposed new method for emotion recognition based on PPM was produced to recognize Ekman's six basic emotions (Anger, Disgust, Fear, Happiness, Sadness and Surprise). The samples produced in the paper resulted in an accuracy of 96%, precision of 0.96 and recall of 0.99 at classifying Ekman's 6 emotions in LiveJournal blogs.

Identification of gene function in biological publications was a paper produced by Mahoui et al. [90]. The paper describes the utilisation of text encoding and prediction by PPM to identify gene functions within abstracts of biomedical papers. The samples produced in the paper resulted in precision, recall for identification of gene functions 0.89 and 0.85; 0.85 and 0.94.

Liu and Teahan [84] produced English-Chinese bilingual sentence alignment and achieved 94% at identification of satisfactory and unsatisfactory English-Chinese translations. Teahan and Aljehane [148] produced Grammar-Based Pre-Processing for PPM. The paper discusses applying a grammar-based pre-processing prior to using PPM. The PPM achieves significantly better compression for different natural language texts compared to other well-known compression methods. The results from the paper had an improvement in compression between 11% and 20% on the Calgary Corpus.

'Pre-processing for PPM: Compressing UTF-8 encoded natural language text' was a paper produced by Teahan and Alhawiti [147]. The paper details research of several new universal preprocessing techniques that are described to improve PPM compression of UTF-8 encoded natural language text. The results produced in the paper show an improvement in compression over standard PPM (e.g, 25% for Arabic, 35% for Russian and 32% for Persian).

4.5 The Tawa Toolkit

The Tawa toolkit [150] implements PPM models and can easily be applied to a broad range of text mining and NLP applications (see Figure 4.1). Some applications of Tawa include *train*, *classify*, *encode* and *decode*. The algorithms and pseudo-code of the encoding, decoding, training and six other applications are described in detail in [150]. Other details, such as the implementation aspects and search algorithms applied in the toolkit, are discussed.

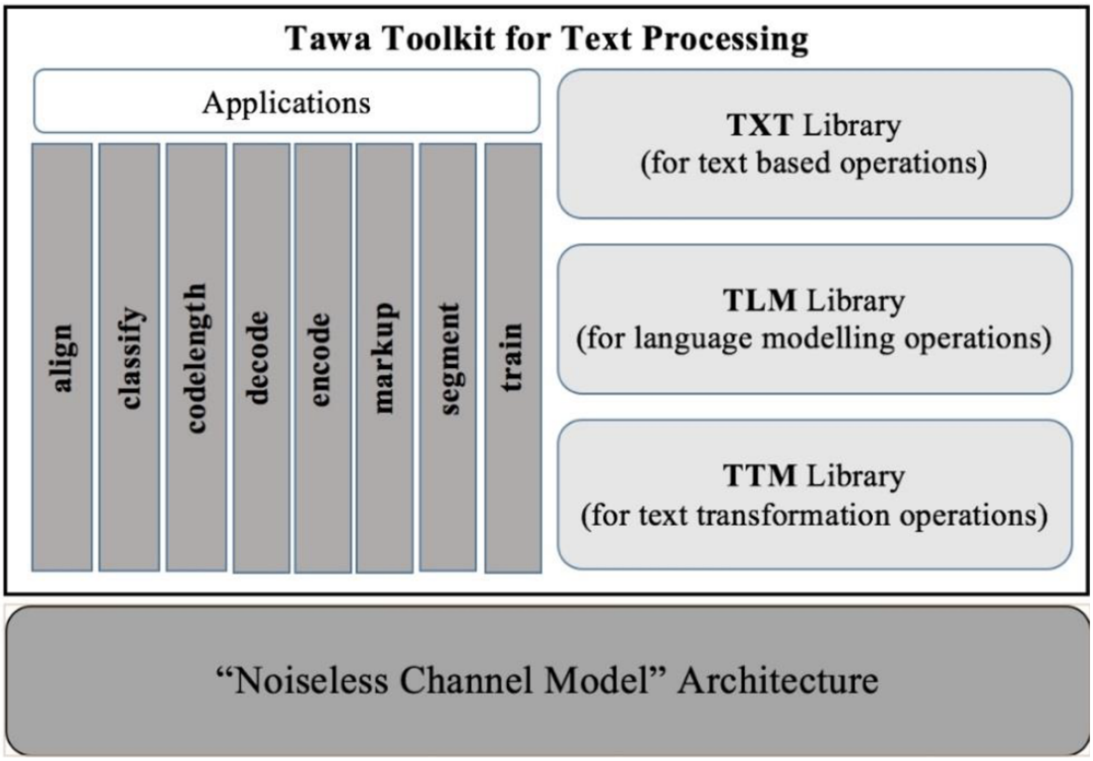

Tawa Toolkit for Text Processing
Applications
align
classify
codelength
decode
encode
markup
segment
train
TXT Library
(for text based operations)
TLM Library
(for language modelling operations)
TTM Library
(for text transformation operations)
"Noiseless Channel Model" Architecture

Figure 4.1: A overview of the Tawa Toolkit and its design [150]

The *train* and *segment* applications of Tawa are used within this project. This is discussed in section 4.6 and 4.7.

4.6 Overview of the Merlin System

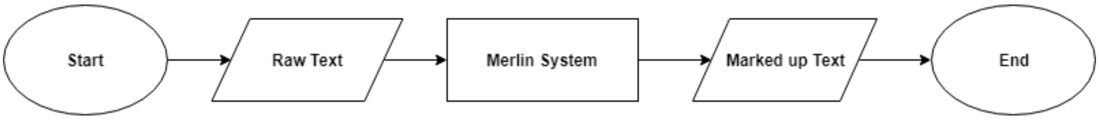


Figure 4.2: The overview of the start to end process of the Merlin system.

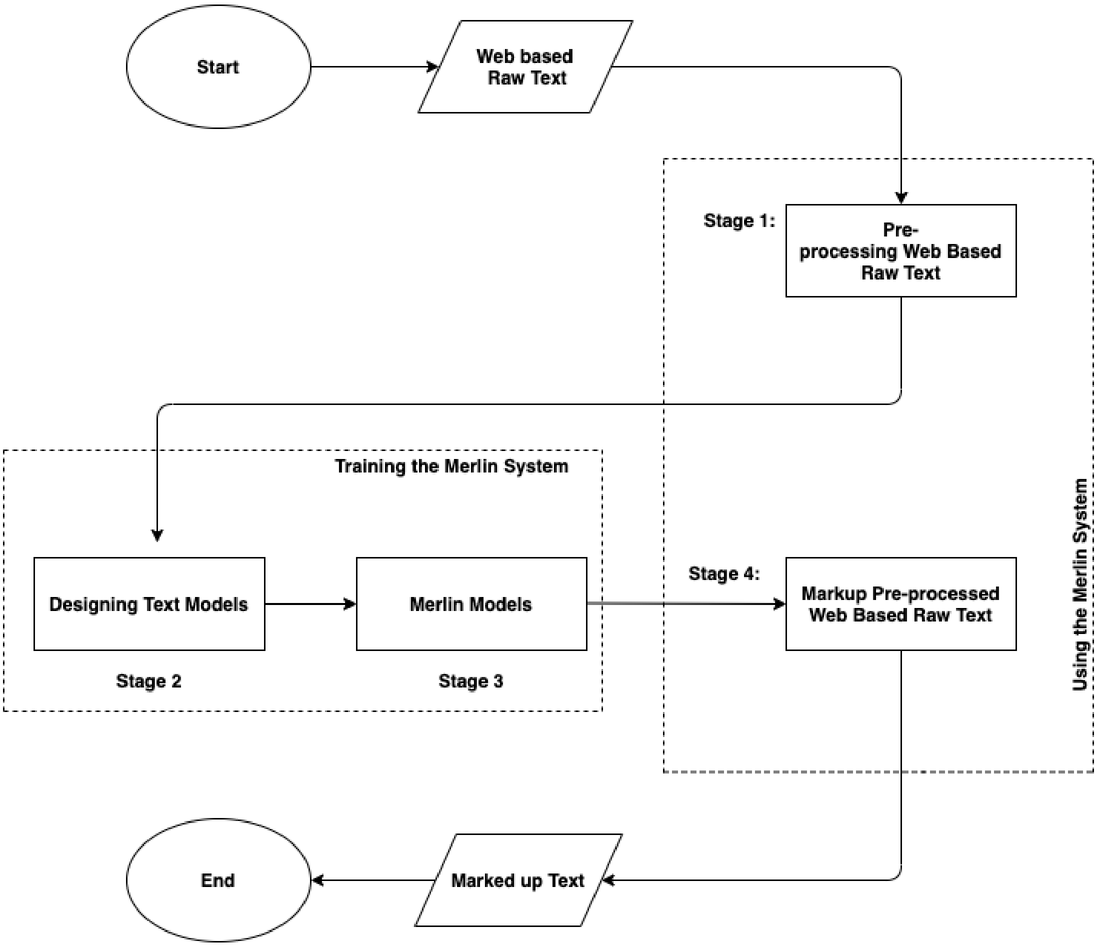


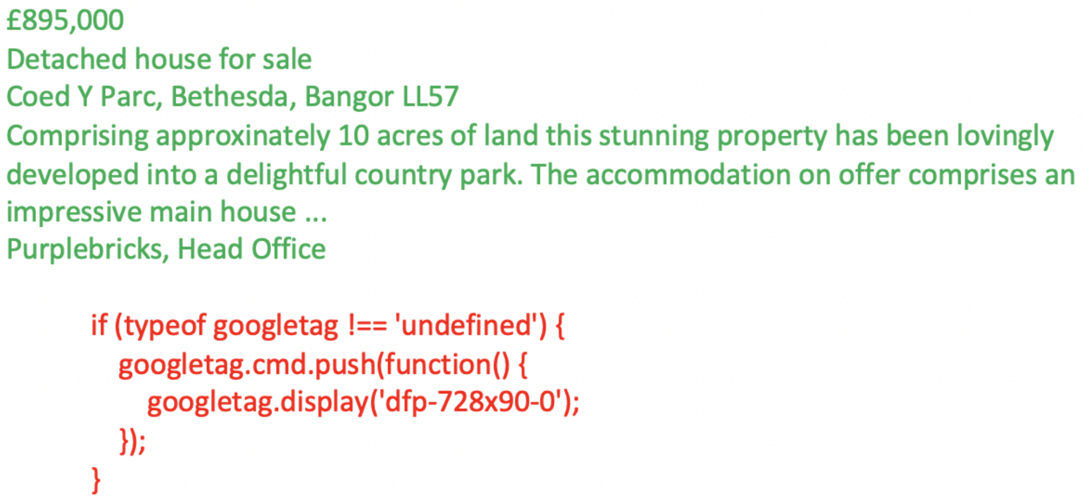
Figure 4.3: The Merlin System workflow process, showing the different stages from the start to the end.

The system that has been developed has been named Merlin. The overview of the system which processes raw text to produce marked up text is shown in figure 4.2. The Merlin system has two processes *using the Merlin System* and *training the Merlin System*.

The detailed specifics of the workflow of the Merlin system as shown in Figure 4.3 starts with the Web based Raw Text and consists of four stages. The first stage (Stage 1) is a pre-processing stage. The second stage (Stage 2) involves the designing of the text used for training the models. The third stage (Stage 3) involves the process of training of the text file models. The final stage (Stage 4) involves the markup process of the pre-processed web based raw text. More details about each stage are described below.

4.6.1 Pre-Processing the Raw Text

While preparing the data for annotating the unstructured data, some errors such as back-end code (e.g., CSS, Javascript and jQuery) needed to be removed. It was identified as causing “noise” within the data, complicating the process, as shown in Figure 4.4. In Figure 4.4, the green highlighted text indicates correct unstructured data, meaning this is what is needed for the Merlin system. The red text in this instance is the “noise”, the pre-processing has picked up Javascript code which is not wanted for the Merlin system. Software in Python, was used to gather related housing and job text from the websites.



£895,000
Detached house for sale
Coed Y Parc, Bethesda, Bangor LL57
Comprising approxinately 10 acres of land this stunning property has been lovingly developed into a delightful country park. The accommodation on offer comprises an impressive main house ...
Purplebricks, Head Office

```
if (typeof googletag !== 'undefined') {
   googletag.cmd.push(function() {
      googletag.display('dfp-728x90-0');
   });
}
```

Figure 4.4: Noise within the data shown by an example data set that shows in green the correct data being captured and in red data that is noise.

4.6.2 Design Text Models

In order to identify the different entities in the unstructured data using the compression-based approach, text models by the Tawa toolkit [36] need to be created. These models are created by training on relevant text that is specific to the kind of textual data being identified. Section 4.6 provides details of how these text models were implemented.

4.7 Implementing The Merlin System

Tawa is a compression-based toolkit based on the API designed by Clearly and Teahan [36]. It provides several tools, including for mining, classifying and transforming unstructured text. The aim of Tawa is to help simplify the design and implementation of applications that require texual models, such as text classification, word or language segmentation amongst other text mining capabilities. The toolkit protects users from modelling details and the probability estimation process. As of 2018 [150], Tawa consists of eight main applications, *align*, *classify*, *codelength*, *decode*, *encode*, *markup*, *segment* and *train*.

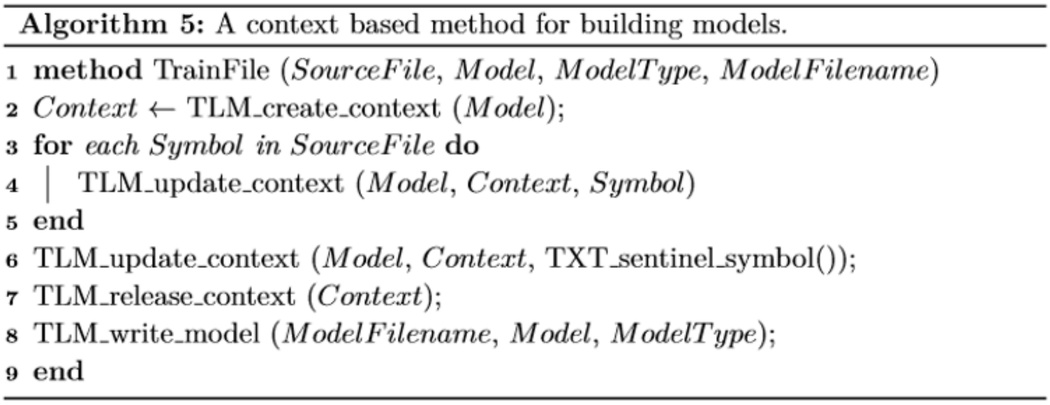
Merlin makes use of two applications within the Tawa toolkit those being building the models and language segmentation. The following section will provide more detail.

4.7.1 Training Models

The Tawa toolkit has library calls used for the implementation of the applications. An example of one of the library calls is *TrainFile*, the algorithm shown below (see Figure 4.5) is used to train models from a source file text. Tawa processes each character sequentially (line 3 of 4.5), using the Tawa library method called *LM_update_context in* (line 4 of 4.5. The objective of this method is to train the compression-based language models [150].

There are two different models that can be used in this method — a static model or a dynamic model. The static model will not change once it has been created and it will always return the same probability estimates. A dynamic model will change when processing more files, requires various arguments such as the *SourceFile* which contains the specific text used to train the *Model*, which is created separately.

The *ModelType* argument is used to specify the models type, which could be static or dynamic. The argument *ModelFileName* is used to create the model output file by calling *TLM_write_model* (see line 8 of Figure 4.5) which can be loaded latter by calling *TLM_load_model*. It is important to note that the size of the model is usually significantly smaller than the size of the text file [150].



```
Algorithm 5: A context based method for building models.
1 method TrainFile ($SourceFile$, $Model$, $ModelType$, $ModelFilename$)
2 $Context$ ← TLM_create_context ($Model$);
3 for each Symbol in SourceFile do
4     TLM_update_context ($Model$, $Context$, $Symbol$)
5 end
6 TLM_update_context ($Model$, $Context$, TXT_sentinel_symbol());
7 TLM_release_context ($Context$);
8 TLM_write_model ($ModelFilename$, $Model$, $ModelType$);
9 end
```

Figure 4.5: The algorithm used for training a model [150]

In order to get the models, the text files used for training required. The following text files have been created for the purpose of providing training data for the models.

In terms of training the housing-based models, the system will need to have a set of text files that contain the following information, as decided by the company partners:

Item	Description
Companies	A list of all registered companies in the United Kingdom.
Currency	A list of prices.
Housing descriptions	A text file containing many housing descriptions.
Telephone	A list of telephones from the selected housing websites.
Street names	A list of all registered streetnames in the United Kingdom.
Towns	A list of all registered towns in the United Kingdom.

Table 4.2: Training data used for building the Housing-based models

In terms of training the jobs-based models, the system will need to have a set of text files that contain the following information, as decided by the company partners:

Item	Description
Location	A list of locations in the United Kingdom.
Job descriptions	A text file containing many job descriptions.
Companies	A list of all registered companies in the United Kingdom.
Currency	A list of prices.
Job titles	A text file containing many titles.

Table 4.3: Training data used for building the Jobs-based models

The data sourced for the creation of each of the training files are from the following:

- **Companies** – The list of UK companies has been obtained from the Companies House [1]. The list contains the latest companies registered as of June 2019.

[1] https://www.gov.uk/government/organisations/companies-house

- **Street names and Towns** – The list of UK street names and towns was obtained via a purchased licence which is held by Royal Mail with the latest data from April 2019.

- **Housing Descriptions** – The text used to train the Housing Descriptions was obtained from hundreds of pages of extracted results from Zoopla, Rightmove and OnTheMarket websites.

- **Telephone Numbers** – The list of telephone numbers was obtained from hundreds of pages extracted from Zoopla, Rightmove and OnTheMarket websites.

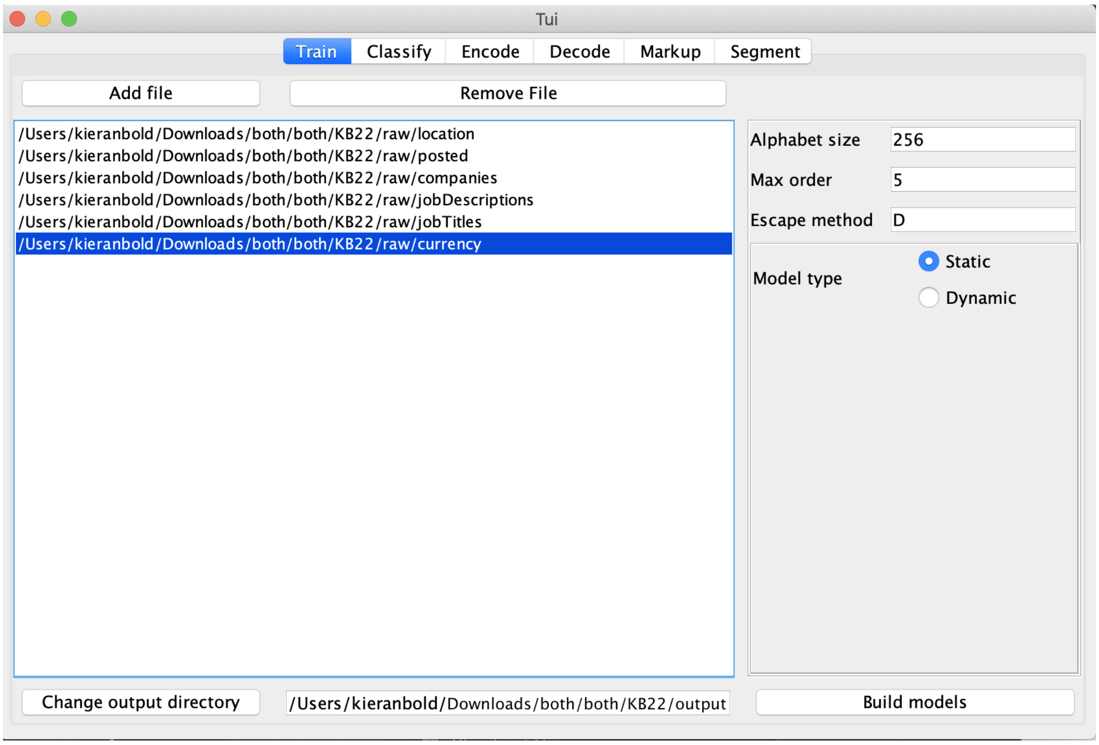
- **Job Descriptions and Job titles** – The text used to train the Job Descriptions and Job Titles was obtained from hundreds of pages of extracted results from Indeed and Reed websites.

A sample of data can be shown in Table 4.4 for each of the training files.

Item	Data type	Samples of training data	Size of files
Companies	Text	!Nspired Investments Ltd. !Obac Uk Limited "1St Rate" Psychology Services Ltd "309" West End Lane Management Limited "A Taste Of" Tours Limited "A.K.Welding Service" Ltd	51.88 MB
Currency	Text	£1 £2 £3 £4 £5	112.43MB
Housing descriptions	Text	A spectacular four bedroom family home which has undergone home improvements to the highest of standards and offers a separate office/potential annexe space. This superb property which is stood behind wrought iron gates is situated in a secluded setting and overlooks breathtaking views. There is ...	1.33MB
Job descriptions	Text	Examples of some of the day to day activities you should expect include communicating with the Director, managing formal correspondence, booking travel,.. Customers want great products at great value which they can buy easily and it's our job to deliver this in the right way for them....	653.39KB
Telephone	Text	024 7511 8874 01322 584475 01656 220944 024 7511 8874 0208033 7635	2.02KB
Street names	Text	Union Street Carnegies Brae Crimon Place Broad Street Albyn Place Broad Street	21.48MB
Locations / Towns	Text	Cardiff Cardiff City Aberdeen Aberfeldy Abergavenny Manchester M1	309.79KB

Table 4.4: Samples of each type of data used to create the models

The text files are then loaded into an application named TUI (see Figure 4.6. TUI stands for Text Understanding Interface. It is a tool based on TAWA currently developed at Bangor University [150]. Once the ‘Build models’ button has been pressed, this then creates a static order 5 PPM model, using escape method D. As stated in section 4.7 a model provides the user with predictions for the sequence of symbols and a model can be static or dynamic. Dynamic models keep on being updated when further files are processed at a later time and a static model does not change, therefore it will always give the same probability estimates [150] [35] [97].



/Users/kieranbold/Downloads/both/both/KB22/raw/location
/Users/kieranbold/Downloads/both/both/KB22/raw/posted
/Users/kieranbold/Downloads/both/both/KB22/raw/companies
/Users/kieranbold/Downloads/both/both/KB22/raw/jobDescriptions
/Users/kieranbold/Downloads/both/both/KB22/raw/jobTitles
/Users/kieranbold/Downloads/both/both/KB22/raw/currency

Alphabet size 256
Max order 5
Escape method D
Model type ● Static ○ Dynamic

/Users/kieranbold/Downloads/both/both/KB22/output

Figure 4.6: TUI Training Application Developed by Dr. William Teahan at Bangor University

Once the processing of the models have been completed, six models were generated using an order 5 PPMD character-based compression scheme using the respective training data. These models are as follows:

- Location model
- Posted model
- Companies model
- jobDescriptions

- jobTitles model

- Currency model

4.8 Language Segmentation

The Tawa toolkit has a segment tool that can be used to perform language segmentation. Language segmentation involves marking up the text to indicate where "code switching" has occurred between different languages. However the language segmentation tool provided in the toolkit has a much wider application than just finding the boundaries between languages i.e., where code-switching occurs. It can also be used in a new way to demarcate between different styles of languages and even discover where named entities such as names and dates occur in the text, and this is the solution that has been adopted for this project. This use of PPM-based text mining in this way to perform identification of named entities is novel and has not previously been published or evaluated in the literature.

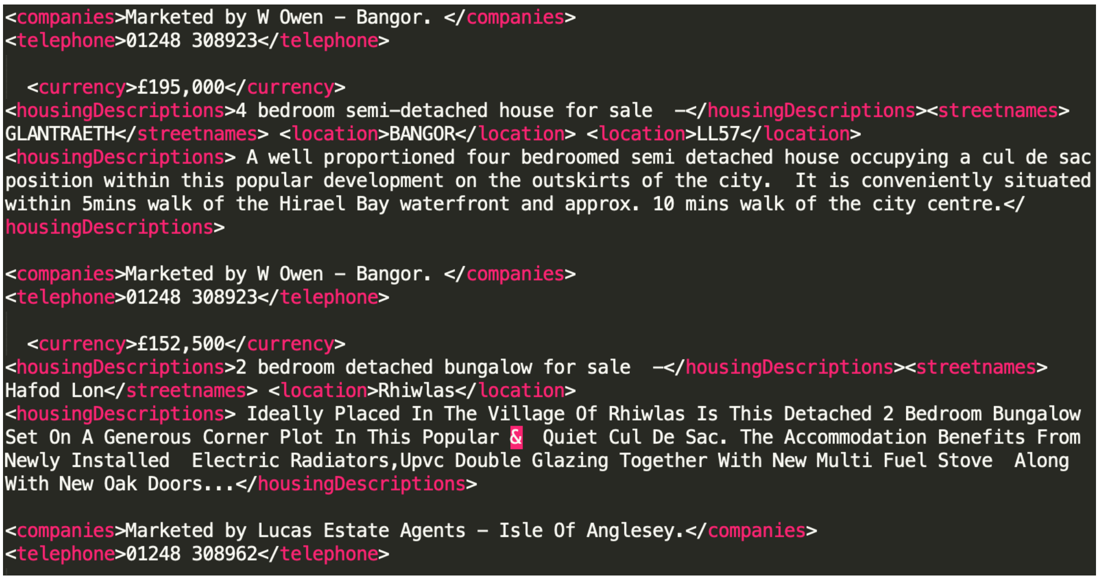
The specific approach that the toolkit uses for language segmentation is as follows. Essentially, the method determines which model (from a set of specific models passed to the algorithm) that compresses each subsequence of the text best in order to minimise the overall compression of the text. The Viterbi algorithm is used to perform the language segmentation. The Viterbi algorithm is an algorithm that was developed by Viterbi (1967) to find the most probable path via a trellis-based search [55] [120].

The algorithm involves generating multiple search paths for each character being processed. This means that the method searches for all possible segmentations of the text based on the models passed to the algorithm. The Viterbi algorithm ensures that the best compression codelength is returned when the search is finished [150].

4.8.1 Creating Ground Truth Files

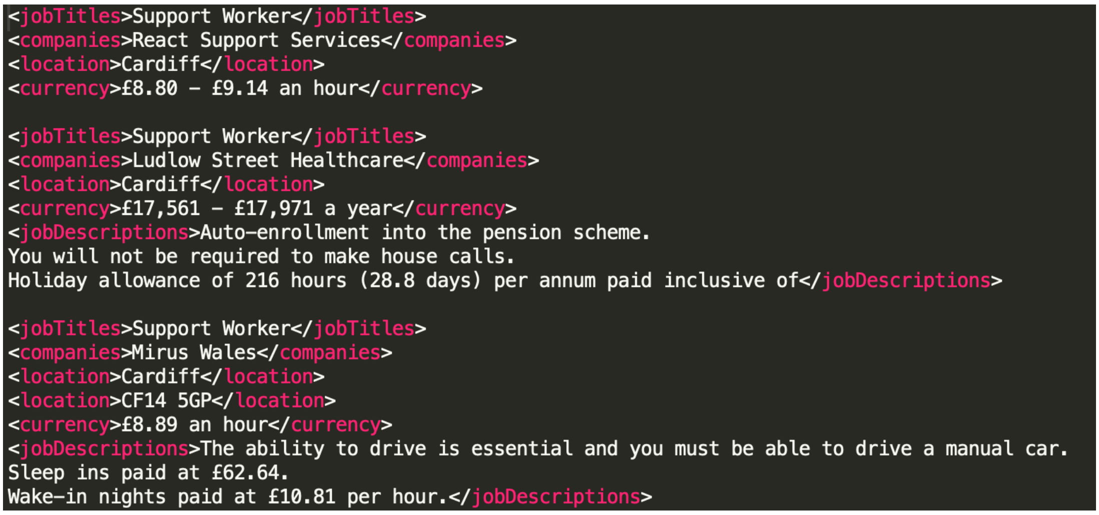
Ground truth files were created in order to evaluate the Merlin system. This involved manually marking up the sample data from the extracted text from the listings and going through line by line adding tags to each of the relevant models. The sample data was obtained from listings from various housing and job websites.

Samples of the ground truth data that was used in the evaluation in Figures 4.6 and 4.7. Originally the extracted data had the text without the tags at the front and end of each line. Going through each line, the manual process of adding the tags were performed. An example being ‘Marked by W Owen - Bangor.’ would be classed as a ‘company’ - so the ‘<companies> </companies>’ tag was wrapped around that text line.



```
<companies>Marketed by W Owen - Bangor. </companies>
<telephone>01248 308923</telephone>

  <currency>£195,000</currency>
<housingDescriptions>4 bedroom semi-detached house for sale  -</housingDescriptions><streetnames>
GLANTRAETH</streetnames> <location>BANGOR</location> <location>LL57</location>
<housingDescriptions> A well proportioned four bedroomed semi detached house occupying a cul de sac
position within this popular development on the outskirts of the city.  It is conveniently situated
within 5mins walk of the Hirael Bay waterfront and approx. 10 mins walk of the city centre.</
housingDescriptions>

<companies>Marketed by W Owen - Bangor. </companies>
<telephone>01248 308923</telephone>

  <currency>£152,500</currency>
<housingDescriptions>2 bedroom detached bungalow for sale  -</housingDescriptions><streetnames>
Hafod Lon</streetnames> <location>Rhiwlas</location>
<housingDescriptions> Ideally Placed In The Village Of Rhiwlas Is This Detached 2 Bedroom Bungalow
Set On A Generous Corner Plot In This Popular &  Quiet Cul De Sac. The Accommodation Benefits From
Newly Installed  Electric Radiators,Upvc Double Glazing Together With New Multi Fuel Stove  Along
With New Oak Doors...</housingDescriptions>

<companies>Marketed by Lucas Estate Agents - Isle Of Anglesey.</companies>
<telephone>01248 308962</telephone>
```

Figure 4.7: Ground Truth Sample — Housing



```
<jobTitles>Support Worker</jobTitles>
<companies>React Support Services</companies>
<location>Cardiff</location>
<currency>£8.80 - £9.14 an hour</currency>

<jobTitles>Support Worker</jobTitles>
<companies>Ludlow Street Healthcare</companies>
<location>Cardiff</location>
<currency>£17,561 - £17,971 a year</currency>
<jobDescriptions>Auto-enrollment into the pension scheme.
You will not be required to make house calls.
Holiday allowance of 216 hours (28.8 days) per annum paid inclusive of</jobDescriptions>

<jobTitles>Support Worker</jobTitles>
<companies>Mirus Wales</companies>
<location>Cardiff</location>
<location>CF14 5GP</location>
<currency>£8.89 an hour</currency>
<jobDescriptions>The ability to drive is essential and you must be able to drive a manual car.
Sleep ins paid at £62.64.
Wake-in nights paid at £10.81 per hour.</jobDescriptions>
```

Figure 4.8: Ground Truth Sample — Jobs

The ground truth data used 5 pages of extracted listings from each of the housing and job websites. This meant that 50 results were being tagged per website. The ground truth data was randomly obtained through selecting various housing and job websites for the purpose of the evaluation performed in this chapter. The process of marking up each unstructured text file took time to add the tags manually to each line to the appropriate data. The tags were added manually by the developer, going through line by line and attaching the relevant tag to that sentence. To allow for easier readability whilst tagging the text, a text-editor called Sublime was used.

The tags that were used for housing are the following:

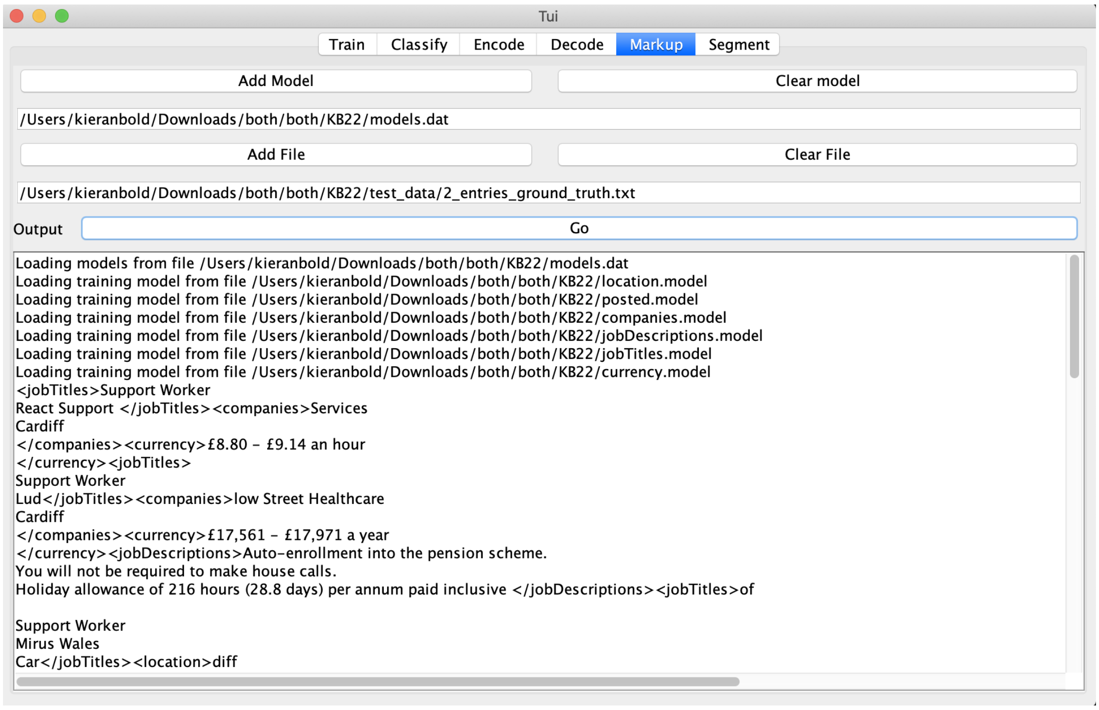
- companies
- currency
- housingDescriptions
- location
- streetnames
- telephone

The tags that were used for jobs are the the following:

- companies
- currency
- jobDescriptions
- jobTitles
- location
- posted

4.8.2 Markup of Pre-processed Web Based Raw Text

Using the pre-processed web based raw text, the markup process involves using the text models and the segmentation tool to tag information relevant to what has been trained. The web-based raw text was pre-processed using effective methods of extraction which is discussed in Chapter 6, section 6.7. Figure 4.9 shows the screenshot of sample markup output from the application:



/Users/kieranbold/Downloads/both/both/KB22/models.dat

/Users/kieranbold/Downloads/both/both/KB22/test_data/2_entries_ground_truth.txt

```
Loading models from file /Users/kieranbold/Downloads/both/both/KB22/models.dat
Loading training model from file /Users/kieranbold/Downloads/both/both/KB22/location.model
Loading training model from file /Users/kieranbold/Downloads/both/both/KB22/posted.model
Loading training model from file /Users/kieranbold/Downloads/both/both/KB22/companies.model
Loading training model from file /Users/kieranbold/Downloads/both/both/KB22/jobDescriptions.model
Loading training model from file /Users/kieranbold/Downloads/both/both/KB22/jobTitles.model
Loading training model from file /Users/kieranbold/Downloads/both/both/KB22/currency.model
<jobTitles>Support Worker
React Support </jobTitles><companies>Services
Cardiff
</companies><currency>£8.80 – £9.14 an hour
</currency><jobTitles>
Support Worker
Lud</jobTitles><companies>low Street Healthcare
Cardiff
</companies><currency>£17,561 – £17,971 a year
</currency><jobDescriptions>Auto-enrollment into the pension scheme.
You will not be required to make house calls.
Holiday allowance of 216 hours (28.8 days) per annum paid inclusive </jobDescriptions><jobTitles>of

Support Worker
Mirus Wales
Car</jobTitles><location>diff
```

Figure 4.9: TUI Markup Application Sample Output

4.8.3 Improving the Results

The results from the initial experiments showed that some of the larger models (produced with much larger training data) overwhelmed the smaller ones, and consequently become better at predicting the English language that was common across all testing data. This is a known issue with using PPM for segmentation [9]. To rectify this, the larger models needed to be reduced in size. An iterative approach was used to reduce the size of the models. Firstly the models were reduced to half size by removing odd numbered lines in the file. This was repeated until there were no changes in the segmentation classifications generated using the models, or the overall classification accuracy decreased.

Reference	Variant 1	Variant 2	Variant 3	Variant 4	Variant 5	Variant 6
<companies>	50MB	50MB	50MB	50MB	50MB	50MB
<currency>	3.6MB	3.6MB	3.6MB	3.6MB	3.6MB	3.6MB
<jobDescriptions>	2.0MB	2.0MB	2.0MB	2.0MB	2.0MB	2.0MB
<jobTitles>	575KB	363KB	223KB	223KB	200KB	125KB
<location>	128KB	128KB	128KB	1.8MB	1.8MB	1.8MB
<posted>	3.7KB	3.7KB	3.7KB	3.7KB	3.7KB	3.7KB
Accuracy (%)	0.9127	0.9132	0.8848	0.8906	0.8303	0.9113
Recall (%)	0.7658	0.7739	0.7115	0.7240	0.6257	0.7783
Precision (%)	0.6756	0.6846	0.6578	0.8308	0.7263	0.8512
F1 Score (%)	0.7179	0.7265	0.6836	0.7737	0.6723	0.8131

Table 4.5: Scaling of the files for Jobs in Reed

The different variations of the training sizes for the models is used for each variant of the experiment (labelled Variant 1 to 10 in Tables 4.5 & 4.6).

Variant 6 was used from Table 4.5 on the experiments for evaluation. Each iteration improved results slightly and the final variant produced the best output.

Reference	Variant 7	Variant 8	Variant 9	Variant 10
<companies>	27.2MB	13.6MB	6.8MB	6.8MB
<currency>	117.9MB	117.9MB	117.9MB	117.9MB
<housingDescriptions>	2KB	2KB	2KB	1.4MB
<streetnames>	22.5MB	22.5MB	22.5MB	22.5MB
<location>	328KB	328KB	328KB	11KB
<telephone>	2KB	2KB	2KB	2KB
Accuracy (%)	0.8432	0.8305	0.8493	0.9113
Recall (%)	0.6530	0.7584	0.7915	0.7783
Precision (%)	0.6970	0.7213	0.7781	0.8512
F1 Score (%)	0.6743	0.7394	0.7847	0.8131

Table 4.6: Scaling of the files for Jobs in Zoopla

Another approach was used when models shared a significant amount of duplicate text. For example, company names often contain the name of a town or city where the company is based. To balance the models, the company model was modified by removing all words which were shared in the location model.

4.9 Evaluation of the Merlin System Results

This section discusses results from a series of experiments that were performed to evaluate the performance of the Merlin system.

Each tag generated by the system on test data was compared against the ground truth. From this information, confusion matrices were constructed and then used to calculate the accuracy, recall, and precision to determine the suitability of the various classification models. If there are no improvements against previously marked up text, the model is left as is, and other models are modified as described in Section 4.5.1. This is repeated until the maximum possible accuracy, recall, and precision values are attained.

4.9.1 Housing Website Experiments

The first set of experiments investigated how well the Merlin system performs with Zoopla, Rightmove and OnTheMarket website data. To do this, ten different queries were used and the results have been averaged. The queries used for the housing website are shown in Table 4.7:

Query
Bangor, Gwynedd
Cardiff
Sheffield, South Yorkshire, £10,000 to £40,000
London, £100,000 to £300,000, 3 beds
Dundee, £400,000 to £800,000, 4 beds
Wrexham, £100,000 to £140,000, 2 beds
Surrey, £120,000 to £180,000
West Yorkshire, Flats
Fife, £125,000, 2 beds
Northern Ireland

Table 4.7: Queries used for evaluating Merlin's performance on Housing Websites

4.9.2 Experiment 1: Evaluating the markup of Zoopla website data

Table 4.8 lists results for Zoopla housing website data with using Table 4.7 as the searching criteria. The Merlin system output was compared against ground truth data that was manually created for the test data. This was done by creating a confusion matrix that compared the predicted output to the ground truth data. The results from Zoopla data was the following:

Query	**Acc. (%)**	**Recall (%)**	**Prec. (%)**	**F1 Score (%)**
Bangor, Gwynedd	84.0	79.0	77.0	78.0
Cardiff	85.0	72.0	86.0	78.3
Sheffield, South Yorkshire, £10,000 to £40,000	83.0	70.0	84.0	76.3
London, £100,000 to £300,000, 3 beds	91.0	84.0	89.0	86.4
Dundee, £400,000 to £800,000, 4 beds	89.0	81.0	90.0	85.2
Wrexham, £100,000 to £140,000, 2 beds	81.5	63.3	70.4	66.7
Surrey, £120,000 to £180,000	88.2	76.5	88.5	82.1
West Yorkshire, Flats	86.8	70.4	73.9	72.1
Fife, £125,000, 2 beds	85.6	70.2	69.3	69.7
Northern Ireland	82.7	70.6	81.6	75.7
Average	86.5	77.2	85.2	77.0

Table 4.8: Merlin Performance Results of Zoopla housing data

The confusion matrix can be seen in figure 4.10. The average accuracy from Zoopla housing data for table 4.8, was 86.5%, average recall 77.2%, average precision of 85.2% and average F1-Score of 77.0%. As shown in the figure, 31 occurrences of prediction based a streetname when it should have been location. 150 occurrences of predicated housingDescriptions when these should have been companies.

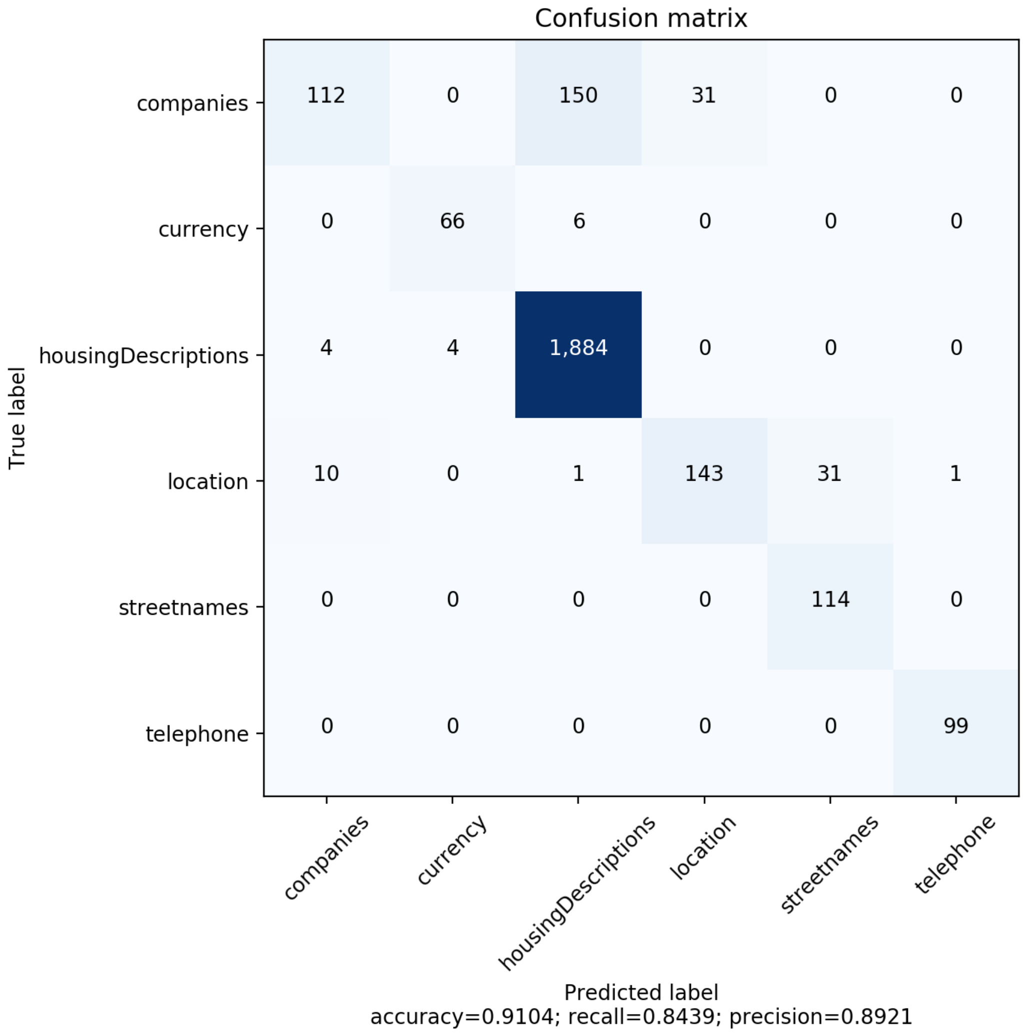


Figure 4.10: Confusion matrix for the markup output produced by the Merlin system for the Zoopla housing data

Several mistakes made by the Merlin system are readily apparent as shown in Figure 4.10. Notably, streetnames and locations are being tagged incorrectly. Overall performance is excellent as indicated by the results in Figure 4.10.

4.9.3 Experiment 2: Evaluating the markup of OnTheMarket website data

Table 4.9 lists results for OnTheMarket housing website data with using Table 4.7 as the searching criteria. The Merlin system output was compared against ground truth data that was manually created for the test data. This was done by creating a confusion matrix that compared the predicted output to the ground truth data. The results from OnTheMarket data was the following:

Query	**Acc.** (%)	**Recall** (%)	**Prec.** (%)	**F1-Score** (%)
Bangor, Gwynedd	94.0	74.0	90.0	81.2
Cardiff	92.2	73.9	87.4	80.1
Sheffield, South Yorkshire, £10,000 to £40,000	84.0	66.0	81.0	72.7
London, £100,000 to £300,000, 3 beds	92.0	78.0	88.0	82.6
Dundee, £400,000 to £800,000, 4 beds	89.0	78.0	79.0	78.4
Wrexham, £100,000 to £140,000, 2 beds	93.0	70.4	88.4	78.4
Surrey, £120,000 to £180,000	85.1	71.0	85.7	77.7
West Yorkshire, Flats	94.6	78.8	89.6	83.9
Fife, £125,000, 2 beds	92.2	73.9	87.4	80.1
Northern Ireland	89.2	72.2	84.8	78.0
Average	90.2	73.8	85.0	79.2

Table 4.9: Merlin Performance Results of OnTheMarket housing data

The confusion matrix can be seen in figure 4.11. The average accuracy from OnTheMarket housing data for table 4.9, was 90.2%, average recall 73.8%, average precision of 85.0% and average F1-Score of 79.2%. The figure shows 144 predicted occurrences were inaccurately predicated and these should have been companies. Additionally 72 of these should have been location, 87 occurrences should have been currency and 53 occurrences should have been companies. This is clear that the models had some confusion, primarily the housingDescriptions model.

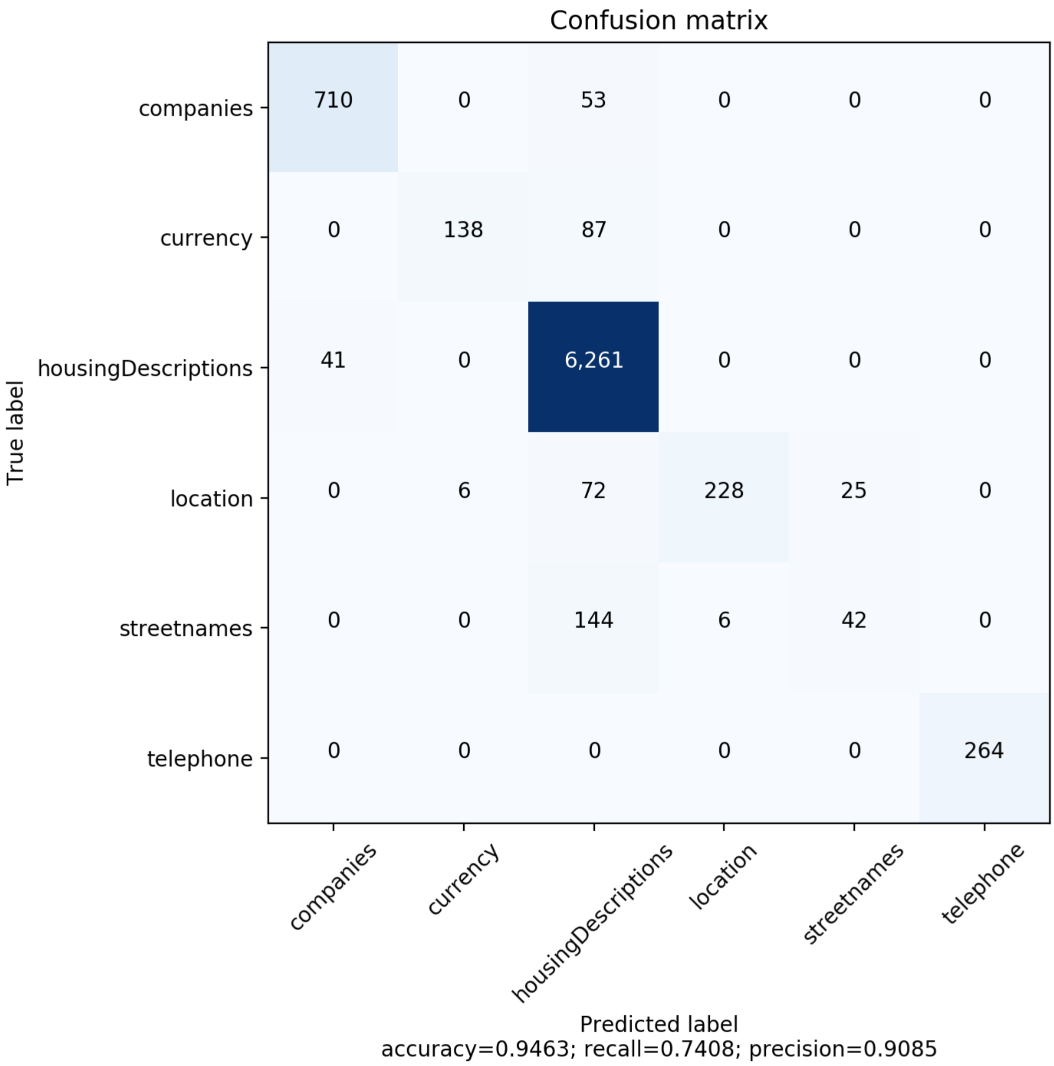


Figure 4.11: Confusion matrix for the markup output produced by the Merlin system for the OnTheMarket housing data

Several mistakes made by the Merlin system are shown in shown in Figure 4.11. Notably, streetname is being tagged as a currency. However, overall performance is excellent as indicated by the results in Figure 4.11.

4.9.4 Experiment 3: Evaluating the markup of Rightmove website data

Table 4.10 lists results for Rightmove's housing website data with using Table 4.7 as the searching criteria. The Merlin system output was compared against ground truth data that was manually created for the test data. This was done by creating a confusion matrix that compared the predicted output to the ground truth data. The results from Rightmove data was the following:

Query	**Acc. (%)**	**Recall (%)**	**Prec. (%)**	**F1-Score (%)**
Bangor, Gwynedd	91.0	69.0	91.0	78.4
Cardiff	83.0	70.0	72.0	70.9
Sheffield, South Yorkshire, £10,000 to £40,000	79.0	65.0	66.0	65.4
London, £100,000 to £300,000, 3 beds	91.0	65.0	77.0	70.4
Dundee, £400,000 to £800,000, 4 beds	93.3	78.7	91.0	84.4
Wrexham, £100,000 to £140,000, 2 beds	82.0	57.3	68.9	62.6
Surrey, £120,000 to £180,000	91.7	75.7	86.2	80.6
West Yorkshire, Flats	80.6	50.3	60.8	55.1
Fife, £125,000, 2 beds	91.2	70.7	82.4	76.1
Northern Ireland	83.5	53.0	67.0	59.1
Average	87.4	69.4	79.4	70.3

Table 4.10: Merlin Performance Results of Rightmove housing data

The confusion matrix can be seen in figure 4.12. The average accuracy from Rightmove housing data for table 4.10, was 87.4%, average recall 69.4%, average precision of 79.4% and average F1-Score of 70.3%. The confusion matrix shows that 110 occurrences of companies predicated should have been streetnames. Additionally 12 occurrences of predicated housingDescriptions should have been currency.

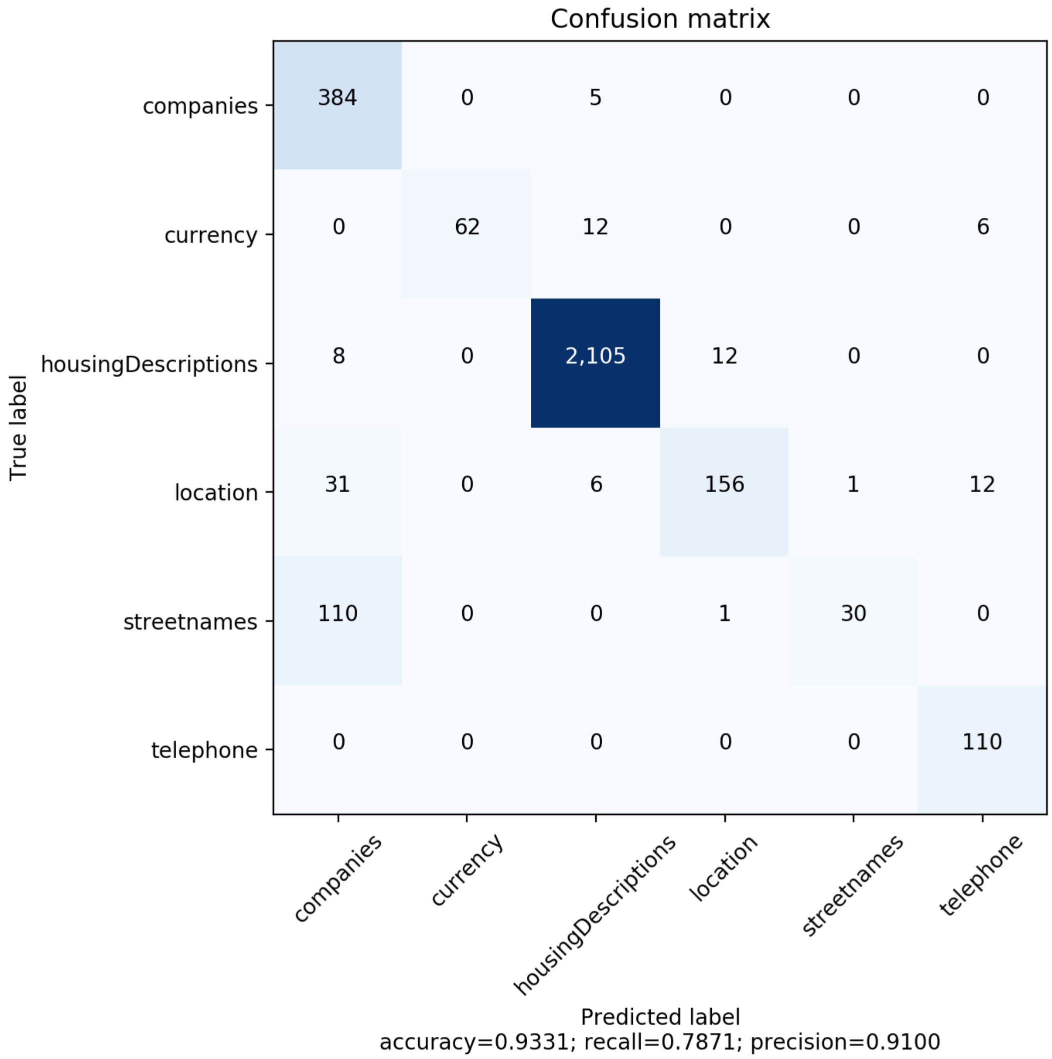


Figure 4.12: Confusion matrix for the markup output produced by the Merlin system for the Rightmove housing data

Several mistakes made by the Merlin system are apparent as shown in Figure 4.12. Notably, currency is being tagged as a housing description. However, overall performance is excellent as indicated by the results in Figure 4.12.

4.9.5 Job Website Experiments

The first set of experiments investigated how well the Merlin system performs with Indeed and Reed website data. To do this, ten different queries were used and the results have been averaged. The queries used for the job websites are shown in Table 4.11:

Query
Bangor, Gwynedd
Cardiff, Support Worker
Sheffield, Customer Service
London, Accountant
Dundee, Finance Manager
Birmingham, Delivery Driver
Wolverhampton, GP
Stoke on Trent, Paramedic
Welshpool, Food Production
Milton Keynes, System Analyst

Table 4.11: Queries used for evaluating Merlin's performance on Job Websites

4.9.6 Experiment 4: Evaluating the markup of Reed website data

Table 4.12 lists results for Reed's job website data with using Table 4.11 as the searching criteria. The Merlin system output was compared against ground truth data that was manually created for the test data. This was done by creating a confusion matrix that compared the predicted output to the ground truth data. The results from Reed's data was the following:

Query	**Acc.** **(%)**	**Recall** **(%)**	**Prec.** **(%)**	**F1-Score** **(%)**
Bangor, Gwynedd	86.0	73.0	82.0	77.2
Cardiff, Support Worker	87.0	79.0	78.0	78.4
Sheffield, Customer Service	89.0	86.0	88.0	86.9
London, Accountant	88.0	77.0	78.0	77.4
Dundee, Finance Manager	92.0	85.0	89.0	86.9
Birmingham, Delivery Driver	80.9	74.6	74.2	74.4
Wolverhampton, GP	83.8	79.1	77.6	78.3
Stoke on Trent, Paramedic	87.4	76.5	82.0	79.2
Welshpool, Food Production	85.6	78.3	82.9	80.5
Milton Keynes, System Analyst	85.8	78.6	81.8	80.2
Average	84.8	80.0	83.0	79.9

Table 4.12: Merlin Performance Results of Reed jobs data

The confusion matrix can be seen in figure 4.13. The average accuracy from Reeds jobs data for table 4.12, was 88.4%, average recall 80.0%, average precision of 83.0% and average F1-Score of 79.9%. 50 predicated occurrences were classed as jobTitles although they should have been companies. Additionally 31 occurrences of jobDescriptions should have been companies.

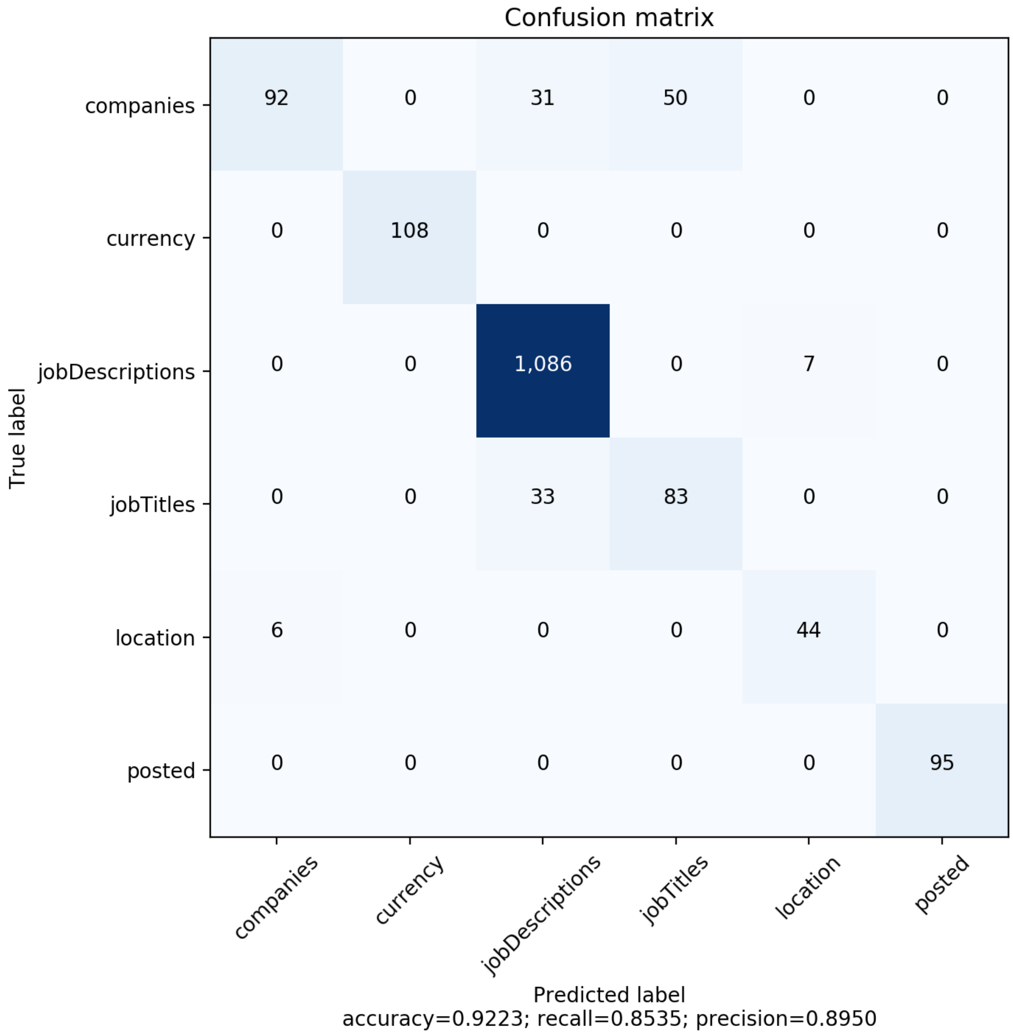


Figure 4.13: Confusion matrix for the markup output produced by the Merlin system for the Reed job data

Several mistakes made by the Merlin system are apparent as shown in Figure 4.13.Notably, location is being tagged as a company. However, overall performance is excellent as indicated by the results in Figure 4.13.

4.9.7 Experiment 5: Evaluating the markup of Indeed website data

Table 4.13 lists results for Indeed's job website data with using Table 4.11 as the searching criteria. The Merlin system output was compared against ground truth data that was manually created for the test data. This was done by creating a confusion matrix that compared the predicted output to the ground truth data. The results from Indeed's data was the following:

Query	**Acc.** (%)	**Recall** (%)	**Prec.** (%)	**F1-Score** (%)
Bangor, Gwynedd	89.0	77.0	83.0	79.8
Cardiff, Support Worker	90.0	77.0	87.0	81.6
Sheffield, Customer Service	83.0	62.0	82.0	70.6
London, Accountant	88.0	62.0	88.0	72.7
Dundee, Finance Manager	83.0	70.0	69.0	69.4
Birmingham, Delivery Driver	89.2	73.6	85.7	79.2
Wolverhampton, GP	85.0	71.0	80.0	75.2
Stoke on Trent, Paramedic	80.2	66.3	83.2	73.8
Welshpool, Food Production	83.8	77.9	82.3	80.0
Milton Keynes, System Analyst	85.1	77.1	80.8	78.9
Average	86.6	69.6	81.8	76.1

Table 4.13: Merlin Performance Results of Indeed jobs data

The confusion matrix can be seen in figure 4.14. The average accuracy from Indeeds jobs data for table 4.13, was 86.6%, average recall 69.6%, average precision of 81.8% and average F1-Score of 76.1%. 70 occurrences of jobTitles predicated should have been companies. Additionally 14 occurrences of jobTitles should have been jobDescriptions.

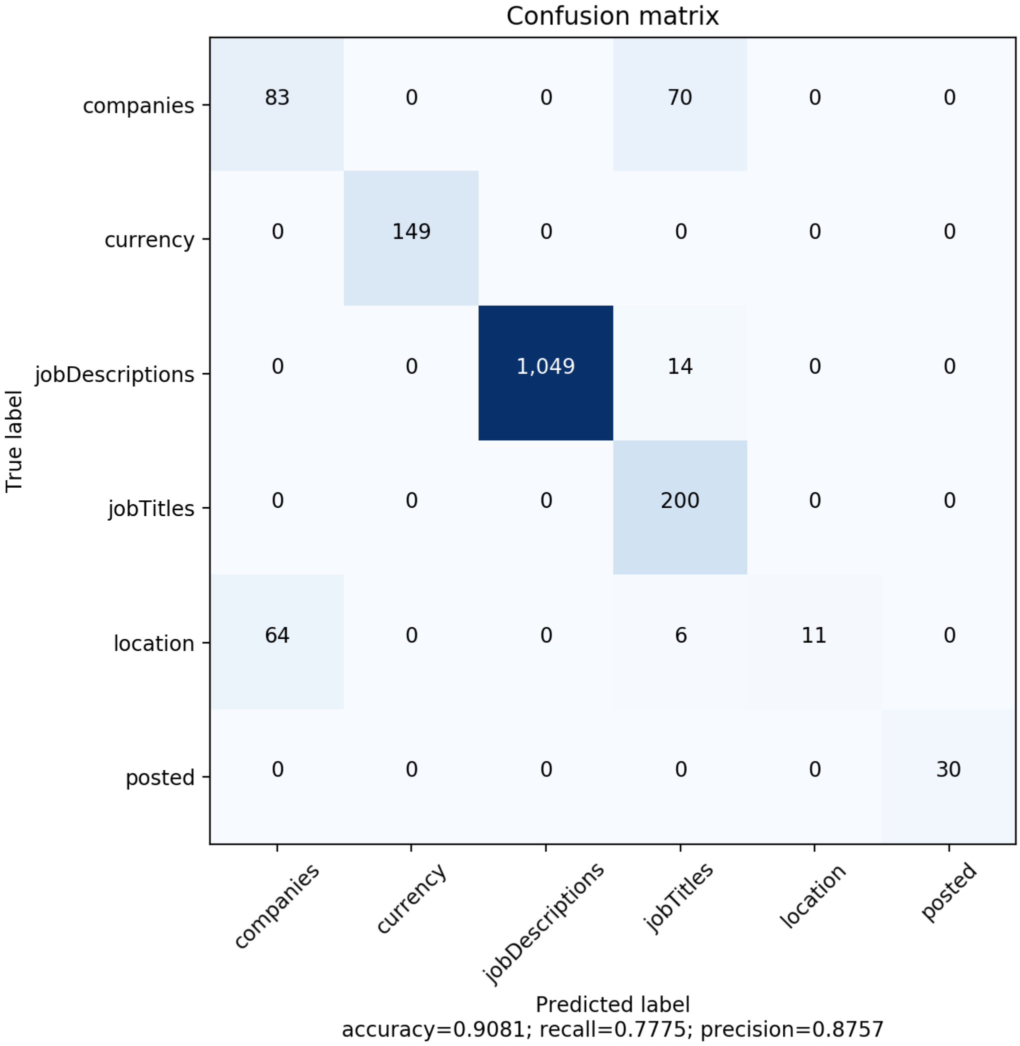


Figure 4.14: Merlin Performance Result of Indeed job data

Several mistakes made by the Merlin system are apparent as shown in Figure 4.14. Notably, job titles and posted tags are being tagged incorrectly. However, overall performance is excellent as indicated by the results in Figure 4.14.

4.9.8 Both Housing and Job Website Results

This section describes a further experiment that involved merging the best result from the housing website, that being the query “Bangor, Gwynedd” from OnTheMarket from Table 4.9 and the best result from the job website, that being the query “Cardiff, Support Worker” from Indeed 4.13, to see how well the Merlin system can cope with two sets of different sources of data. Taking these two best results, an experiment was performed to see how well the two top performing results would perform when they are merged together.

As this section is looking at the best two results from both housing and jobs, a singular precision, accuracy, recall and F1-Score was produced. Figure 4.15 indicates accuracy of 82.0%, a recall of 65.0% and precision of 77.0%. The F1-Score for this is 70.4%. The outcome from the confusion matrix shows that the predictions of the jobTitles,jobDescriptions and housingDescriptions models have predicted a large amount of occurrences incorrectly. An example would be housingDescriptions predictions, 255 predicted occurrences should have been location, 134 occurrences should have been streetnames.

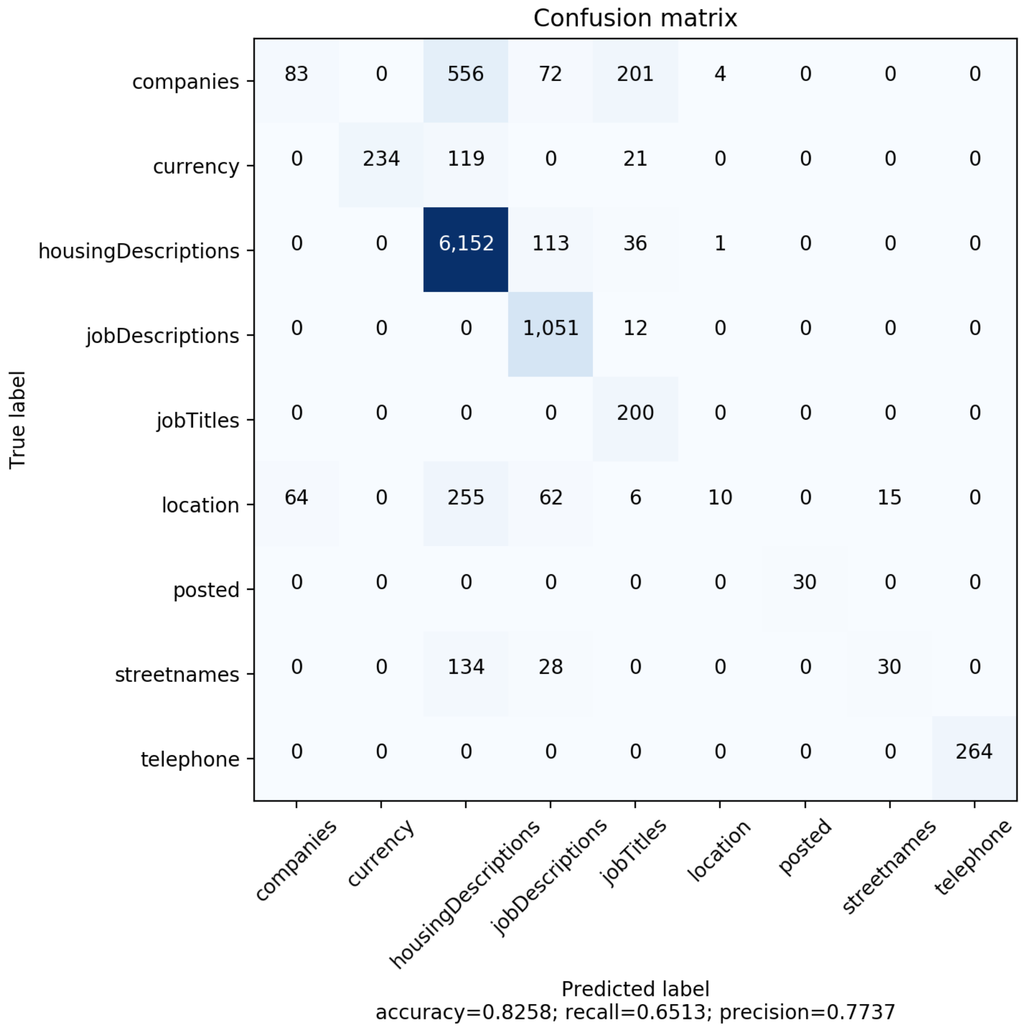


Figure 4.15: Confusion matrix for the markup output produced by the Merlin system for the housing and job data

As anticipated, these have dipped in performance as there occurrences coming from both housing and jobs models, in particular the housing and jobs descriptions model that are getting incorrectly tagged. This is where a job description is getting partly tagged as a housing description and job description and vice versa although the overall performance is still excellent.

High accuracy, precision and recall rates from the evaluation experiments for the different job and housing websites as shown in the Table 4.14.

Website	Average Acc. (%)	Average Recall (%)	Average Prec. (%)	Average F1-Score (%)
Zoopla	86.5	77.2	85.2	77.0
OnTheMarkeet	90.2	73.8	89.6	83.9
Rightmove	87.4	69.4	79.4	70.3
Reed	84.8	80.0	83.0	79.9
Indeed	86.6	69.6	81.8	76.1

Table 4.14: Performance measures of the Merlin system for the jobs and housing data

4.10 Comparison of Merlin System with the spaCy Natural Language Toolkit

This section describes experiments to compare the well-performing spaCy Natural Language Toolkit to the Merlin system's performance based on the previous experiments.

spaCy's [2] tagger, parser, text categoriser and other components are powered by statistical models. The decision these components make for example such as which part-of-speech tag to assign, or whether a word is a named entity, is produced by prediction based on the model's current weight values. The weight values are estimated based on examples the model has seen during training.

In order to train a model, the developer needs to have training data to train the models using neural networks. In this instance, the same training data was used as for the Merlin system. The developer then needs to label the models that are being used to predict the class labels.

[2] https://spacy.io/usage/training

Training is then an iterative process in which the model's predictions are compared against the reference annotations in order to estimate the gradient of the loss. The gradient of the loss is used to calculate the gradient of the weights through backpropagation. The gradients indicate how the weight values should be changed so that the model's predictions become more similar to the reference labels over time.

In order to know how the model is performing and whether it is learning the right patterns, the developer needs not just training data but evaluation data.

If the user were to test the model with the data it was trained on, you will have no idea how well it is generalising. In this instance as stated we are using the same ground truth data that was used to train the Merlin system for the evaluation data

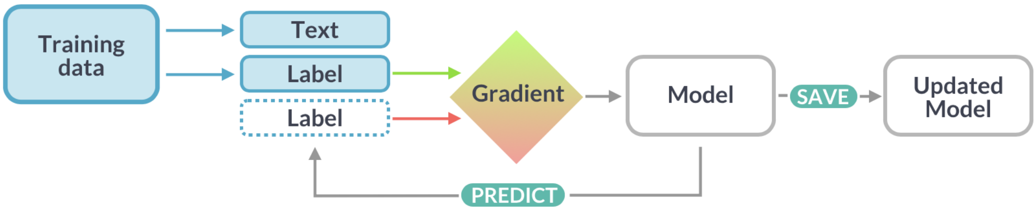


Figure 4.16: spaCy's training pipeline for named entity recognition

Figure 4.16 demonstrates the spaCy's training pipeline [3]:

- Training data: Examples of the training data and their annotations.
- Text: The input text the model should predict a label for.
- Label: The label the model should predict.
- Gradient: The direction and rate of change for a numeric value. Minimising the gradient of the weights should result in predictions that are closer to the reference labels on the training data.

spaCy does more than just memorise examples. It also comes up with the theory that can be generalised across unseen data.

[3] https://spacy.io/usage/training

spaCy [100] uses deep learning for implementing NLP models, which is summarised as "embed, encode, attend, predict" [109]. spaCy's approach to text is that it is inserted in the model in the form of a unique numerical value (ID) for every input that can represent a token of a corpus or a class of the NLP task. At the embedding stage, features such as the prefix, the suffix, the shape and the lower form of a word are used for extraction of hashed values that reflect word similarities.

spaCy is an open-source library that provides natural language processing tools for the Python programming language. Jiang et al. [74] indicates that this tool performs second best among four well-established open-source NER tools regarding F-measure.

spaCy's NER tool extracts named entities in eighteen categories: persons, nationalities or religious groups, facilities, organisations, geopolitical entities, locations, products, events, works of art, law documents, languages, dates, times, percentages, money, quantities, ordinals, and cardinals [77]. spaCy also allows the developer to train and update components of the developer's own data and to integrate custom models.

The following subsections discuss results from a series of experiments that were performed to evaluate the performance of spaCy.

Precision, recall and F1 Score calculated from the confusion matrices were used in order to perform the direct comparison with the Merlin system.

4.10.1 Experiment 6: Evaluating the markup of Zoopla website data using spaCy

In the Merlin System, the best result for Zoopla housing data was 84.0% recall, 89.0% precision and 86.4% F1 score.

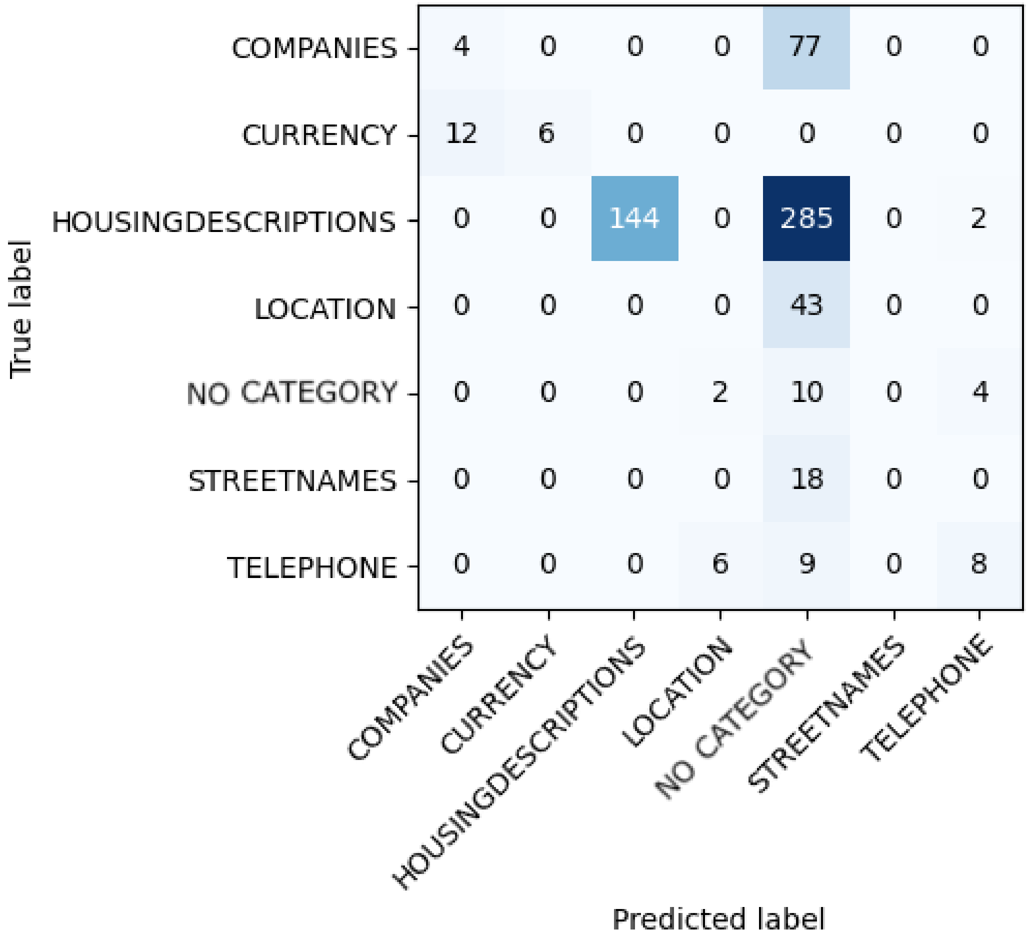


Figure 4.17: Confusion matrix for the markup output produced by the spaCy system for the Zoopla housing data.

The confusion matrix for the spaCy comparison can be seen in figure 4.17. The overall results from Zoopla housing data was 50.0% precision, 1.82% recall and 3.5% F1 score.

There were many mistakes made by spaCy on the Zoopla housing data as seen in the confusion matrix. Housing descriptions, companies and locations are unable to be predicted accurately, which has resulted in them being labelled as “no category”. This is where spaCy could not predict the text against any of the models. No street names or locations have been identified correctly.

4.10.2 Experiment 7: Evaluating the markup of OnTheMarket data using spaCy

In the Merlin System, the best result for OnTheMarket housing data was 74.0% recall, 90.0% precision and 81.2% F1 score.

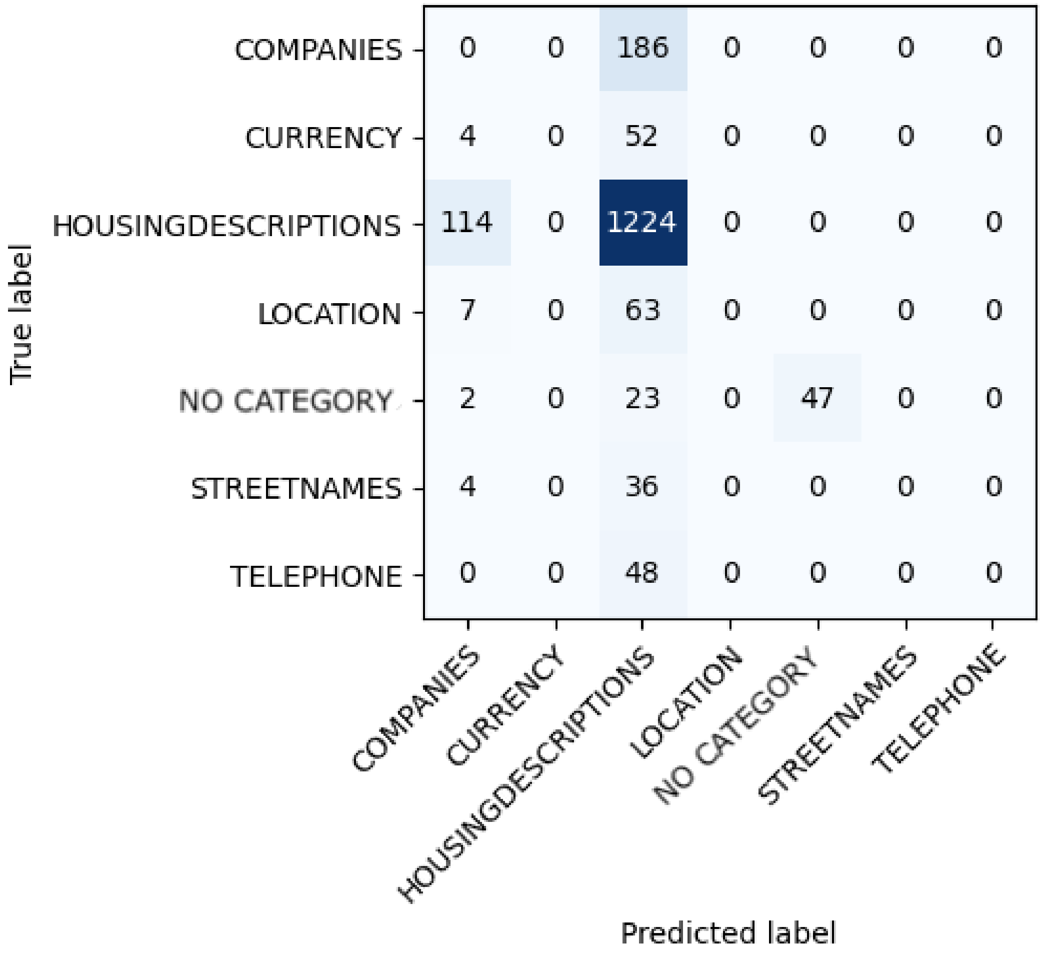


Figure 4.18: Confusion matrix for the markup output produced by the spaCy system for the OnTheMarket housing data.

The confusion matrix for the spaCy comparison can be seen in figure 4.18. The results from OneTheMarket housing data was 10.0% precision, 1.71% recall and 2.93% F1 score.

There were many mistakes made by the spaCy performance on the OnTheMarket housing data, such as streetnames, telephones and location being identified under housing descriptions. Streetnames, telephones, currency and companies did not get predicted. A few instances spaCy predicted companies as streetnames, location and currency.

4.10.3 Experiment 8: Evaluating the markup of Rightmove data using spaCy

In the Merlin System, the best result for Rightmove housing data was 78.7% recall, 91.0% precision and 84.4% F1 score.

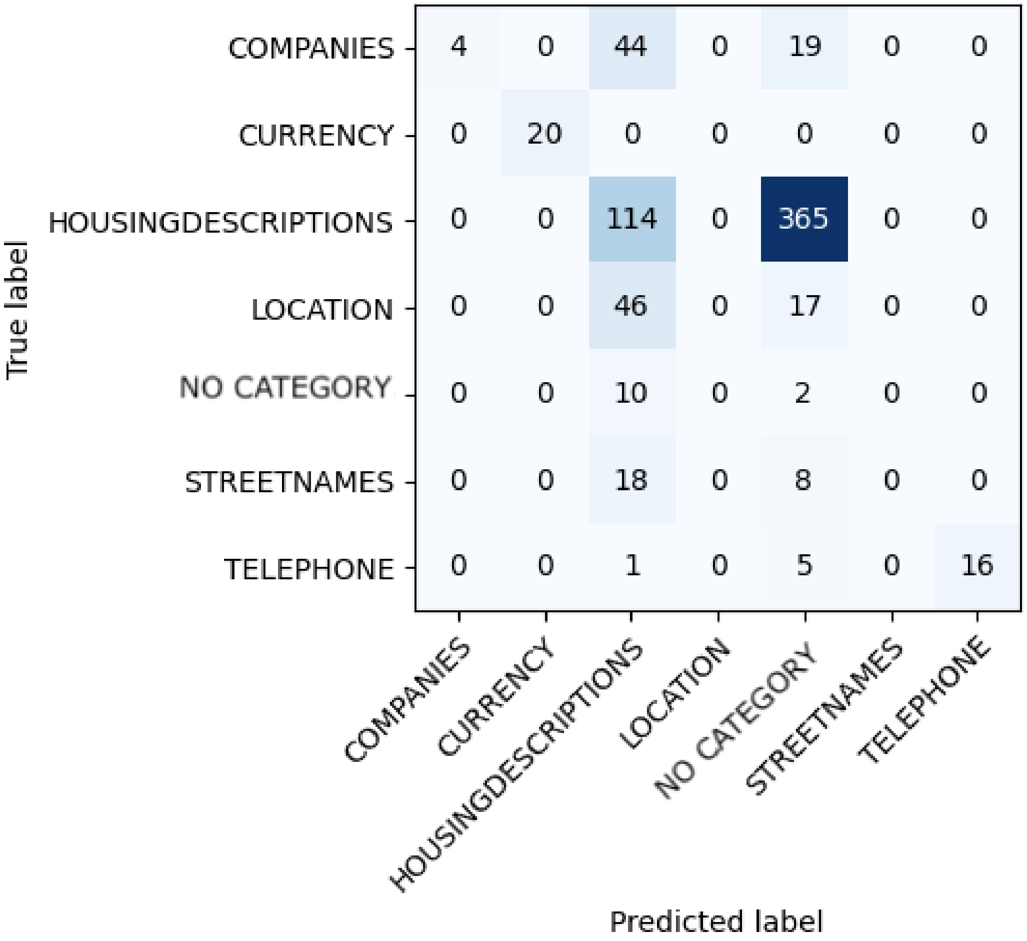


Figure 4.19: Confusion matrix for the markup output produced by the spaCy system for the Rightmove housing data.

The confusion matrix for the spaCy comparison can be seen in figure 4.19. The results from Rightmove housing data was 10.0% precision, 1.43% recall and 2.50% F1 score.

There were many mistakes made by the spaCy performance on the Rightmove housing data, such as housing descriptions not being able to be tagged correctly. No streetnames or locations were predicted correctly.

4.10.4 Experiment 9: Evaluating the markup of Reed data using spaCy

In the Merlin System, the best result for Reed jobs data was 75.0% recall, 89.0% precision and 86.9% F1 score.

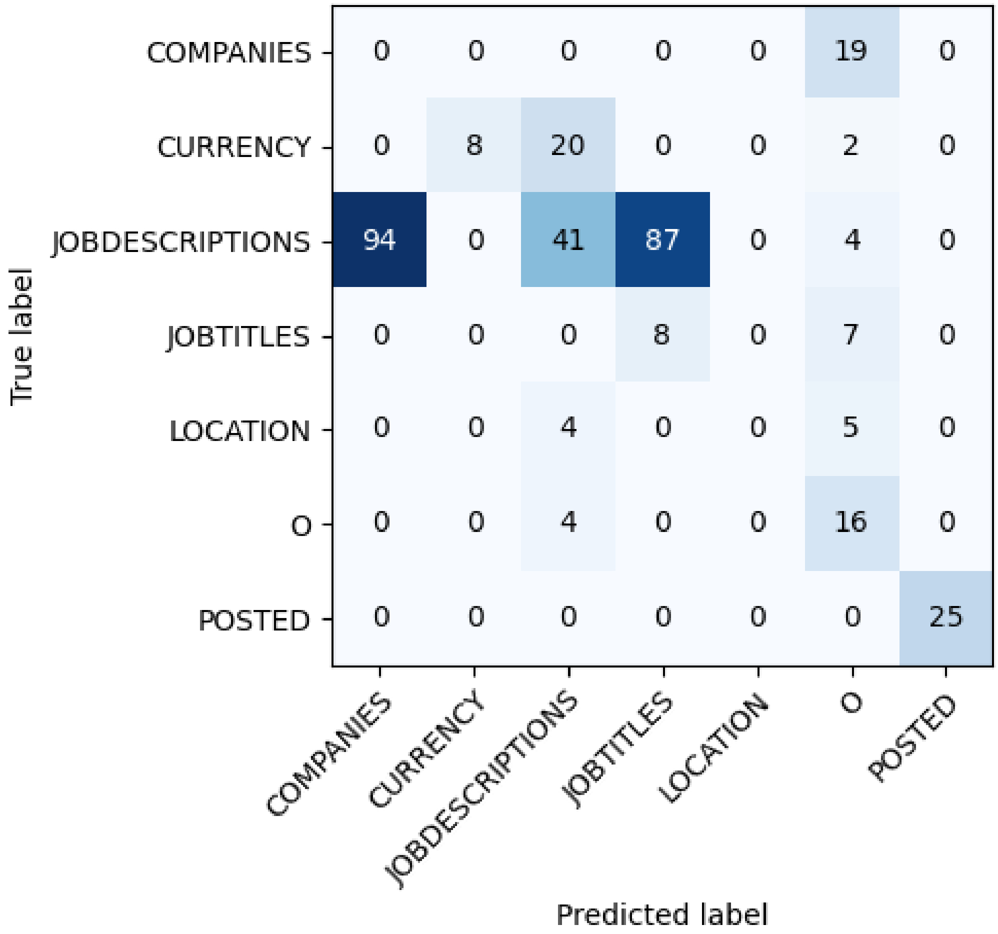


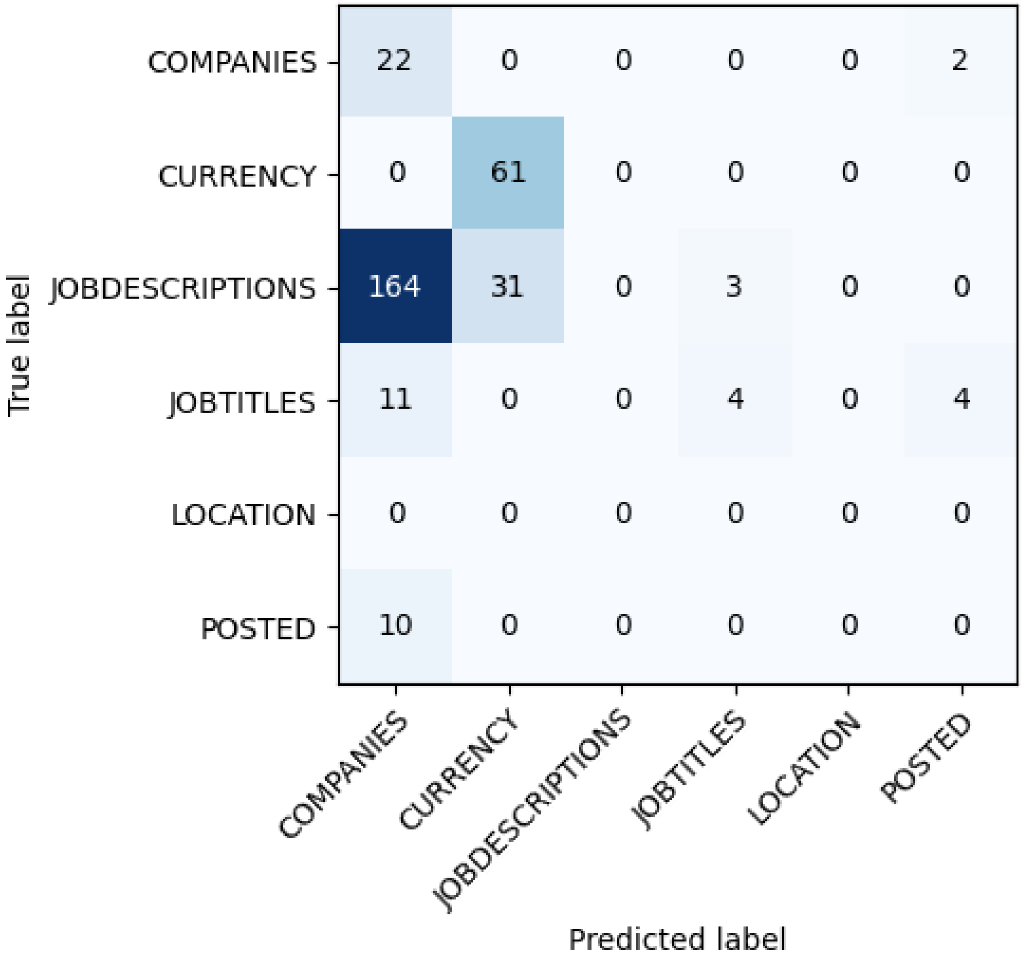
Figure 4.20: Confusion matrix for the markup output produced by the spaCy system for the Reed jobs data

The confusion matrix for the spaCy comparison can be seen in figure 4.20. The results for spaCy from Reed job data was 36.0% precision, 26.47% recall and 30.51% F1 score.

spaCy's processing of the jobs data for Reed has done significantly better than all the housing data experiments. spaCy predicted a large number of job descriptions correctly and posted although some predictions included job descriptions as companies. spaCy predicted no locations and made the mistakes in job descriptions and no category. Many of the categories were predicted under housing descriptions such as the streetnames and companies.

4.10.5 Experiment 10: Evaluating the markup of Indeed data using spaCy

In the Merlin System, the best result for Indeed jobs data was 77.0% recall, 87.0% precision and 81.6% F1 score.



	COMPANIES	CURRENCY	JOBDESCRIPTIONS	JOBTITLES	LOCATION	POSTED
COMPANIES	22	0	0	0	0	2
CURRENCY	0	61	0	0	0	0
JOBDESCRIPTIONS	164	31	0	3	0	0
JOBTITLES	11	0	0	4	0	4
LOCATION	0	0	0	0	0	0
POSTED	10	0	0	0	0	0

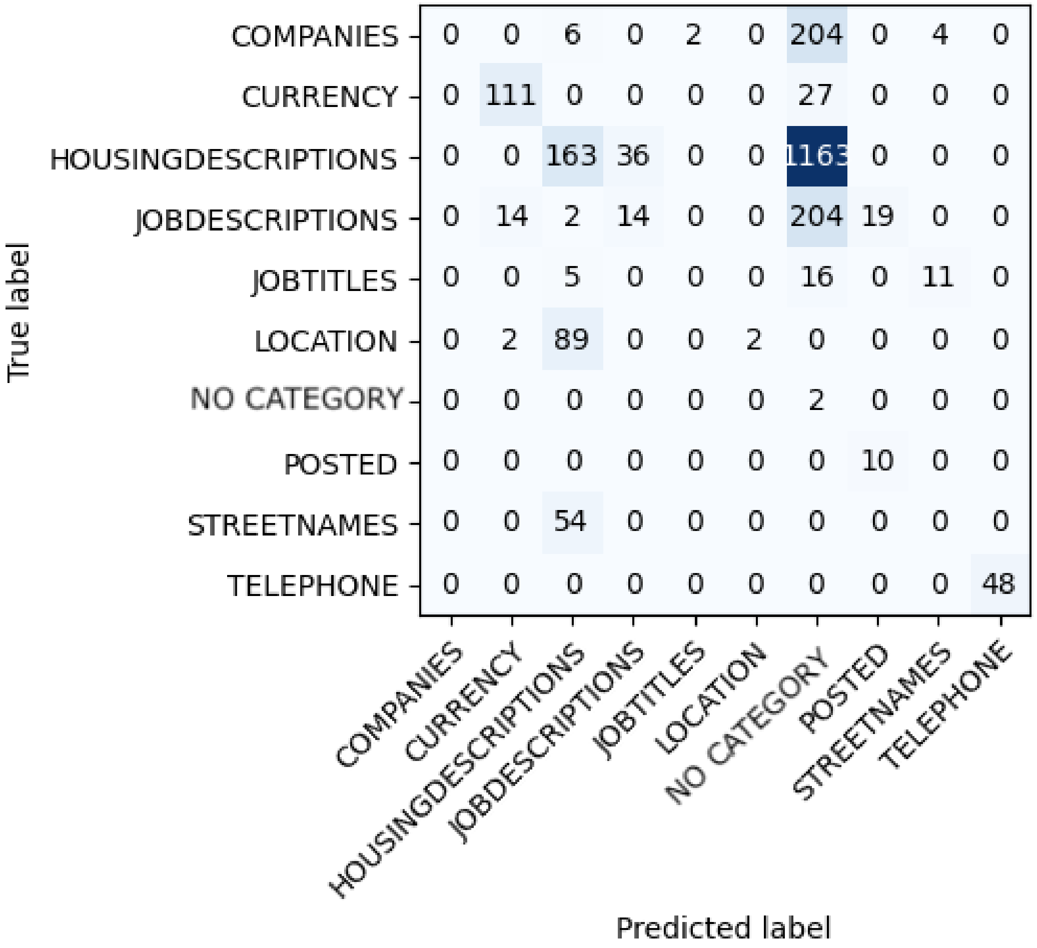
Figure 4.21: Confusion matrix for the markup output produced by the spaCy system for the Indeed jobs data

The confusion matrix for the spaCy comparison can be seen in figure 4.21. The results from Indeed job data was 40.48% precision, 28.33% recall and 33.33% F1 score.

spaCy's processing of the jobs data for Indeed again has done significantly better than all the housing data experiments. As shown in figure 4.21. spaCy predicted a large number of companies and currency correctly although it struggled with everything else. Many of the categories predicted by spaCy came under companies, such as the job descriptions. No locations, posted or job descriptions were predicted by spaCy.

4.10.6 Both Housing and Job Website Results for spaCy

In the Merlin System, the best result for both housing and job data was 65.0% recall, 77.0% precision and 70.4% F1 score.



	COMPANIES	CURRENCY	HOUSINGDESCRIPTIONS	JOBDESCRIPTIONS	JOBTITLES	LOCATION	NO CATEGORY	POSTED	STREETNAMES	TELEPHONE
COMPANIES	0	0	6	0	2	0	204	0	4	0
CURRENCY	0	111	0	0	0	0	27	0	0	0
HOUSINGDESCRIPTIONS	0	0	163	36	0	0	1163	0	0	0
JOBDESCRIPTIONS	0	14	2	14	0	0	204	19	0	0
JOBTITLES	0	0	5	0	0	0	16	0	11	0
LOCATION	0	2	89	0	0	2	0	0	0	0
NO CATEGORY	0	0	0	0	0	0	2	0	0	0
POSTED	0	0	0	0	0	0	0	10	0	0
STREETNAMES	0	0	54	0	0	0	0	0	0	0
TELEPHONE	0	0	0	0	0	0	0	0	0	48

Figure 4.22: Confusion matrix for the markup output produced by the spaCy system for the housing and job data

The confusion matrix for the spaCy comparison can be seen in figure 4.22. The results for both housing and jobs data was 47.15% precision, 26.36% recall and 33.82% F1 score.

spaCy predicted many housing descriptions as no category, meaning it could not predict the text. In this instance, spaCy could not recognise the majority of housing descriptions.

There were also mistakes also made for companies and job descriptions. spaCy could not predict any job titles or companies. In the housing descriptions, spaCy predicted a portion of streetnames and location when and these were wrong.

4.11 Summary and Discussion

This chapter introduced an approach called the Merlin system for the automatic annotation of data scraped from the web. The method uses character-based PPM language models implemented by the Tawa toolkit [150] to segment the text into different classes of text. The chapter discusses the overview of how Merlin works, the implementation of each step and the evaluation of the web mining system.

The Merlin system works well overall. The experiments have shown that with the iterations of balancing the size of the PPM models, this allows models that have been built from smaller training text not to be overwhelmed by the models that have been built from a much larger training data.

The results in this chapter highlighted how there needs to be a balance between each model, to avoid the chance of one model classifying everything over another or for a model to be ignored. For example, despite the location model having been trained from the text containing many locations, the larger companies model would often dominate in the classification as it also included many location names. This was offset by balancing the data and by removing the location names.

The investigation concluded that the Merlin process works well at automatically tagging the data.

As a comparison to the Merlin system, annotation using spaCy was also performed. The results show spaCy produced very poor results.

There were many instances where spaCy could not predict what the text was and provided no markup for the data, primarily for the housing data when the housing and jobs data were combined.

spaCy additionally took a considerable amount of time to train compared to the compression-based approach taken by the Merlin system. The times taken for both Merlin and spaCy can be seen in Table 4.15.

The Merlin system outperforms spaCy's performance in terms of standard measures of precision, recall and F1 score (see Table 4.16). Additionally the amount of time to train the models is considerably less when using Merlin compared to spaCy.

Website	**Merlin System**	**spaCy**
	Time Taken to Train	**Time Taken to Train**
Zoopla	8 minutes	3 days, 2 hours and 23 minutes
OnTheMarket	7 minutes	2 days, 3 hours and 22 minutes
Rightmove	9 minutes	2 days, 5 hours, 42 minutes
Reed	10 minutes	3 days, 10 minutes
Indeed	10 minutes	2 days, 5 hours and 13 minutes
Housing and Jobs	10 minutes	3 days, 6 hours and 27 minutes

Table 4.15: Comparison of the Merlin System and spaCy time taken to train the data. This shows that the Merlin system performed better in time taken to train and tag the data compared to spaCy.

The direct comparison of using the Merlin system to spaCy performance on the same data is shown in Table 4.16.

Website	**Merlin System**				**spaCy**		
	Precision (%)	**Recall (%)**	**F1 Score (%)**		**Precision (%)**	**Recall (%)**	**F1 Score (%)**
Zoopla	89.0	84.0	86.4		50.0	1.82	3.5
OnTheMarket	90.0	74.0	81.2		10.0	1.71	2.93
Rightmove	91.0	78.7	84.4		10.0	1.43	2.50
Reed	89.0	75.0	86.9		36.0	26.47	30.51
Indeed	87.0	77.0	81.6		40.48	28.33	33.33
Housing and Jobs	77.0	65.0	70.4		47.15	26.36	33.82

Table 4.16: Comparison of the Merlin System and spaCy classifer performance output measures. This shows that the Merlin system performed better in all experiments compared to spaCy.

The comparison of the Merlin system and spaCy shows why the Merlin system using PPM was used for the prototypes discussed in subsequent chapters with better performance results and a much quicker time to train the models. The comparison between the F1 Scores shows a significant difference in performance, for example only 3.5% for spaCy compared to 86.4% for Merlin on the Zoopla data; and for the combined Housing and Jobs data, spaCy achieved an F1 score of 33.82% compared to 70.4% for Merlin.

The modification and balancing of models to provide a greater accuracy in classification could be automated rather than using the manual approach described in this chapter for the Merlin system. By automatically reducing or increasing the size of models, better optimised classifications could be made. Additional pre-processing steps could also be used to improve classification to remove further noise.

Chapter 5

Developing and Evaluating Prototype Alpha

5.1 Introduction

This chapter discusses the development and evaluation of the initial Alpha prototype. This includes the methodology Five Design Sheets [115] [117] that was used to consider alternative design ideas.

The Alpha prototype explores data extraction of local points of interest within the North Wales area. The Alpha prototype also features the retrieval of local points of interest, cinema and weather information. The aim of this prototype is to provide proof of concept for extracting data. The company partners wanted an initial prototype to discuss the possibility of extracting information and these areas were chosen.

The database of the developed prototype stores information about points of interest, which is obtained with different web scraping techniques. The information provides the name, where it is located and the category that has been defined for the point of interest.

The web scraping techniques include document parsing and HTML parsing. The application requires the user to enter a search query into the text box. The query is then used to search the database and return the points of interest that match the criteria entered. The results returned will then appear in a table which is displayed to the user. This prototype provides weather information based on the entered query, local points of interest and cinemas. The information may include highest rated, a random selection or recently updated listings of points of interest.

5.2 Five Design Sheets

Five Design Sheets [115][116] is a design methodology where the developer creates five design-sheets that describe the design of the project through ongoing interaction with the client. By the end of the design process, the developer will have produced five sheets of paper with many designs and information associated with them. The aim of this method is to provide a structured process for the developer to follow to design a software tool. The Five Design Sheets methodology enables a developer to create and sketch ideas, discuss them with the clients and then refine the ideas to a solution that is workable. This is where the client would be involved with the ideation and the creation process.

The Five Design Sheets methodology contains several parts which are as follows:

- **Five sheets:** one brainstorm sheet, three design sheets and one realisation sheet.

- **Five stages:** The developer and client would meet, the developer would brainstorm some ideas, create three design-sheets, discuss with the client and then a realisation design is generated. This realisation design would then be implemented using techniques discussed by the developer.

- **Five parts to brainstorm:** In this process, the developer ideate the ideas, filter the ideas down followed by categorising and combine any similar designs created. The user would then refine the design and question what has been created.

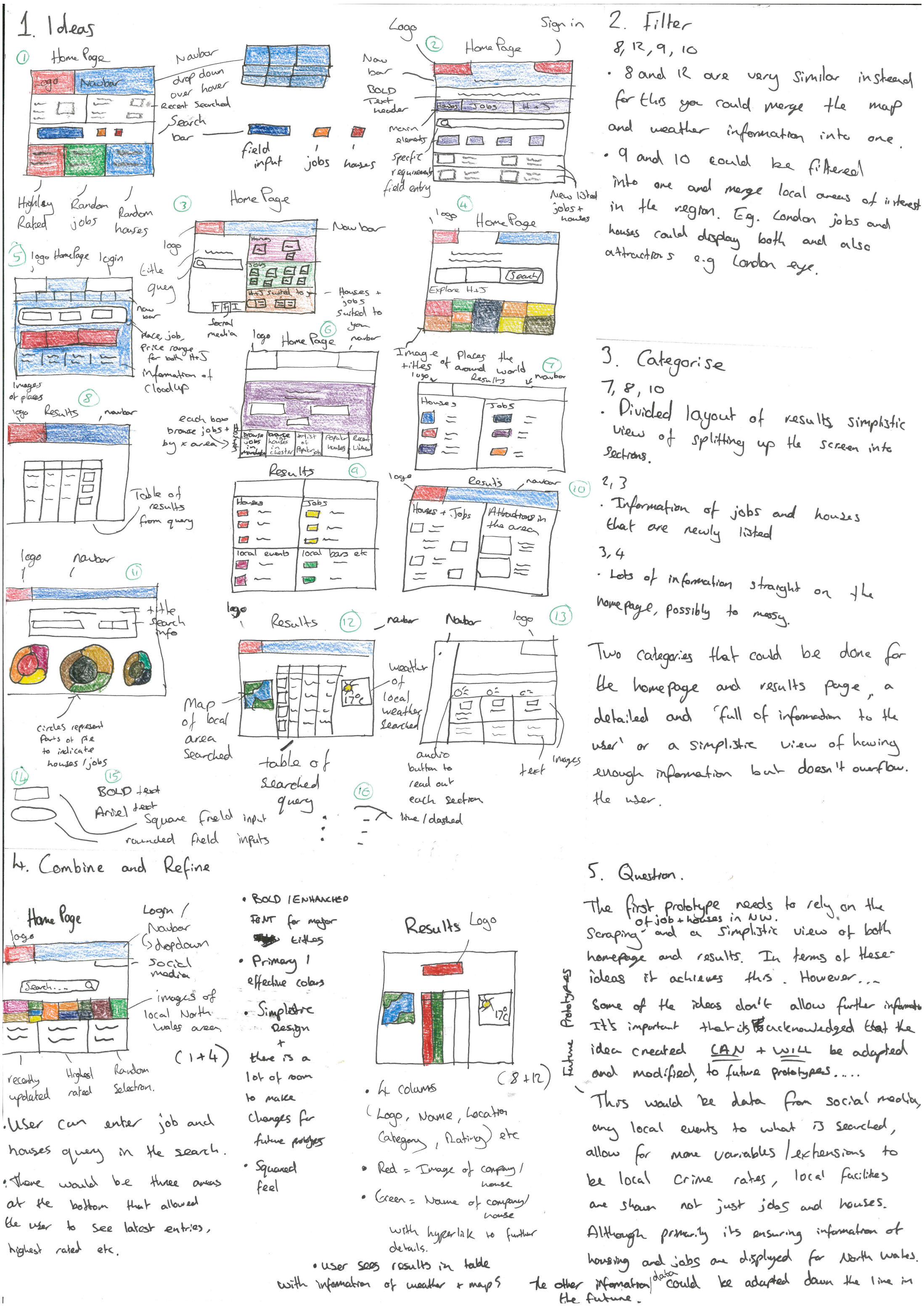
- **Five parts to each sheet:** Layout of the design: this is the vision of what the final visualization would look like. Meta-information should be included, such as title, author, date, sheet number and the task. Focus of the design: this could be key techniques or a novel approach that has been used to create this design. Operations are iincluded which are sketches and some brief descriptions of how the user operates or controls the interface. A discussion of the advantages and disadvantages of the technique are also included in the sheet [115].

5.3 Design of the Alpha Prototype

To devise a design that meets the requirements of the project outline, there needs to be a methodology that allows the developer to approach different designs from multiple mini-designs and concepts. [115].

Five Design Sheets was adopted as it allowed for an iterative process of design and allowed the team to collaborate with regard to the final design.

5.3.1 Sheet 1 — brainstorm



2. Filter

8, 12, 9, 10

- 8 and 12 are very similar instead for this you could merge the map and weather information into one.
- 9 and 10 could be filtered into one and merge local areas of interest in the region. Eg. London jobs and houses could display both and also attractions e.g London eye.

3. Categorise

7, 8, 10

- Divided layout of results simplistic view of splitting up the screen into sections.

2, 3

- Information of jobs and houses that are newly listed

3, 4

- Lots of information straight on the homepage, possibly to messy.

Two categories that could be done for the homepage and results page, a detailed and 'full of information to the user' or a simplistic view of having enough information but doesn't overflow. the user.

4. Combine and Refine

- User can enter job and houses query in the search.
- There would be three areas at the bottom that allowed the user to see latest entries, highest rated etc.

- BOLD / ENHANCED FONT for major titles
- Primary / effective colours
- Simplistic Design + there is a lot of room to make changes for future protypes
- Squared feel

- 4 columns (Logo, Name, Location Category, Rating) etc
- Red = Image of company / house
- Green = Name of company / house with hyperlink to further details.
- user sees results in table with information of weather + maps

5. Question.

The first prototype needs to rely on the Scraping of job + houses in NW. and a simplistic view of both homepage and results. In terms of these ideas it achieves this. However...

Some of the ideas don't allow further informatio It's important that its ~~to~~ acknowledged that the idea created CAN + WILL be adapted and modified, to future prototypes....

This would be data from social media, any local events to what is searched, allow for more variables / extensions to be local crime rates, local facilities are shown not just jobs and houses. Although primarily its ensuring information of housing and jobs are displayed for North Wales. the other information / data could be adapted down the line in the future.

Figure 5.1: Sheet 1 produced during the Five Design Sheets process for the Alpha prototype

The idea of the first sheet that was produced during the Five Design Sheets design process, as shown in figure 5.1, is to consider the data and compose initial designs. This stage is the 'brainstorming' sheet. The process allows the developer to put multiple ideas in a large space of A3 paper. This sheet is where the developer would generate all possible ideas. This encourages the developer to be creative and imaginative in this process.

There are five steps in this sheet:

1. **Generate Ideas.** This is where the developer sketches out as many ideas as possible. These may be mini-ideas or may be comprehensive and complete ideas or ideas that aren't fully materialized or simple ideas or concepts that may be unrealistic or even crazy.

2. **Filter the ideas.** This step takes the ideas the developer has made and starts to remove any duplicates. Any ideas made should be put together.

3. **Categorize the sketches.** This step orders and categorises the sketches into mini-ideas. Any concepts that are similar should be placed together.

4. **Combine and Refine.** This step starts to organise the mini-ideas into bigger solutions. Multiple views can occur here, and there could be different aspects with the same data shown with multiple views.

5. **Questions.** This stage is the final part, where the developer would reflect on what has been created and ask themselves is this the solution that the client wants, and is it fit for the purpose of what is needed to be created? Does it answer the specification requirements asked by the user?

For example, considering point one of the five steps, this involves sketching different overall views of the homepage and results page, alongside with information about how the elements could be placed and what materials could be used.

Taking each of these considerations, the numbering below corresponds to the various points of Sheet 1 shown in figure 5.1.

1. The overall homepage should display a mixture of random jobs and housing along with highly rated local points of interest. There is a drop down hover feature for

the navigation bar, that shows the different features available. This would include recent searches from the results page. These are shown on the homepage to allow accessibility to see what the user has recently searched.

2. There is a search field with specific requirements based on the searching criteria. The homepage shows newly listed data. This includes bold text on the categories to emphasis each part.

3. The search field and title are on the left side of the page. The navigation and logo are at the top of the web page. There is a highlighted information in three sections on the right side of the web page.

4. There are image tiles of places around the world, and different type of cities to allow a shortcut to specific areas of information. The logo and title are at the top and the search field in the center.

5. There is a blue background, with a square navigation bar. The information about Cloodup is broken up in sections at the bottom of the page. There are three titles which indicate the main elements of the website.

6. The bottom of the webpage is split into five sections. The sections are showing the popular listings, the recently recently viewed listings, the browsing of housing in an area and browsing of jobs in an area alongside popular attractions.

7. The webpage is split into two sections. These sections show different sets of information.

8. The results are shown in a table format. This displays all available information from each section based from the query.

9. The webpage is split into equal parts showing housing, jobs, local events and local points of interest e.g., bars.

10. The search results show attractions in the area on the right hand side. There is a mixture of housing and jobs on the left side of the webpage with a pattern of how they are presented.

11. The webpage shows recently searched information in pie charts, which shows the significance of each part. This is where you can hover over a section of the chart in order to expand the information.

12. The results webpage has local weather displayed on the right side and a map of the local area searched on the left side. There is a table of queried search results in the center of the web page.

13. There are audio buttons reading out different text sections meaning it is accessible. The navigation bar is listed on the left side of the webpage and logo on the top right of the webpage.

14. There are square fielded inputs and rounded field inputs considered.

15. The text is bold with Ariel font being used as it provides visual accessibility in reading the text.

16. There are bullet points and dashed points for information.

Further point two to five involves the filtering stage. Points 8, 12, 9 and 10 can be filtered:

1. Points 8 and 12 are very similar. The possibility of merging the map and weather information into one.

2. Points 9 and 10, a possibility of filtering into one and merge local areas of interest in the region. e.g., London jobs and houses could display both and also attractions e.g., London eye.

Considering the five steps from this sheet, points three to five involve categorising the ideas. Points 7, 8 and 10 can be categorised according to the layout of results, providing a simplistic view of splitting up the screen into sections. Points 2 and 3 can be categorised by the information of jobs and houses that are newly listed. Points 3 and 4 include a large amount of information from different areas on the homepage which could be considered messy. Two categories that may be a possibility for the homepage and result page are a detailed and ‘full of information to the user’ or a simplistic view of housing enough information but does not overload the user.

Point four involves combining and refining the ideas made. In this instance, points 1 and 4 for the homepage were combined:

- The user can enter job and houses query in the search.
- There would be three areas at the bottom that allow the user to see the latest entries and highest rated.
- There would be bold text and enhanced font for highlighted entries.
- There would be primary colours used.
- A simplistic design and the ability to make changes for the future.

Ideas 8 and 12 were also combined and refined for the results page:

- There would be four colours used one for the logo, name, location, category and rating.
- The red colour would be the image of the house.
- The green colour would be the name of the company or house.
- If the user were to click the information, it is hyperlinked to further information.
- The user would see results in table with information and information about weather and a map of the searched area.

Questioning

The final point from the brainstorming sheet involves questioning. The first prototype needs to rely on the scraping of information. The reason for using North Wales for the first prototype is because it provides a convenient sample, and an area known to the author. The prototype design needs to have a simplistic view within the homepage and results page. In terms of the ideas discussed, this initial design achieves this goal as the prototype can gather local points of interest when searching for an area. It is important that the idea can be adapted for future developments.

Future prototypes might include data from social media events or local crime rates, points of interest and school ratings. This would provide more information about the user to the area they are searching for.

The prototypes for this project primarily involve housing and jobs and anything further could be an additional benefit for the future.

5.3.2 Sheets 2, 3, 4 — Initial Designs

After the first design sheet, the design in collaboration with the client creates three individual design sheets to place three ideas that were generated from the initial Sheet 1 brainstorming exercise. The reason why there are three sheets and not one is because it facilitates a more thorough discussion that goes into the detail of every element and ideas that have been created. Without it, this would mean that there would be too few designs. However, if there are too many designs, it would waste the client's time. It could be that there are only two designs that are reasonable / realistic. However, it is better to create a third design even if it seems unfeasible. This is because the client might be able to take aspects of each sheet and might be able to extend ideas created through discussion. The client could potentially see the idea of the application further than the developer may see it.

The content on these sheets contains:

1. **The Layout of the Design.** This provides the vision of what the final design prototype would look like. This would be a sketched version of the application.

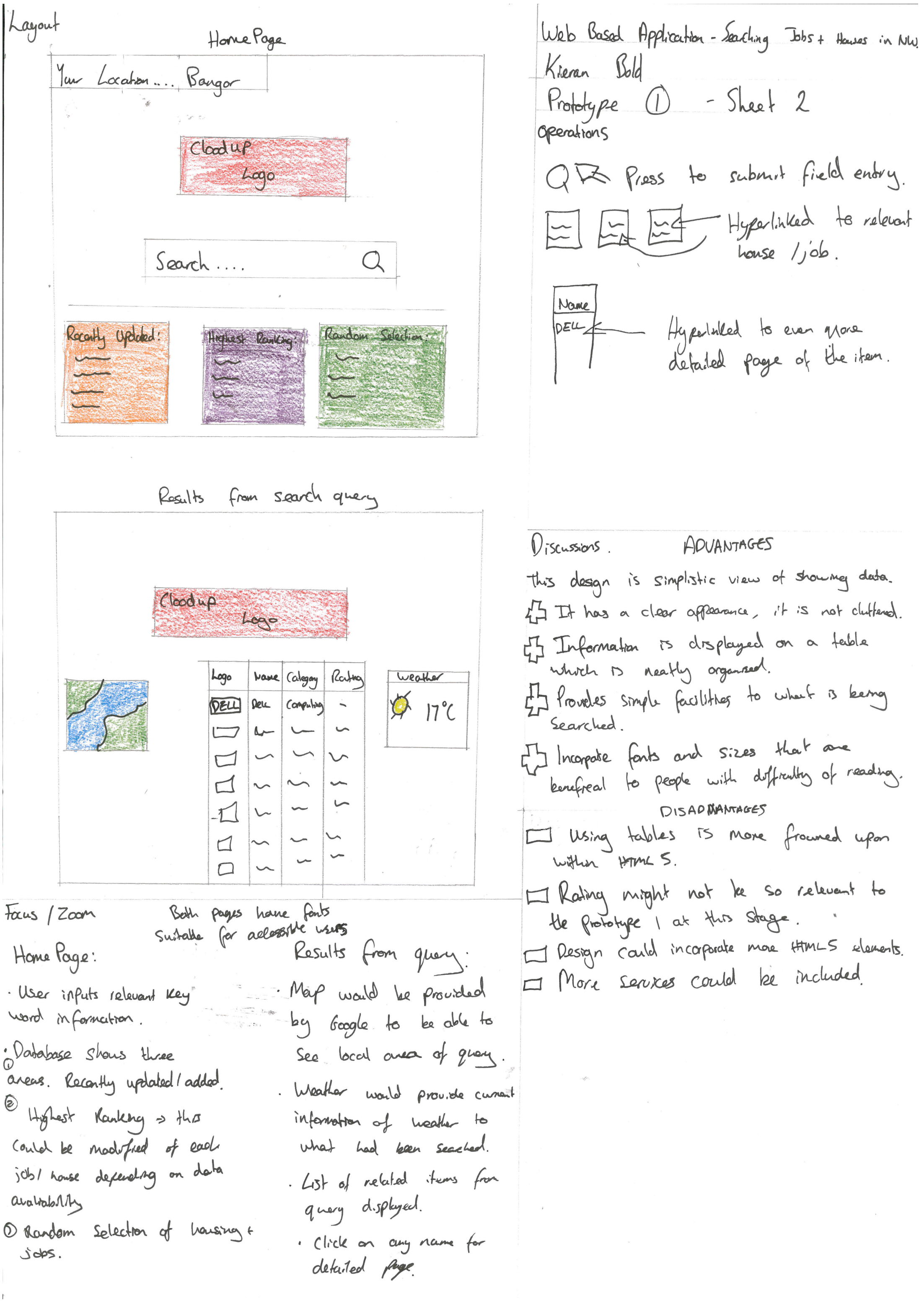
2. **Focus.** The developer may need to display data in multiple aspects or views. There could be some specific parts that the developer might want to focus on or zoom into.

3. **Operations.** This is where sketches and brief descriptions of how the developer would operate around the application / user interface.

4. **Discussion.** The developer discusses the advantages and disadvantages of the technique considered. This is a brief discussion of the designs.

5. **Meta-information.** The developer should include a title, authors, date, sheet number and what the task is.

Sheet 2 — Initial Designs



Web Based Application - Searching Jobs + Houses in NW.
Kieran Bold
Prototype ① - Sheet 2
Operations

Press to submit field entry.

Hyperlinked to relevant house /job.

Hyperlinked to even more detailed page of the item.

Discussions. ADVANTAGES

This design is simplistic view of showing data.

- It has a clear appearance, it is not cluttered.
- Information is displayed on a table which is neatly organised.
- Provides simple facilities to what is being searched.
- Incorpate fonts and sizes that are benefical to people with difficulty of reading.

DISADVANTAGES

- Using tables is more frowned upon within HTML 5.
- Rating might not be so relevent to the Prototype 1 at this stage.
- Design could incorporate more HTML5 elements.
- More services could be included.

Focus / Zoom

Both pages have fonts suitable for accessible users

Home Page:

- User inputs relevant key word information.
- Database shows three areas. ① Recently updated / added. ② Highest Ranking → this could be modified of each job / house depending on data availability ③ Random Selection of housing + jobs.

Results from query:

- Map would be provided by Google to be able to see local area of query.
- Weather would provide current information of weather to what had been searched.
- List of related items from query displayed.
- Click on any name for detailed page.

Figure 5.2: Sheet 2 produced during the Five Design Sheets process for the Alpha prototype

On figure 5.2, Sheet 2 produced from the Five Design Sheets process provides the overall view of the initial prototype and then discusses the layout of the homepage and results page:

Homepage

- The user user inputs related keyword information.
- The database shows three sections. The main highlights are recently uploaded and added.
- There is also a highest ranking. This could be modified of each job / house depending on data availability.
- The ability of a random selection of jobs and housing would appear.

Results from query

- The map would be provided by Google's API that would visualise the local area based on the query.
- Weather would provide current and forecast weather to the area being searched by the user.
- The list of related information would come from the query.
- The table of information would result from the user clicking on any item for a detailed further page of information clicked.

The operations that can be performed as defined in the layout from Sheet 5.2 are the following:

- The user would have the ability to submit the query by pressing the button next to the input field.
- The information will be hyperlinked to related information in the square boxes.
- Each individual element in the square boxes are hyperlinked to detailed pages from the basic information shown.

The discussion section from sheet 5.2 is that the design uses the KISS principle [114] to show the information.

Additionally the advantages and disadvantages of this design are:

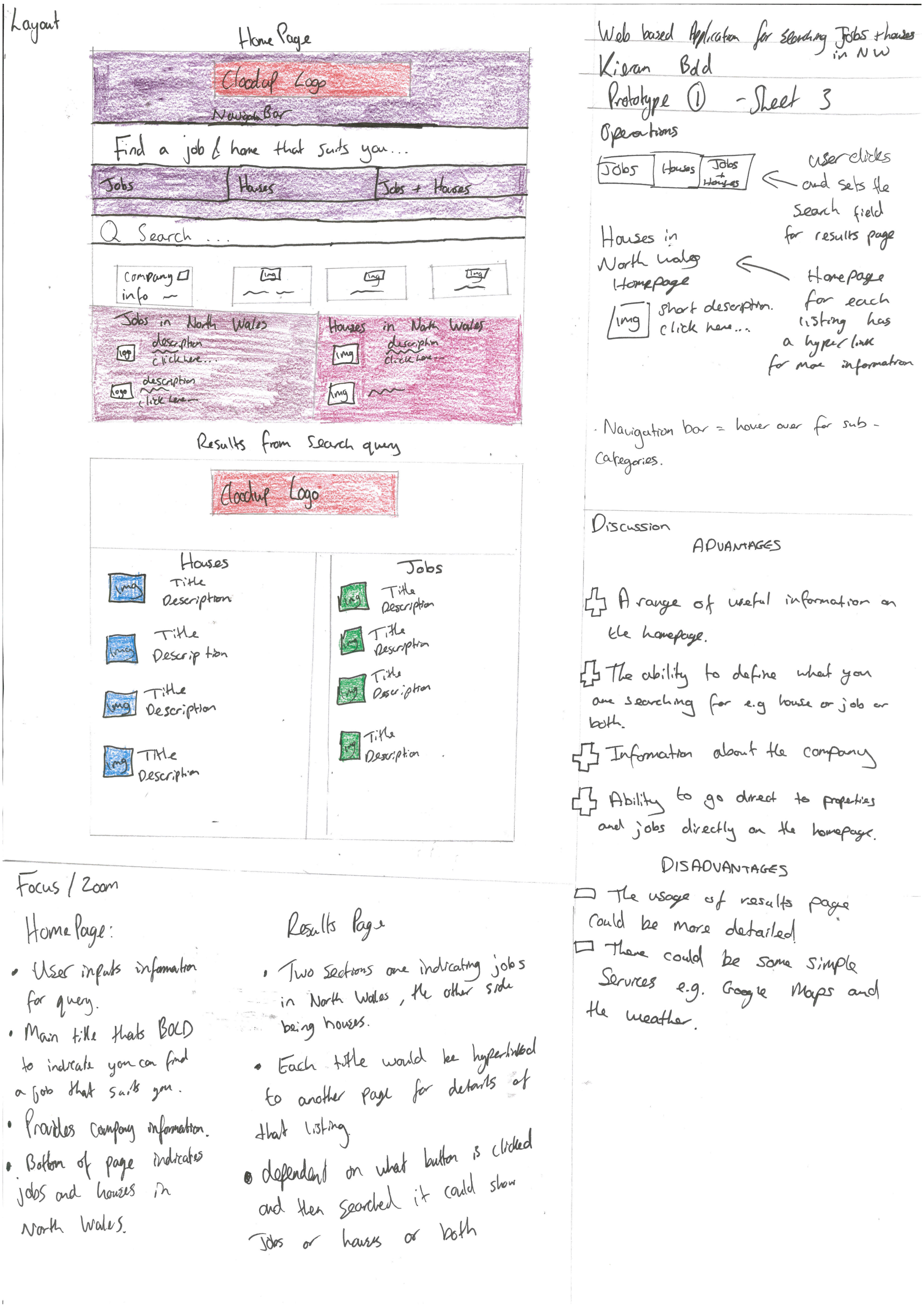
Advantages

- It has a clear appearance, and it is not cluttered.
- The information displayed is in a table which is neatly organised.
- It provides local attractions, such as cinemas, restaurants and bars of what is being searched.
- It incorporates fonts and sizes that are helpful for the visually impaired.

Disadvantages

- The rating of points of interest, e.g., cinemas, shops and cafes, might not be so relevant to Alpha prototype at this stage.
- The design could incorporate more HTML 5 elements.
- There could be more services included in the prototype.

Sheet 3 — Initial Designs



Web based Application for searching Jobs + houses in NW

Kieran Bdd

Prototype ① - Sheet 3

user clicks and sets the search field for results page

Homepage for each listing has a hyper link for more information

· Navigation bar = hover over for sub-categories.

Discussion

ADVANTAGES

- A range of useful information on the homepage.
- The ability to define what you are searching for e.g house or job or both.
- Information about the company
- Ability to go direct to properties and jobs directly on the homepage.

DISADVANTAGES

- The usage of results page could be more detailed
- There could be some simple services e.g. Google Maps and the weather.

Focus / Zoom

Home Page:

- User inputs information for query.
- Main title thats BOLD to indicate you can find a job that suits you.
- Provides company information.
- Bottom of page indicates jobs and houses in North Wales.

Results Page

- Two sections one indicating jobs in North Wales, the other side being houses.
- Each title would be hyperlinked to another page for details of that listing
- dependent on what button is clicked and then searched it could show Jobs or houses or both

Figure 5.3: Sheet 3 produced during the Five Design Sheets process for the Alpha prototype

As shown in Figure 5.3, Sheet 4 from the Five Design Sheets provides the overall view of the prototype and then discusses the layout of the homepage and results page:

Homepage

- The user inputs information in the search query box.
- The main title that is bold will indicate to the user that you can find a information that suits you.
- It provides company information.
- The bottom of the page indicates jobs and housing in North Wales.

Results page

- There are two sections — one jobs in North Wales and the other side being housing.
- Each title would be hyperlinked to another web page for details of the listing.
- Dependent on what button is clicked and then searched, it could show jobs, housing or both.

The operations defined in the layout from Sheet 5.3 are the following:

- The user clicks the buttons and sets the search field for the results page.
- The homepage for each listing has a hyperlink for more information.
- The navigation bar allows for hovering to show the sub-categories.

From the discussion section from Sheet 5.3, the advantages and disadvantages of the design are:

Advantages

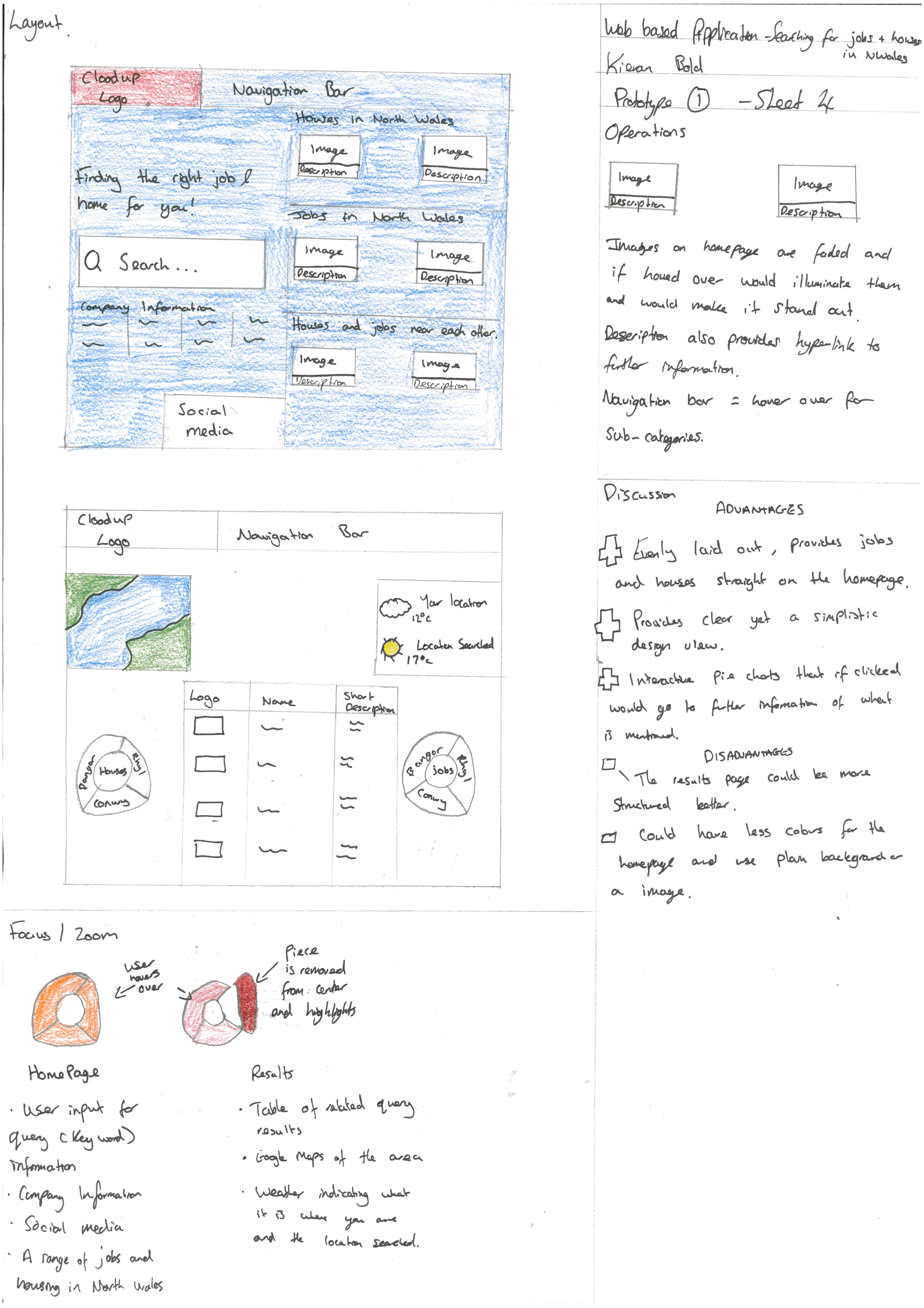
- There is a range of useful information straight on the homepage to catch the users' attention.

- There is the ability to define what you are searching for with the option filters.
- The prototype will have the ability to go directly to the properties and jobs from the homepage.

Disadvantages

- The usage of results page could be more detailed.
- There could be some simple services for example, weather.

Sheet 4 — Initial Designs



Web based Application - Searching for jobs + houses in NWales

Kieran Bold

Prototype ① - Sheet 4

Layout

Operations

Images on homepage are faded and if hoved over would illuminate them and would make it stand out.

Description also provides hyper-link to further information.

Navigation bar = hover over for Sub-categories.

Discussion

ADVANTAGES

- Evenly laid out, provides jobs and houses straight on the homepage.
- Provides clear yet a simplistic design view.
- Interactive pie charts that if clicked would go to further infomation of what is mentioned.

DISADVANTAGES

- The results page could be more structured better.
- Could have less colours for the homepage and use plain background or a image.

Focus / Zoom

HomePage

- User input for query (Key word) Information
- Company Information
- Social media
- A range of jobs and housing in North Wales

Results

- Table of related query results
- Google Maps of the area
- Weather indicating what it is where you are and the location searched.

Figure 5.4: Sheet 4 produced during the Five Design Sheets process for the Alpha prototype

As shown in figure 5.4, Sheet 4 from the Five Design Sheets provides the overall view of the prototype and then details the layout of the homepage and results page:

Homepage

The design for the homepage includes the following:

- User input for query (keyword) information.
- Company information.
- Social media.
- A range of jobs and housing in North Wales.

Results page

The design for the results page includes the following:

- A table of related query results.
- A Google Maps of the area.
- Localised weather of the searched criteria.

The operations section from Sheet 5.4 are the following:

- Images on the homepage faded and if hovered over would illuminate.
- A description also provides a hyperlink to further information.
- A navigation bar can be hovered over for sub-categories.

Concerning Sheet 5.4, the advantages and disadvantages of the design are:

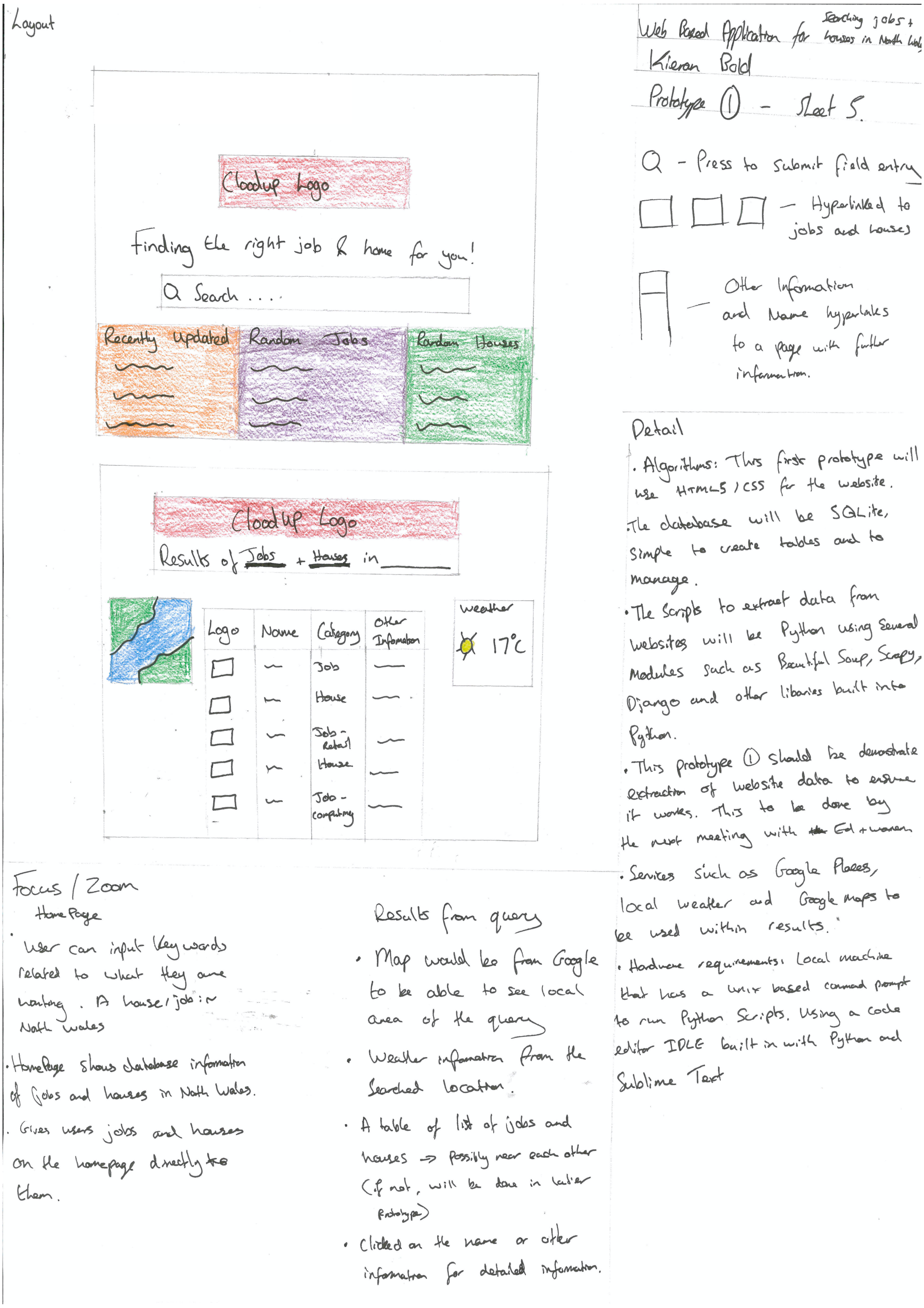
Advantages

- Evenly laid out, the prototype provides jobs and houses straight on the homepage.
- The prototype design provides a clear yet a simplistic design view.
- The interface provides interactive pie charts that if clicked would navigate to further information.

Disadvantages

- The results page could be better structured.
- The prototype design could have less colours for the homepage and instead have a plain background or image.

5.3.3 Sheet 5 — Realization



Web Based Application for Searching jobs + houses in North Wales
Kieran Bold
Prototype ① - Sheet 5.

Q - Press to submit field entry

□ □ □ - Hyperlinked to jobs and houses

Other Information and Name hyperlinks to a page with further information.

Detail

- Algorithms: This first prototype will use HTML5 / CSS for the website.
- The database will be SQLite, simple to create tables and to manage.
- The Scripts to extract data from websites will be Python using several Modules such as Beautiful Soup, Scrapy, Django and other libaries built into Python.
- This prototypee ① should be demonstrate extraction of website data to ensure it works. This to be done by the next meeting with ~~the~~ Ed + women
- Services such as Google Places, local weather and Google maps to be used within results.
- Hardware requirements: Local machine that has a unix based command prompt to run Python Scripts. Using a code editor IDLE built in with Python and Sublime Text

Focus / Zoom

Home Page

- User can input key words related to what they are wanting. A house / job in North wales
- HomePage shows database information of Jobs and houses in North Wales.
- Gives users jobs and houses on the homepage directly to them.

Results from query

- Map would be from Google to be able to see local area of the query
- Weather information from the Searched location.
- A table of list of jobs and houses → possibly near each other (if not, will be done in later prototype)
- Clicked on the name or other information for detailed information.

Figure 5.5: Sheet 5 produced during the Five Design Sheets process for the Alpha prototype

The designer in collaboration with the client then progresses onto the final sheet which is the realization sheet. The developer would consider at this stage what the visulisation tool could look like. The designer is in collaboration with the client would also look into what specific features and services that the product will have and also how the users operates it e.g., operations / functions that may be used. The noted difference between this sheet and the other sheets is that discussion is different with detail. The details section should include detailed discussion of how the product would work or how it will be created. Details that could be included are:

1. **Description** of what the algorithms are going to be used.

2. **Any dependencies.** This would be if there are any software libraries that would be needed to build the tool you are using.

3. **Estimates** of the cost or time to build this product, this could also man-months required of effort into the product.

4. **Specific requirements.** This is such as a details of any materials and quantities that may be required for example hardware requirements.

Figure 5.5 presents the realisation of the design for the initial prototype. This was arrived after the Five Design Sheets process. Using the information above, the elements are the following:

Homepage

- The user can input keywords related to what they are requiring. Concerning a house or job in North Wales.

- The Homepage shows database information of jobs and housing in North Wales.

- The final sheet gives the user jobs and housing directly on the homepage with the ability to go directly to each listing for further detail.

The elements for the results from the query are as follows:

- The map would be from Google's API to be able to see local area.

- The weather information will come from the search query.

- There is a table displaying the jobs and houses information.
- There is the ability to click on the name for detailed information.

The operations for figure 5.5 are:

- You can submit the search field with the criteria.
- There are hyperlinks to the jobs and houses throughout the homepage and results.
- There is other information and hyperlinks to further information on the pages.

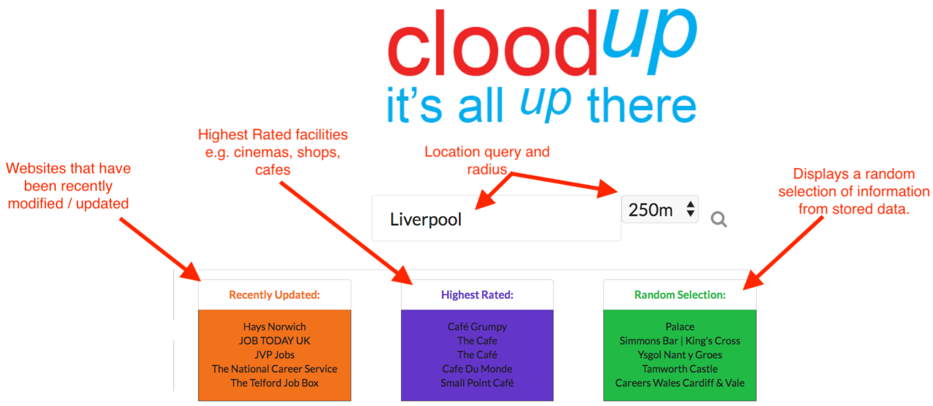
The detail for figure 5.5 is as follows:

- Algorithms: This first prototype will use HTML5/CSS for the website.
- The database will be SQLite. It's simple to use, manage and create tables.
- The scripts to extract data from the websites will be implemented in Python using modules such as BeautifulSoup, Scrapy and Django. The reasoning for using Python is that it provides an easy-to-use framework to build upon.
- This prototype should demonstrate extraction of website data to ensure it works.
- Services such as Google Places, local weather and Google Maps will be used within the results.
- The hardware requirements are: a Local machine that has a Unix based command interface; interactive development environment IDLE that is built within Python; and using Sublime Text for general code.

5.3.4 Design Outcome

The feasibility of the approach depends on whether the data extraction and fusion between two data sources is going to be a viable product. The data that will be used in this prototype will be produced from Google Places, which will do a search of a place and it's points of interest from a specified location query. This could be adapted in the future to include further data within the area.

Website Design: The design process has resulted in a simplistic two-page design, one being the homepage of where the user would input the search query (see Figure 5.6), the other page being the results from the query (see Figure 5.7). The Five Design Sheet methodology for this has been followed through, taking into consideration different design concepts and also ensuring accessibility standards have been followed e.g., setting a minimum font size to ensure people with visual impairment can view the websites [1]. The pages that will be created use HTML5, CSS and Semantic UI. Semantic UI allows fluidity of the design instead of having static buttons, navigation bar and so on.


cloodup
it's all up there
Websites that have been recently modified / updated
Highest Rated facilities e.g. cinemas, shops, cafes
Location query and radius
Displays a random selection of information from stored data.
Liverpool
250m

Recently Updated:

Hays Norwich
JOB TODAY UK
JVP Jobs
The National Career Service
The Telford Job Box

Highest Rated:

Café Grumpy
The Cafe
The Café
Cafe Du Monde
Small Point Café

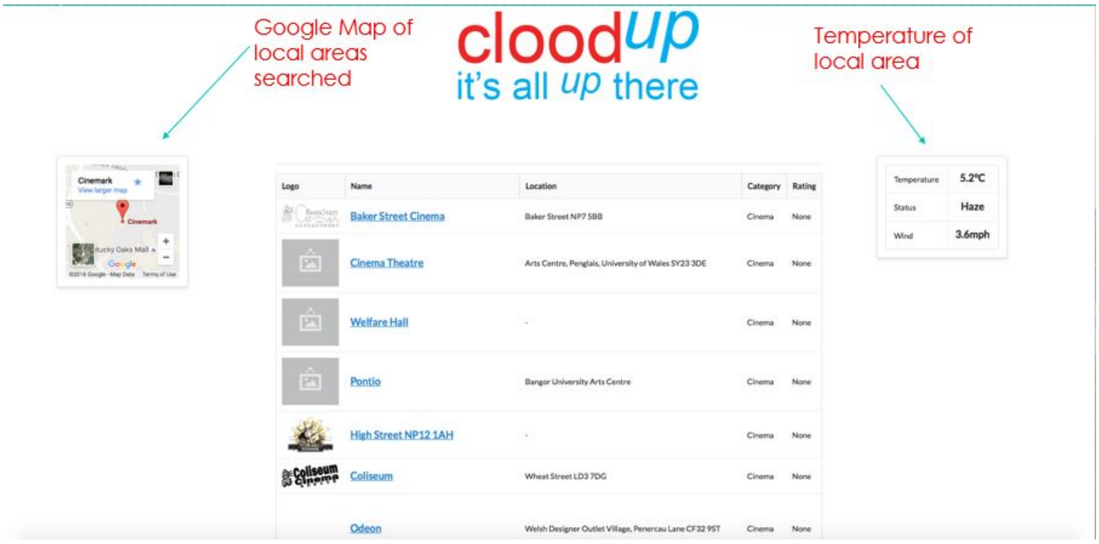
Random Selection:

Palace
Simmons Bar | King's Cross
Ysgol Nant y Groes
Tamworth Castle
Careers Wales Cardiff & Vale

Figure 5.6: Homepage of Alpha prototype

The homepage of the website provides the logo of the business, a search field and three boxes. The three boxes indicate websites that have been recently updated, these include "Recently Updated", "Highest Rank" and "Random selection". This is triggered when searching is done prior to searching. The highest rating element would rank them in order. Random selection is where each time the homepage is refreshed, new results would appear.

[1] http://www.bbc.co.uk/guidelines/futuremedia/accessibility/html/



Logo	Name	Location	Category	Rating
	Baker Street Cinema	Baker Street NP7 5BB	Cinema	None
	Cinema Theatre	Arts Centre, Penglais, University of Wales SY23 3DE	Cinema	None
	Welfare Hall	-	Cinema	None
	Pontio	Bangor University Arts Centre	Cinema	None
	High Street NP12 1AH	-	Cinema	None
	Coliseum	Wheat Street LD3 7DG	Cinema	None
	Odeon	Welsh Designer Outlet Village, Penercau Lane CF32 9ST	Cinema	None

Temperature	5.2°C
Status	Haze
Wind	3.6mph

Figure 5.7: Results Page of Alpha prototype showing the results of the query, the location searched and the temperature of the area

The results page of the website provides a table of scraped results. It also includes a Google Map of the area being searched and the temperature of the local area. The Google Map was made small as it only needed to provide the specific area entered. The temperature of the area was used as the company partners felt that this would be a good contribution to show dependent on the activity the person is searching for. The website uses a range of different services to provide a platform that extracts points of interest such as cafes, cinemas, local areas of interest and schools.

5.3.5 Summary

The company partners found the Five Design Sheet design process to be lengthy. Mr. Edwin Smith said *"I had difficulty breaking down elements of this design as I already had pictured what I wanted."* Edwin reflected that the Five Design Sheets provides an opportunity to look into the finer detail that you could ignore. Mr. Warren Greveson said *"I appreciate the process of the Five Design Sheets. It does allow a project to be given a broader scope and for ideas to be flown more easily. I would however not use this myself in the future as I prefer to work over several iterations and processes"*.

When these comments were made from the company partners, a discussion was held on why the developer had found using the Five Design Sheet methodology helpful and allowed for further creativity and opportunity. The insights gained from this early prototype led to the design for the next prototype. Additionally it provided the opportunity to include different elements that were not considered initially by the company partners.

These pages were presented to the founder of the Five Design Sheets, Professor Jonathan Roberts of Bangor University. He stated that the design could have have gone even further by breaking down the individual elements of the ideas on sheet one, instead of drawing a whole web page and considering elements. If the approach of using Five Designs Sheets were to be done again, the focus on more individual aspects related to the design would be investigated more thoroughly.

5.4 Implementation of Alpha Prototype

There are many ways of performing data extraction from the web. This section focuses on creating a prototype that investigates extracting information from a few websites that hold information about cinemas or local areas of interest. This has been done for proof of concept for extracting elements from websites. The prototype has been called Alpha for reference. The Alpha prototype also involved setting up an essential structure for a website and allowing the user to understand how extraction is done. The Alpha prototype explored the different methods of data extraction that can be used.

5.4.1 Use-Case Model Survey

The Alpha prototype has two different types of users: Admin and User.

Admin

The Admin does the maintenance, configuration and installation of the system. Figure 5.8 is the Use-case diagram of the Admin user.

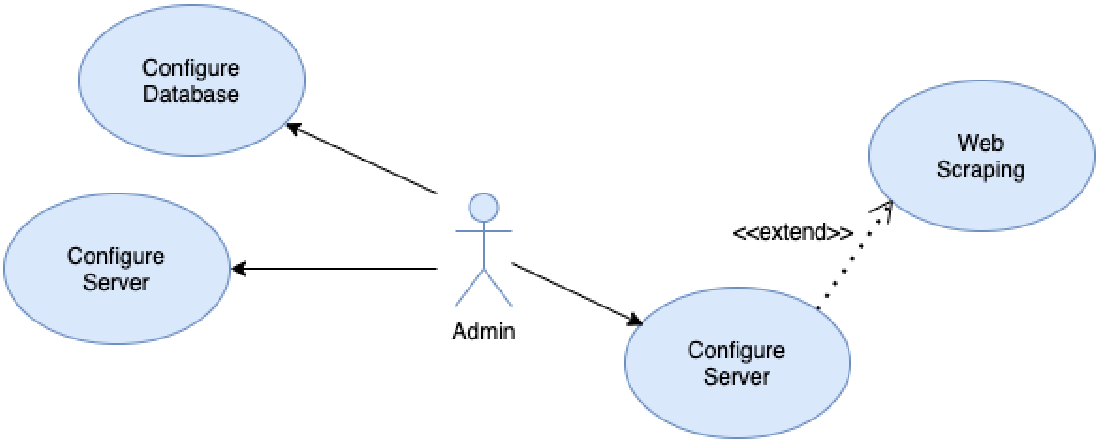


Figure 5.8: Alpha Prototype — Admin Use-case diagram

The use-case scenarios as illustrated in the figure is as follows:

- Configure Database: This involves the configuration of the database settings and maintenance.
- Configure Server: Maintenance of the software, running operations.
- Populate Database: Add new points of interest to the existing database tables.
- Control of web scraping techniques: This is used to retrieve points of interest.

User

The user has access to the main functions of the system. Figure 5.9 provides the Use-case diagram for the User.

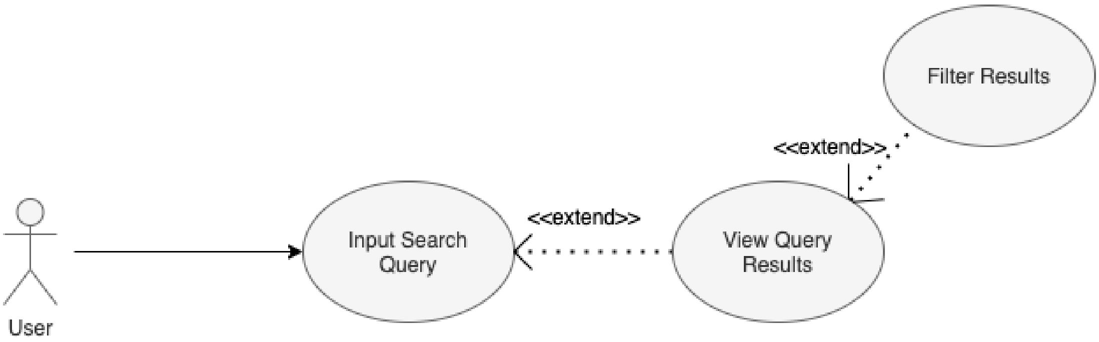


Figure 5.9: Alpha Prototype — User Use-case diagram

The use-case scenarios as illustrated in the figure is as follows:

- Input Search Query: A user can submit a search query to the Alpha prototype interface.

- View Query Results: Then the user can view results that the query returns.

- Filter Results: The user can apply filters to tailor the returned results.

5.4.2 Architecture Diagram for the Alpha prototype

The architecture diagram, Figure 5.10, gives an overview of how the Alpha prototype works. Each column represents a tier of the MVC design pattern. The boxes inside each column represent the processes of the Alpha prototype.

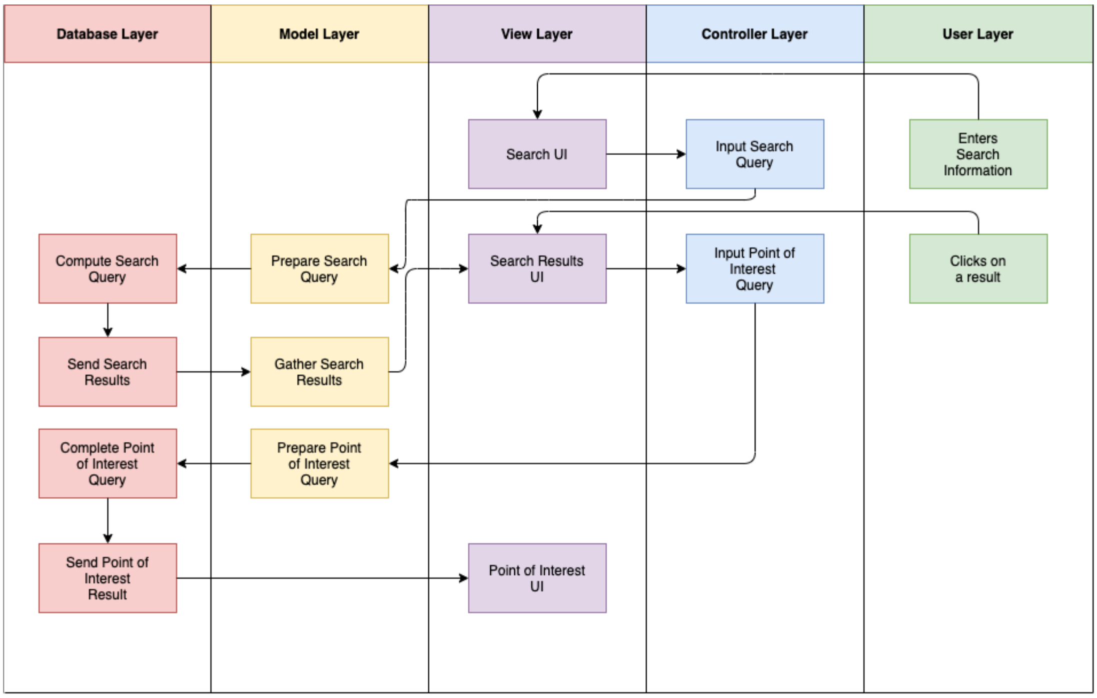


Figure 5.10: Alpha Prototype — Diagram displaying the architecture of the MVC model

5.4.3 Home Page of the Alpha Prototype

The home page of the Alpha prototype is the landing page that the user will be interacting with. The elements for this prototype that the home page will need to contain are:

- a form where the user can supply a search query;

- search button which is used to start the query;

- three individual tables that display the recently updated locations, highest rated locations and a random selection of different places.

5.4.4 Results Page of the Alpha Prototype

The Results page of the Alpha prototype displays the query that the user submitted. The Results page template has the following elements included:

- google maps displaying the location of the criteria that matches the search;
- weather information based on the location queried;
- a table that contains all the results found from the query entered.

5.4.5 Component Diagram Process

The Alpha prototype has five steps for the system process. The component diagram of this prototype can be found in Figure 5.11.

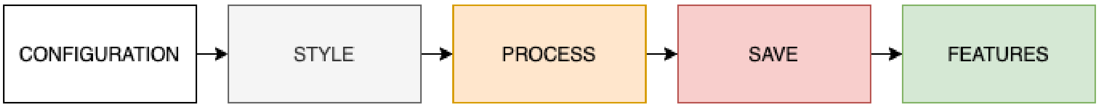


Figure 5.11: Alpha Prototype — Component Diagram Process

- **Configuration:** The admin would ensure that configuration files are setup for the Alpha prototype to run and maintained.
- **Style:** This involves the styling of the Alpha prototype web pages.
- **Process:** This involves the methods of how the Alpha prototype is processed.
- **Save:** This involves saving any new information from web scraping techniques into the database.
- **Features:** This involves any additional functionality or features used within the Alpha prototype such as weather and temperature services.

5.4.6 Managing Web Pages of the Alpha Prototype

The *urlopen* [2] module within Python's library takes a web page address input and returns the web page content in a Python string. This string contains all of the HTML elements containing the class tags, which are needed to be displayed.

The Alpha prototype uses information provided by `http://ukcinemas.org.uk` which is a website that contains information about cinemas located in the United Kingdom. Cinemas were used as part of the Alpha prototype as they provide many locations and details, such as name, ranking and meta information. The website is structured and an excellent candidate to experiment with web scraping for this prototype.

If a user were to enter an unrelated query to the cinemas' website, it would then use Google Place's API to populate the database with results found.

5.4.7 Connection for the Database for the Alpha Prototype

As Django is built-in to Python, upon installation, a file that contains the settings is created for the database. If the developer needs to change the database type, they would need to change the file's connection setting. If the project were to be taken outside of the localhost, it would require the project's settings to be changed. The connection details for the Alpha prototype are in table 5.1.

Engine:	The database engine used.
Name:	The name of the database.
User:	The username used to connect to the database.
Password:	The password for the database.
Host:	The location of the database.
Port:	The port used to connect to the database.

Table 5.1: Database configuration settings for Alpha Prototype

When the database is connected and configured, it can then be accessed. Python's list data structure allows a user to query the database and store all of the data from the query in a list. A major benefit of Django is that when queries are constructed, they are

[2] https://docs.python.org/3/library/urllib.html

performed on the application side and then translated to work with the database that has been chosen.

5.4.8 Styling the Web Pages of the Alpha Prototype

The web pages are linked to the styling sheets via the template language. This website uses Semantic UI [3], which is a framework that is used to design web pages.

The tagging system is characterised into different sections. This allows different design functions to be placed onto an element by inputting a keyword to the HTML class name. The advantage of Semantic UIs is that all of the styles are independent. This approach adopts the keyword-based styling instead of structure, parent classes or parent tags.

5.4.9 Processing and Saving: the data for the Alpha Prototype

A database model is created by a function called PointsOfInterest for the Alpha Prototype, which allows for the database to save facility information about a specific cinema venue. The Alpha prototype venue variables include name, location, description, category, rating, website, logo and last updated listings have been created to store information.

The homepage returns the top 5 points of interest ranked from lowest to largest. Updating objects or even inserting objects is done by constructing a data set that matches the table of requirements and then saving the data.

As the data is being processed for the website that is extracting information about cinemas, the software also populates the database from the search query.

When all the data has been retrieved, the data needs to be saved to the connected database. Data is stored in a list with a total of points of interest and number of categories. The data can be saved into the database by looping through the list and saving each list index to the respective table.

5.4.10 Modules of the Alpha Prototype

Figure 5.12 shows the main modules that were used to implement the Alpha prototype.

[3] https://semantic-ui.com

Figure 5.12: Overview of Modules in Alpha prototype

Google Maps

The Google Maps feature is connected to a web service Python wrapper module of googlemaps [4]. The service allows the interface to have a map by supplying it with coordinate or place information. The information that is then supplied allows interactivity between the map and the user. The location query that the user enters is coded to find the place, and then this is paired with the map address and it is then supplied back.

Google Places

Google Places [5] is an API service that returns information about places using HTTP requests. The Places API provides establishments, geographic locations and points of interest. Parameters can be passed to the API such as the name of the facility, the category of the facility or location. The information that can be returned can vary from basic information such as name and location to detailed information such as opening and closing times.

Open Weather API

OpenWeatherMAP [6] is a service that provides weather data for free, including the current weather, forecast and also historical data. It's implemented using PyOWM [7] which is a client Python library for the OpenWeatherMap API. It allows easy consumption of OWM weather data from Python applications by a simple object model. The API would return a JSON file that contains different information, including weather status, temperature and humidity.

[4] https://pypi.python.org/pypi/googlemaps/
[5] https://console.developers.google.com
[6] http://openweathermap.org/api
[7] https://pyowm.readthedocs.io/en/latest/

5.5 Evaluation of the Alpha Prototype

This section evaluates the Alpha prototype that have been created for the project. The Alpha prototype investigated into extracting different local areas of interest with a keyword query. The Alpha prototype used the Google Places API to retrieve data relevant for the search. The prototype was created as experimentation to explore extraction methods utilising an API.

The evaluation of the Alpha Prototype uses keywords entered by the developer and then an relevance is calculated from the results.

Figure 5.2 lists the results from the evaluation. The table indicates the keywords that were used to query the prototype, how many results were found and how many results from the results found were used for the sample.

Additionally the table mentions the relevance and the average score from the sample. The results indicated that some search results found results outside of the region specified; for example, the query ‘schools in Colwyn Bay’ gave results of schools outside of the Colwyn Bay area, although they were still related to schools.

Further design implementations and features would need to be incorporated for a more precise search result.

Keywords	**Results Found**	**Results Used**	**Relevance**	**Accuracy (%)**
cinema	1000+	100	100 out of 100	100
cinema UK	300	30	20 out of 30	66
schools Colwyn Bay	20	20	19 out of 20	95
shops in Bangor	60	20	12 out of 20	60
coffee shops in London	400	30	21 out of 30	70
garden centre Llandudno	15	15	13 out of 15	86
car garages Bangor	25	25	23 out of 25	92
pubs in Rhyl	40	40	40 out of 50	80
restaurants in Conwy	300	100	80 out of 100	80
leisure centres centres in conwy	20	20	18 out of 20	90
church gwynedd	120	50	35 out of 50	70
schools Bangor	70	30	20 out of 30	66
banks in llandudno	30	30	28 out of 30	93
pet store in abergele	20	20	18 out of 20	90
bicycle stores in mochdre	12	12	8 out of 12	66
beach in conwy	4	4	4 out of 4	100
hospitals gwynedd	6	6	6 out of 6	100
dentist holyhead	7	7	6 out of 7	100
hotel in abersoch	120	120	70 out of 120	58
supermarkets in llandudno	24	24	20 out of 24	83

Table 5.2: The Alpha Prototype evaluation of queries searched with details and the calculated relevance.

5.6 Summary of Alpha Prototype

The Alpha prototype has demonstrated that data extraction methods conducted make it possible to extract facility data from Google Places, which can be incorporated into the information provided by this project's website. However this was not felt necessary for the initial prototypes that were developed for this project as it was out of scope and has been left for future work.

The results from Table 5.2 indicate a good range of accuracy results that are being retrieved from Alpha prototype evaluation queries. The lowest accuracy was 58% and the highest being 100%. These results are promising as it demonstrates proof of concept of retrieving and displaying relevant results to a user search query. These methods will now be enhanced and improved, incorporating them into the following chapters for housing and jobs.

Chapter 6

Developing and Evaluating Prototype Beta I

6.1 Introduction

This chapter discusses the development and evaluation of the Beta I prototype that was produced for this project. This includes the methodology that was used for this prototype and the suitability of its design for this project.

The Agile Design process involves taking tasks and breaking them into small increments with minimal planning. Iterations using this design process are done within short times. Agile design [93] [51] allows you to build a product through communication, collaboration, and small but rapid iterations in order to sustain agility that allows the development team to adapt to a changing environment. This model relies heavily on the customers being apart from the team so that changing requirements are adapted to.

Agile development has several main components, which are:

- Selection of the project and coming up with the vision.

- Initiation of the project by obtaining the stakeholder participation, funding and creators team.

- Rapid development to create iterations that meet the changing needs of the stakeholders. There are release iterations that may not be fully completed or functional, although the designers believe that it is good enough for the purpose of the users.

- Release of the product (end game).

- Production — ensure that there is continued development, maintenance and support for the system.

- Retirement — remove the prototype when no longer needed.

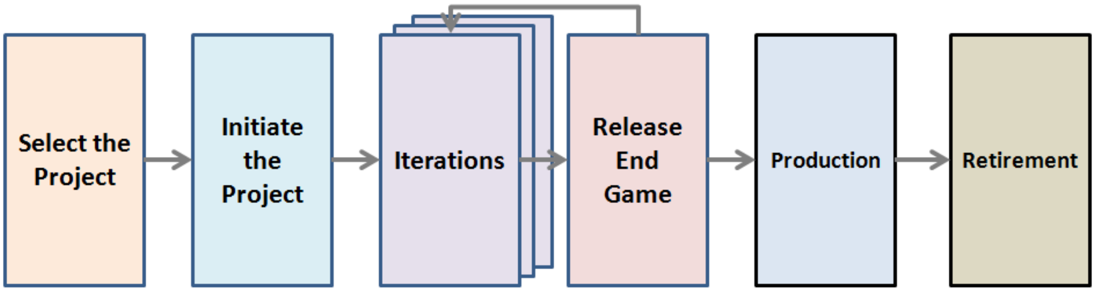


Figure 6.1: Flow of Agile Design

6.2 Beta I Prototype — Agile Design process

The Beta I prototype uses Agile Design as stated in section 6.1. As a result, all the stakeholders could rapidly change the needs of the prototype. The iterations do not need to be completed in full and they can be created if it is not fully completed or functional.

The company partners wanted to take what was made in the Alpha prototype but allow for a dynamic perspective of visualising the data. The vision they had was to display the information on a map instead of a traditional table / listing. This approach is used with some housing websites as an option to view housing on a map. However with jobs, at present no job website has a map functionality. We collaborated together and drew mock-up elements of the websites:

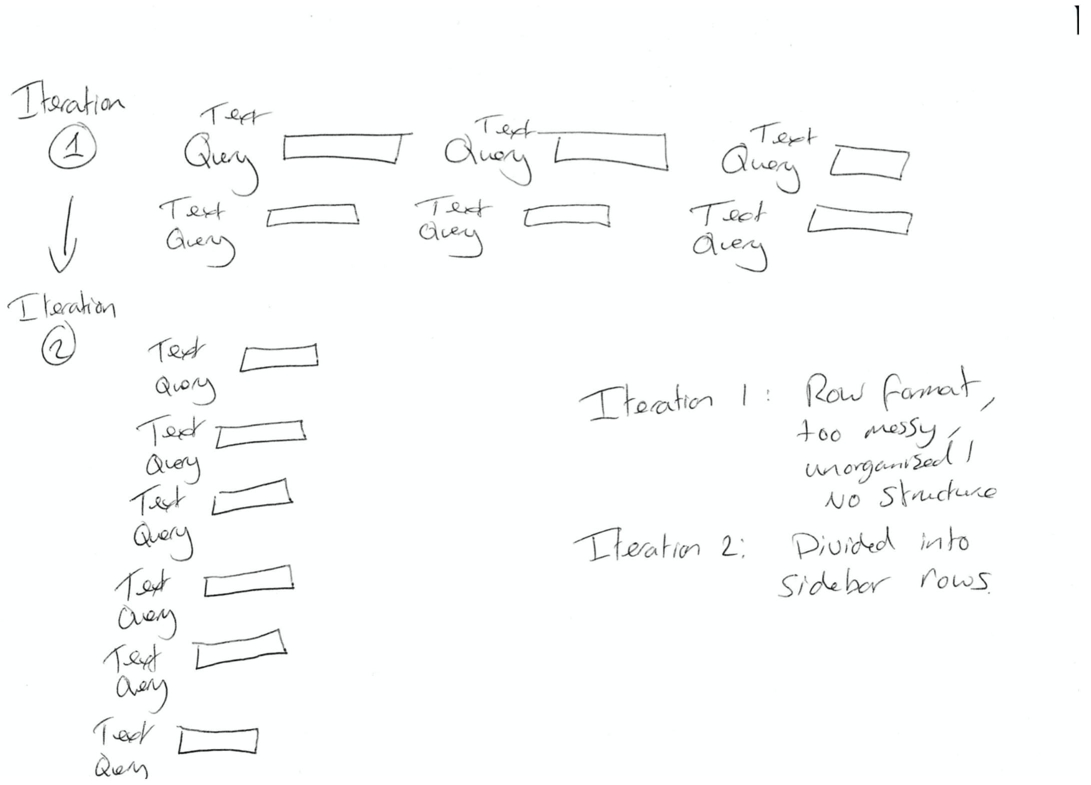


Figure 6.2: Design Beta I Prototype showing the iteration of Form options

Referring to figure 6.2, the first iteration of design looked into how the user would search. We sketched out a table with a search query, with some fields. The company partners felt that this would not be most suitable as tables should be avoided with HTML 5.

We then decided to put the query fields on the left side of the webpage in its own separated dividers, resulting in the code being more HTML5 friendly. It also looked better not having a table.

The company wanted to use the Beta I Prototype as an experiment to have both the map and search field on the same web page. This would mean that the user would have the ability to search for the criteria on the left hand side and then it would produce the results on the same page. They wanted to see if it was possible to use Google Earth, a platform that you can download and search for criteria and then zoom into the area. This resulted in the following design:

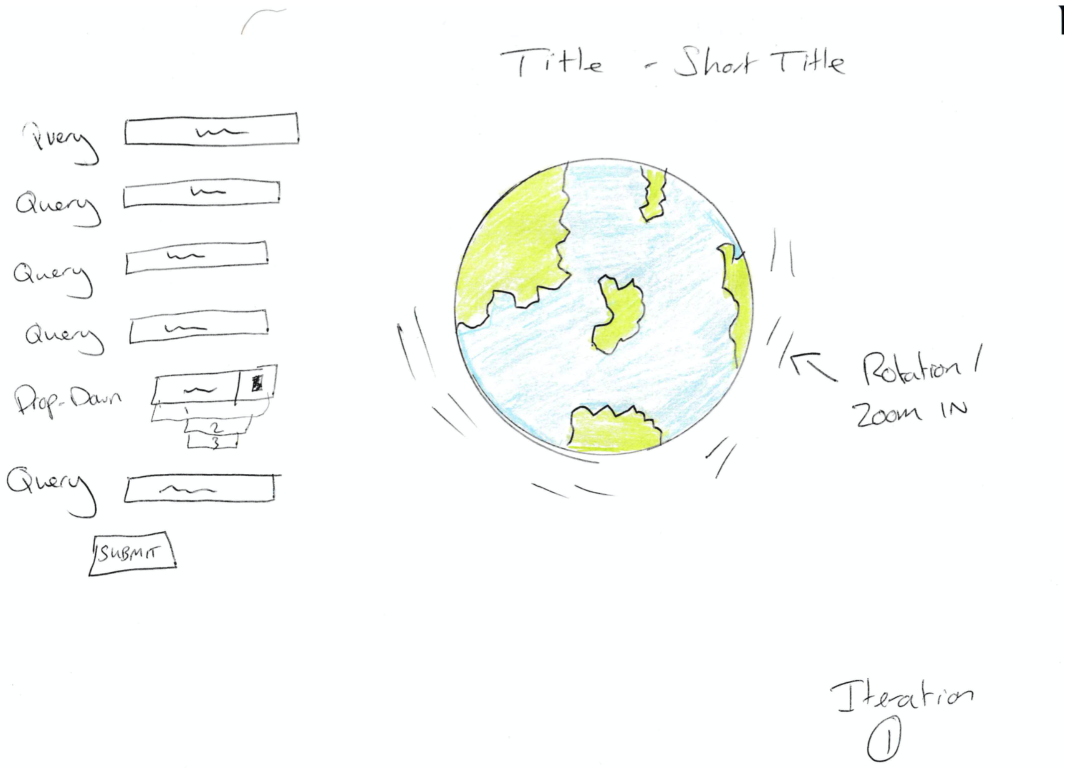


Figure 6.3: Beta I Prototype — Iteration 2 — 3D map

Figure 6.3 shows the webpage with a 3D map. This required an investigation to see if it was viable to use Google Earth through websites. It was discovered that at present, Google Earth is only available to download and use through through a computer. It did not allow for cross compatibility from Desktop to Smartphone either.

This led to another iteration. Instead of using a 3D globe, we thought of using a 2D map of the area on the right hand side of the search field. This can be seen in Figure 6.4.

Using a query and map solution has been used by many researchers in the past. Dynamic queries are used to cope with information overload. Having the ability to select, and adjust the information by means of an interface [133].

A geographical visualisation has been created for the Beta I prototype. This is a form of information visualisation in which principles from geographic information systems have been integrated into the output of the dataset [86].

Using a geographic visualisation places the emphasis onto using a map and provides a dynamic display that the users can manipulate with a dynamic query. This results in the map changing in response to changes of the data and the actions from the user [87].

Figure 6.5 is using multiple views for information visualisation. On the left hand side the query is displayed and upon submission of the submit button, the results load directly on the map on the right hand side. As the data-set is large, it was important to load the data in a way that the information is accessible upon clicking on detail-on-demand windows. This will avoid having potential problems with memory usage [153].

Ahlbery's [3][2] approach to information visualisation has used toggle boxes, drop-downs and detail-on-demand windows, the theory being to use these type of accessibility options to enhance user experience. Detail-on-demand windows allow the overviews of data that is important to the user to be provided only when they are requested dynamically. The Beta I prototype in this instance allows for the specific housing and job query to be highlighted upon searching the query as shown in Figure 6.4. Detail-on-demand windows are demonstrated in Figure 6.5 and selecting on the pins would generate specific housing or job information windows when they are requested.

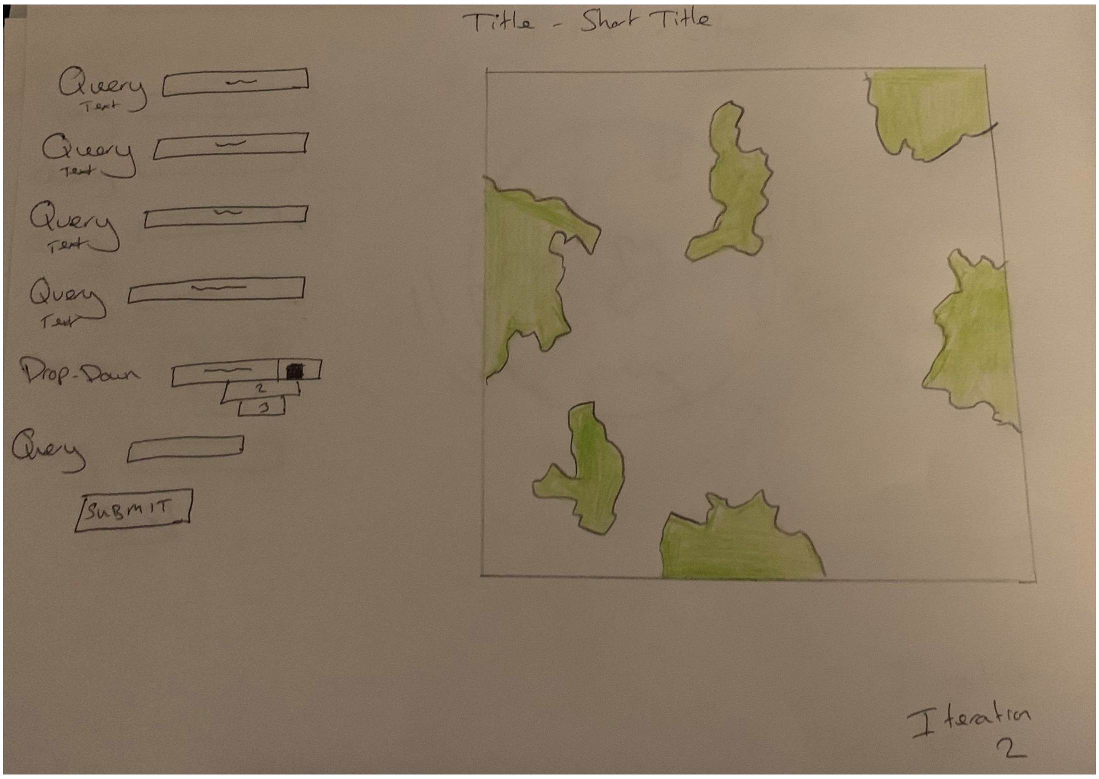


Figure 6.4: Beta I Prototype — Iteration 2 — 2D map

On the results page from jobs and housing websites, the web page has a wealth of information that you can view, although on a map, it isn't possible to view everything in a presentable manner. The information would need to be condensed and presentable to the user. As well, a decision on how you pinpoint the information on the map was needed.

Referring to Figure 6.5, a decision was made to use a red circle to pinpoint the information for this prototype. The reason why red was used as there was a motivation to use this for accessibility. If the user were to click the red circle, an information window will appear listing the title, short description, location and a link to the specific web listing. This allows for basic information to be provided to the user and the ability for them to view the listing further on the website in question.

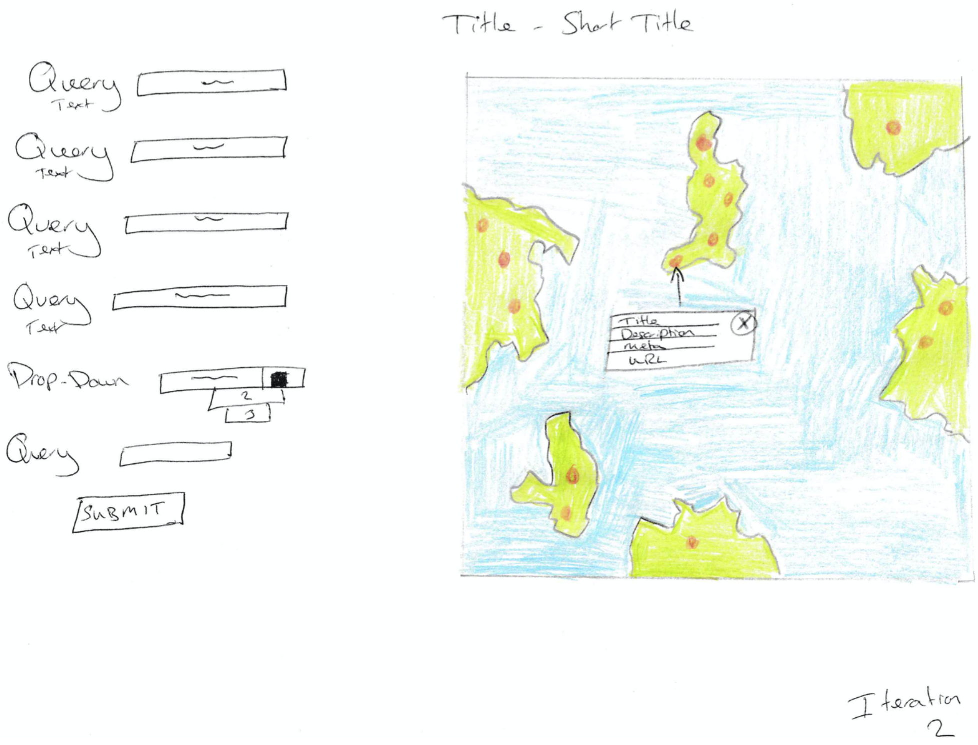
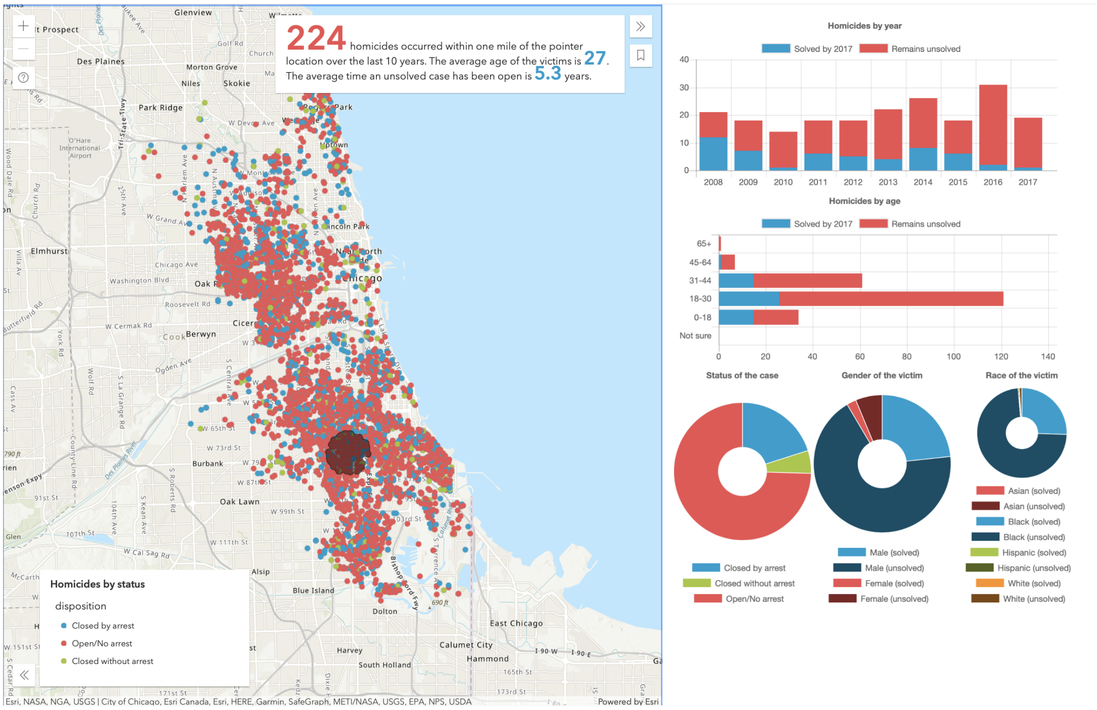


Figure 6.5: Beta I Prototype — Iteration 2 — 2D map, infowindow and red circles

6.2.1 Design Outcome

The vision set in this prototype was to have everything on one page, the ability to search and to view the results at the same time instead of having to have multiple pages to get to the results. At this stage, the source of the jobs and housing elements are on two separate websites although the design element is the same for both.

ArcGIS API for JavaScript [1] was used for this prototype as shown in Figure 6.7. The reasoning behind this was because the demo examples illustrate that it is possible to produce impressive results. An example of this can be shown in Figure 6.6 where the sample demonstrates how distance-related queries are used to process geographical-based statistics with the results produced in a series of charts. This app displays homicide data from 50 U.S. cities between 2007 and 2017, gathered by The Washington Post [2].



224 homicides occurred within one mile of the pointer location over the last 10 years. The average age of the victims is 27. The average time an unsolved case has been open is 5.3 years.

Homicides by status

disposition

● Closed by arrest

● Open/No arrest

● Closed without arrest

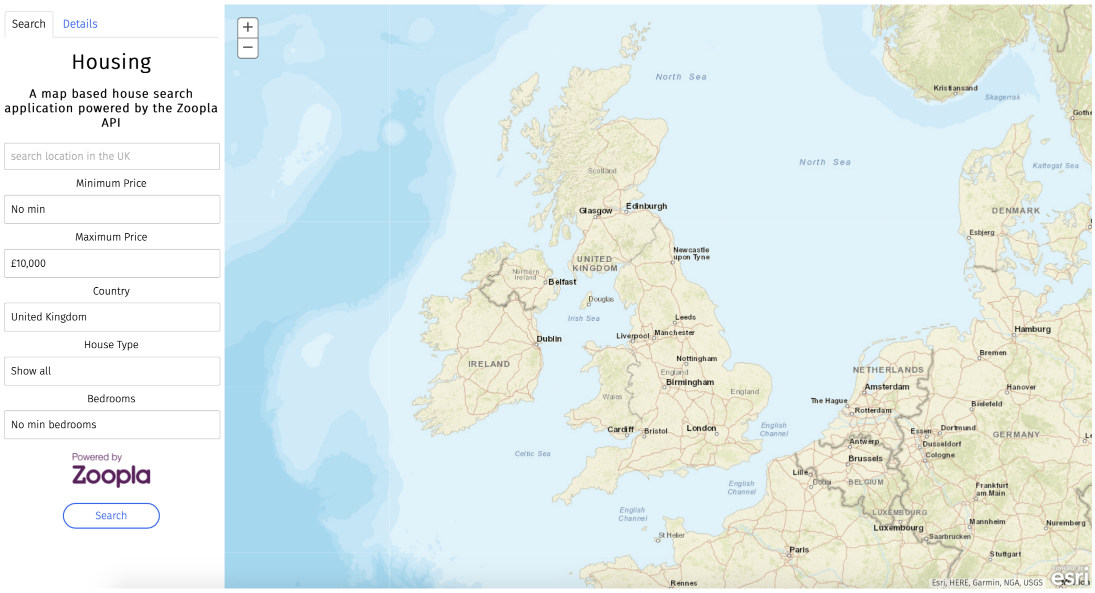
Esri, NASA, NGA, USGS | City of Chicago, Esri Canada, Esri, HERE, Garmin, SafeGraph, METI/NASA, USGS, EPA, NPS, USDA

Powered by Esri

Figure 6.6: Query statistics client—side by distance

[1] https://developers.arcgis.com/javascript/

[2] https://www.washingtonpost.com/graphics/2018/investigations/unsolved-homicide-database/



Housing

A map based house search application powered by the Zoopla API

search location in the UK

Minimum Price

No min

Maximum Price

£10,000

Country

United Kingdom

House Type

Show all

Bedrooms

No min bedrooms

Powered by Zoopla

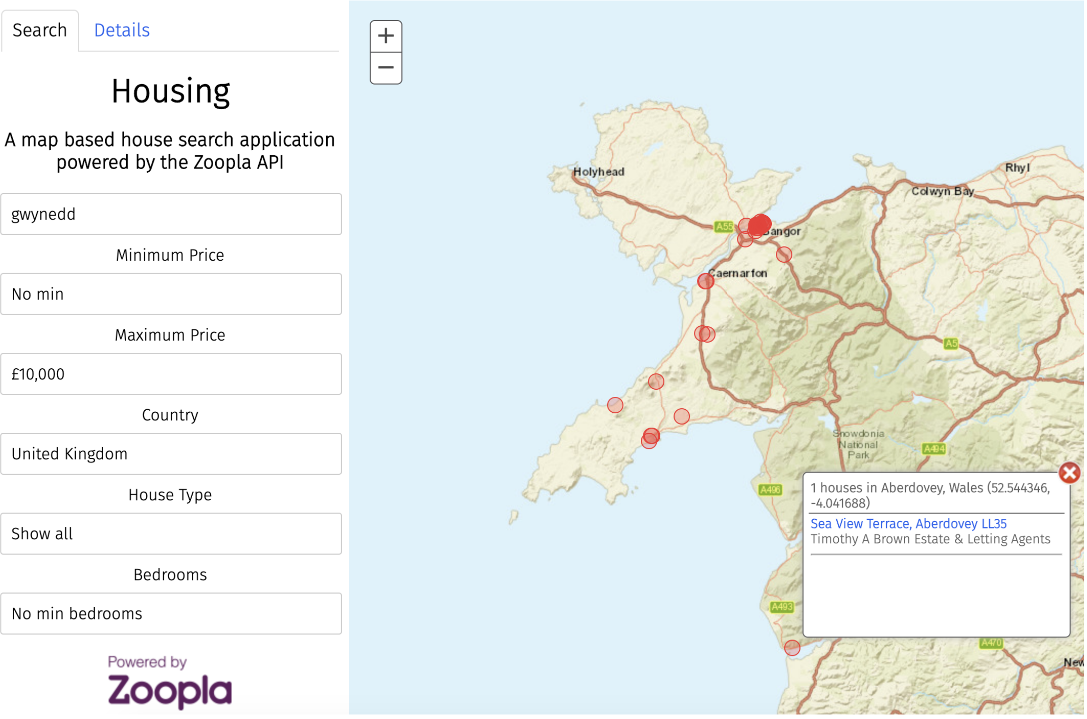
Search

Figure 6.7: Beta I Prototype — Design Overview

As shown in figure 6.7, the website for the Beta I prototype has the search area on the left hand side of the web page, including the ability to:

- Search for a location in the UK.
- Minimum price of the property.
- Maximum price of the property.
- Property type — housing or flat.
- Number of bedrooms.

In figure 6.8, once a search is performed, the results will then be pinned on the map with red circles and if clicked will show basic information of the listing and a hyperlink to the specific web page of the listing.



Housing

A map based house search application powered by the Zoopla API

gwynedd

Minimum Price: No min

Maximum Price: £10,000

Country: United Kingdom

House Type: Show all

Bedrooms: No min bedrooms

Powered by Zoopla

1 houses in Aberdovey, Wales (52.544346, -4.041688)

Sea View Terrace, Aberdovey LL35

Timothy A Brown Estate & Letting Agents

Figure 6.8: Beta I Prototype Design — Searching

6.2.2 Conclusion of the Beta I design process

Using agile design, several iterations of the Beta I design prototype were created and the design methodology used achieved the creation of a design to display the housing and job results on a map based interface. There has been the process of breaking down the elements of the website design into small increments.

Agile design was much preferred by the company partners. Mr. Edwin Smith said: “I found this much easier to work with, it allows collaboration and many iterations of the processes as required. It allowed for rapid prototyping and design development.” Agile Design allowed for more frequent discussions with the company partners and allowed for there to be multiple iterations and additions as the design was being made. The company partners were happy with this design although for the Beta II prototype, a further design was required due to the change in structure of the website, using the Merlin system instead of the APIs. Additionally the company partners preferred the icons to be further distinctive to make it clear to the user what a job and housing is from the results. This is discussed in the next section.

6.3 Implementation of Beta I Prototype

Some housing and job websites have APIs that are publicly available to use and, at the time of writing, were free of charge without restrictions. Two available APIs that were investigated were Zoopla for housing website data and Indeed for job website data. These APIs provide documentation which describe what elements are included in the API and how they can be adapted to a map overlay with latitude and longitude data [3][4].

The housing and jobs have been created on separate websites to make it easier to explore the effectiveness of the individual APIs. A benefit from creating this prototype is that the code used has been adaptable and reusable for both APIs. Beta I has a design objective from the previous section of following Agile Design.

6.3.1 ESRI Maps

On both the housing and jobs website element, ESRI maps have been used. The ArcGIS API for Javascript [5] provides developers the ability to build compelling web applications that allow an interactive user experience with 2D and 3D visualizations.

In this prototype, an ESRI map is initialised along with the key elements of what is required for displaying the results on the map. A function has been created as part of the ESRI required modules ArcGIS API. The elements are the creation of the map, the information window, the layering on the map, the symbol pins and then the event to add the points onto the map.

Figure 6.9 shows the creation of the ESRI Maps functionality within the Beta I prototype.

[3] https://developer.zoopla.co.uk

[4] https://uk.indeed.com/?r=us

[5] https://developers.arcgis.com/javascript/

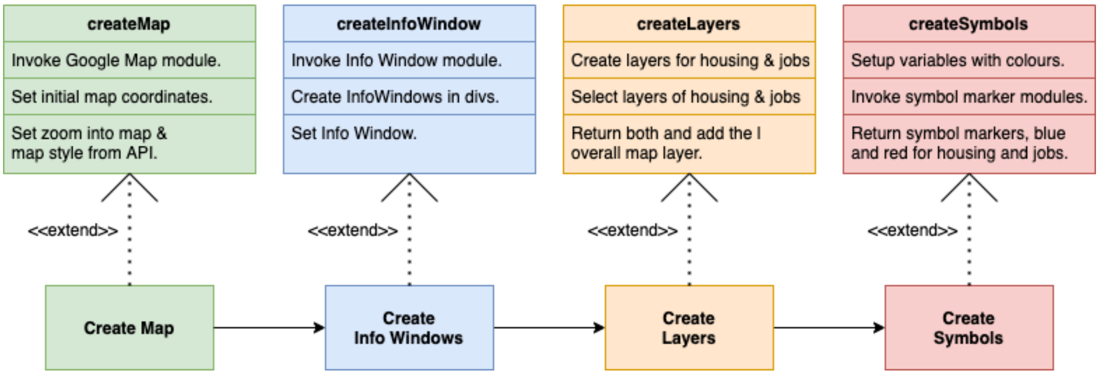

createMap
Invoke Google Map module.
Set initial map coordinates.
Set zoom into map & map style from API.
createInfoWindow
Invoke Info Window module.
Create InfoWindows in divs.
Set Info Window.
createLayers
Create layers for housing & jobs
Select layers of housing & jobs
Return both and add the l overall map layer.
createSymbols
Setup variables with colours.
Invoke symbol marker modules.
Return symbol markers, blue and red for housing and jobs.
<<extend>>
<<extend>>
<<extend>>
<<extend>>
Create Map
Create Info Windows
Create Layers
Create Symbols

Figure 6.9: ESRI Maps Initialization and Options within Prototype Beta I

Each of the element process from Figure 6.9 are as follows:

1. Create Map: a function is created to invoke the Google Map module and set configuration settings on the map and the layout style.

2. Create Info Windows: a function is created to invoke the Info Window module and adding the capability of adding infowindows to the HTML structure.

3. Create Layers: a function is created to create the different layers for both housing and jobs, and to be able to return the results of the list arrays and display them on the map.

4. Create Symbols: a function that sets up the symbols for both housing and jobs, invoking the marker modules and returning these markers on both the housing and jobs listings.

Figure 6.10 shows the process of the ESRI Maps functionality following from the creation steps within the Beta I prototype.

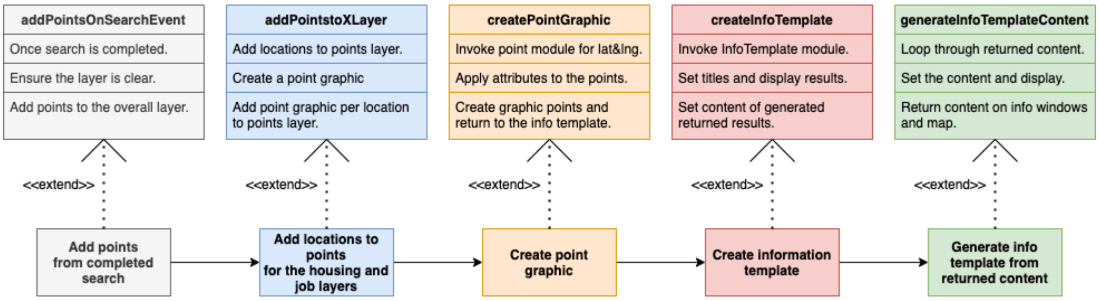

addPointsOnSearchEvent
Once search is completed.
Ensure the layer is clear.
Add points to the overall layer.
addPointstoXLayer
Add locations to points layer.
Create a point graphic
Add point graphic per location to points layer.
createPointGraphic
Invoke point module for lat&lng.
Apply attributes to the points.
Create graphic points and return to the info template.
createInfoTemplate
Invoke InfoTemplate module.
Set titles and display results.
Set content of generated returned results.
generateInfoTemplateContent
Loop through returned content.
Set the content and display.
Return content on info windows and map.
<<extend>>
<<extend>>
<<extend>>
<<extend>>
<<extend>>
Add points from completed search
Add locations to points for the housing and job layers
Create point graphic
Create information template
Generate info template from returned content

Figure 6.10: ESRI Maps Process Flow within the Prototype Beta I

Each of the functionality element processes from Figure 6.10 are as follows:

1. Add points from completed search: a function is created to state when the search is completed to then ensure the layer is clear, the function then adds the points to the layer.

2. Add locations to the points for the housing and job layers: a function is set to add locations to the points, the function then creates the point graphics for the location coordinates and displays the points on the overall layer.

3. Create point graphic: a function created that invokes the point module for the latitude and longitude, this then applies the attributes to the points and returns the created graphic points.

4. Create information template: this created the content that will be displayed from the returned array lists, such as the titles and then it returns the generated returned results.

5. Generate info template from returned content: the created function loops through the returned content, sets the content and then applies the information to infowindows. Additionally this function returns the content onto the map.

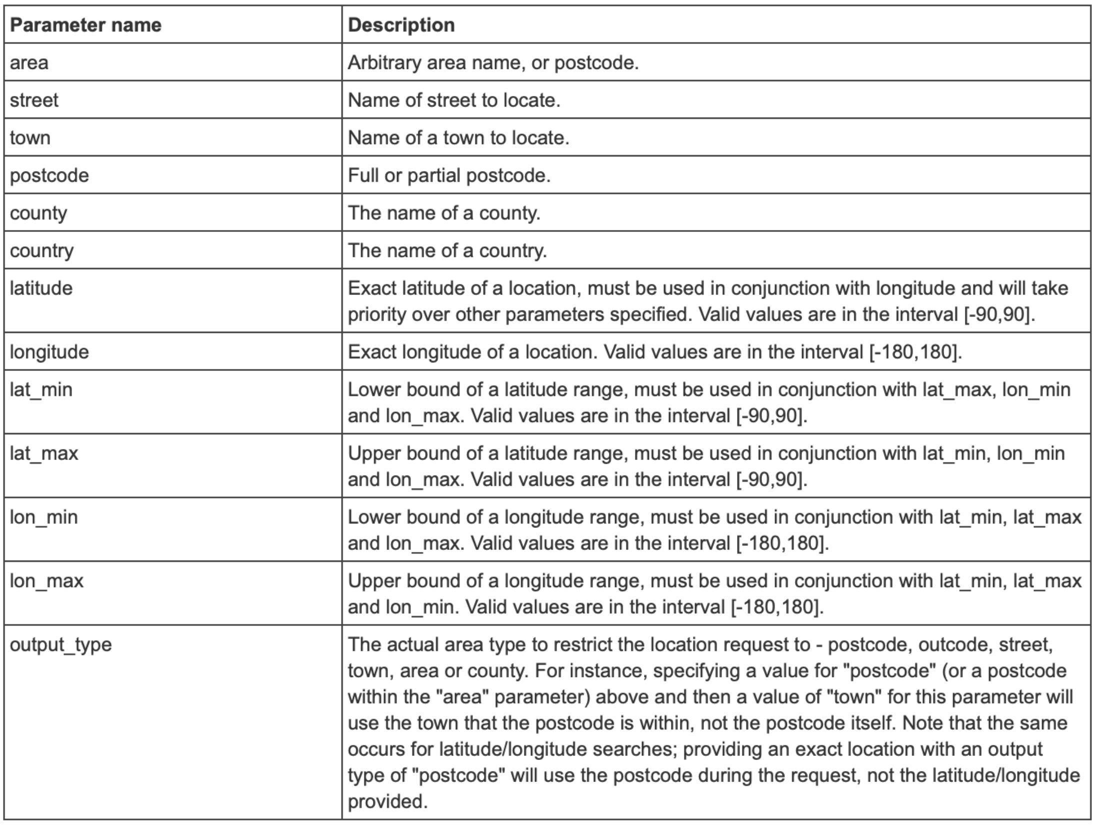
6.3.2 Zoopla API

Zoopla launched an open API to allow developers to create applications using local data of over 27 million homes. There are over 1 million sale and rental listings available in this data, comprising 15 years of sale price data [6].

Zoopla's API requires the user to sign up and create an API key to use the available data. The methods for accessing the data are via standard API methods and protocols, all of which use the GET method of HTTP protocols.

Zoopla's API provides standard input parameters, such as keywords, although several different methods can be used. If several parameters are specified, a higher chance of better results would be produced.

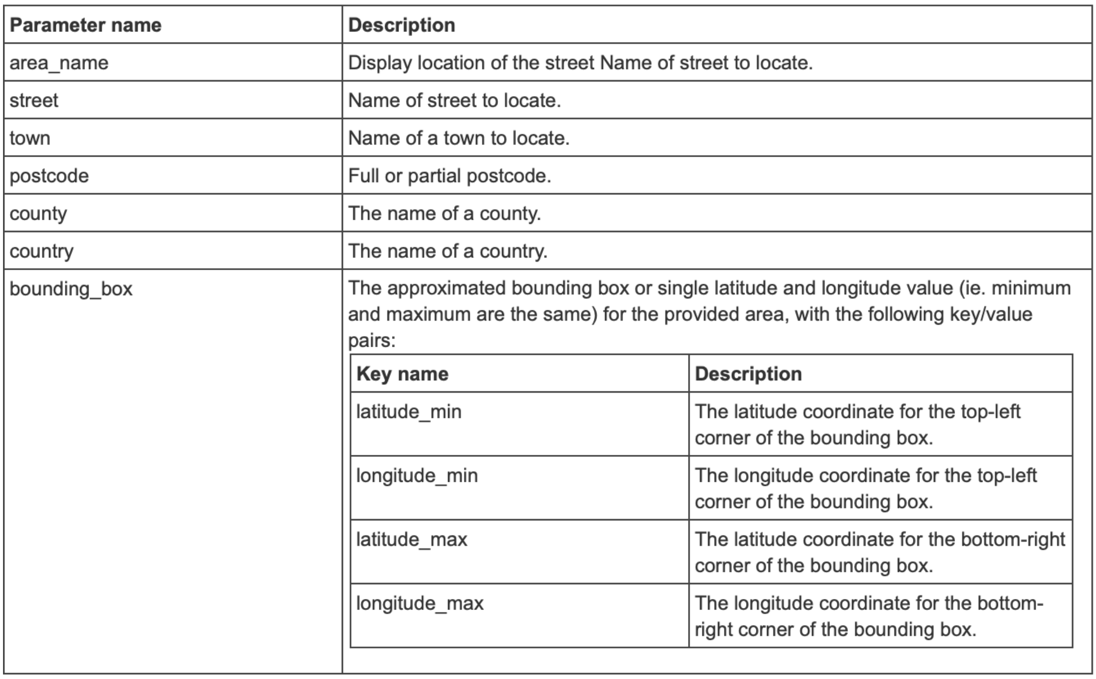
[6] https://developer.zoopla.co.uk



Parameter name	Description
area	Arbitrary area name, or postcode.
street	Name of street to locate.
town	Name of a town to locate.
postcode	Full or partial postcode.
county	The name of a county.
country	The name of a country.
latitude	Exact latitude of a location, must be used in conjunction with longitude and will take priority over other parameters specified. Valid values are in the interval [-90,90].
longitude	Exact longitude of a location. Valid values are in the interval [-180,180].
lat_min	Lower bound of a latitude range, must be used in conjunction with lat_max, lon_min and lon_max. Valid values are in the interval [-90,90].
lat_max	Upper bound of a latitude range, must be used in conjunction with lat_min, lon_min and lon_max. Valid values are in the interval [-90,90].
lon_min	Lower bound of a longitude range, must be used in conjunction with lat_min, lat_max and lon_max. Valid values are in the interval [-180,180].
lon_max	Upper bound of a longitude range, must be used in conjunction with lat_min, lat_max and lon_min. Valid values are in the interval [-180,180].
output_type	The actual area type to restrict the location request to - postcode, outcode, street, town, area or county. For instance, specifying a value for "postcode" (or a postcode within the "area" parameter) above and then a value of "town" for this parameter will use the town that the postcode is within, not the postcode itself. Note that the same occurs for latitude/longitude searches; providing an exact location with an output type of "postcode" will use the postcode during the request, not the latitude/longitude provided.

Figure 6.11: Zoopla API Input Parameters

Figure 6.11 shows the available input parameters that the user can develop to request the API to retrieve from the search form. The Beta I primarily uses the first eight options from the figure.



<table>
<tr><th>Parameter name</th><th>Description</th></tr>
<tr><td>area_name</td><td>Display location of the street Name of street to locate.</td></tr>
<tr><td>street</td><td>Name of street to locate.</td></tr>
<tr><td>town</td><td>Name of a town to locate.</td></tr>
<tr><td>postcode</td><td>Full or partial postcode.</td></tr>
<tr><td>county</td><td>The name of a county.</td></tr>
<tr><td>country</td><td>The name of a country.</td></tr>
<tr><td>bounding_box</td><td>The approximated bounding box or single latitude and longitude value (ie. minimum and maximum are the same) for the provided area, with the following key/value pairs:
<table>
<tr><th>Key name</th><th>Description</th></tr>
<tr><td>latitude_min</td><td>The latitude coordinate for the top-left corner of the bounding box.</td></tr>
<tr><td>longitude_min</td><td>The longitude coordinate for the top-left corner of the bounding box.</td></tr>
<tr><td>latitude_max</td><td>The latitude coordinate for the bottom-right corner of the bounding box.</td></tr>
<tr><td>longitude_max</td><td>The longitude coordinate for the bottom-right corner of the bounding box.</td></tr>
</table>
</td></tr>
</table>

Figure 6.12: Zoopla API Output Parameters

Zoopla's API would then provide a standard set of parameters for output that is generated containing information about the matched data according to the input parameters provided, as shown in figure 6.12.

6.3.3 Zoopla API Integration

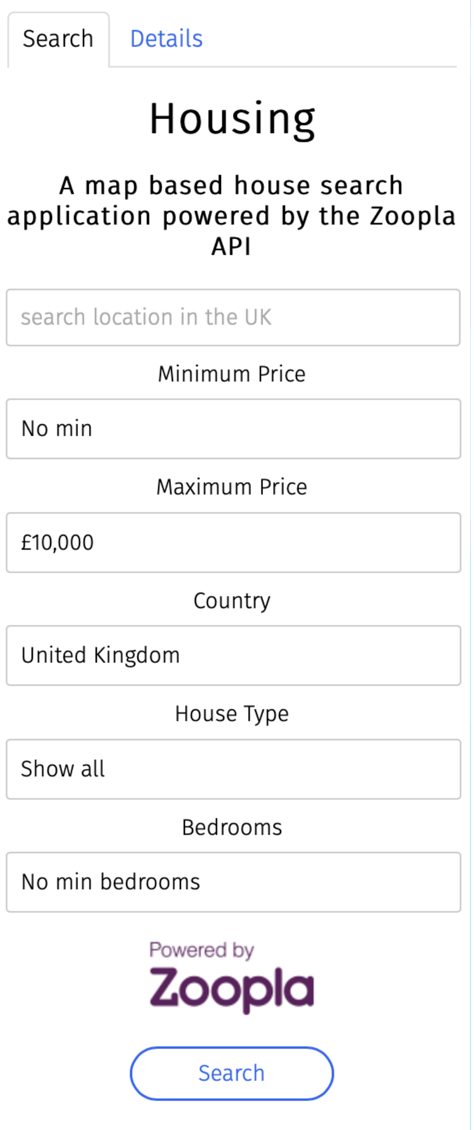
The Beta I prototype has been developed using the parameters shown in Figure 6.12. All the parameters except *bounding_box* have been used to request the API to search property listings. For example, using the postcode parameter, the user might enter 'LL57'. The response would be the following, as shown in table 6.1.

Begin tag	Text	End tag
<listing>		</listing>
<agent_address>	282 High Street, Bangor	</agent_address>
<agent_logo>	https://st.zoocdn.com/zoopla_static_agent_logo_(640059).png	</agent_logo>
<agent_name>	Beresford Adams - Bangor Sales	</agent_name>
<agent_phone>	01248 308913	</agent_phone>
<category>	Residential	</category>
<country>	Wales	</country>
<country_code>	gb	</country_code>
<county>	Gwynedd	</county>
<description>	Rich in history and bursting at the seams with character, the former Hillgrove School is a rare and exciting opportunity for any	</description>
<details_url>	https://www.zoopla.co.uk/for-sale/details/54144846?utm_source=v1:5bnJADncHQDN6geyKa1ztDj-OUc5xa-Y&amp;utm_medium=api	</details_url>
<displayable_address>	Ffriddoedd Road, Bangor, Gwynedd LL57	/displayable_address>
<first_published_date>	03/02/2020 21:12	</first_published_date>
<floor_plan>	https://lc.zoocdn.com/3f1e89c402fbec0d9ed27a3b825131b2cd329bdb.jpg	</floor_plan>
<image_150_113_url>	https://lid.zoocdn.com/150/113/bb5a119d2ebf67e7a0b4a747037871ae11aef718.jpg	</image_150_113_url>
<image_354_255_url>	https://lid.zoocdn.com/354/255/bb5a119d2ebf67e7a0b4a747037871ae11aef718.jpg	</image_354_255_url>
<image_50_38_url>	https://lid.zoocdn.com/50/38/bb5a119d2ebf67e7a0b4a747037871ae11aef718.jpg	</image_50_38_url>
<image_645_430_url>	https://lid.zoocdn.com/645/430/bb5a119d2ebf67e7a0b4a747037871ae11aef718.jpg	</image_645_430_url>
<image_80_60_url>	https://lid.zoocdn.com/80/60/bb5a119d2ebf67e7a0b4a747037871ae11aef718.jpg	</image_80_60_url>
<image_caption>	Front	</image_caption>
<image_url>	https://lid.zoocdn.com/354/255/bb5a119d2ebf67e7a0b4a747037871ae11aef718.jpg	</image_url>
<last_published_date>	21/05/2021 11:45	</last_published_date>
<latitude>	53.22418	</latitude>
<listing_id>	54144846	</listing_id>
<listing_status>	sale	</listing_status>
<location_is_approximate	0	</location_is_approximate>
<longitude>	-4.141412	</longitude>
<num_bathrooms>	1	</num_bathrooms>
<num_bedrooms>	6	</num_bedrooms>
<num_floors>	1	</num_floors>
<num_recepts>	1	</num_recepts>
<outcode>	LL57	</outcode>
<post_town>	Bangor	</post_town>
<price>	1200000	</price>
<date>	03/02/2020 19:40	</date>
<percent>	0%	</percent>
<price>	1500000	</price>
<date>	12/09/2020 18:57	</date>
<direction>	down	</direction>
<percent>	-20%	</percent>
<price>	1200000	</price>
<direction>	down	</direction>
<last_updated_date>	12/09/2020 18:57	</last_updated_date>
<percent>	-20%	</percent>
<property_type>	Detached house	</property_type>
<short_description>	Rich in history and bursting at the seams with character, the former Hillgrove School is a rare(...)	</short_description>
<status>	for_sale	</status>
<street_name>	Bangor Gwynedd	</street_name>
<thumbnail_url>	https://lid.zoocdn.com/80/60/bb5a119d2ebf67e7a0b4a747037871ae11aef718.jpg	</thumbnail_url>

Table 6.1: Zoopla API Listing Output

As shown in table 6.1, the API has produced an XML tagged structure output with the response from the query entered, which in this instance was ‘LL57’. The API produces XML or JSON formatted responses so the developer can pick the selected elements that are required for the application’s purpose.

With respect to the prototype created, the user has to fill out a search form, and when it is submitted, the query is sent to the API.



Housing

A map based house search application powered by the Zoopla API

search location in the UK

Minimum Price: No min

Maximum Price: £10,000

Country: United Kingdom

House Type: Show all

Bedrooms: No min bedrooms

Figure 6.13: Beta I Sidebar Homepage for housing

In figure 6.13, on the sidebar of the homepage, the user has different options that they can use for the query. Suppose the user were to type ‘Gwynedd’ for the location. It would then take the parameter ‘area’ from figure 6.11 and perform a call to the API.

The reason for using the area parameter is because it is the required option to select from the API documentation.

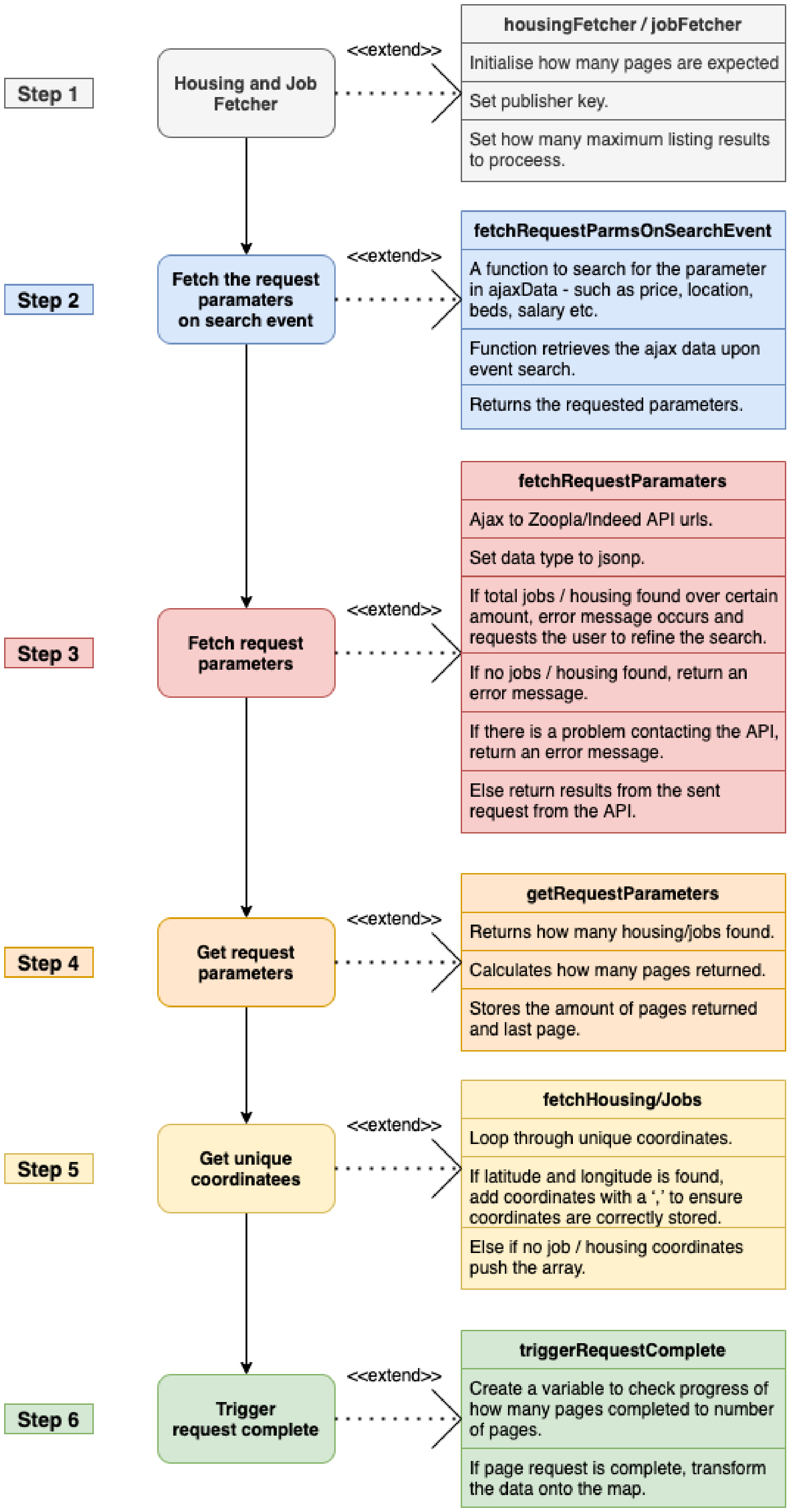

Step 1
Housing and Job Fetcher
<<extend>>
housingFetcher / jobFetcher
Initialise how many pages are expected
Set publisher key.
Set how many maximum listing results to proceess.
Step 2
Fetch the request paramaters on search event
<<extend>>
fetchRequestParmsOnSearchEvent
A function to search for the parameter in ajaxData - such as price, location, beds, salary etc.
Function retrieves the ajax data upon event search.
Returns the requested parameters.
Step 3
Fetch request parameters
<<extend>>
fetchRequestParamaters
Ajax to Zoopla/Indeed API urls.
Set data type to jsonp.
If total jobs / housing found over certain amount, error message occurs and requests the user to refine the search.
If no jobs / housing found, return an error message.
If there is a problem contacting the API, return an error message.
Else return results from the sent request from the API.
Step 4
Get request parameters
<<extend>>
getRequestParameters
Returns how many housing/jobs found.
Calculates how many pages returned.
Stores the amount of pages returned and last page.
Step 5
Get unique coordinatees
<<extend>>
fetchHousing/Jobs
Loop through unique coordinates.
If latitude and longitude is found, add coordinates with a ',' to ensure coordinates are correctly stored.
Else if no job / housing coordinates push the array.
Step 6
Trigger request complete
<<extend>>
triggerRequestComplete
Create a variable to check progress of how many pages completed to number of pages.
If page request is complete, transform the data onto the map.

Figure 6.14: The component process of how the searching and retrieving is performed in the Beta I Prototype

An example using the above with the code created for this prototype is shown in Figure 6.14. A function has been created to identify the API version, how big the page sizes are and the API key. The way this prototype has been created uses standard implementation techniques although the way of how the data is processed and displayed is unique to this project.

Upon the user pressing the 'Search' button from Figure 6.13, this then takes what the user has entered and sets it into 'ajaxData'; in this instance 'Gwynedd' would be the location parameter as shown in step 2 of Figure 6.14.

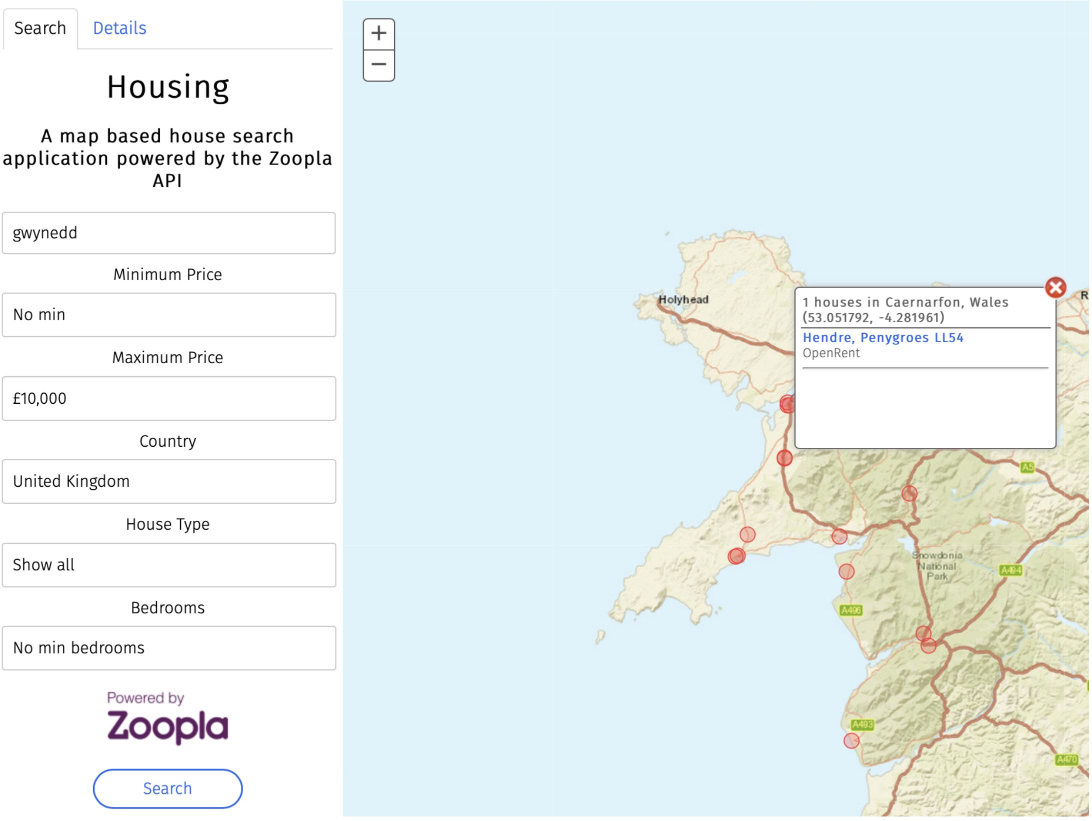
The parameters are then sent to Zoopla's API using the JSON format. Step 3 of Figure 6.14 process involves the results being returned. If more than 3000 house listings are found, an error message will appear requesting that the search be refined and made more specific. If there are no housings from the listing, an error message indicating there is no housing available; otherwise, it will fetch the results.

Step 4 of the process from Figure 6.14, checks how many pages are returned and then stored.

For step 5 of the process, a function has been created to ensure the coordinates are split and stored correctly.

The final step, step 6, involves fetching the pages with an AJAX call. Once all the pages have been finished, the results would be stored and produced on the map.

The results are then presented back to the user as shown in Figure 6.15. Pins are loaded onto the map and the user can click on the circles and the info window shows the information of the listing for that coordinate. In this figure, the user has queried 'Gwynedd' with a maximum price of '£10,000' and the Alpha prototype has returned all listings with the relevant inputs.



Housing

A map based house search application powered by the Zoopla API

gwynedd

Minimum Price

No min

Maximum Price

£10,000

Country

United Kingdom

House Type

Show all

Bedrooms

No min bedrooms

1 houses in Caernarfon, Wales (53.051792, -4.281961)

Hendre, Penygroes LL54

OpenRent

Figure 6.15: Beta I Example Result from sample query for housing

6.3.4 Indeed API

Indeed’s API allows developers to create applications that search Indeed’s job listings on their own platform. Indeed’s API requires the user to sign up and gain a publisher key to use the data available.

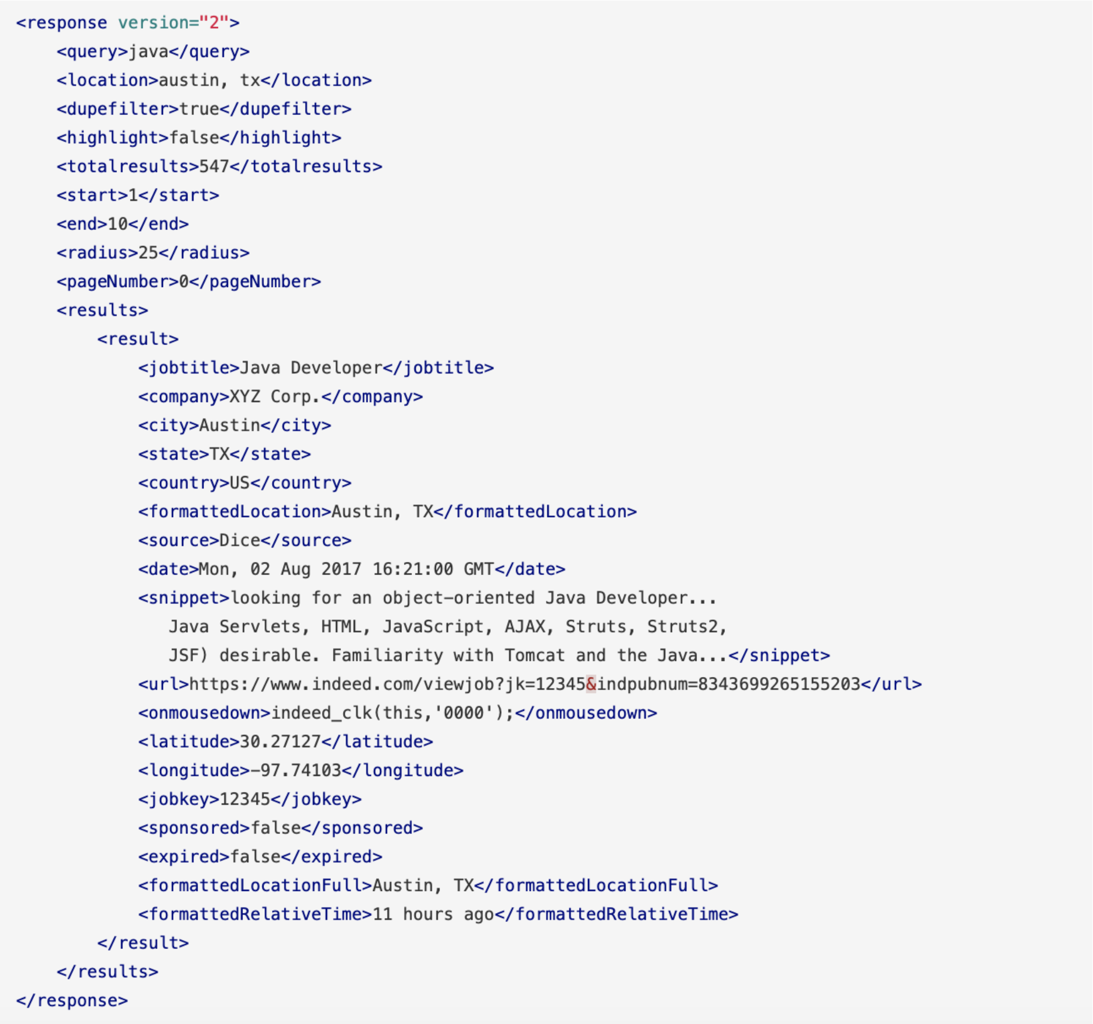
The methods for accessing the data involve standard API methods and protocols, using the GET method of HTTP protocols. Indeed’s API provides a range of request parameters for the developer to use as shown in Table 6.2 in order to use Indeed’s listings.

Parameter	Description
q	This stands for Query. The URL displays the expected formatted for the parameter, an example: q=developer or q=java+developer
l	This stands for Location. A post code, city, state or region combination can be used. An example: l=Conwy, UK
radius	This is the distance from the search location. The unit for this parameter is local to the country searching.
jt	This is the job type. The user can specify the following: fulltime, parttime, contract, internship or temporary.
salary	This is the amount of salary per year the user is requesting.
limit	This is the maximum number of results to return per query.
co	This is the country the user is searching for.
latlong	This is the latitude and longitude information.

Table 6.2: Indeed API Parameters

6.3.5 Indeed API Integration

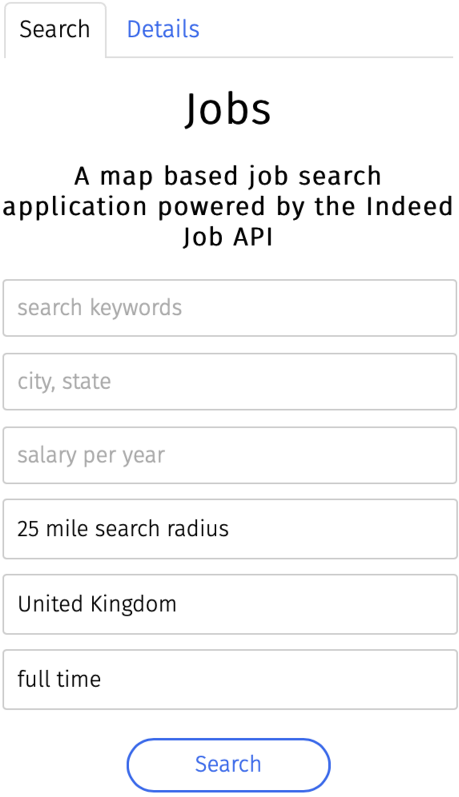
The Beta I prototype has been developed that uses all the parameters in Table 6.2 to request the API to search job listings. For example, the parameters required to search a request for Java developers (q) in Austin, TX (l) using the Indeed API parameters as shown in Figure 6.2.



```xml
<response version="2">
    <query>java</query>
    <location>austin, tx</location>
    <dupefilter>true</dupefilter>
    <highlight>false</highlight>
    <totalresults>547</totalresults>
    <start>1</start>
    <end>10</end>
    <radius>25</radius>
    <pageNumber>0</pageNumber>
    <results>
        <result>
            <jobtitle>Java Developer</jobtitle>
            <company>XYZ Corp.</company>
            <city>Austin</city>
            <state>TX</state>
            <country>US</country>
            <formattedLocation>Austin, TX</formattedLocation>
            <source>Dice</source>
            <date>Mon, 02 Aug 2017 16:21:00 GMT</date>
            <snippet>looking for an object-oriented Java Developer...
               Java Servlets, HTML, JavaScript, AJAX, Struts, Struts2,
               JSF) desirable. Familiarity with Tomcat and the Java...</snippet>
            <url>https://www.indeed.com/viewjob?jk=12345&indpubnum=8343699265155203</url>
            <onmousedown>indeed_clk(this,'0000');</onmousedown>
            <latitude>30.27127</latitude>
            <longitude>-97.74103</longitude>
            <jobkey>12345</jobkey>
            <sponsored>false</sponsored>
            <expired>false</expired>
            <formattedLocationFull>Austin, TX</formattedLocationFull>
            <formattedRelativeTime>11 hours ago</formattedRelativeTime>
        </result>
    </results>
</response>
```

Figure 6.16: Indeed API Listing Output

Figure 6.16 lists the output returned from the query input. The API produces XML or JSON formatted responses, so the developer can easily select which elements are required for the application.


Search

Details

Jobs

A map based job search application powered by the Indeed Job API

search keywords

city, state

salary per year

25 mile search radius

United Kingdom

full time

Search

Figure 6.17: Beta I Sidebar Homepage for jobs

In figure 6.17, the sidebar of the jobs homepage is shown. This allows the user to enter different options for the query.

An example of using the above query with the code created for this prototype is similar to the previous housing section.

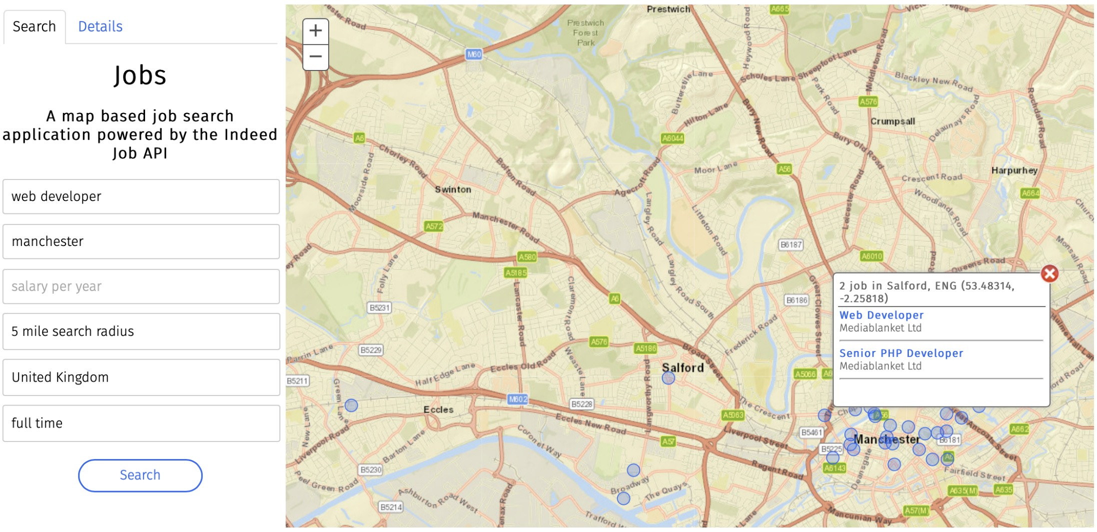
A function is created to identify the version of the API and initialise the jobFetcher. This is shown in part 1 of Figure 6.14.

Once the user has submitted the form, part 2 of Figure 6.14 would execute, where the searched input on the form would be fetched.

The parameters are then sent to Indeed's API in the format of JSON. Part 3 of Figure 6.14 processes the results returned. If there are more than 20,000 results returned, it will require the query to be more specific. If no jobs are found, an error message will inform the user of this; otherwise, it would fetch the request's jobs.

Part 4 of Figure 6.14 involves checking how many results have been returned and stores them. Part 5 of Figure 6.14 gets the coordinates and splits them in order to place them onto a map. Part 6 of Figure 6.14 finalises fetching the pages.

The results are then presented back to the user, as shown in figure 6.18.



Jobs

A map based job search application powered by the Indeed Job API

web developer

manchester

salary per year

5 mile search radius

United Kingdom

full time

2 job in Salford, ENG (53.48314, -2.25818)

Web Developer
Mediablanket Ltd

Senior PHP Developer
Mediablanket Ltd

Figure 6.18: Beta I Prototype Example Result from query for jobs

6.4 Developing methods for scraping web content involving accommodation and jobs

Beta I explored different methods for scraping web content involving accommodation and job websites relying on vendor APIs to produce the information. A problem occurred during the development of the prototype with the vendors tightening up the usage of their APIs. It called for the project to explore alternative approaches that remove the reliance on APIs since external factors could come into play, such as charges or the removal of the API itself, and as a result would be out of our control.

The creation of an alternative be-spoke structured system to extract elements from the housing and job websites using regular expressions was then explored.

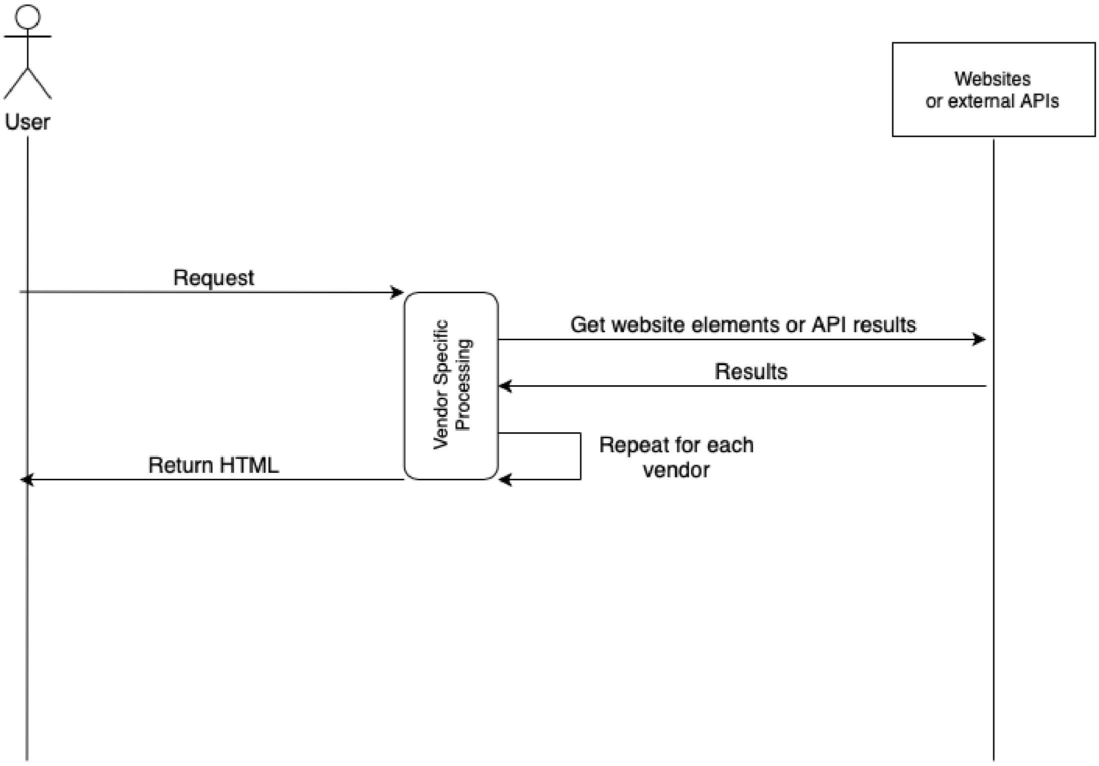


Figure 6.19: An Activity Diagram showing the process of how the created system gathers elements from websites or external APIs with created scripts.

Figure 6.19 illustrates the process that was developed for the be-spoke system. The process is explained as follows:

1. The user sends the request.

2. PHP Scripts have been created with different methods for scraping different elements from multiple websites come into operation.

3. The scripts send requests to the tagged HTML elements or external APIs.

4. The results are returned.

5. The process returns the HTML to the user and repeats this cycle for both housing and job websites.

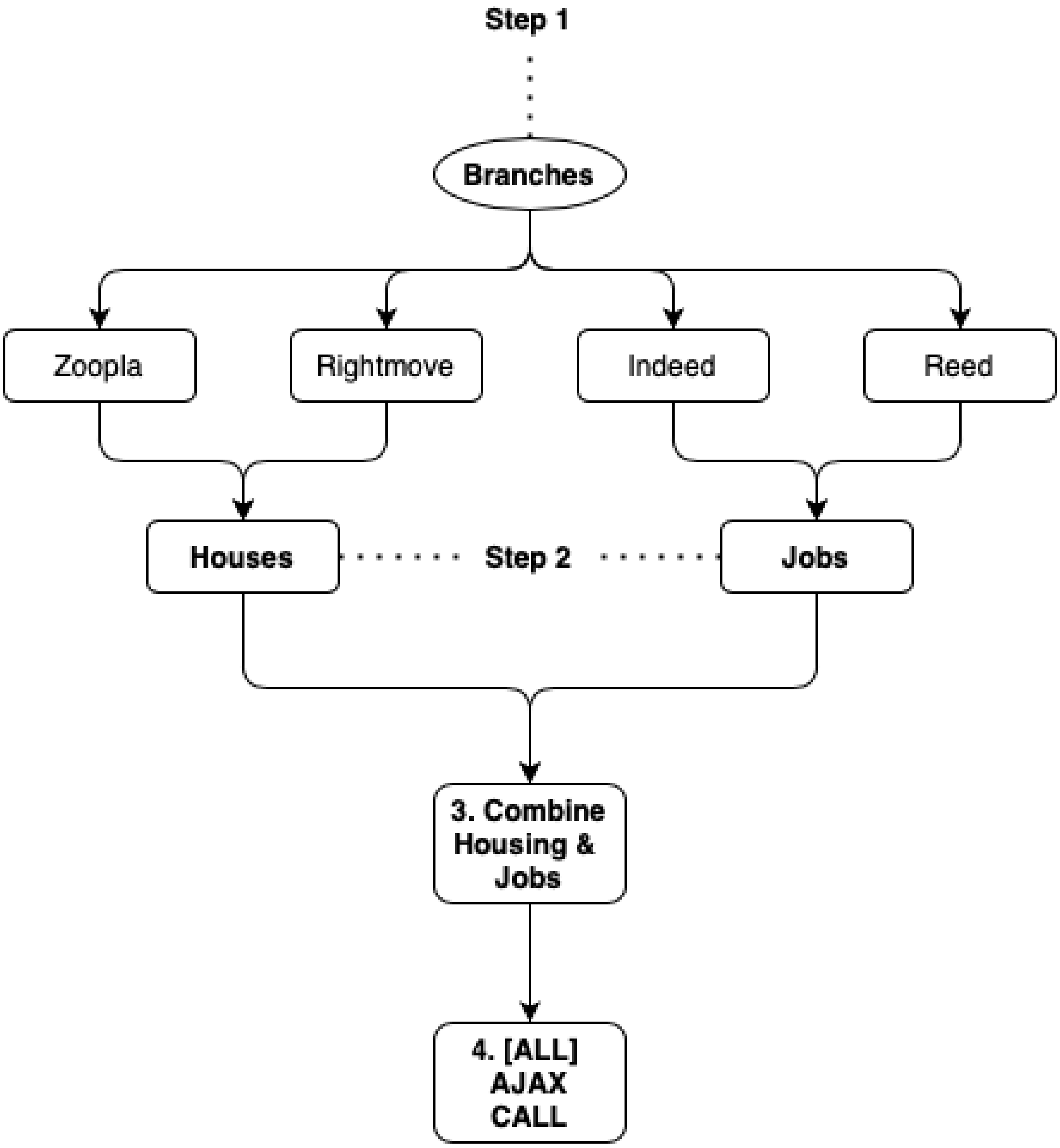


Figure 6.20: This structure explains the steps taken of the created system to gather jobs and housing.

Extending from figure 6.19, a system was created as shown in figure 6.20 to demonstrate the process of how the be-spoke process would work. The process for this is the following:

1. Step 1 shows the the branches of the system, the individual websites for housing and jobs. These files consist of regular expressions and parsing XML data.

2. Step 2 brings all the websites for jobs together and all the websites for housing together. The input is a postcode, and the output is an array of objects for jobs and an array of objects for housing.

3. Step 3 combines both housing and jobs arrays together. This combined array is used to produce the output to the user.

4. Step 4 is an AJAX call bringing the previous steps together, then an output is produced of housing and jobs.

The prototype software created for this process is as follows:

Step one from Figure 6.20 lists the individual websites. The created system structure has its own file for each housing and job website, and includes regular expressions within them to allocate and retrieve the elements from the HTML source.

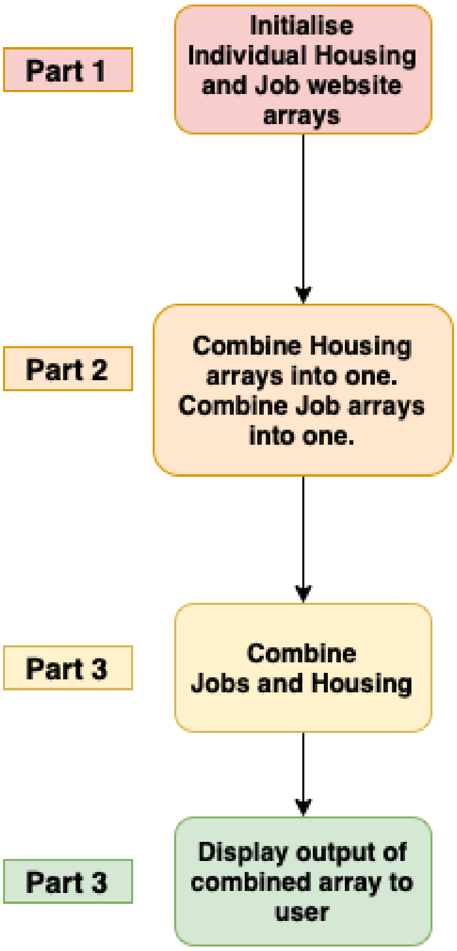

Part 1
Initialise Individual Housing and Job website arrays
Part 2
Combine Housing arrays into one. Combine Job arrays into one.
Part 3
Combine Jobs and Housing
Part 3
Display output of combined array to user

Figure 6.21: The process of scraping web content involving jobs and housing

Figure 6.21 shows the flow of the methods that were developed for scraping web content. Step 1 of the process does the following:

- A function is created with the a postcode parameter, that is passed as a parameter to the function.

- Initialised the URL, defined the postcode parameter in the URL structure. Executed the URL and created an array to store the returned data.

- Regular expressions are created to match the patterns to the source code of the individual job and housing websites. The returned matches would then be stored into the array.

- A loop is set up to cycle through the returned elements from the regular expressions. The loop then assigns the array into distinguishable parts e.g., amount, title, description, salary etc.

- The loop is then returned and pushed to part 2.

In step 2 of Figure 6.21, a jobs file and a housing file has been created and the function for it gathers the individual arrays of the housing websites into one array list and the jobs websites into one array list.

In step 3 of Figure 6.21, this stage combines all housing and jobs from the two dependency files in the previous step.

In step 4 of Figure 6.21, this would display the outputted loop to the developer, the developer could then incorporate into a Google Map.

The problem with this created system is that the developer must specify fixed regular expressions patterns to match the HTML source code. If the source code or elements of the website were to change, this would result in the regular expressions and selected <div> HTML elements become broken.

An example of where the regular expression patterns have become broken was when Indeed did a re-design of its website in May 2019. When the site owners decided to take this action, it resulted in the elements changing, meaning the structure had changed. This then resulted in our system we had designed breaking. The previously designed regular expressions would no longer match returning null information. This is a problem with relying on API's.

Figure 6.22, is the created method process in Python for the Beta I prototype.

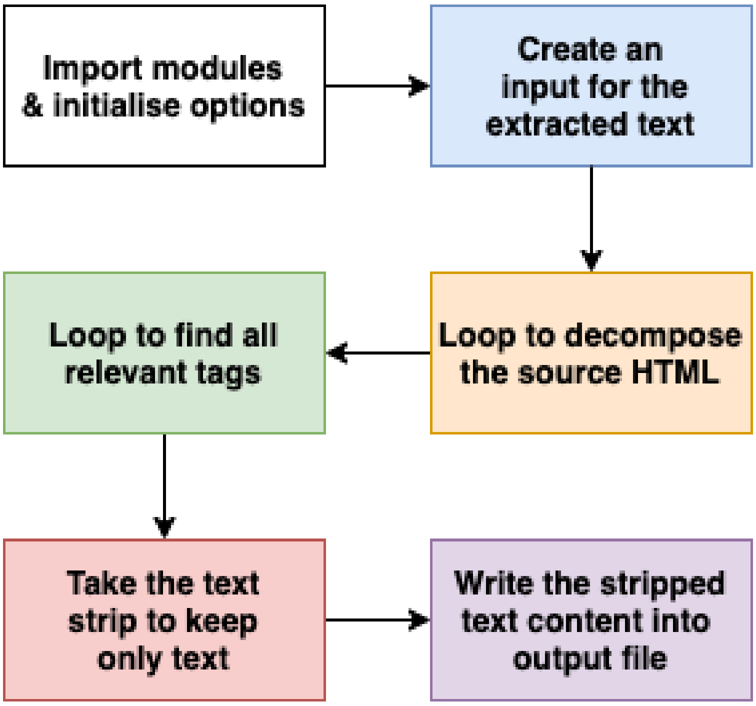

Import modules & initialise options
Create an input for the extracted text
Loop to decompose the source HTML
Loop to find all relevant tags
Take the text strip to keep only text
Write the stripped text content into output file

Figure 6.22: An additional web scraping process created for Beta I prototype for extracting elements from web pages

Table 6.3, is the output from the method process from Figure 6.22. This results in the title, company, location, date posted and currency being scraped from the web content.

Sample Output	Source Code
Test Engineer Ardia Solutions £40k - £55k Per Year Bangor Test Engineer Remote/Bangor Up to 55k On behalf of our rapidly growing client, we are recruiting for an enthusiastic Test Engineer. As Test Engineer you will be thoroughly testing products to identify potential problems and defects, optimising quality, and ensuring regulatory compliance.This role offers remote working but you wi...	<h2 class="card-title" title="**Test Engineer**" name="card_title">Test Engineer</h2> </a><h3 name="card_companyname" >**Adria Solutions**</h3> <span name="card_job_location" class="card-job-location">**Bangor**</span> </div><div class="d-md-none results-mata-holder set-margin"> <div class="meta-info-sm" name="card_location"><span>Bangor </span></div><div class="meta-info-sm" name="card_salary"><span><div class= "card-salary" name="card_salary"> <span>**£40k**<span aria-label="to">**-** </span> **£55k Per Year**</span></div></span> </div></div> <div name="card-job-description" class="col-12 results-card-description"> <div name="sanitizedHtml" class="sanitizehtmlcomponent__ SanitizedContent-sc-1s5wjr7-0 kVyGpI"> **Test Engineer Remote/Bangor Up to 55k** <span> **On behalf of our rapidly growing client, we are recruiting for an enthusiastic Test Engineer. As Test Engineer you will be thoroughly testing products to identify potential problems and defects, optimising quality, and ensuring regulatory compliance.**</span><span> **This role offers remote** **working but you wi...** </span></div>

Table 6.3: This shows a comparison of the source code and a sample after using the web scraping process on Monsters website. Highlighted areas in bold on the source code indicate the elements taken for the sample output

6.4.1 Issues with the Beta I prototype

We found that there were many issues with using the external APIs when creating our system and creating web extraction methods for the project.

There have been complications with the source code changing from time caused by the continuing development of the housing and job websites, resulting in our created system breaking easily and it not being future proof.

There has also been the issue of the APIs being restrictive over time. Indeed and Zoopla have changed policy on how each API can now be used since the project started in 2016.

There were also complications with our requests sometimes being blocked by the websites. This was due to the user agent being unidentifiable to the website and the server rejecting our requests. The pace of gathering results was also another factor; the speed of the requests became a problem with the server preventing further access. This resulted in 403 forbidden requests at points, requiring substantial trial and error when trying to web scrape the content. Bots were allowed under the websites' terms and conditions at the time of this project.

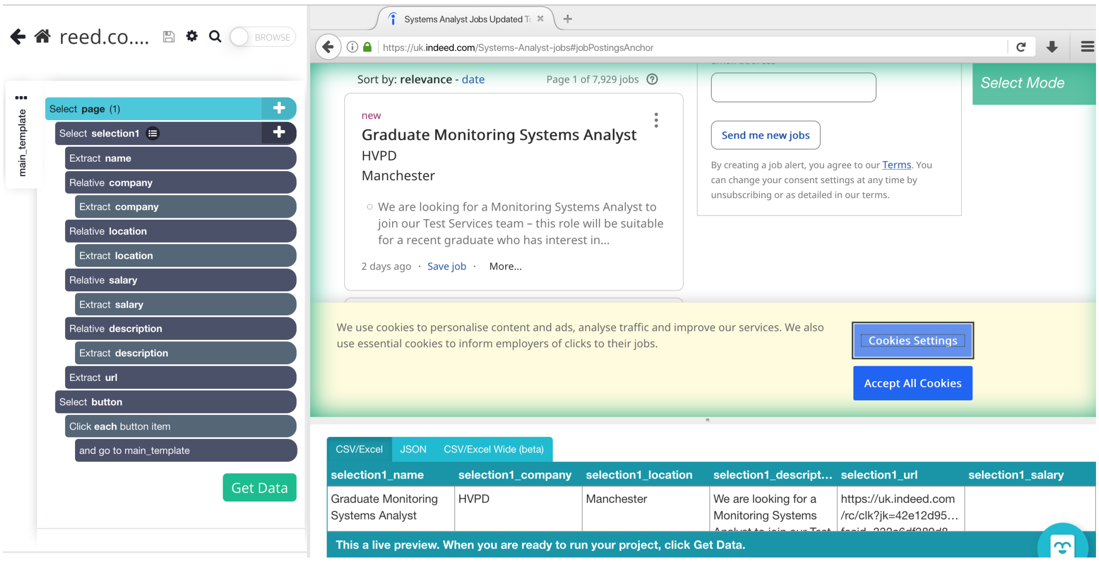
Despite these complications, these methods were eventually successful at scraping the web content, although they cannot be solely relied on due to the external factors outside of our control.

For this reason, we investigated the use of the third-party web scraping application called ParseHub.

6.4.2 ParseHub

ParseHub is a free third-party web scraper that can extract data by clicking on the elements the user requires [7].

[7] https://parsehub.com


reed.co....
BROWSE
main_template
Systems Analyst Jobs Updated T
https://uk.indeed.com/Systems-Analyst-jobs#jobPostingsAnchor
Select Mode
Get Data
CSV/Excel
JSON
CSV/Excel Wide (beta)

Select page (1)
Select selection1
Extract name
Relative company
Extract company
Relative location
Extract location
Relative salary
Extract salary
Relative description
Extract description
Extract url
Select button
Click each button item
and go to main_template

Sort by: relevance - date   Page 1 of 7,929 jobs

new
Graduate Monitoring Systems Analyst
HVPD
Manchester

○ We are looking for a Monitoring Systems Analyst to join our Test Services team – this role will be suitable for a recent graduate who has interest in...

2 days ago · Save job · More...

Send me new jobs

By creating a job alert, you agree to our Terms. You can change your consent settings at any time by unsubscribing or as detailed in our terms.

We use cookies to personalise content and ads, analyse traffic and improve our services. We also use essential cookies to inform employers of clicks to their jobs.

Cookies Settings
Accept All Cookies

selection1_name	selection1_company	selection1_location	selection1_descript...	selection1_url	selection1_salary
Graduate Monitoring Systems Analyst	HVPD	Manchester	We are looking for a Monitoring Systems	https://uk.indeed.com /rc/clk?jk=42e12d95...	

This a live preview. When you are ready to run your project, click Get Data.

Figure 6.23: ParseHub Interface

Figure 6.23, provides a screenshot of the interface of ParseHub. On the sidebar, the user can load up the website and select the web page elements and extract the text. The extracted elements can be shown in a preview under the CSV/Excel output on the bottom of the screen.

Using Parsehub allowed us to save time on the bulk amount of web scraping, which further enhanced the methods created for the Beta I Prototype.

6.5 Summary of Beta I Prototype Design

The Beta I prototype has demonstrated that external APIs give the possibility of using their own listings in a structured manner, meaning that it does not require any manual scripts to extract elements by using regular expressions. However during the implementation, both APIs from Zoopla and Indeed now require payment to use them. This resulted in the realisation from the project team that this API-based solution would not be suitable as there were too many external factors would be out of our control.

An alternative way was made by creating our own system for scraping individual website elements using a language modelling approach as described in Chapter four. It was noted however that this approach was still not future proof and could break if a website owner from either jobs or housing websites were to change the structure of the website.

The scope of the project only required to create a mockup prototype of housing and jobs together using static data. The use of trying to use live data was out of scope and has been left for future work.

6.6 Evaluation of the Beta I Prototype

This section evaluates the Beta I Prototype that has been created for the project. The Beta I prototype evaluation involved the website created with APIs and poses questions to sample users to find out preferences of specific options that the company partners wanted to investigate.

The questionnaire was devised from a discussion with the company partners and supervisor involvement. The questionnaire was conducted once the prototype was ready for evaluation.

The evaluation methodology used for the Beta I prototype uses open-ended style questions with 5-point Likert scale style questions. The System Usability Scale questions were altered in this questionnaire to find out answers in relation to housing and job queries so further adaptions could be made to the Beta II prototype. Ethics approval was granted for the questionnaires, the reference for this is: CSEE-P-2019-CG-012.

For the evaluation of the Beta I prototype, the creation of a mock-up of the accommodation and job website using vendors APIs was presented to university students to gain qualitative and quantitative data to analyse how students feel about using this given prototype. Students were selected as a suitable representative for the questionnaire because they are often seeking housing each year and also potentially looking for part-time employment because they could be finishing their degree and end up seeking both employment and housing opportunities.

A total of 50 respondents responded to this questionnaire. All respondents were University students, with the target population being 3rd year undergraduate students and Postgraduate students. An equal proportion of male and females were asked to complete this questionnaire.

The data storage and website used was using Microsoft forms, with excellent data security as a result. The form used required as minimal personal information as possible, being designed so it does not ask who they are. As it is held anonymously, the data cannot be removed later on as there is not a way to confirm that they did submit that questionnaire. If the user decides part way through to no longer take part, they can just close the browser window and it will be abandoned.

Using a questionnaire with this prototype was a convenient way of collecting comparable data from a large number of individuals although the questionnaire can only produce valid and meaningful results if the questions are precise and clear, and additionally they are asked consistently across the respondents.

Using a Likert-scale questionnaire was used for measuring user preferences with relative ease [106]. The respondents can specify their level of agreement or disagreement on an agree-disagree scale for a series of statements [19]. The Likert scale questions can capture the users feelings for a particular item. The results of analysis of multiple items then reveals a pattern that has scaled properties [83] [72] [28]. The questionnaire uses a 5-part Likert and it is used throughout the questionnaire.

There were three tasks for the user to complete and for them to fill out the questionnaire:

- Task 1 — Jobs: Complete the task of searching for a job and then return to the questionnaire to answer the following questions.

- Task 2 — Housing: Complete the task of searching for a house and then return to the questionnaire to answer the following questions.

- Task 3 — Housing and Jobs: Complete the task of searching for a house and job together and then return to the questionnaire to answer the following questions.

The questions asked in the Beta I prototype were:

1. What do you consider the main advantage this website has over existing job and housing websites?

2. How useful would it be if the search included job and housing results from countries worldwide?
3. How useful would it be to able to establish distances between a job and a housing location.
4. Was it easy to get started on this search?
5. Was it easy to do the search on this topic?
6. Are you satisfied with your search results?
7. Did you have enough time to do an effective search?
8. Please write down any other comments that you have about your searching experience with this information retrieval system here. Thank you.
9. How easy was it to perform a search on the website?
10. How easy was it to use this website?
11. How well did you understand how to use the website?
12. Do you agree with the following statement: I think that I would like to use this website frequently.
13. Do you agree with the following statement: I found the website unnecessarily complex?
14. Do you agree with the following statement: I thought the website was easy to use.
15. Do you agree with the following statement: I think that I would need the support of a technical person to be able to use this website?
16. Do you agree with the following statement: I felt very confident using the website.

17. How much did you like using the website?

18. Is there any platform you are aware of that you can search requiring both job-based and housing-based websites son a single platform?

19. What did you like about the website?

20. What did you dislike about the website?

21. How much experience have you had searching on commercial websites?

22. How often do you conduct a search on a job website?

23. How often do you conduct a search on a housing website?

24. Have you used any of the following websites?

25. For job-based searching, which of the websites would you prefer to use?

26. For housing-based searching, which of the websites would you prefer to use?

6.6.1 Beta I Prototype Questionnaire Evaluation — Results

Table 6.4 shows the results from the Likert style questionnaire, providing how many respondents responded to each of the Likert options. In this section, a discussion for each of the questions in the questionnaire will be examined in turn. Additionally the median, mean, standard deviation and minimum and maximum response for each question has been calculated in Table 6.5.

Question	Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree
2	0	0	2	18	30
3	0	0	2	7	17
4	0	0	0	10	40
5	0	0	0	9	41
6	0	0	2	17	31
7	0	0	0	3	47
9	0	0	0	12	38
10	0	0	3	19	28
11	0	0	0	17	33
12	0	0	3	27	20
13	46	2	2	0	0
14	0	0	1	19	30
15	47	0	0	3	0
16	0	0	0	21	29
17	0	0	0	32	18
21	0	1	4	21	24

Table 6.4: Beta I Prototype Likert Style Question Results

Question	Median	Mean	Standard Deviation	Minimum Value	Maximum Value
2	5	4.56	0.57	3	5
3	5	4.52	0.64	3	5
4	5	4.89	0.40	4	5
5	5	4.82	0.38	4	5
6	5	4.58	0.57	3	5
7	5	4.94	0.24	3	5
9	5	4.76	0.43	4	5
10	5	4.54	0.57	3	5
11	5	4.66	0.47	4	5
12	4	4.34	0.59	3	5
13	1	1.00	0.00	1	1
14	5	4.58	0.53	3	5
15	1	1.18	0.71	1	4
16	5	4.58	0.49	4	5
17	4	4.36	0.48	4	5

Table 6.5: Beta I Prototype Questionnaire Statistics

Q1: What do you consider the main advantage this website has over existing job and housing websites?

The purpose of this question was to explore the users' thoughts on what they believed the main advantage of the website was over existing sites out there. This question was open-ended (non-Likert-scale) for the user to provide a longer text-based response.

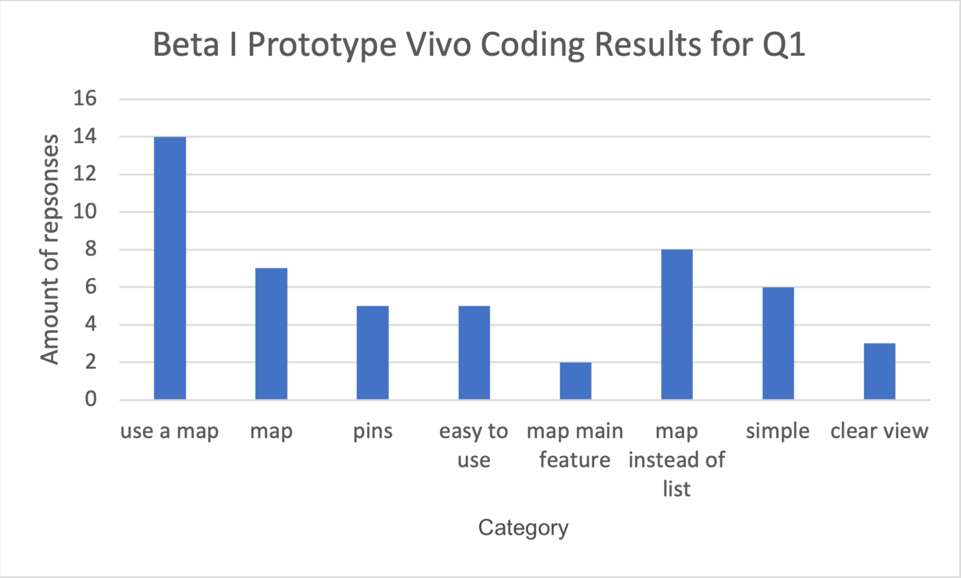


Figure 6.24: Beta I Prototype Vivo Coding Results for Q1

An inductive approach was used on the open-ended questions from the questionnaire using vivo coding. The vivo coding method has the ability to read through the data and name codes based on words and phrases utilised by the participant [92]. Reviewing the comments received from the question, participants found the main advantages of this prototype were “use a map”, “map”, “map instead of list” and “simple” are the higher ranked phrases used by the comments in the responses provided. It was evident in the results shown that users found it convenient to have the ability to interact with a map-based interface instead of a list. A few of the comments are below:

- ‘Using a map instead of a listing result.’
- ‘Nice and simple, easy to use, and having a map instead.’
- ‘It is refreshing and nice to see something easy to use compared to the usual listing of results on other websites.’

It was clear that users enjoyed the view of a map instead of a list with the ability to see and explore further details by clicking on the circle pins than they would having produced in a static list.

Q2: How useful would it be if the search included job and housing results from countries worldwide?

Since there is no single search service that provides a search service with housing and jobs together, this question explored whether the respondent felt that finding results from another country would be beneficial if the user was planning to move abroad. A Likert scale was used in this question and the breakdown is discussed in Table 6.5. 29 out of 50 participants rated this the maximum.

The results show from Table 6.5 show that there is a desire for a website that can provide both housing and jobs from a worldwide aspect.

Q3: How useful would it be to able to establish distances between a job and a housing location?

The question posed was to ask if a user would like the ability to pin where a job is located to where a house is located to show the distance between both of them, a factor which could encourage them based on the time factor to get to work and home. A Likert scale was used in this question and the breakdown is discussed in Table 6.5. 17 out of 50 participants rated this the maximum.

Q4: Was it easy to get started on this search?

The question asked was to explore how easy the user felt the website was to use. 40 out of 50 participants had rated this the maximum value, being extremely easy to use. A Likert scale was used in this question and the breakdown is shown in Table 6.5.

Q5: Was it easy to do the search on this topic?

The question's purpose was to determine if the participant managed to find it easy to search for jobs and housing. 41 out of 50 participants had rated this the maximum value, being very easy to use. A Likert scale was used in this question and the breakdown is discussed in Table 6.5.

Q6: Are you satisfied with your search results?

The question's purpose was to determine if the participant was satisfied with the overall search results that they were querying. A Likert scale was used in this question and the breakdown is discussed in Table 6.5. 31 out of 50 participants had rated this the maximum value, being very satisfied.

Q7: Did you have enough time to do an effective search?

The questionnaire set out a time frame to complete all questions and use the website within 15-20 minutes. This question was designed to ask if the participant felt that they had enough time to do effective searching for housing and jobs.

A Likert scale was used in this question and the breakdown is discussed in Table 6.5. 47 out of 50 participants had rated this the maximum value, indicating that they had enough time for the search.

Q8: Please write down any other comments that you have about your searching experience with this information retrieval system here. Thank you.

An optional question was posed by asking participants if they had any further comments about their searching experience. Out of 50 participants, 13 participants left comments.

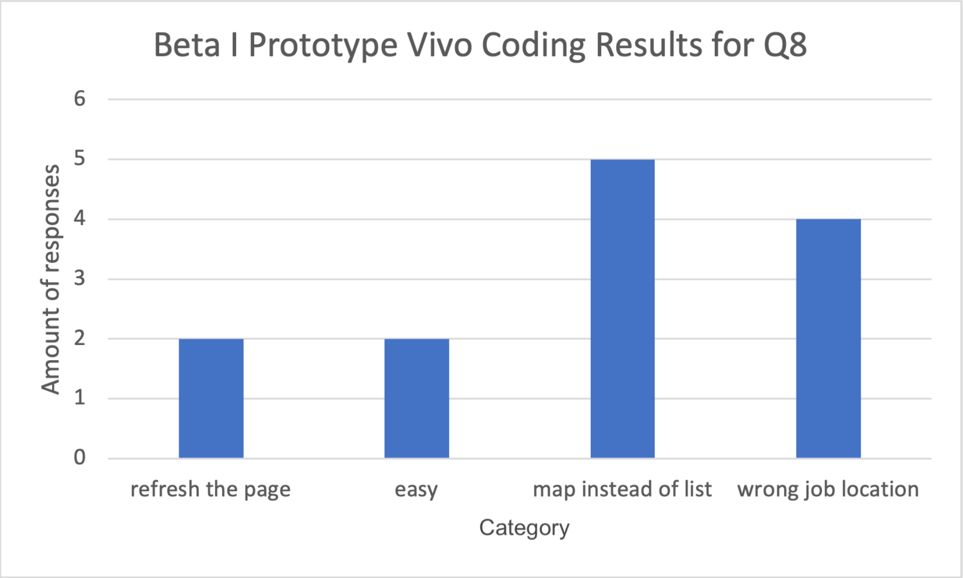


Figure 6.25: Beta I Prototype Vivo Coding for Q8

The vivo coding results in figure 6.25 shows the amount of responses to the inductive labels. It indicates that the website was preferred with a “map instead of a list” although sometimes the results had the “wrong job location”. A few of the comments are listed below. This question was posed as an open-ended question.

- ‘Sometimes, it was a little slow to put the information on the map. Sometimes it was wrong for jobs.’
- ‘Nice to see the ability to show the information on the map instead of a listing.’
- ‘It works well, the housing places well the jobs not so much.’
- ‘I noticed the odd job I searched for being pinned incorrectly to where it should be, although it was in the general area.’

Q9: How easy was it to perform a search on the website?

The question’s purpose was to find out how easy it was for the participant to perform a search on the website.

38 out of 50 participants had rated this the maximum value, that is being extremely easy to perform a search. The results indicate that most people found it very easy to perform a search on the website. A Likert scale was used in this question and the breakdown can be shown in Table 6.5.

Q10: How easy was it to use this website?

The question’s purpose was to find out how easy it was to use the website: the general overall aspect of the search, appearance, and usage. This factor will be essential to see how many participants felt it was easy to use.

A Likert scale was used for this question and the breakdown is discussed in Table 6.5. 28 out of 50 participants had rated this the maximum value, that is being extremely easy to perform a search. The results indicate that the website was easy to use overall.

Q11: How well did you understand how to use the website?

The question was to find out how the participant understood how to use the website, meaning if they managed to use the website well enough.

33 out of 50 participants had rated this the maximum value of 5, extremely easy to understand the website. A Likert scale was used for this question and the breakdown is discussed in Table 6.5.

Q12: I think that I would like to use this website frequently

The question's purpose was to find out if the participant would frequently use this website. This will depend on how the participant searches for jobs and housing in their own lives, but it will provide an insight on whether they are willing to use this website specifically frequently.

A Likert scale was used for this question and the breakdown is discussed in Table 6.5. Results indicate that the majority of participants would frequently use this website.

Q13: I found the website unnecessarily complex?

The question's purpose was to determine if the participant found this website unnecessarily complex, resulting in difficulties in managing or searching for what they have queried. This indication is important to see if drastic changes are required to a future prototype.

A Likert scale was used for this question and the breakdown is discussed in Table 6.5. Results indicate that almost all participants did not find it unnecessarily complex.

Q14: I thought the website was easy to use.

The question's purpose was to find out if the participant found this website easy to use. It is essential to understand if significant changes need to be made or not for the future prototype.

A Likert scale was used for this question and the breakdown is discussed in Table 6.5. Results indicate that 30 out of 50 participants strongly agree that the website was easy to use.

Q15: I think that I would need the support of a technical person to be able to use this website?

The question's purpose was to find out if the participant needed the support of someone technically minded to use this website. No matter who uses the website, they should find it easy and straightforward to use without requiring the help of a technical person.

A Likert scale was used for this question and the breakdown is discussed in Table 6.5. Results indicate that 47 out of 50 participants strongly disagree that they would not need a technical person, compared with 3 people feeling that they would require a technical person to help them with the website.

Q16: I felt very confident using the website

The question's purpose was to find out if the participant felt confident in using the website. The prototype's purpose is to ensure that the person using it is confident and does not feel stressed or frustrated using it.

A Likert scale was used for this question and the breakdown is discussed in Table 6.5. All participants felt confident with using the website.

Q17: How much did you like using the website?

The question's purpose was to find out if the participant liked using the website. The prototype's likability is essential to ensure that people will come back to it.

Overall, all participants liked the website, which strongly supports the case for future versions using a similar front end for the user.

Q18: Is there any platform you are aware of that you can search requiring both job-based and housing-based websites on a single platform?

The question's purpose was to determine if the participant knew of any platform that the user can search for both jobs and housing on one website. Research has shown no existing website does this, but it would be essential to catching it here if a website is known. The results indicated that no one knew of a website that does this.

Q19: What did you like about the website?

The question's purpose was to find out if the participant wished to leave a comment about whether they liked the website. The vivo coding results in figure 6.26 has 'easy to use', 'map feature' and 'simple to search' being the most frequently stated comments about whether they liked the website.

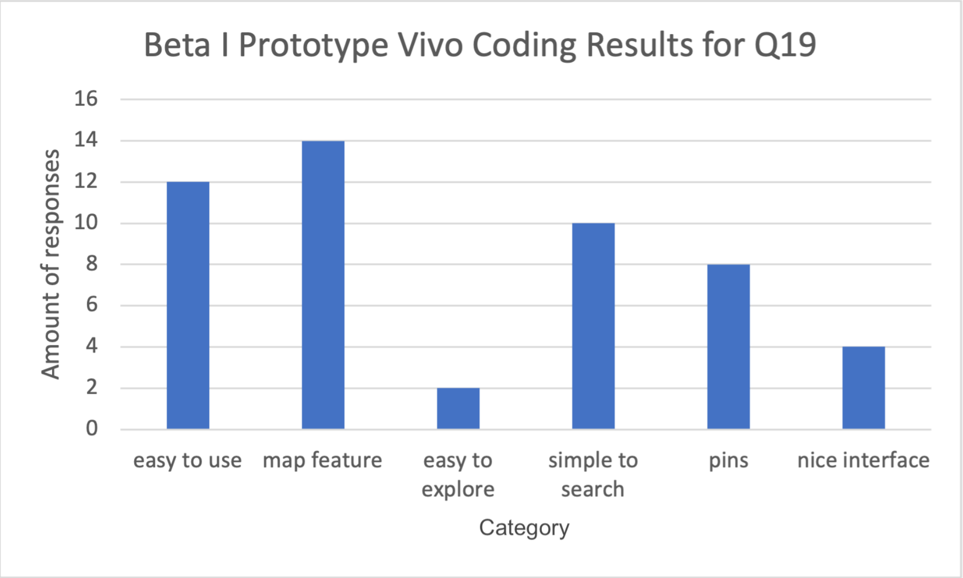


Figure 6.26: Beta I Prototype Vivo Coding for Q19

Below are a few comments provided as answers to this question:

- 'The map, the way it shows it in a simple way and easy to read and see.'

- 'The pins, the map, the simplicity.'

- 'Finding jobs and housing using a map, something I have not seen before.'

Q20: What did you dislike about the website?

The question's purpose was to find out if the participant wished to leave a comment about whether they disliked something about the website.

The vivo coding results in figure 6.27 has 'zoom in manually' and 'sometimes incorrect jobs' being the most frequently stated comments about whether they disliked the website.

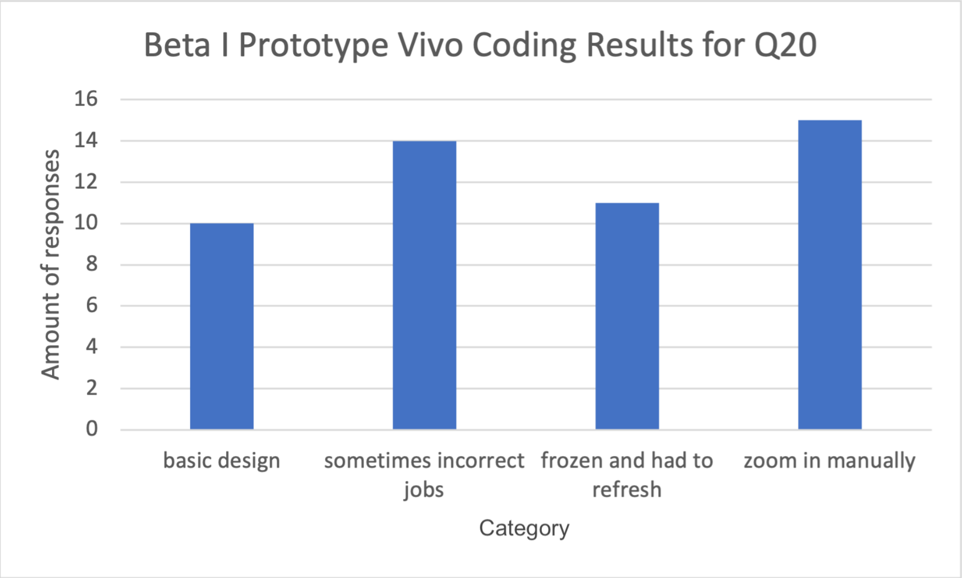


Figure 6.27: Beta I Prototype Vivo Coding for Q20

The comments noted that sometimes the website would freeze, or it would have to be refreshed, and people would have to zoom in on the map instead of it automatically doing it upon the request:

- 'Some jobs were misplaced, not many but a few.'
- 'I had to zoom into the area I searched for.'
- 'Having to refresh sometimes as it would break.'

Q21: How much experience have you had searching on commercial websites?

The question's purpose was to find out how much experience the participant has in using commercial job and housing websites.

As the table 6.4 shows, the results vary depending on the experience with looking for a job or house using commercial websites.

Q22: How often do you conduct a search on a job website?

The question's purpose was to find out how often a participant performs a search on a job website. The options provided were: Never, a few times a year, at least once a month, at least once a week.

The results were the following:

- Never — 0%.
- A few times a year — 40.00% (20 respondents).
- At least once a month — 14.00% (7 respondents).
- At least once a week — 46.00% (23 respondents).

Q26: How often do you conduct a search on a housing website?

The question's purpose was to find out how often a participant performs a search on a housing website. The options provided were: Never, a few times a year, at least once a month, at least once a week.

The results indicated the following:

- Never — 2% (1 respondent).
- A few times a year — 50.00% (25 respondents).
- At least once a month — 14.00% (7 respondents).
- At least once a week — 34.00% (17 respondents).

Q23: Have you used any of the following websites?

The question's purpose was to find out how often does a participant use different job and housing websites.

Website	Never	Rarely	Very Often	Total
Zoopla	1	12	37	50
Indeed	0	9	41	50
Monster	1	12	37	50
Jobsite	0	15	35	50
Google Jobs	48	1	1	50
Total Jobs	10	16	24	50
Primelocation	11	14	25	50
Rightmove	1	13	36	50
OnTheMarket	2	12	36	50
Reed	4	17	29	50
Glassdoor	11	18	21	50
Fish 4 Jobs	13	18	19	50

Table 6.6: Beta I Prototype Respondent Results for Q23

The results in figure 6.6 indicate that Zoopla and Indeed are highest ranked, meaning these websites are used very often. This compares with Google Jobs, whose website is rarely being used by the participants.

Q24: For job-based searching, which of the websites would you prefer to use?

The question's purpose was to find out which website the participant prefers to use. The options and results are shown in Table 6.7:

Website	**Respondents**
Indeed	28
Jobsite	16
Reed	9
Monster	6
Total Jobs	5
Fish 4 Jobs	5
Glassdoor	5
Google Jobs	0
Total	50

Table 6.7: Beta I Prototype Respondent Results for Q24

Results in Table 6.7 show that 56% of responses use Indeed. Following that, Jobsite received 32% of responses. Notably, Google Jobs did not receive any response although at the time of this questionnaire, Google Jobs was a new addition to Google.

Q25: For housing-based searching, which of the websites would you prefer to use?

The question's purpose was to find out which housing-based website the participant prefers to use. The options and results are shown in Table 6.8:

The breakdown of each website is listed below:

Website	**Respondents**
Primelocation	10
Rightmove	21
Zoopla	21
OnTheMarket	14
Total	50

Table 6.8: Beta I Prototype Respondent Results for Q25

Table 6.8 shows that both Rightmove and Zoopla had 21 responses each. Primelocation received the lowest amount of responses with 10.

6.7 Discussion and Findings

The Beta I prototype investigated using existing APIs that are available to the public. Originally they were free to use but during the start of the evaluation process, both APIs became pay to use. Although after emailing both API companies explaining the research nature of this project, they granted us temporary access to finish off the research. We investigated using these together to provide a solution for the project. However, it was soon realised that we could not control these APIs, and a potential payment scheme could be required for these APIs at any point in the future.

The project required an understanding of what existing APIs provide and found out how users felt about how the data was presented to them.

A questionnaire was created and sent out to 3rd year university students and postgraduate students to evaluate the Beta I prototype. The questionnaire involved a few tasks-based search queries, searching for a job, searching for a house and then searching for both housing and jobs. Those tasks were for the users to search for specific jobs and housing.

A series of survey questions were then presented to users. The users were asked to provide a rate of 1 (for strongly disagreed) up to 5 (for strongly agrees).

The questions asked have provided thorough detail on the likes, dislikes of the Beta I prototype. The questions allow for us to be able to consider changing different elements and features for the following Beta II prototype.

The findings of the survey found details on which websites respondents prefer for housing and jobs and have provided detail on how often the participants use housing and job websites. These details have been helpful as they give an insight on what direction is needed in the Beta II prototype.

Chapter 7

Developing and Evaluating Prototype Beta II

7.1 Introduction

This chapter discusses the development and evaluation of the Beta II prototype that was produced for this project. This includes the methodology that was used for this prototype and the suitability for this project. This prototype uses Agile Design as stated in section 6.1. For the final prototype design, the company partners wanted to have a homepage which the user could enter search criteria and then have the information display onto another page with a map. Additionally, it was important for both jobs and housing information to be brought together into one website.

Focus and context visualisation is used for the Beta II prototype. First, the user has an overview of the website, and upon submitting the information for the form, a detailed view of the information is provided [24]. Search user interfaces such as Google have a homepage which is as simple as possible, providing a search field and submit button. Upon submission, the results are then displayed on another page. Jobs and Housing websites that provide listings also do the same process of separating out the results.

An interactive map has been used to visualise the data for this Beta II prototype. The user has the ability to interactively manipulate the data. The map uses the presentation method of coloured shaped signs to demonstrate the different jobs and housing presented to the user [13]. The prototype uses a collaborative method of GIS of bringing geographic information to an information system with using jobs and housing [17].

An important consideration was that the input parameters on the page needed to be displayed in a clean presentable way, meaning it is not cluttered for the user to enter the input. There were iterations of the forms performed, one putting the fields next to each other in a row and the final outcome of having it in a divider with columns. This can be seen in Figure 6.2.

When the user has entered a query, the results then load in the Google Map. The results are displayed using pins scattered on the map. Figure 7.1 shows the different iterations used on the pin styles.

Once the user clicks on one of the coloured pins, an information window appears listing the title, short description, location and a link to the specific web listing. A red pin with the letter H indicates a house and a green pin with the letter J indicates a job. The reason behind this is because the company partners wanted the map to display different types of information. This allows for the clear separation of jobs and housing. The information searched in the query was also discussed in different iterations. This is shown in figure 7.1.

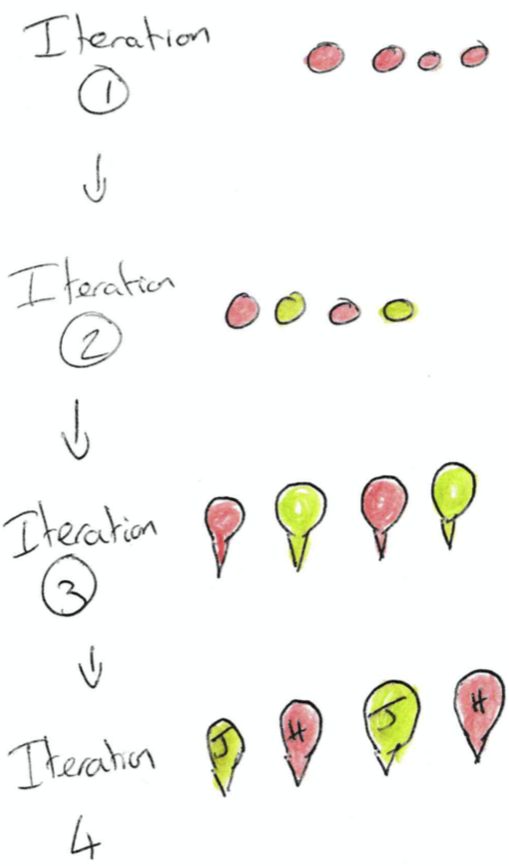
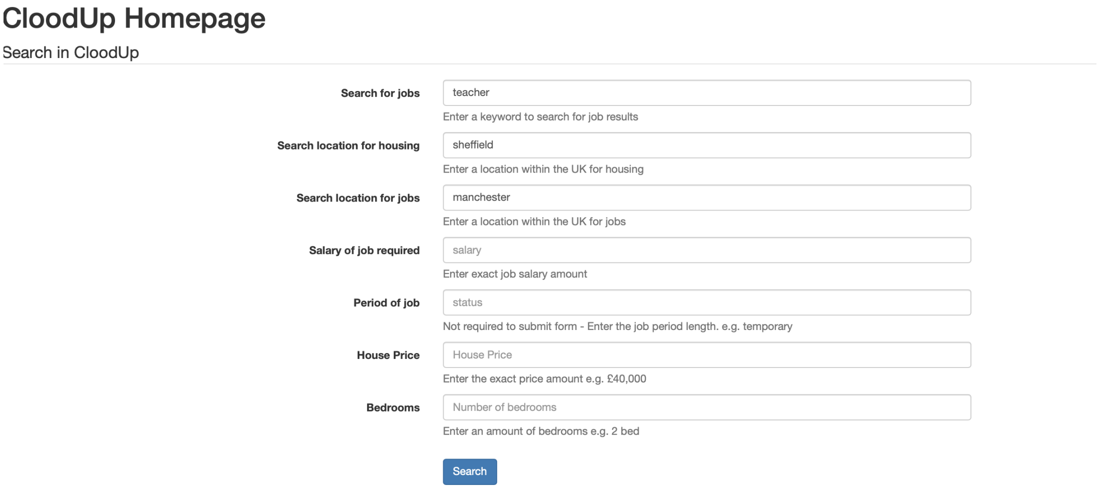


Figure 7.1: Beta II Prototype Design — Sketch of different Iterations of Pin Styles

7.1.1 Design Outcome

The vision in this final design prototype was to have a homepage for entering the search query and the ability in the future to expand the homepage with different elements such as news, events, company information and discovery of places.

The Google Maps API was used for this prototype. This is because it provides a bigger diverse set of tools and options to use.



CloodUp Homepage

Search in CloodUp

Search for jobs: teacher
Enter a keyword to search for job results

Search location for housing: sheffield
Enter a location within the UK for housing

Search location for jobs: manchester
Enter a location within the UK for jobs

Salary of job required: salary
Enter exact job salary amount

Period of job: status
Not required to submit form - Enter the job period length. e.g. temporary

House Price: House Price
Enter the exact price amount e.g. £40,000

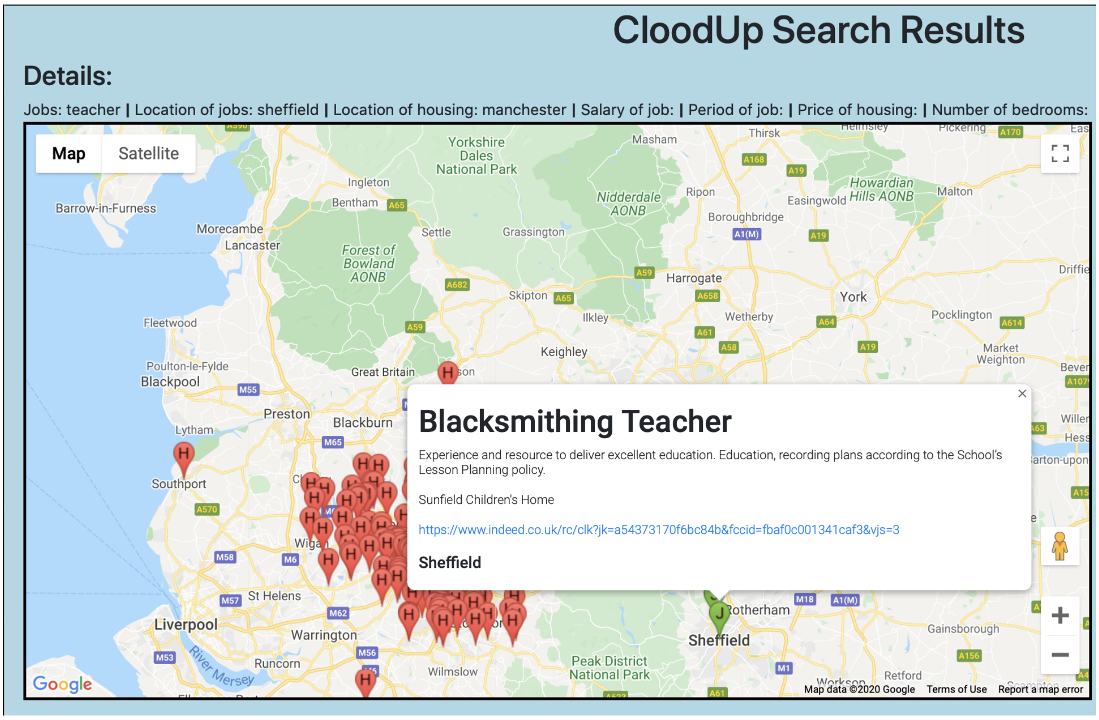
Bedrooms: Number of bedrooms
Enter an amount of bedrooms e.g. 2 bed

Figure 7.2: Beta II Prototype Design — Homepage

In figure 7.2, the homepage shows the overview of the form that the user has to fill in to get results. This includes:

- Search terms for jobs
- Search terms for housing
- Search location for jobs
- Salary of job required
- Period of job
- House Price Range
- Number of Bedrooms

In figure 7.3, once the search has been performed, the results will then be pinned on a map with red and green pins showing jobs and housing. If the user clicks on one of the pins, it will show an infowindow with basic information of the listing and a hyperlink to the specific web page listing. It also provides the details of the form to remind the user what they have requested. The decision behind this was because compared to other services as discussed in Chapter 3, the information windows do not provide details of the results on the map and only the pins.



CloodUp Search Results

Details:

Jobs: teacher | Location of jobs: sheffield | Location of housing: manchester | Salary of job: | Period of job: | Price of housing: | Number of bedrooms:

Blacksmithing Teacher

Experience and resource to deliver excellent education. Education, recording plans according to the School's Lesson Planning policy.

Sunfield Children's Home

https://www.indeed.co.uk/rc/clk?jk=a54373170f6bc84b&fccid=fbaf0c001341caf3&vjs=3

Sheffield

Figure 7.3: Beta II Prototype Design — Results

7.1.2 Conclusion from the design process

The final prototype takes the work from Beta I Prototype and provides the ability to search for jobs and housing on one website. It ensures that the search functionality is shown together on the homepage with a clear separation of results. This prototype was developed using agile development which provides the flexibility and freedom of iterations without having to implement many components. The design is simplistic and can be developed with more advanced design techniques, but this website design provides results of both housing and jobs with a clear difference between each.

7.2 Component Diagram Process

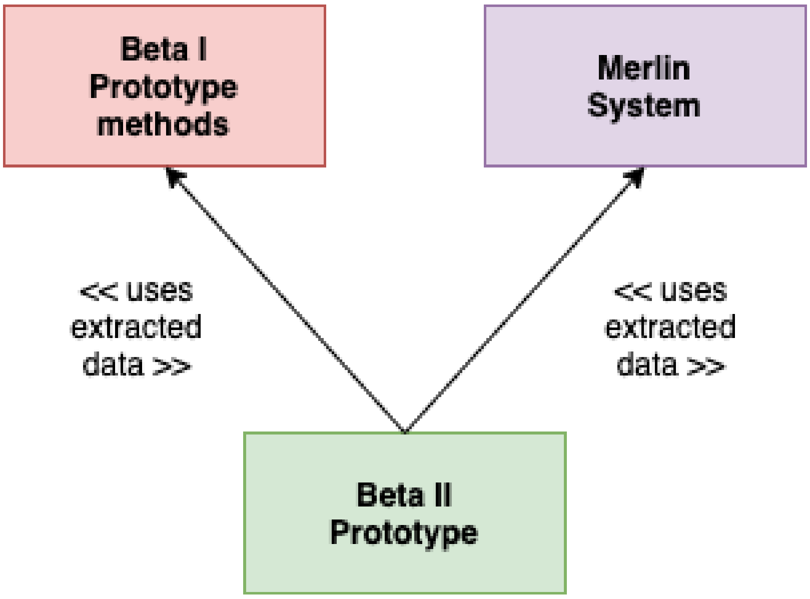


Figure 7.4: Component Diagram showing Beta II usage of data for the prototype

The final prototype uses a mixture of methods created in Beta I and tagged results in the Merlin system as shown in Figure 7.4. The final prototype uses a database that has records of housing and job listings. Beta II has a design objective from the previous chapter of following Agile design. The user fills out a search form, and SQL statements are used to retrieve the information.

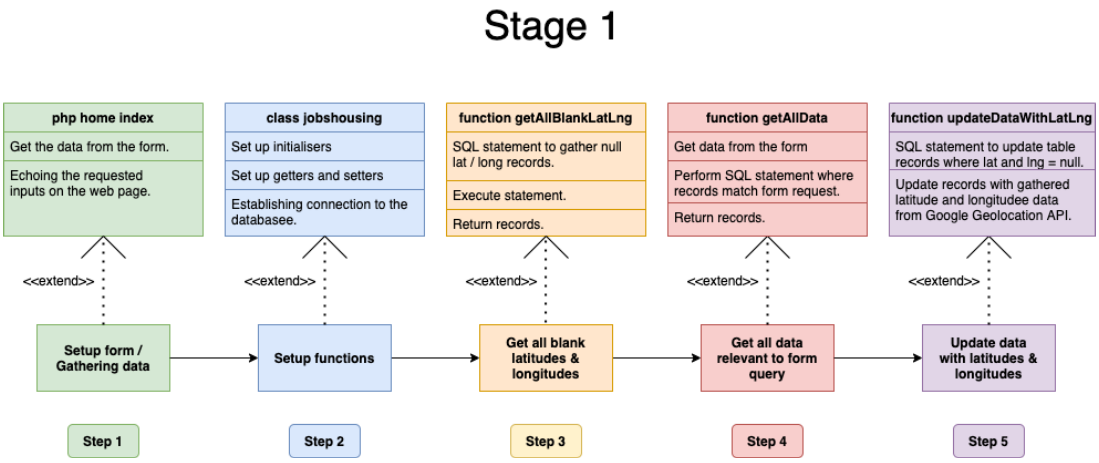

Stage 1
php home index
Get the data from the form.
Echoing the requested inputs on the web page.
class jobshousing
Set up initialisers
Set up getters and setters
Establishing connection to the databasee.
function getAllBlankLatLng
SQL statement to gather null lat / long records.
Execute statement.
Return records.
function getAllData
Get data from the form
Perform SQL statement where records match form request.
Return records.
function updateDataWithLatLng
SQL statement to update table records where lat and lng = null.
Update records with gathered latitude and longitudee data from Google Geolocation API.
<<extend>>
<<extend>>
<<extend>>
<<extend>>
<<extend>>
Setup form / Gathering data
Setup functions
Get all blank latitudes & longitudes
Get all data relevant to form query
Update data with latitudes & longitudes
Step 1
Step 2
Step 3
Step 4
Step 5

Figure 7.5: Stage 1: The process overview of the Beta II prototype's searching and saving functionality

The Beta II prototype has two stages. Figure 7.5 discusses the first stage. The first part of the Beta II's functionality comes from the unique created process created as shown in Figure 7.5 which comprises five steps.

Figure 7.5 step 1 describes the method that has been used in Beta II's prototype to display the results. The functionality starts by obtaining the data entered on the search form and then echos the requested inputs onto the webpage. The purpose of this is to show what the user had entered and display it to the user on the results page. This is done as as reminder for the user to remember what they had queried on the form.

Figure 7.5, step 2 performs the initialisation, including the get and set functions and establishes a connection with the database.

Figure 7.5, step 3 performs a function that gathers all records in the table where there are null latitude and longitude. Additionally, another function gets any information from the database relevant to the queried inputs.

Figure 7.5, step 4, sets the latitude and longitude of the data with null values. Additionally, the map is loaded here.

Figure 7.5, step 5, updates the data with the latitude and longitudes from the Google Geolocation API and saves the new coordinates to the database.

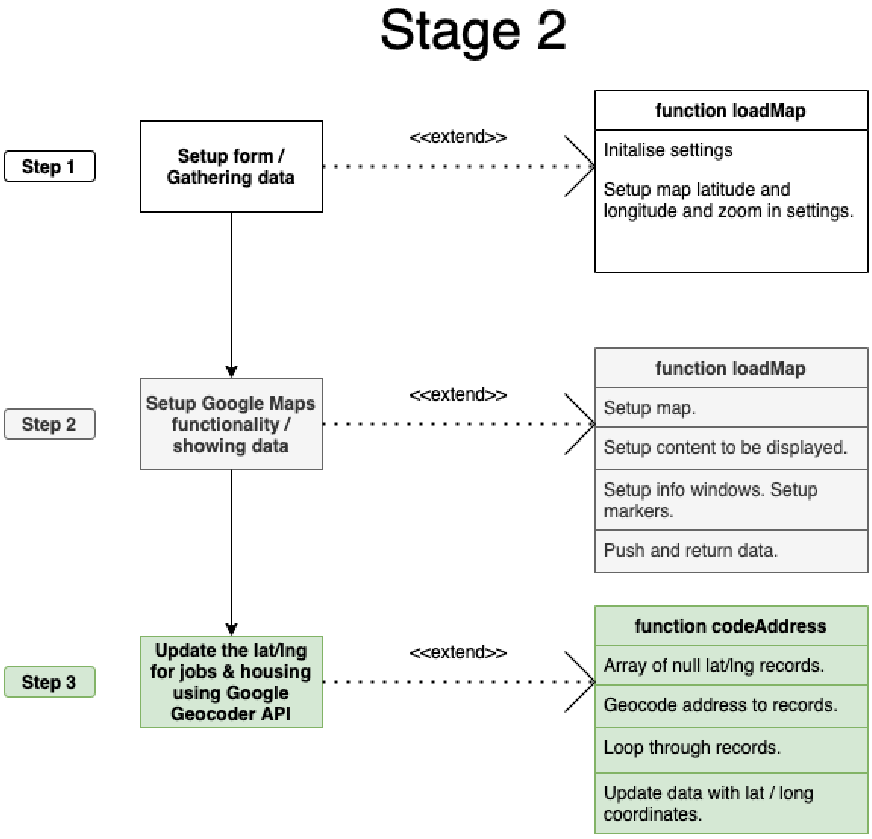

Stage 2
Step 1
Setup form / Gathering data
<<extend>>
function loadMap
Initalise settings
Setup map latitude and longitude and zoom in settings.
Step 2
Setup Google Maps functionality / showing data
<<extend>>
function loadMap
Setup map.
Setup content to be displayed.
Setup info windows. Setup markers.
Push and return data.
Step 3
Update the lat/lng for jobs & housing using Google Geocoder API
<<extend>>
function codeAddress
Array of null lat/lng records.
Geocode address to records.
Loop through records.
Update data with lat / long coordinates.

Figure 7.6: Stage 2: The process overview of the Beta II prototype setup and updating functionality

In Figure 7.6 shows the Beta II prototypes second stage. The second stage of the Beta II prototype involves setting up the Google Map, showing the pins, infowindows and updating the null addresses with Google's Geocoding API [1]. Google Geocoding API is a service that provides geocoding and reverse geocoding of addresses. In the instance where listings may only provide a brief location, this API can process the latitude and longitude of such location to put it onto a map.

In Figure 7.6, step 1, the function loads the maps initial settings of the map and geocoder.

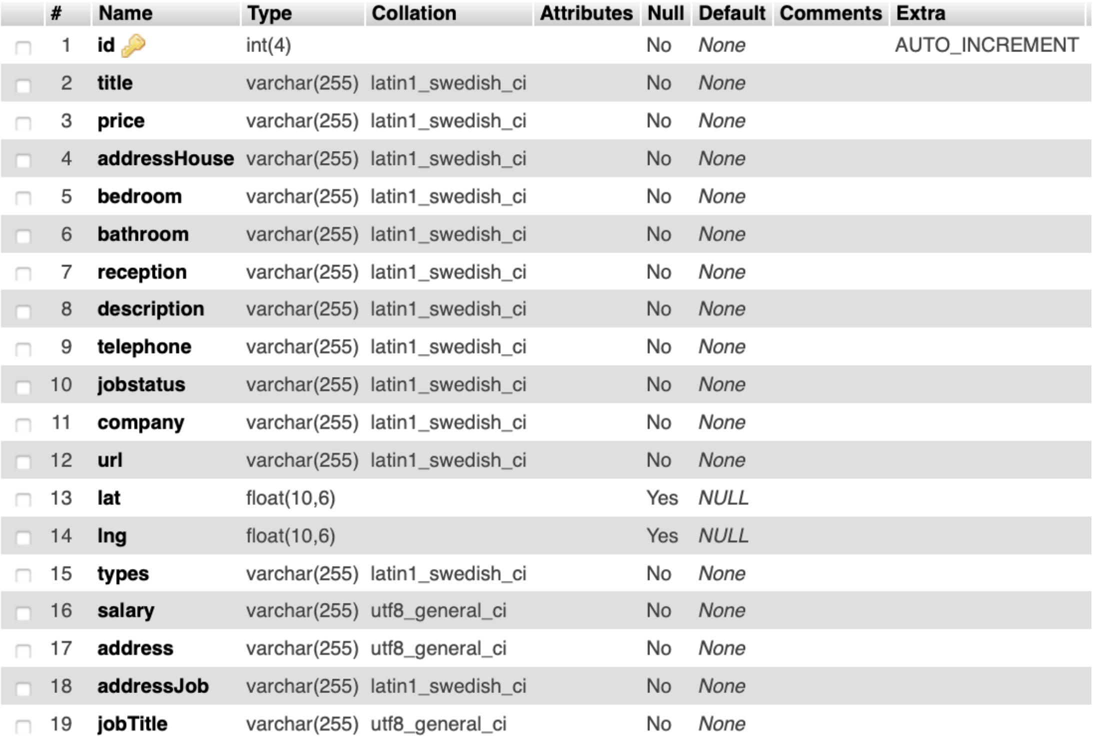
In Figure 7.6, step 2, sets up the markers, infowindows and icons. Step 2 also defines how the infowindows will operate and what imagery is being used for the icons of housing and jobs.

In Figure 7.6, step 3, a created function calls the Geocoder API and requests the latitude and longitude of the null entries from the database. This function then updates the

[1] https://console.developers.google.com

relevant column in the table. Additionally a further function is created as an AJAX call to update the data with the latitude and longitudes of rows in the database that had null latitude and longitude values.

The records that were created from the output from the Merlin system and other methods created in Chapter 6 are stored on a database on an InnoDB storage engine. The database has 43,493 records. 22,493 records are jobs, and 21,000 are housing. The structure of the database can be seen in figure 7.7.



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
□	1	**id** 🔑	int(4)			No	*None*		AUTO_INCREMENT
□	2	**title**	varchar(255)	latin1_swedish_ci		No	*None*		
□	3	**price**	varchar(255)	latin1_swedish_ci		No	*None*		
□	4	**addressHouse**	varchar(255)	latin1_swedish_ci		No	*None*		
□	5	**bedroom**	varchar(255)	latin1_swedish_ci		No	*None*		
□	6	**bathroom**	varchar(255)	latin1_swedish_ci		No	*None*		
□	7	**reception**	varchar(255)	latin1_swedish_ci		No	*None*		
□	8	**description**	varchar(255)	latin1_swedish_ci		No	*None*		
□	9	**telephone**	varchar(255)	latin1_swedish_ci		No	*None*		
□	10	**jobstatus**	varchar(255)	latin1_swedish_ci		No	*None*		
□	11	**company**	varchar(255)	latin1_swedish_ci		No	*None*		
□	12	**url**	varchar(255)	latin1_swedish_ci		No	*None*		
□	13	**lat**	float(10,6)			Yes	*NULL*		
□	14	**lng**	float(10,6)			Yes	*NULL*		
□	15	**types**	varchar(255)	latin1_swedish_ci		No	*None*		
□	16	**salary**	varchar(255)	utf8_general_ci		No	*None*		
□	17	**address**	varchar(255)	utf8_general_ci		No	*None*		
□	18	**addressJob**	varchar(255)	latin1_swedish_ci		No	*None*		
□	19	**jobTitle**	varchar(255)	utf8_general_ci		No	*None*		

Figure 7.7: Beta II: Overview of the Beta II Prototype Database Structure

7.3 Summary of the Beta II Prototype

The Beta II prototype has demonstrated that using the data gathered from the Merlin System and the Beta I methods of extracting web content, has successfully been able to place both housing and jobs listings onto pins onto a map. This is significant as this has met the requirements for the project. Mr. Edwin Smith, the company partner said *"It is remarkable to have such a prototype that finally brings together housing and jobs onto a simple but effective map based website. This is an exciting time as now we as Cloodup will be able to take this prototype to investors to show them how beneficial and useful it would be to have this as the future for the jobs and housing market."*

7.4 Evaluation of the Beta II Prototype

This section evaluates the Beta II Prototype that has been created for the project. The Beta II prototype was evaluated with the system usability scale methodology [26] in order to gain insight into the users' likes and dislikes. The questionnaire was completed by the user once they had interacted with the prototype during the experimental evaluation. 50 respondents participated in the questionnaire.

For the evaluation of the Beta II prototype, the creation of a combined accommodation and job website has been presented to the same students who took part in the evaluation of the previous prototype.

The criteria of the respondents and why they were selected is the same as for the Beta I prototype as discussed in section 6.6. The data sorted followed the same principles as discussed in section 6.6. Ethics approval was granted for the questionnaire, the reference for this is CSEE-P-2019-CG-012. Students were not timed to complete each task, although they took an average of 10 minutes to complete each task and to answer the questionnaire.

The System Usability Scale (SUS) [27] set of 10 questions were used to evaluate the usability of the Beta II prototype. This methodology was used in order to gain insights into the users' likes and dislikes.

7.4.1 Beta II Prototype — Results from Experimental Evaluation

This section discusses the results from the experimental evaluation for the Beta II Prototype. The experimental evaluation comprised three tasks — searching for jobs, searching for housing and searching for both jobs and housing.

There were three tasks for the user to complete and for them to fill out the questionnaire:

- Task 1 — Jobs: Complete the task of searching for a job and then return to the questionnaire to answer the following questions.

- Task 2 — Housing: Complete the task of searching for a house and then return to the questionnaire to answer the following questions.

- Task 3 — Housing and Jobs: Complete the task of searching for a house and job together and then return to the questionnaire to answer the following questions.

The questions used for the following tasks came from the System Usability Scale methodology [26]:

1. I think that I would like to use this system frequently.

2. I found the system unnecessarily complex.

3. I thought the system was easy to use.

4. I think that I would need the support of a technical person to be able to use this system.

5. I found the various functions in this system were well integrated.

6. I thought there was too much inconsistency in this system.

7. I would imagine that most people would learn to use this system very quickly.

8. I found the system very cumbersome to use.

9. I felt very confident using the system.

10. I needed to learn a lot of things before I could get going with this system.

Task 1 for Beta II Prototype — Jobs

The first task for the Beta II prototype was for the participant to devise their own queries to search for jobs on the website. Figure 7.1 shows the detailed for breakdown this evaluation.

The SUS questions alternate in their forms, with odd-numbered questions requesting a positive affirmation and even ones a negative affirmation. This specific form of questions using a Likert scale uses a score out of 40; however, it is commonly multiplied by 2.5 to achieve a score out of 100, although it is not a percentage. Subsequent testing by Sauro of the scale that uses over 500 trials resulted in an average score of 68, meaning anything above 68 is above average [126].

The Beta II prototype for the jobs task scored an average of 76.60 / 100 for the SUS. This score places it on the Good/Excellent boundary [18]. In order to get an Excellent, a score of 80 / 100 must be achieved.

Question	**Strongly Disagree**	**Somewhat Disagree**	**Neither Agree Nor Disagree**	**Somewhat Agree**	**Strongly Agree**
1	0	1	24	20	5
2	28	11	11	0	0
3	0	0	7	39	4
4	46	4	0	0	0
5	0	0	8	36	6
6	0	15	23	12	0
7	0	0	8	31	11
8	41	5	3	1	0
9	0	0	4	27	19
10	48	2	0	0	0

Table 7.1: Results of the SUS Evaluation of the Jobs section of the website. Each cell lists the number of respondents selecting that option for each question. Odd numbered questions have a positive phrasing, meaning ‘strongly agree’ is best. Even-numbered questions are negatively phrased, therefore disagree is the desired answer.

Task 2 for Beta II Prototype — Housing

The second task for the Beta II prototype was for the participant to devise their own queries to search for housing on the website. Figure 7.3 shows the detailed breakdown for this evaluation. The same methodology has been used here as before in Task 1.

The Beta II prototype for the housing section scored an average of 90.65 / 100 for the SUS. This score places it on the Excellent category [18].

Question	Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree
1	0	0	11	31	8
2	45	5	0	0	0
3	1	0	0	27	22
4	46	4	0	0	0
5	0	0	1	45	4
6	48	2	0	0	0
7	0	0	0	33	17
8	48	2	0	0	0
9	0	0	0	6	44
10	46	4	0	0	0

Table 7.2: Results of the SUS Evaluation of the Housing section of the website. Each cell lists the number of respondents selecting that option for each question. Odd numbered questions have a positive phrasing, meaning 'strongly agree' is best. Even-numbered questions are negatively phrased, therefore disagree is the desired answer.

Task 3 for Beta II Prototype — Jobs & Housing

The third task for the Beta II prototype was for the participant to devise their own queries to search for query both housing and jobs on the website. Figure 7.3 shows the detailed breakdown for this evaluation. The same methodology that was used for Task 1 and Task 2 has been done here.

The final prototype for the housing section scored an average of 94.26 / 100 on the SUS. This score places it on the Excellent category [18].

Question	Strongly Disagree	Somewhat Disagree	Neither Agree Nor Disagree	Somewhat Agree	Strongly Agree
1	0	0	2	29	19
2	50	0	0	0	0
3	0	0	0	2	48
4	50	0	0	0	0
5	0	1	0	9	40
6	11	24	16	0	0
7	0	0	0	6	44
8	50	0	0	0	0
9	0	0	0	1	49
10	50	0	0	0	0

Table 7.3: Results of the SUS Evaluation of the Housing and Jobs section of the website. Each cell lists the number of respondents selecting that option for each question. Odd numbered questions have a positive phrasing, meaning ‘strongly agree’ is best. Even-numbered questions are negatively phrased, therefore disagree is the desired answer.

Final Prototype Feedback

Respondents were also asked to provide open-ended comments for the following questions. This section discusses the feedback received.

What did you like about the website?

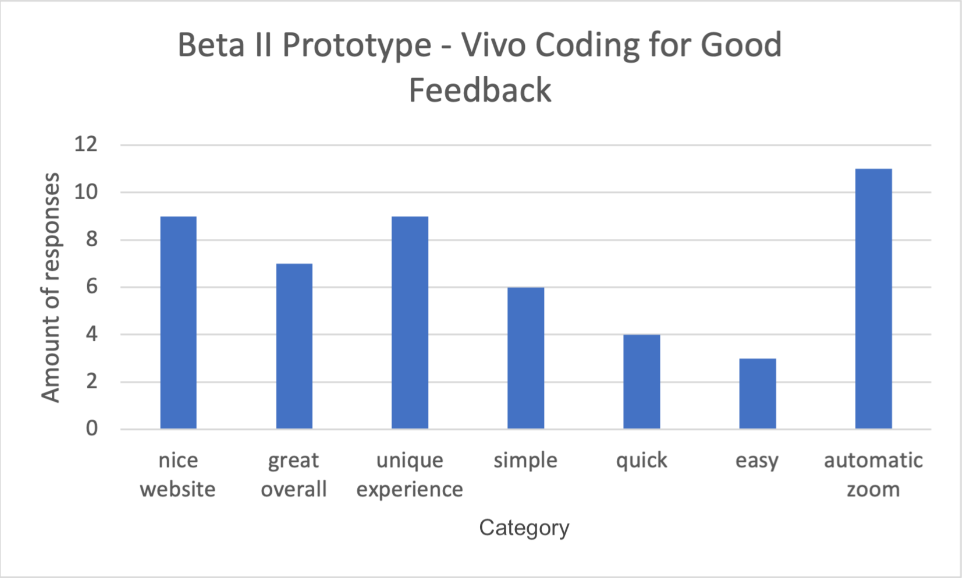


Figure 7.8: Beta II Prototype Results for Good Feedback question

Using the comments provided by the users, Vivo Coding has been adaopted to showcase the frequently used words within the comments. The Vivo Coding can be seen in Figure 7.8. Users indicated that they felt the website was ‘great’, ‘easy’, ‘nice’ to use overall:

- ‘It was great to navigate and manage. Nice bubbles of information.’
- ‘The zooming in the area after the search, the information provided was good and not overpowering.’
- ‘Overall a fantastic website to use. Zoom in, search, compare, look, explore.’

What did you dislike about the website?

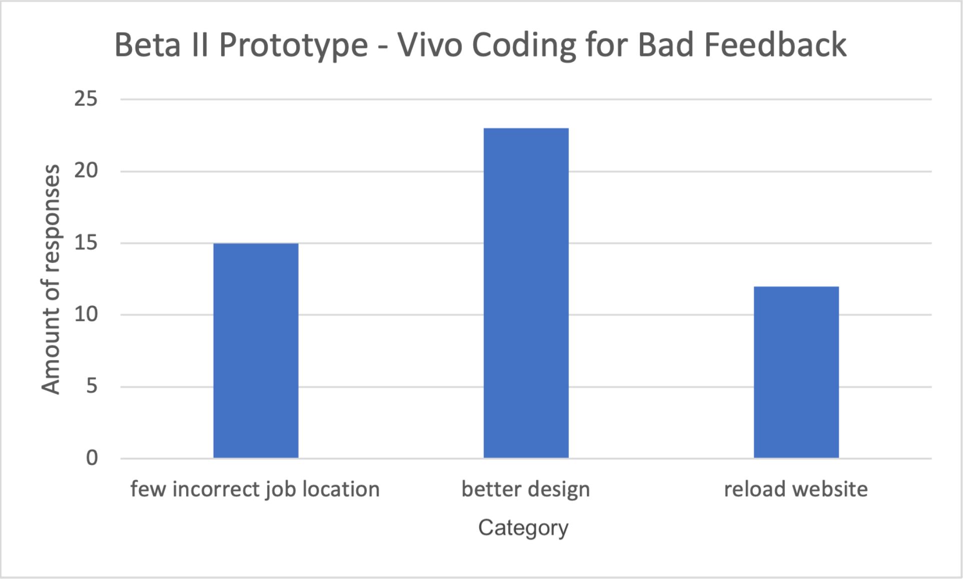


Figure 7.9: Beta II Prototype Results for Bad Feedback question

Using the comments provided by the users, Vivo Coding has been adopted to showcase the comments. This can be seen in figure 7.9. Users indicated that they felt the website could have 'better' 'design' and that sometimes the 'job' 'data' location was 'wrong'.

- 'Some inconstancy within the jobs element of the website, where some information was shown incorrectly.'

- 'The website design seemed a bit dull and could do with a fresh update.'

- 'Sometimes I had to refresh when the data did not load.'

What feature would you like to see introduced?

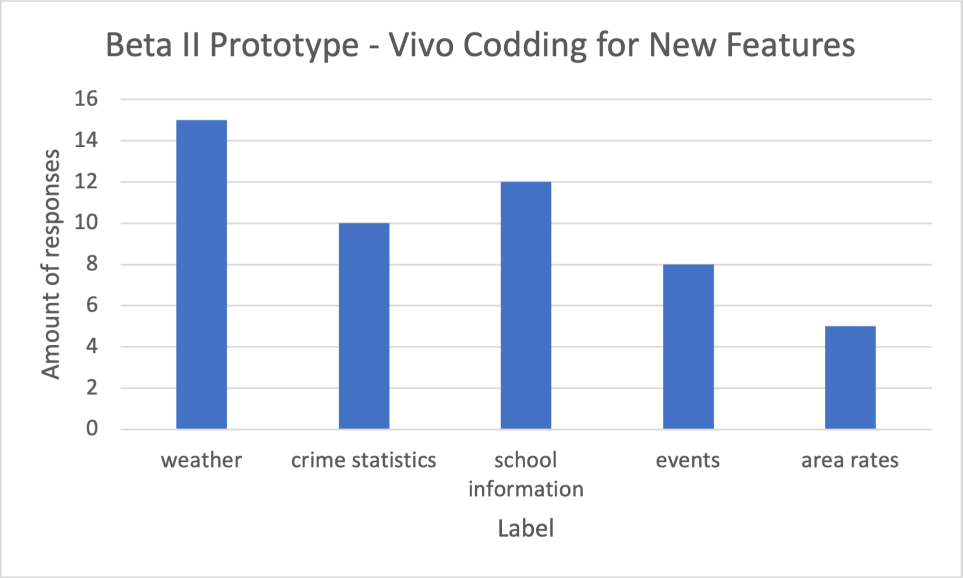


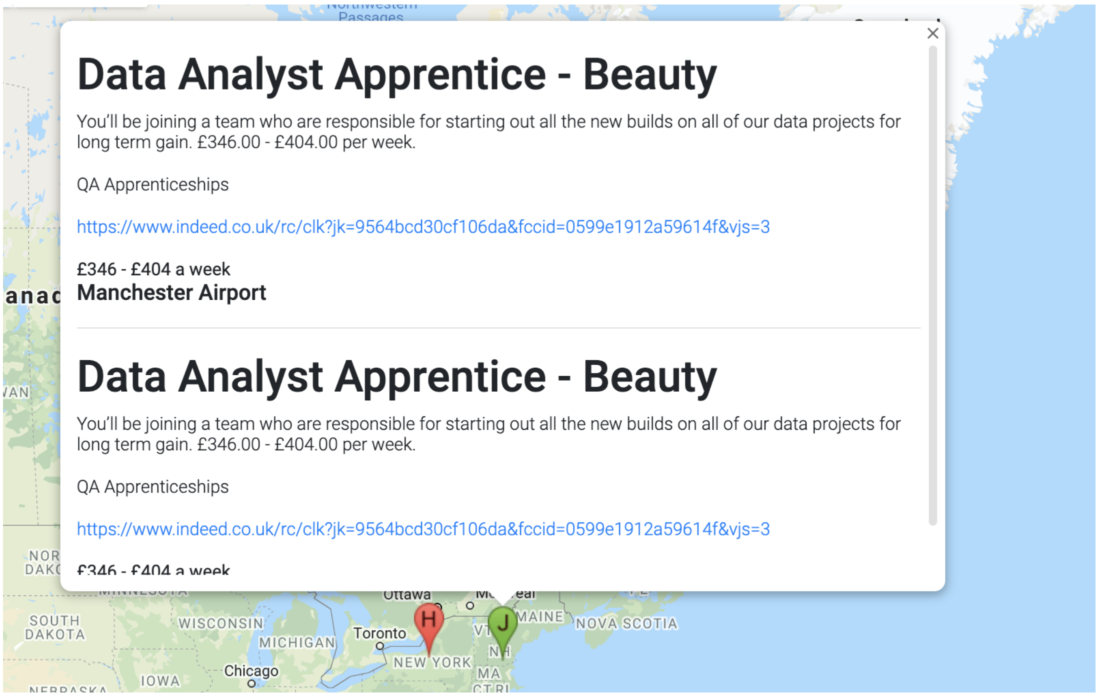
Figure 7.10: Beta II Prototype Results for Feature Feedback question

Using the comments provided by the users, Vivo Coding has been produced to showcase what was frequently mentioned. The results can be seen in figure 7.10. Users indicated that they would like to see 'weather', 'crime', 'school' information as future features to be added.

7.5 Limitations of Prototypes

This section discusses the limitations of the prototypes that have been highlighted from the feedback. There were some instances where job listings would be tagged incorrectly by Google's Geocoder API from the Beta II prototype.

An example of this would be where the API would select the wrong location of what is being provided; this can be shown in figure 7.11.



Data Analyst Apprentice - Beauty

You'll be joining a team who are responsible for starting out all the new builds on all of our data projects for long term gain. £346.00 - £404.00 per week.

QA Apprenticeships

https://www.indeed.co.uk/rc/clk?jk=9564bcd30cf106da&fccid=0599e1912a59614f&vjs=3

£346 - £404 a week

Manchester Airport

Data Analyst Apprentice - Beauty

You'll be joining a team who are responsible for starting out all the new builds on all of our data projects for long term gain. £346.00 - £404.00 per week.

QA Apprenticeships

https://www.indeed.co.uk/rc/clk?jk=9564bcd30cf106da&fccid=0599e1912a59614f&vjs=3

£346 - £404 a week

Figure 7.11: Beta II Location Problem

Figure 7.11 shows two jobs listed in the USA. The location provided by the data extraction methods only had 'Manchester Airport' as the location available. Google's Geocoding API took this location information and provided the latitude and longitude. The latitude and longitude provided in this case have placed these jobs in the USA.

It was highlighted by the company partner Mr. Edwin Smith that when entering an amount of search query into the form, sometimes the outputted results wouldn't retrieve the exact data being required. Search results may bring back no results or missed results.

Mr. Edwin Smith said:

"In some instances if you enter a price as 5000 as opposed to 50,000 you may get different results. If you put the same job criteria in for jobs but add permanent to one of them, you get a different result. If I set house price at 400,000, I get houses listed at 1,400,000 and 3,400,000. If I search for jobs and housing in Sheffield with no other criteria, I get lots of jobs and houses, one of which is Head of Delivery with a salary of £60,290. If I search Jobs only and input this criteria i.e., Sheffield - Head of Delivery - £60,290, I get nothing even though the job exists. If I then go back and add Sheffield to the housing criteria, the job appears."

Reviewing Mr. Edwin Smith's comments, the reasoning for these instances is because of how the SQL statement is setup to search the database with the inputted query. Additionally, the listing data in the database is not standardised.

An example of this is where within the database there is a column to identify if the job is 'Permanent / Temporary'. Not all job websites that were scrapped provide that information on their listings.

This meant that if a user were to enter either Permanent or Temporary as an option on the input form, it would miss out other potential listings within the database if they had a null field in that column.

The limitation behind this is due to how the implementation of the code was done. This is because the SQL statement requires on finding an option of either of those two statements or it being left blank, if it's specified, that means some listings would not be included with the returned output results.

7.6 Discussion and Findings

A questionnaire was created and sent out to University students to evaluate the prototype Beta II. The 50 university students selected were 3rd year undergraduate and postgraduate students. The students were approached by email and through face-to-face discussions on asking if they could use the prototype and ask A few tasks-based search queries provided as part of the experimental evaluation were for the users to search for jobs and housing.

A series of questions were then presented to users. The users were asked to provide a rate of 1 (for strongly disagreed) up to 5 (for strongly agrees).

The Beta II prototype was given to the same users that were surveyed in the evaluation of Beta I prototype. The questionnaire involved three tasks. The tasks given to the users were to search queries for just a job, then a housing query followed by both housing & jobs query.

The results showed that users had a stronger liking towards the final Merlin-based Beta II prototype than the API-based Beta I Prototype. The evaluation also highlighted that the jobs information obtained from the online jobsites causes problems regarding the location being too broad and not specific enough. The implications would be that the jobs would need to be further refined for future developments to make the results more accurate to the user.

Chapter 8

Lessons Learnt

8.1 Alpha Prototype

The Alpha prototype was created as a foundation to the overall thesis. The Alpha prototype allowed for a mechanism to identify the scope of the overall project. The approach for the Alpha project was to use agile software development to allow the flexibility of amending, creating and discussions with the company partners on a regular, quick basis to allow for rapid development.

The communication for the Alpha prototype was discussed with the company partners and my supervisor, of which we met every month face to face to discuss the latest workings on the Alpha prototype. Additionally a weekly round up email was sent to the company partners and the supervisor to provide latest thoughts and development. As a research student and developer of this project, the recommendation is to have as much communication with the stakeholders and relevant parties is key. This allows not only yourself to feel confident about what you are doing, but it also allows for the individual to express any concern and openly discuss how you are feeling. Starting on a new project and a new research student can be formidable, but having the approach of breaking individual parts down is key.

In the Alpha prototype, a decision was made to take a “quick and dirty” software development approach. This was a technical decision to allow to explore ways of tackling the approach of extracting some point of interest data but in a way that can be presented.

The requirements set for the Alpha prototype came from the company partners and supervisor, to create a mockup prototype capable of extracting data from the Internet. The decision that was made here allowed for the student to explore ways of creating a platform that can perform extraction on points of interest but in a way that was “quick and dirty”.

The technical decision made for the Alpha prototype was to use Python as the main source of coding this prototype with using the modules of BeautifulSoup, Scrapy and Django. Using these modules allowed the flexibility of creating a prototype that could demonstrate the proof of concept working and allows the company partners to see that the overall project of creating a website that brings housing and jobs together in some aspect is possible.

As a new project is approached, not only the development needs to be considered but how to go about designing the approach for that prototype. Despite in this instance wanting a quick and dirty approach to coming up with a solution to show the concept and idea would work, it is also important to allow the design aspect of the prototype to be thoroughly considered.

The approach taken in the Alpha prototype was to use a methodology that allowed to drill down into each element and to detail each working part of the prototype. The company partners had the vision of one overall view but this was difficult to understand thoroughly without drawing it out. Additionally they found it difficult to expand their horizon and view of what it should be. The Five Design Sheets allowed for all this functionality, where the idea of the company partners could be visualised together face to face within a meeting and then drilling down into the working parts.

The Five Design sheets allowed for further ideas to be conceptualised and after the process was completed with thanks to the methodology, the realisation from the company partners was that what they wanted to do was not possible due to the technology constraints of Google Earth.

The suggested recommended for design would be to take the Five Design sheets methodology and perform this on the initial prototype. This process allowed for the student to check if what the company partners wanted to do were possible. If this process did not happen, the idea the company partner would have been approached and whilst trying to implement this, would have been found to not work and would have had time wasted. The Five Design sheets allowed time to be saved and opened the eyes of the company partners of why using such an approach of the Five Design Sheets is important to software development.

As a proof of concept for the initial prototype, with housing and jobs brought together on a single website, it is even important to test the proof of concept prototype to see how well the existing modules and techniques worked at bringing together of points of interest. Using standard known evaluation techniques such as precision and accuracy to gather how well the results are performing is key. The testing and evaluation of the initial prototype can help lead the way to using either similar implementation methods for further prototypes or to raise attention that something different needs to be done.

The Alpha prototype produced results that ranged from 58% to 100%, dependent on what was searched. This resulted in the next prototype needing to develop new techniques for the implementation in order to retrieve results more efficiently. Having a proof of concept prototype for an overall project similar to this is important, as it allows for exploration, development and for a chance where mistakes can be made and there are chances to make corrections through communication, development and evaluation of the prototype. It was also a chance to work with the company partners on the designs and different approaches researched.

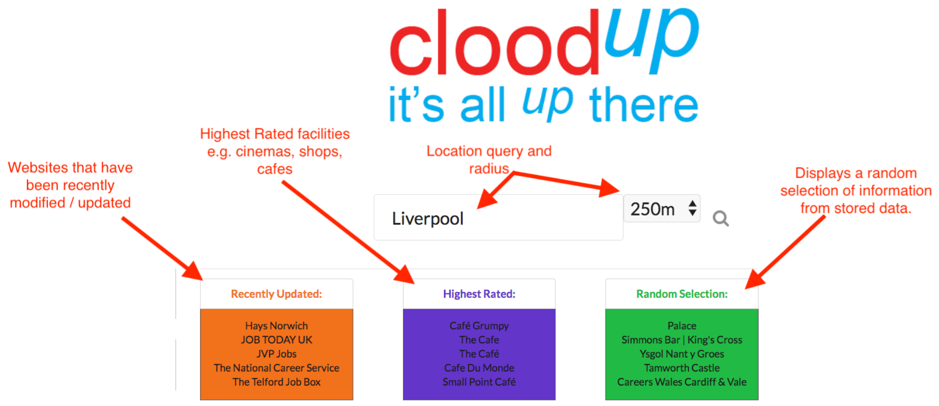
The Alpha prototype itself is split into five technical parts, as shown in the figure below:

Alpha Prototype	
Category	**Description**
Data Input	The data used for this prototype is from the Google Places API, using local points of interest in the area. The data was used as a proof of concept.
Data Output	The data output provides local points of interest such as information about local cafes, cinemas and shops.
Storage	The results of from the API are inserted into an SQL database and is regularly updated with new listings when the user calls the API.
Interaction	The user interacts with the Alpha prototype within a web page, allowing the user to interact with a table and drill down further into the specific entry.
Visualisation	The user enters a search query, upon submitting, the results display in a table format with columns and rows. The results are then displayed to the user. The user has the ability to filter the returned results.

Table 8.1: Alpha Overview of technical elements

8.1.1 Data Input

These technical elements will now be discussed in turn.


cloodup
it's all up there
Websites that have been recently modified / updated
Highest Rated facilities e.g. cinemas, shops, cafes
Location query and radius
Displays a random selection of information from stored data.
Liverpool
250m
Recently Updated:
Highest Rated:
Random Selection:

Hays Norwich
JOB TODAY UK
JVP Jobs
The National Career Service
The Telford Job Box

Café Grumpy
The Cafe
The Café
Cafe Du Monde
Small Point Café

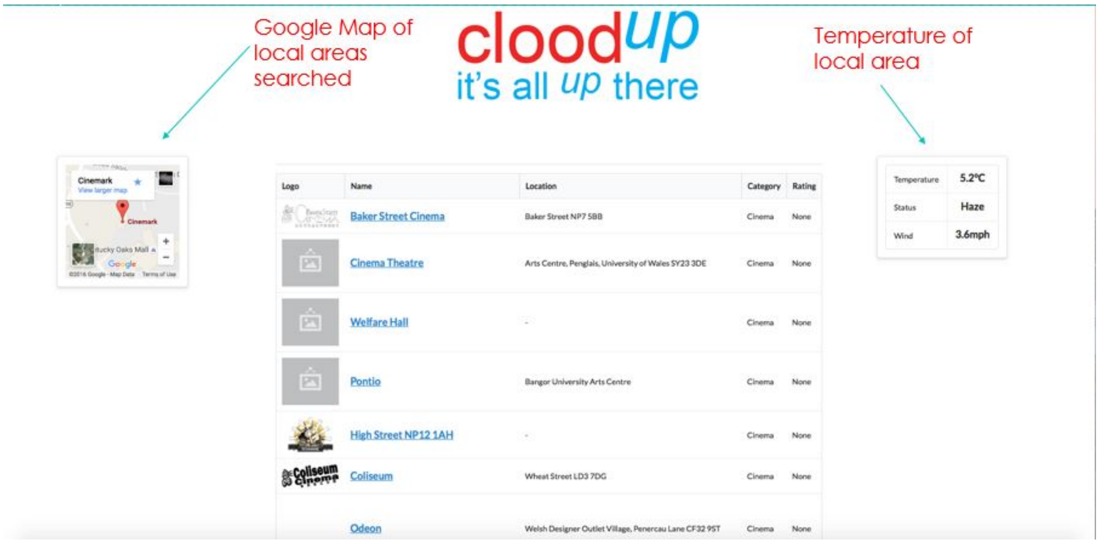
Palace
Simmons Bar | King's Cross
Ysgol Nant y Groes
Tamworth Castle
Careers Wales Cardiff & Vale

Figure 8.1: Homepage of Alpha prototype.

The Alpha prototype uses data available from Google Places API to gather local points of interest that is queried from the input. In order to query the API itself, a homepage had to be created as displayed in Figure 8.1. To query the API, the search field is filled by the user. The methodology “keep it simple” was used in the instance of creating the prototype, along with the quick and dirty methodology approach. This ensured that the core functionality and purpose could be achieved to continue development. The chosen data selected was points of interest with Google Places API as it was similar in context to the overall project abstract. As the API was already structured, all this required was an input from the user.

The lessons learnt from the data input solution in the Alpha prototype were that the search query only worked well for short keyword terms. If the user were to express a sentence for the query, the results would not provide the search results that they were looking for. The current implementation only looked for short keywords and it was found that users would like to express their search with a further advanced search. Going forward to the second prototype, it was clear that the input of a field needed to be expanded with further options, which provides a better user experience with the results that were shown.

8.1.2 Data Output


Google Map of local areas searched
cloodup
it's all up there
Temperature of local area
Cinemark
View larger map
Cinemark
Kentucky Oaks Mall
Google
©2016 Google · Map Data Terms of Use

Logo	Name	Location	Category	Rating
	Baker Street Cinema	Baker Street NP7 5BB	Cinema	None
	Cinema Theatre	Arts Centre, Penglais, University of Wales SY23 3DE	Cinema	None
	Welfare Hall	-	Cinema	None
	Pontio	Bangor University Arts Centre	Cinema	None
	High Street NP12 1AH	-	Cinema	None
	Coliseum	Wheat Street LD3 7DG	Cinema	None
	Odeon	Welsh Designer Outlet Village, Penercau Lane CF32 9ST	Cinema	None

Temperature	5.2°C
Status	Haze
Wind	3.6mph

Figure 8.2: Results Page of Alpha prototype

The Alpha prototype provides results to the user on a separate page in a table as shown in Figure 8.2. This provides a table of different points of interest, from the area searched for. The API is queried and a list of results is given. These results have then been placed into a table. The decision to use a table in this initial prototype was to provide one way of displaying the results from the API and allow for the user to drill down further into the specific result. We believed that showing just enough detail initially is required but having the choice to drill further down should be an option.

The lessons learnt from the data output solution in the Alpha prototype were that using the table to display the results although it was organised, did not allow much flexibility with the visualisation of the information shown. The output of the information on some instances were too broad meaning irrelevant information was being output from the search query. Better visualisation of the data output was needed for the Beta I prototype along with a more refined search.

8.1.3 Storage

The Alpha prototype allows for the student to explore different ways of either storing or displaying the data live. In this instance, the results were stored onto an SQL database which meant that we were not constantly calling the API. If the decision was made where we would not store the results, this would mean there would be a cost for calling the API repeatedly. The decision was made to store results into an SQL database and on a periodic basis, there would be a call to update the stored database results. This resulted in reduced costs involved with paying for the use of the API.

The lessons learnt from the storage solution of the Alpha prototype were that this allowed the ability to store results locally and saved calling the API each time to retrieve the results. The downside of this though was the method required periodic cleanup of the results and the information on the listings would change. The Alpha prototype provided a demonstration of the need to the balance between the costing of using APIs and storing data to avoid calling the relevant APIs for the same query with the exact result.

8.1.4 Interaction

The Alpha prototype allows for the user to enter a query, interact with the recently updated, highest rated and a random selection of places. When the user enters the query, we wanted the ability for the user to look at a mini map of the area that they have searched for and to know what the temperature and weather is at that location. The decision here to show the results in a table and for the user to interact with the results by drilling down into a specific result was taken to make things clear and simple. Proof of concept and the ability to show retrieval works was key in this aspect of design of the Alpha prototype.

The lessons learnt from the Interaction solution of the Alpha prototype were the user would see a table of results and could drill further into the specific result. However some of the results did not provide precise information. The ability for the user to explore and search an area was highlighted here as just having a table of results did not give a conceptual understanding of the query entered. The next prototype clearly required having a more precise and accurate results that would make it easier to interact with. Interacting with results that were not relevant to what was input in this prototype was confusing.

8.1.5 Visualisation

The Alpha prototype was simple in its design and visualisation, to keep to rapid agile software development, “quick and dirty” and “keep it simple” methodologies. The focus of proof of concept was on placing the results onto a table or basic HTML.

The lessons learnt from the Visualisation solution were that it is important to have a better way of presenting the data to the user, using a map that can be interacted with along with consolidating the information into better spaces and views.

8.2 Beta I Prototype

The Beta I prototype now focused on the task of looking at existing APIs for housing and jobs that were available, using scripts to automatically scrape HTML elements from websites. Beta I continued from the Alpha prototype with an emphasis on using agile software development although changing the way information is retrieved and displayed.

The communication for the Beta I prototype was similar in context to the Alpha prototype. We would meet up once a month and have weekly roundup emails to discuss progression.

The Beta I prototype requirements was now to use any free APIs currently available for both jobs and housing and then produce a design to display the results from the structured data.

As development of the Beta I prototype occurred, both APIs started to become pay per use. This resulted in further discussions concerning whether we should create a prototype that we had more control over, and these considerations needed to be made for the final prototype. At this stage, we made the decision to investigate methods of extraction from the APIs by creating Python scripts that extract elements from website and stores the information locally.

The technical decision was made for the Beta I prototype to use JavaScript, PHP and HTML5 for further web development and to move away from the pre-built modules in Python. This resulted in further development and the ability to create functions without the constraints of what was available in the pre-built modules in Python.

The design used in the Beta I prototype was agile design, as this allowed for a quicker turnover time of workflow which suited the company partners better. The Five Design Sheets produce good collaboration on starting and discussing ideas.

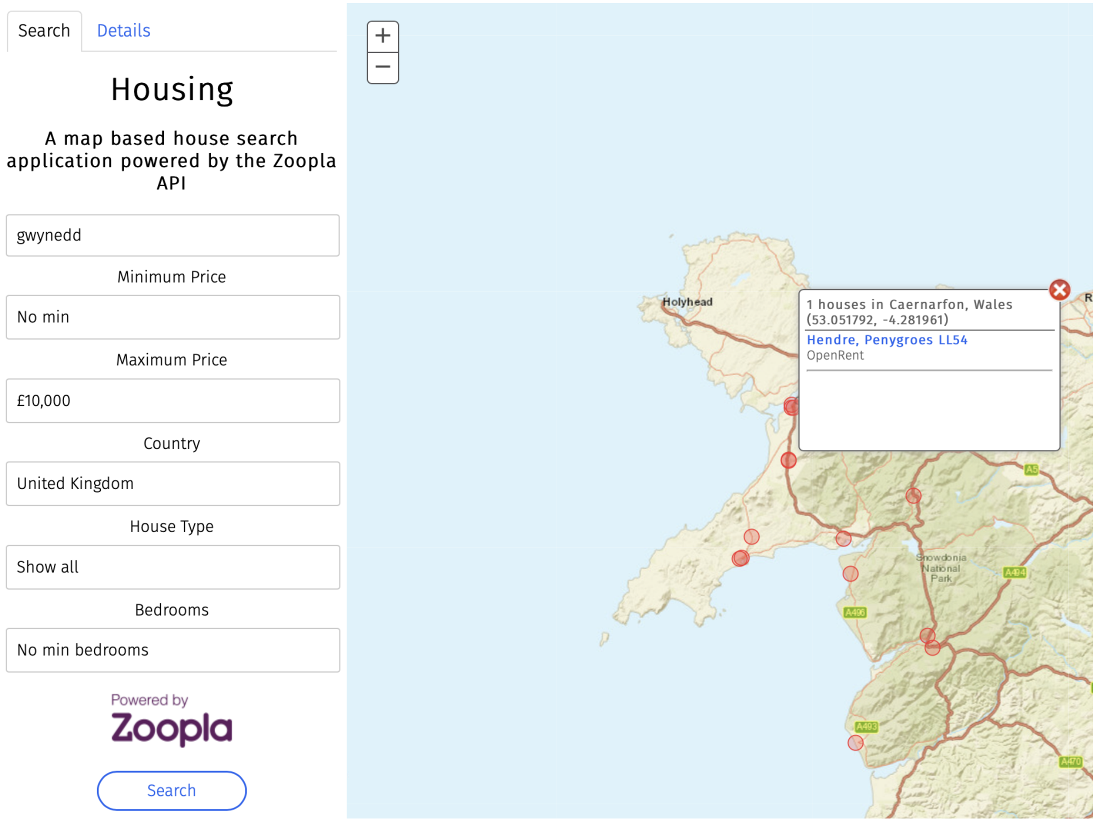
The results from Beta I prototype were encouraging that using an API with structured data results was a good idea and for Beta II, the consideration of creating our own web mining software was important for the project moving forward to move away from the reliability of other external sources.

The Beta I prototype is split into five technical parts, as shown in the figure below:

Beta I Prototype	
Category	**Description**
Data Input	The data used is from a range of housing and job websites using public APIs. The APIs have structured data using json output. Additionally regular expressions was explored to extract website elements.
Data Output	The data output provides structured data of housing and jobs.
Interaction	The Beta I prototype allows the user to explore an interactive map by dragging and zooming into out of different areas of the UK. The user fills out a form and submits it for the data to be processed and returned.
Visualisation	The user is presented with a form on the left side of a webpage and the map loaded to the right side of the page. The results are displayed with circular pins distinctive with different colours and upon pressing, info windows are displayed with the returned results.
Storage	The results of the API are live and not stored. The regular expressions used on the housing and job websites are saved and placed into a database which is then linked to the results.

Table 8.2: Beta I Overview of technical elements.

8.2.1 Data Input



Housing

A map based house search application powered by the Zoopla API

gwynedd

Minimum Price: No min

Maximum Price: £10,000

Country: United Kingdom

House Type: Show all

Bedrooms: No min bedrooms

1 houses in Caernarfon, Wales
(53.051792, -4.281961)

Hendre, Penygroes LL54
OpenRent

Figure 8.3: Beta I Input and Output

The Beta I prototype uses data retrieved from public free APIs for housing and job websites. In order to query the API itself, a homepage had to be created as displayed in 8.3. To query the API, the search field is filled by the user. As the API was structured, it was ready to be output in any way the developer wanted.

Additionally, the user would enter a query and the results would be shown to them in a text format. This was an additional step towards the end of the Beta I prototype required when the two APIs that were used became pay to use.

The lessons learnt from the Data Input solution from the Beta I prototype were that compared to the Alpha prototype, the ability to have further searching criteria (i.e. minimum/maximum price, house type and bedrooms) allowed for the user to have a better experience expressing what they would want to find. The final prototype also needed to have integration of jobs.

8.2.2 Data Output

The Beta I prototype provides results to the user on the same page when they press the search button. The API is queried and a set of results is given in a json format. These results have then been placed into pins with infowindows as we felt to keep it clean and easy to use.

The lessons learnt from the Data Output solution from Beta I prototype were that the results provided clear and concise results related to the data input. The next challenge for the final prototype were to include jobs into the form submission.

8.2.3 Storage

The results for the Beta I are live and not stored coming from the two APIs for housing and jobs. This was done for evaluation purposes so that the accuracy, precision and recall could be precisely calculated. The decision was taken to display these non-live results for the Beta II prototype which then would not require using the API which had become pay to use.

The lessons learnt from the Storage solution for the Beta I prototype were that having live data being used instead of storing it allowed for the results to be dynamic and not static. It also provided the ability to check accuracy, precision and recall metrics to evaluate the quality of the API data. At this stage the APIs were free to use and the restrictions were limited.

8.2.4 Interaction

The interaction for the Beta I prototype are all on one page, meaning the map and the search field are together. The user has the ability to zoom in and out, providing a more detailed look at the local area of what has been searched. Additionally pins were used to highlight the location of the housing. Info Windows were used to contain a wide range of information about the relevant listing.

The lessons learnt from the Interaction solution for Beta I prototype were to use less information in the Info Windows and having better pins to identify housing and jobs.

8.2.5 Visualisation

The visualisation for the Beta I prototype uses a map to display the results on a map and use pins with information windows to display the content at the relevant latitude and longitude coordinates. The lessons learnt from the Visualisation solution for Beta I was that users preferred to have the map results on separate pages.

8.3 Beta II Prototype

The Beta II prototype was the final prototype produced for this thesis. The prototype uses methods used in Beta I and has a database of static listings of housing and jobs. Beta II led from the Beta I prototype and has similar functionalities for searching and displaying the results. The main difference between the prototypes is that the search form and map are separated on two pages.

The final prototype used a static database which has many housing and job listings. When the user submits a search, it queries the database and returns the relevant results.

The design decisions made for the final prototype of the Beta II prototype follows the same principles made in the Beta I prototype. The design that was used was agile design, which allowed for quicker designs and development through changes being made incrementally to the Beta I prototype.

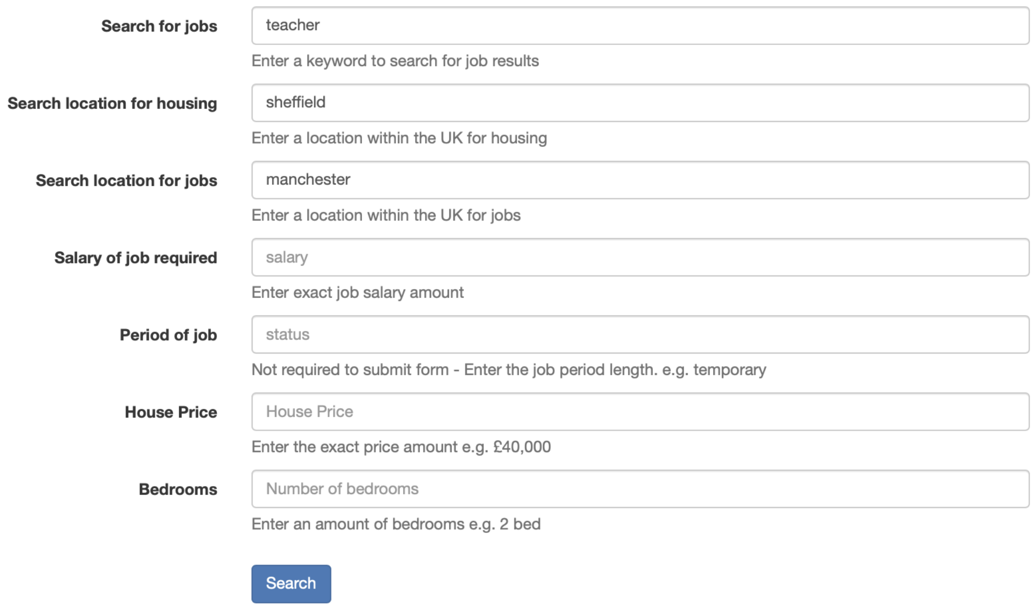
The Beta II prototype moved away from needing to use APIs, but took advantage of the methods used in the Beta I prototype to store a large quantity of listings for housing and jobs.

The Beta II prototype is split into five technical parts, as shown in the figure below:

Beta II Prototype	
Category	**Description**
Data Input	The data used is used is from data extracted from housing and job websites, including APIs and using third party software to gather bulk listings. The Merlin system was also used of tagging done with housing and job data.
Data Output	The data output is structured data that has been saved into SQL database, the input is queried from the user and the results are retrieved from the database.
Interaction	The Beta II prototype allows the user to explore an interactive map by dragging and zooming into out of different areas of the UK. The user fills out a form and submits it for the data to be processed and returned.
Visualisation	The user is presented with a form on the homepage and upon submission of the form, a results page with a map is presented to the user with the results. The results are displayed with circular pins distinctive with different colours and upon pressing, info windows are displayed with the returned results.
Storage	The results of the data that has been processed is stored into an SQL database which is called when the user submits the form. The data used is static and not live.

Table 8.3: Beta II Overview of technical elements.

8.3.1 Data Input



Search for jobs: teacher
Enter a keyword to search for job results

Search location for housing: sheffield
Enter a location within the UK for housing

Search location for jobs: manchester
Enter a location within the UK for jobs

Salary of job required: salary
Enter exact job salary amount

Period of job: status
Not required to submit form - Enter the job period length. e.g. temporary

House Price: House Price
Enter the exact price amount e.g. £40,000

Bedrooms: Number of bedrooms
Enter an amount of bedrooms e.g. 2 bed

Figure 8.4: Beta II Prototype Design — Homepage.

The Beta II prototype uses data extracted from APIs, third party software, the Merlin system and other created scripts that have extracted website data from housing and jobs. The data used is static.

The data was chosen from housing and job websites in the UK. Using custom made scripts was then used to extract the text from the websites, this was done to move away from the now paid for API's. The Merlin system was then used with different models to organise the different types of text that needs to be classified, such as price, description and title. The Merlin system used ground truth data created from samples of job and housing listings to do a comparison to see how well the output performed. The models are then trained based on the text files for the different categories.

There have been many lessons learnt in the final Beta II prototype which has evolved substantially the first two prototypes. In figure 8.4, the user was provided with a further enhanced search compared to the previous prototype. The search input was expanded to include salary, house price and bedrooms.

8.3.2 Data Output

The Beta II prototype provides results to the user on a separate page compared to Beta I. The results appear on a map and the database is called with the results. These results have then been placed into pins with info windows. The data was put into different columns within a table on database, with the relevant sections of a job or house e.g., job title, price, description which is then retrieved and displayed.

The lessons learnt from the data output solution that the results provided a further detailed result related to the data input. This solution included both jobs and housing on the form submission.

8.3.3 Storage

The Beta II has a database that has stored many listings of housing and jobs. As the APIs turned to being paid for during the development of these prototypes, using a database to store queried results allowed for less frequent calls to the API overall. The results are static in this final prototype.

During this project there had been blockages through free APIs becoming paid and their need to move away from a need of services available through APIs. The project involved exploring different ways of gathering these results and through the Merlin system, this performed well to be stored in the database in a structured way.

The lesson learnt for the Storage solution for the final prototype is that although there is a storage mechanism for the final prototype, it was the best way forward for the company partners to keep this prototype more affordable with storing the results, instead of constantly calling an API.

8.3.4 Interaction

The interaction for the Beta II prototype was split into two web pages. The search form was initially presented to the user and upon submitting the form, the results were then presented. This did provide a split between the company partners and from the evaluations made, as there was slightly more preference for having the interaction that was performed in Beta I. The company partners wanted the ability to express more of a homepage for the future and not having the map as it were in Beta I.

The lesson learnt for the Interaction solution for this final prototype was having better defined pins to identify the different housing and jobs. The interaction from the company has been key to the interaction and visualisation of the prototypes.

8.3.5 Visualisation

The Visualisation for the Beta II prototype followed similar suit to the Beta I, the only difference being the separation of the search form and the results into two web pages. The visualisation overall was simple but effective and allowed the ability for future developments on the prototype.

The lesson learnt for the Visualisation solution for this prototype was having the ability to adapt to user feedback. In future development would focus on the ability of putting different visualisations into the output to allow different options to the user.

8.4 Summary and Discussion

The journey throughout the different prototypes has provided many lessons. Taking the approach of separating out the work provided into prototypes allows the developer to develop and progress at a steady pace. It gives the insight of the need to come up with designs first before jumping straight into the implementation of the work.

The development of each of the prototypes have come from the reflection of the evaluations taken and comments received. Future development would take different approaches of design so the user could have a choice of options to view the data input and output differently.

The different approaches used for the design and implementation has allowed flexibility. Using a mixture of agile development, quick and dirty and keep it simple methodologies have allowed for different approaches to be taken with the work.

The ability to communicate with the company partners and have frequent meetings over work allowed for frequent amendments and learning to new directions.

Following through a software engineering process, it is important for the company partners to provide continuing guidance, with important insights and lessons learnt in order to further develop the work.

Chapter 9

Conclusions and Future Work

This chapter summarises this thesis and the conclusions. The next section will summarise the main work and the findings. The following section revisits the aim and objectives and research questions from the first chapter to see if they have been met. The final two sections explore the limitations of the current work and then the future work.

9.1 Summary

The concept of creating a jobs and housing website was explored and three different prototypes created: Alpha prototype, Beta I prototype and Beta II prototype. A system was also created for web mining of job and housing information using PPM called Merlin along with a comparison with an existing state of the art named entity recognition software spaCy. The results were significantly better using Merlin compared to spaCy in terms of performance and training time. A discussion of the lessons learnt has been provided concerning the design, development and process of the Alpha, Beta I and Beta II prototypes.

9.2 Review of Aim & Objectives

In the first chapter, the aims and objectives were set out:

To create a web-based service that will allow users to search within a given geographical region both for accommodation and employment opportunities.

The specific objectives of this thesis were:

- *Produce a literature review of web services and technologies that provide capabilities related to the proposed web service.*

- *Develop and evaluate a named entity tagger relating to jobs and accommodation that is able to identify the unstructured text of HTML source code and is able to identify and tag the unstructured data appropriately.*

- *Compare the performance of the created named entity tagger to the performance of an existing state of the art named entity recognition software.*

- *Design and implement prototypes using a web-based geographical search interface that enables filtering of search criteria relating to jobs and accommodation using dynamic filters.*

- *Evaluate the usability and effectiveness of the different web-based interfaces.*

The first objective was achieved in chapter 3. A review of existing web services on housing and job websites was conducted, including features and services and a discussion on existing web services capabilities.

The second objective was achieved in chapter 4, creating a web mining system for job and housing information using PPM called Merlin.

The third objective was achieved in chapter 4, which conducted a comparison of the same housing and job data on an existing state of the art named entity recognition software called spaCy.

The fourth objective and fifth objective was achieved in chapters 5, 6 and 7. These chapters discuss the design, implementation and evaluation of the prototypes created.

9.3 Review of Research Questions

The research questions designed for this study were the following:

- *What is the best way of extracting and then fusing information obtained from housing and job sites such as sites like Rightmove and Zoopla into a singular web-based platform?*

- *What is an effective interface design for such a platform? e.g., Would a geographical based interface be more appealing to potential clients for such a service?*

The first research question was addressed in chapter 6. The implementation chapter explored existing vendor APIs and developing effective methods for scraping web content involving accommodation and jobs alongside other resources. The answers to the research question for this are that the best ways of fusing information obtained were a mixture of regular expressions and different libraries that could retrieve specific elements.

The second research question was addressed in chapters 5, 6 and 7. The design methodologies used and collaborative discussion on combining housing and jobs listing onto one website using a map-based interface was achieved. The evaluation of the prototypes against users preferred the way presented on being able to see a map compared to the traditional list and grid search results. The answers to the research question for this are that an effective interface design for a platform being a geographically based interface would be more appealing to users for searching and exploring quickly across a map as discussed in sections 6.6 and 7.4.

9.4 Limitations of the Work

We have encountered a few limitations whilst creating the prototypes. These limitations are discussed below:

- The prototypes created use static information, which is a snapshot of data from a period of time. This means that the data is not crawled dynamically and updated with the latest housing and job listings.

- The methods created in the prototypes are not sustainable to futureproof external factor changes, meaning if housing or job websites were to change their design, the methods of retrieval would break. Additionally, if websites were to adopt new blocking techniques, this could make it challenging to retrieve results. To make this more sustainable in the long term, a partnership with the relevant companies would be necessary. If these companies were to make changes to their systems, knowing in advance would help a developer more quickly adapt to those changes.

- The location of job listings is limited to what is being provided by the search listings websites, meaning the results for jobs are broader to an area than a specific street.

9.5 Future Work

In this thesis, several potential directions for future work have been identified that could significantly enhance the scope and efficacy of the research. Firstly, incorporating points of interest, such as cafes and shops, into the queried map overlay would provide users with a more comprehensive understanding of the area in question. Additionally, integrating crime and education statistics into the system could further enrich the user's experience by offering a broader range of relevant data. Secondly, refining the text processing methodology to eliminate noise from housing and job information might lead to improved results for the Merlin system's classification performance. Thirdly, leveraging the data utilised in training the Merlin system could offer valuable opportunities to annotate larger corpora, thereby serving as a ground truth for training the PPM models, ultimately enhancing their performance. Lastly, expanding the application of the PPM approach to Named Entity Recognition (NER) to encompass other types of web-based data could potentially lead to novel insights and improved understanding of various data sources, thus contributing to the advancement of this field.

References

[1] T. Agata, M. Nozue, N. Hattori and S. Ueda, 'A measure for evaluating search engines on the world wide web: Retrieval test with expected search length,' *Library and Information Science*, no. 37, pp. 1–11, 1997 (p. 11).

[2] C. Ahlberg, 'Spotfire: An information exploration environment,' *ACM SIGMOD Record*, vol. 25, no. 4, pp. 25–29, 1996 (p. 137).

[3] C. Ahlberg and E. Wistrand, 'Ivee: An information visualization and exploration environment,' in *Proceedings of Visualization 1995 Conference*, IEEE, 1995, pp. 66–73 (p. 137).

[4] N. Ahmed and W. J. Teahan, 'Using compression to find interesting one-dimensional cellular automata,' *Complex & Intelligent Systems*, vol. 6, no. 1, pp. 123–146, 2020 (p. 56).

[5] M. M. Alamri and W. J. Teahan, 'Automatic correction of arabic dyslexic text,' *Computers*, vol. 8, no. 1, p. 19, 2019 (p. 58).

[6] S. Alkahtani, W. Liu and W. J. Teahan, 'A new hybrid metric for verifying parallel corpora of arabic-english,' *arXiv preprint arXiv:1502.03752*, 2015 (p. 59).

[7] I. S. Alkhazi and W. J. Teahan, 'Classifying and segmenting classical and modern standard arabic using minimum cross-entropy,' *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 4, 2017 (p. 59).

[8] A. Almahdawi and W. J. Teahan, 'Emotion recognition in text using ppm,' in *International Conference on Innovative Techniques and Applications of Artificial Intelligence*, Springer, 2017, pp. 149–155 (p. 59).

[9] A. J. Almahdawi and W. J. Teahan, 'A new Arabic dataset for emotion recognition,' in *Intelligent Computing-Proceedings of the Computing Conference*, Springer., 2019, pp. 200–216 (pp. 25, 72).

[10] M. Altamimi and W. J. Teahan, ‘Gender and authorship categorisation of arabic text from twitter using PPM,’ *International Journal of Computer Science and Information Technologies*, vol. 9, pp. 131–140, 2017 (p. 58).

[11] P. M. Andersen, P. J. Hayes, S. P. Weinstein, A. K. Huettner, L. M. Schmandt and I. B. Nirenburg, ‘Automatic extraction of facts from press releases to generate news stories,’ in *Third Conference on Applied Natural Language Processing*, Trento, Italy: Association for Computational Linguistics, Mar. 1992, pp. 170–177. DOI: 10.3115/974499.974531 (p. 23).

[12] R. K. Ando, T. Zhang and P. Bartlett, ‘A framework for learning predictive structures from multiple tasks and unlabeled data.,’ *Journal of Machine Learning Research*, vol. 6, no. 11, 2005 (p. 21).

[13] G. L. Andrienko and N. V. Andrienko, ‘Interactive maps for visual data exploration,’ *International Journal of Geographical Information Science*, vol. 13, no. 4, pp. 355–374, 1999 (p. 182).

[14] D. E. Appelt, J. R. Hobbs, J. Bear, D. Israel and M. Tyson, ‘Fastus: A finite-state processor for information extraction from real-world text,’ vol. 93, Jan. 1993, pp. 1172–1178 (p. 20).

[15] A. Arasu, J. Cho, H. Garcia-Molina, A. Paepcke and S. Raghavan, ‘Searching the web,’ *ACM Transactions on Internet Technology (TOIT)*, vol. 1, no. 1, pp. 2–43, 2001 (p. 12).

[16] M. Asahara and Y. Matsumoto, ‘Japanese named entity extraction with redundant morphological analysis,’ in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, Association for Computational Linguistics, 2003, pp. 8–15 (p. 21).

[17] S. Balram and S. Dragicevic, ‘Collaborative geographic information systems: Origins, boundaries, and structures,’ in *Collaborative geographic information systems*, Igi Global, 2006, pp. 1–23 (p. 182).

[18] A. Bangor, P. T. Kortum and J. T. Miller, ‘An empirical evaluation of the system usability scale,’ *Intl. Journal of Human–Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008 (pp. 193, 194).

[19] A. Barua *et al.*, ‘Methods for decision-making in survey questionnaires based on likert scale,’ *Journal of Asian Scientific Research*, vol. 3, no. 1, pp. 35–38, 2013 (p. 165).

[20] J. Bau, E. Bursztein, D. Gupta and J. Mitchell, 'State of the art: Automated black-box web application vulnerability testing,' in *2010 IEEE symposium on security and privacy*, IEEE, 2010, pp. 332–345 (p. 14).

[21] K. Beck, M. Beedle, A. Van Bennekum *et al.*, 'Manifesto for agile software development,' (pp. 3, 25, 27, 28).

[22] D. J. Besemer and P. S. Jacobs, 'Flush: A flexible lexicon design,' in *25th Annual Meeting of the Association for Computational Linguistics*, 1987, pp. 186–192 (p. 20).

[23] E. Bick, 'A named entity recognizer for danish.,' in *LREC*, Citeseer, Lisbon, Portugal: European Language Resources Association, May 2004 (p. 20).

[24] S. Bjork and J. Redstrom, 'Redefining the focus and context of focus+ context visualization,' in *IEEE Symposium on Information Visualization 2000. INFOVIS 2000. Proceedings*, IEEE, 2000, pp. 85–89 (p. 182).

[25] S. Brin and L. Page, 'The anatomy of a large-scale hypertextual web search engine,' *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998 (p. 13).

[26] J. Brooke, 'System usability scale (sus): A quick-and-dirty method of system evaluation user information,' *Reading, UK: Digital Equipment Co Ltd*, vol. 43, pp. 1–7, 1986 (pp. 191, 192).

[27] J. Brooke, 'Sus: A "quick and dirty" usability,' *Usability evaluation in industry*, vol. 189, 1996 (p. 191).

[28] J. Carifio and R. J. Perla, 'Ten common misunderstandings, misconceptions, persistent myths and urban legends about likert scales and likert response formats and their antidotes,' *Journal of social sciences*, vol. 3, no. 3, pp. 106–116, 2007 (p. 165).

[29] L. D. Catledge and J. E. Pitkow, 'Characterizing browsing strategies in the world-wide web,' *Computer Networks and ISDN systems*, vol. 27, no. 6, pp. 1065–1073, 1995 (p. 13).

[30] E. W. Chang, 'Bidding on trespass: Ebay, inc. v. bidder's edge, inc. and the abuse of trespass theory in cyberspace-law,' *AIPLA QJ*, vol. 29, p. 445, 2001 (p. 17).

[31] Y. Chen, W. Wang, Z. Liu and X. Lin, 'Keyword search on structured and semi-structured data,' in *Proceedings of the 2009 ACM SIGMOD International*

Conference on Management of data, Association for Computing Machinery, 2009, pp. 1005–1010 (p. 35).

[32] M. H. Chignell, J. Gwizdka and R. C. Bodner, 'Discriminating meta-search: A framework for evaluation,' *Information processing & management*, vol. 35, no. 3, pp. 337–362, 1999 (p. 11).

[33] C. W. Choo, B. Detlor and D. Turnbull, 'Information seeking on the web–an integrated model of browsing and searching.,' 1999 (p. 13).

[34] G. G. Chowdhury, *Introduction to modern information retrieval*. Facet publishing, 2010 (p. 10).

[35] J. Cleary and I. Witten, 'Data compression using adaptive coding and partial string matching,' *IEEE transactions on Communications*, vol. 32, no. 4, pp. 396–402, 1984 (pp. 52, 68).

[36] J. G. Cleary and W. J. Teahan, 'Unbounded length contexts for PPM,' *The Computer Journal*, vol. 40, no. 2_and_3, pp. 67–75, 1997 (pp. 52, 63).

[37] A. Cockburn and S. Jones, 'Which way now? analysing and easing inadequacies in www navigation,' *International Journal of Human-Computer Studies*, vol. 45, no. 1, pp. 105–129, 1996 (p. 13).

[38] W. W. Cohen and S. Sarawagi, 'Exploiting dictionaries in named entity extraction: Combining semi-markov extraction processes and data integration methods,' in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, New York: Association for Computing Machinery, 2004, pp. 89–98 (p. 20).

[39] K. Collier, *Agile analytics: A value-driven approach to business intelligence and data warehousing*. Addison-Wesley, 2012 (pp. 3, 25).

[40] W. S. Cooper, 'Expected search length: A single measure of retrieval effectiveness based on the weak ordering action of retrieval systems,' *American documentation*, vol. 19, no. 1, pp. 30–41, 1968 (p. 11).

[41] M. Costantino and P. Coletti, *Information extraction in finance*. WIT Press, 2008, vol. 8 (p. 23).

[42] J. Cowie and W. Lehnert, 'Information extraction,' *Communications of the ACM*, vol. 39, no. 1, pp. 80–91, 1996 (p. 23).

[43] S. L. Davis, 'American civil liberties union, et. al. v. janet reno: American library association, inc., et. al. v. united states department of justice, et. al., 929

f. supp. 824 (ed pa. 1996),' *Circles: Buffalo Women's Journal of Law and Social Policy*, vol. 5, no. 1, p. 95, 1997 (p. 16).

[44] G. DeJong, 'Skimming stories in real time: An experiment in integrated understanding (technical report yale/dcs/tr158),' Tech. Rep., 1979 (p. 20).

[45] A. Doupé, M. Cova and G. Vigna, 'Why Johnny can't pentest: An analysis of black-box web vulnerability scanners,' in *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, Springer, 2010, pp. 111–131 (p. 14).

[46] M. G. Dyer and U. Zernik, 'Encoding and acquiring meanings for figurative phrases,' in *24th Annual Meeting of the Association for Computational Linguistics*, Citeseer, 1986, pp. 106–111 (p. 20).

[47] T. E. Endres, 'Advantages of rapid prototyping,' SAE Technical Paper, Tech. Rep., 1999 (p. 3).

[48] H. Erdogan, 'Sequence labeling: Generative and discriminative approaches,' Proc. 9th Int. Conf. Mach. Learn. Appl., 2010, pp. 1–132 (p. 19).

[49] D. Fallows, 'Search engine use,' 2008 (p. 12).

[50] C. Faloutsos, 'Access methods for text,' *ACM Computing Surveys (CSUR)*, vol. 17, no. 1, pp. 49–74, 1985 (p. 12).

[51] J. Ferreira, J. Noble and R. Biddle, 'Agile development iterations and UI design,' in *Agile 2007 (AGILE 2007)*, IEEE, 2007, pp. 50–58 (p. 133).

[52] C. Ferris and J. Farrell, 'What are web services?' *Communications of the ACM*, vol. 46, no. 6, p. 31, 2003 (p. 15).

[53] S. Fine, Y. Singer and N. Tishby, 'The hierarchical hidden markov model: Analysis and applications,' *Machine learning*, vol. 32, no. 1, pp. 41–62, 1998 (p. 21).

[54] M. Fleischman and E. Hovy, 'Fine grained classification of named entities,' in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, Association for Computational Linguistics, 2002, pp. 1–7 (p. 20).

[55] G. D. Forney, 'The viterbi algorithm,' *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973 (p. 69).

[56] M. Fowler, J. Highsmith *et al.*, 'The agile manifesto,' *Software development*, vol. 9, no. 8, pp. 28–35, 2001 (p. 26).

[57] W. B. Frakes and R. Baeza-Yates, *Information retrieval: data structures and algorithms*. Prentice-Hall, Inc., 1992 (p. 10).

[58] A. Fujii, M. Iwayama and N. Kando, ‘Introduction to the special issue on patent processing,’ *Information Processing & Management*, vol. 43, no. 5, pp. 1149–1153, 2007 (p. 11).

[59] U. Gasser, ‘Regulating search engines: Taking stock and looking ahead,’ *Yale JL & Tech.*, vol. 8, p. 201, 2005 (pp. 16, 17).

[60] R. Gorwa and D. Guilbeault, ‘Unpacking the social media bot: A typology to guide research and policy,’ *Policy & Internet*, vol. 12, no. 2, pp. 225–248, 2020 (p. 17).

[61] R. Grishman and R. Kittredge, *Analyzing language in restricted domains: sublanguage description and processing*. Psychology Press, 2014 (p. 20).

[62] R. Grishman and B. Sundheim, ‘Message understanding conference-6: A brief history,’ in *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996 (p. 19).

[63] M. Gupta and M. Bendersky, ‘Information retrieval with verbose queries,’ in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Now Publishers, Inc., 2015, pp. 1121–1124 (p. 10).

[64] S. Gupta, ‘A survey on search engines,’ *Journal for Research Volume*, vol. 2, no. 11, 2017 (p. 15).

[65] G. Gürkaynak, İ. Yılmaz and D. Durlu, ‘Understanding search engines: A legal perspective on liability in the internet law vista,’ *Computer Law & Security Review*, vol. 29, no. 1, pp. 40–47, 2013 (pp. 15, 16).

[66] M. R. Henzinger, R. Motwani and C. Silverstein, ‘Challenges in web search engines,’ in *ACM SIGIR Forum*, ACM New York, NY, USA, vol. 36, 2002, pp. 11–22 (pp. 13, 14).

[67] J. R. Hobbs, ‘The finite string newsletter: Site report: Another from the darpa series, overview of the tacitus project,’ *Computational Linguistics*, vol. 12, no. 3, 1986 (p. 20).

[68] C. Hölscher and G. Strube, ‘Web search behavior of internet experts and newbies,’ *Computer networks*, vol. 33, no. 1-6, pp. 337–346, 2000 (p. 13).

[69] P. G. Howard, *The design and analysis of efficient lossless data compression systems*. Brown University, 1993 (p. 54).

[70] S. Huston and W. B. Croft, ‘Evaluating verbose query processing techniques,’ in *Proceedings of the 33rd international ACM SIGIR conference on Research and*

development in information retrieval, Association for Computing Machinery, 2010, pp. 291–298 (pp. 10, 11).

[71] M. Iwayama, A. Fujii, N. Kando and A. Takano, 'NTCIR patent: A test collection for patent retrieval/classification,' in *Proceedings of Patent Retrieval, workshop of the 23rd international ACM SIGIR conference on Research and development in information retrieval*, 2000 (p. 11).

[72] S. Jamieson, 'Likert scales: How to (Ab)use them?' *Medical education*, vol. 38, no. 12, pp. 1217–1218, 2004 (p. 165).

[73] B. J. Jansen, A. Spink, J. Bateman and T. Saracevic, 'Real life information retrieval: A study of user queries on the web,' in *ACM SIGIR Forum*, ACM New York, NY, USA, vol. 32, 1998, pp. 5–17 (p. 11).

[74] R. Jiang, R. E. Banchs and H. Li, 'Evaluating and combining name entity recognition systems,' in *Proceedings of the Sixth Named Entity Workshop*, 2016, pp. 21–27 (p. 89).

[75] N. R. Al-Kazaz, S. A. Irvine and W. J. Teahan, 'An automatic cryptanalysis of transposition ciphers using compression,' in *International conference on cryptology and network security*, Springer, 2016, pp. 36–52 (pp. 58, 59).

[76] D. V. Khmelev and W. J. Teahan, 'A repetition based measure for verification of text collections and for text categorization,' in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, ACM, 2003, pp. 104–110 (p. 58).

[77] B. Kleinberg, M. Mozes, A. Arntz and B. Verschuere, 'Using named entities for computer-automated verbal deception detection,' *Journal of forensic sciences*, vol. 63, no. 3, pp. 714–723, 2018 (p. 89).

[78] K. Knight, 'Mining online text,' *Communications of the ACM*, vol. 42, no. 11, pp. 58–61, 1999 (p. 22).

[79] V. Krotov and L. Silva, 'Legality and ethics of web scraping,' 2018 (p. 17).

[80] K. K. Lavania, S. Jain, M. K. Gupta and N. Sharma, 'Google: A case study (web searching and crawling),' *International Journal of Computer Theory and Engineering*, vol. 5, no. 2, p. 337, 2013 (pp. 11, 12).

[81] S. Lawrence and C. L. Giles, 'Searching the world wide web,' *Science*, vol. 280, no. 5360, pp. 98–100, 1998 (p. 12).

[82] J. Li, A. Sun, J. Han and C. Li, 'A survey on deep learning for named entity recognition,' *IEEE Transactions on Knowledge and Data Engineering*, 2020 (p. 19).

[83] R. Likert, 'A technique for the measurement of attitudes.,' *Archives of psychology*, 1932 (p. 165).

[84] W. Liu, Z. Chang and W. Teahan, 'Ppm compression-based method for english-chinese bilingual sentence alignment,' in *2nd international Conference on Statistical Language and Speech Processing (SLSP 2014), 14-16 October 2014, Grenoble, France*, 2014 (p. 60).

[85] S. L. Lytinen and A. Gershman, 'Atrans automatic processing of money transfer messages.,' in *AAAI*, vol. 86, Citeseer, 1986, pp. 1089–1093 (p. 20).

[86] A. M. MacEachren, F. P. Boscoe, D. Haug and L. W. Pickle, 'Geographic visualization: Designing manipulable maps for exploring temporally varying georeferenced statistics,' in *Proceedings IEEE symposium on information visualization (Cat. No. 98TB100258)*, IEEE, 1998, pp. 87–94 (p. 136).

[87] A. M. MacEachren and M.-J. Kraak, *Exploratory cartographic visualization: Advancing the agenda*, 1997 (p. 137).

[88] A. Al-Mahdawi, 'Automatic emotion recognition in english and arabic text,' Ph.D. dissertation, Bangor University, 2019 (pp. 53, 54).

[89] M. Mahoney, *Large text compression benchmark*, 2011 (p. 54).

[90] M. Mahoui, W. J. Teahan, A. K. Thirumalaiswamy Sekhar and S. Chilukuri, 'Identification of gene function using prediction by partial matching (ppm) language models,' in *Proceedings of the 17th ACM conference on information and knowledge management*, 2008, pp. 779–786 (p. 60).

[91] C. D. Manning, P. Raghavan and H. Schütze, *Introduction to Information Retrieval*. Cambridge university press, 2008 (p. 24).

[92] J. Manning, 'In vivo coding,' *The international encyclopedia of communication research methods*, vol. 24, pp. 1–2, 2017 (p. 170).

[93] R. C. Martin, *Agile software development: principles, patterns, and practices*. Prentice Hall, 2002 (p. 133).

[94] V. M. Matthew, 'Adaptive weighing of context models for lossless data compression,' *Florida Institute of Technology CS Dept, Technical Report CS-2005-16*, 2005 (p. 54).

[95] K. T. Maxwell, ‘Term selection in information retrieval,’ Ph.D. dissertation, University of Edinburgh, 2016 (p. 11).

[96] S. M. Mirtaheri, M. E. Dinçktürk, S. Hooshmand, G. V. Bochmann, G.-V. Jourdan and I. V. Onut, ‘A brief history of web crawlers,’ *arXiv preprint arXiv:1405.0749*, 2014 (p. 14).

[97] A. Moffat, ‘Implementing the ppm data compression scheme,’ *IEEE Transactions on communications*, vol. 38, no. 11, pp. 1917–1921, 1990 (pp. 52, 53, 68).

[98] V. R. Moffat, ‘Regulating search,’ *Harv. JL & Tech.*, vol. 22, p. 475, 2008 (p. 15).

[99] B. Mohit, ‘Named entity recognition,’ in *Natural language processing of semitic languages*, Springer, 2014, pp. 221–245 (p. 19).

[100] I. Montani, M. Honnibal, M. Honnibal, S. Landeghem, A. Boyd, H. Peters *et al.*, ‘Spacy: Industrial-strength natural language processing in python,’ *Zenodo*, 2021 (p. 89).

[101] R. J. Mooney and R. Bunescu, ‘Mining knowledge from text using information extraction,’ *ACM SIGKDD explorations newsletter*, vol. 7, no. 1, pp. 3–10, 2005 (p. 22).

[102] M. R. Morris, J. Teevan and K. Panovich, ‘A comparison of information seeking using search engines and social networks,’ in *Fourth International AAAI Conference on Weblogs and Social Media*, 2010 (p. 12).

[103] S. Munzert, C. Rubba, P. Meißner and D. Nyhuis, *Automated data collection with R: A practical guide to web scraping and text mining*. John Wiley & Sons, 2014 (p. 18).

[104] D. Nadeau and S. Sekine, ‘A survey of named entity recognition and classification,’ *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007 (pp. 19, 20).

[105] R. Navarro-Prieto, M. Scaife and Y. Rogers, ‘Cognitive strategies in web searching,’ in *Proceedings of the 5th Conference on Human Factors & the Web*, 1999, pp. 43–56 (p. 13).

[106] T. Nemoto and D. Beglar, ‘Likert-scale questionnaires,’ JALT 2013 conference proceedings, 2014, pp. 1–8 (p. 165).

[107] D. L. Olson and D. Delen, *Advanced data mining techniques*. Springer Science & Business Media, 2008 (p. 25).

[108] C. Oppenheim, A. Morris, C. McKnight and S. Lowley, 'The evaluation of www search engines,' *Journal of documentation*, 2000 (p. 11).

[109] E. Partalidou, E. Spyromitros-Xioufis, S. Doropoulos, S. Vologiannidis and K. Diamantaras, 'Design and implementation of an open source greek pos tagger and entity recognizer using spacy,' in *IEEE/WIC/ACM International Conference on Web Intelligence*, 2019, pp. 337–341 (p. 89).

[110] S. M. Peltola, F. P. Melchels, D. W. Grijpma and M. Kellomäki, 'A review of rapid prototyping techniques for tissue engineering purposes,' *Annals of medicine*, vol. 40, no. 4, pp. 268–280, 2008 (p. 3).

[111] D. M. Powers, 'Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation,' *arXiv preprint arXiv:2010.16061*, 2020 (p. 24).

[112] Y. Qi, R. Collobert, P. Kuksa, K. Kavukcuoglu and J. Weston, 'Combining labeled and unlabeled data with word-class distribution learning,' in *Proceedings of the 18th ACM conference on Information and knowledge management*, 2009, pp. 1737–1740 (p. 21).

[113] L. F. Rau, 'Extracting company names from text,' in *[1991] Proceedings. The Seventh IEEE Conference on Artificial Intelligence Application*, IEEE, vol. 1, 1991, pp. 29–32 (p. 20).

[114] B. R. Rich, 'Clarence leonard (kelly) johnson 1910-1990: A biographical memoir,' *Biographical Memoirs*, vol. 67, pp. 221–241, 1995 (p. 111).

[115] J. C. Roberts, C. Headleand and P. D. Ritsos, 'Sketching designs using the five design-sheet methodology,' *IEEE transactions on visualization and computer graphics*, vol. 22, no. 1, pp. 419–428, 2015 (pp. 99–101).

[116] J. C. Roberts, C. J. Headleand and P. D. Ritsos, *Five Design-Sheets: Creative Design and Sketching for Computing and Visualisation*. Springer, 2017 (p. 100).

[117] J. C. Roberts, C. Headleand and P. D. Ritsos, 'Sketching designs using the five design-sheet methodology,' *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 419–428, 2016. DOI: `10.1109/TVCG.2015.2467271` (p. 99).

[118] X. Roche, 'Copying websites,' in *Web Archiving*, Springer, 2006, pp. 93–114 (p. 52).

[119] A. Roy, 'Recent trends in named entity recognition (NER),' *arXiv preprint 2101.11420*, arXiv–2101, 2021 (pp. 19, 21).

[120] M. S. Ryan and G. R. Nudd, 'The viterbi algorithm,' Tech. Rep., 1993 (p. 69).

[121] N. Sager, C. Friedman and M. S. Lyman, *Medical language processing: computer management of narrative data*. Addison-Wesley Longman Publishing Co., Inc., 1987 (p. 23).

[122] P. Sahoo and R. Parthasarthy, 'An efficient web search engine for noisy free information retrieval.,' *The International Arab Journal of Information Technology.*, vol. 15, no. 3, pp. 412–418, 2018 (p. 8).

[123] D. Salomon and G. Motta, *Handbook of data compression*. Springer, 2010 (p. 54).

[124] G. Salton, 'Automatic text processing. addison welsley,' *Reading, Massachusetts*, vol. 4, 1989 (p. 12).

[125] S. Sarawagi *et al.*, 'Information extraction,' *Foundations and Trends® in Databases*, vol. 1, no. 3, pp. 261–377, 2008 (p. 23).

[126] J. Sauro, 'A practical guide to the system usability scale: Background,' *Benchmarks & best practices*, 2011 (p. 193).

[127] F. Sebastiani, 'Machine learning in automated text categorization,' *ACM computing surveys (CSUR)*, vol. 34, no. 1, pp. 1–47, 2002 (p. 22).

[128] F. Sebastiani, 'Text categorization,' in *Encyclopedia of Database Technologies and Applications*, IGI Global, 2005, pp. 683–687 (p. 22).

[129] S. Sekine, R. Grishman and H. Shinnou, 'A decision tree method for finding and classifying names in Japanese texts,' Montre'al, Quebec, Canada: ACL, 1998. [Online]. Available: `https://aclanthology.org/W98-1120` (p. 21).

[130] S. Sekine, K. Sudo and C. Nobata, 'Extended named entity hierarchy.,' in *The International Conference on Language Resources and Evaluation*, European Language Resources Association, May 2002 (p. 21).

[131] T. Seymour, D. Frantsvog, S. Kumar *et al.*, 'History of search engines,' *International Journal of Management & Information Systems (IJMIS)*, vol. 15, no. 4, pp. 47–58, 2011 (p. 14).

[132] X. Shen and C. Zhai, 'Active feedback in ad hoc information retrieval,' in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, 2005, pp. 59–66 (p. 10).

[133] B. Shneiderman, 'Dynamic queries for visual information seeking,' *IEEE software*, vol. 11, no. 6, pp. 70–77, 1994 (p. 136).

[134] C. Silverstein, M. Henzinger, H. Marais and M. Moricz, 'Analysis of a very large altavista query log,' Technical Report 1998-014, Digital SRC, Tech. Rep., 1998 (p. 13).

[135] C. Silverstein, H. Marais, M. Henzinger and M. Moricz, 'Analysis of a very large web search engine query log,' in *ACM SIGIR Forum*, ACM New York, NY, USA, vol. 33, 1999, pp. 6–12 (pp. 11, 13).

[136] D. S. Sirisuriya, 'A comparative study on web scraping,' 2015 (p. 18).

[137] B. Sobel, 'HiQ v. LinkedIn, Clearview AI, and a New Common Law of Web Scraping,' *LinkedIn, Clearview AI, and a New Common Law of Web Scraping*, 2020 (p. 18).

[138] A. Spink, B. J. Jansen, V. Kathuria and S. Koshman, 'Overlap among major web search engines,' *Internet Research*, 2006 (p. 15).

[139] R. K. Srihari, W. Li, T. Cornell and C. Niu, 'Infoxtract: A customizable intermediate level information extraction engine,' *Natural Language Engineering*, vol. 14, no. 1, pp. 33–69, 2008 (p. 23).

[140] S. V. Stehman, 'Selecting and interpreting measures of thematic classification accuracy,' *Remote sensing of Environment*, vol. 62, no. 1, pp. 77–89, 1997 (p. 24).

[141] L. T. Su, H.-l. Chen and X. Dong, 'Evaluation of web-based search engines from the end-user's perspective: A pilot study.,' in *Proceedings of the ASIS Annual Meeting*, ERIC, vol. 35, 1998, pp. 348–61 (p. 11).

[142] J. Suzuki and H. Isozaki, 'Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data,' in *Proceedings of ACL-08: HLT*, 2008, pp. 665–673 (p. 21).

[143] G. Svennerberg, *Beginning Google Maps API 3*. Apress, 2010 (pp. 31, 32).

[144] M.-C. Tang and Y. Sun, 'Evaluation of web-based search engines using user-effort measures,' *Library and Information Science Research Electronic Journal*, vol. 13, no. 2, pp. 1–8, 2003 (p. 11).

[145] T. Tarmom, W. Teahan, E. Atwell and M. A. Alsalka, 'Compression versus traditional machine learning classifiers to detect code-switching in varieties and dialects: Arabic as a case study,' *Natural Language Engineering*, vol. 26, no. 6, pp. 663–676, 2020 (p. 58).

[146] L. Tauscher and S. Greenberg, 'How people revisit web pages: Empirical findings and implications for the design of history systems,' *International Journal of Human-Computer Studies*, vol. 47, no. 1, pp. 97–137, 1997 (p. 13).

[147] W. J. Teahan and K. M. Alhawiti, 'Preprocessing for ppm: Compressing utf-8 encoded natural language text,' *International Journal of Computer Science & Information Technology*, vol. 7, no. 2, p. 41, 2015 (p. 60).

[148] W. J. Teahan and N. O. Aljehane, 'Grammar based pre-processing for ppm,' *Int. J. Comput. Sci. Inf. Secur*, vol. 9, 2017 (p. 60).

[149] W. J. Teahan and D. J. Harper, 'Using compression-based language models for text categorization,' in *Language modeling for information retrieval*, Springer, 2003, pp. 141–165 (p. 58).

[150] W. J. Teahan, 'A compression-based toolkit for modelling and processing natural language text,' *Information*, vol. 9, no. 12, p. 294, 2018 (pp. 52, 59–61, 63, 64, 68, 69, 96).

[151] C. Van Rijsbergen, 'Information retrieval: Theory and practice,' in *Proceedings of the Joint IBM/University of Newcastle upon Tyne Seminar on Data Base Systems*, vol. 79, University of Newcastle-upon-Tyne Computing Laboratory, 1979 (p. 24).

[152] H. Wang, J. Z. Huang, Y. Qu and J. Xie, 'Web services: Problems and future directions,' *Journal of Web Semantics*, vol. 1, no. 3, pp. 309–320, 2004 (p. 15).

[153] M. Q. Wang Baldonado, A. Woodruff and A. Kuchinsky, 'Guidelines for using multiple views in information visualization,' in *Proceedings of the working conference on Advanced visual interfaces*, 2000, pp. 110–119 (p. 137).

[154] Y. Wilks, 'Information extraction as a core language technology,' in *International Summer School on Information Extraction*, Springer, 1997, pp. 1–9 (p. 23).

[155] M. L. Wilson, M. Schraefel and R. W. White, 'Evaluating advanced search interfaces using established information-seeking models,' *Journal of the American Society for Information Science and Technology*, vol. 60, no. 7, pp. 1407–1422, 2009 (p. 37).

[156] I. H. Witten, I. H. Witten, A. Moffat, T. C. Bell, T. C. Bell and T. C. Bell, *Managing gigabytes: compressing and indexing documents and images*. Morgan Kaufmann, 1999 (p. 20).

[157] P. Wu and W. J. Teahan, 'A new PPM variant for chinese text compression,' *Natural Language Engineering*, vol. 14, no. 3, pp. 417–430, 2008 (p. 52).

[158] Z. Xiaojin, ‘Semi-supervised learning literature survey,’ *Computer Sciences TR*, vol. 1530, 2008 (p. 20).