University of Reading

Department of Computer Science

Individual Project - CS3IP16

Automated Data Gathering, Classification and Visualisation Application to Aid Regulatory Affairs' Analysis of Competitive Intelligence in the Pharmaceutical Industry.

Rhian Taylor

24008603

BSc Computer Science with Industrial Year

r.c.taylor@student.reading.ac.uk

Project Supervisor: Dr Pat Parslow

Acknowledgements

I would like to thank Dr Pat Parslow, my supervisor, for the guidance he has given me and the patience he has shown throughout this project. His support and expertise were invaluable and was deeply appreciated.

I would like to thank Ben Willis for his advice regarding web-development and his recommendation to use Bootstrap.

I would like to thank Dr Tom Thorne, my python lecturer, for opening my eyes to the Python programming language. Your lectures demonstrated the applicability of Python to my projects objectives and I learned so much from them that I utilised when building the back-end of this system.

Table of Contents

Chapter 1 Introduction

The aim of this project was to automate the gathering, classification and visualisation of open source drug approval data for the use of Regulatory Affairs teams in the Pharmaceutical industry for competitive intelligence analysis.

This chapter explores the background and motivation of this project, exploring the Pharmaceutical industry and how competitive intelligence is gathered and used. Personal experience of this issue was the primary motivation for this project as it provided insight into a problem that many don't realise exist, let alone were looking to target. The objectives of the project are laid out, summarising the purpose of this project and what it aims to achieve.

1.1 Background

The pharmaceutical industry is one of the most dynamic sectors of world economics [1]. In 2015 the Pharmaceuticals and Biotechnology sector ranked first by overall research and development(R&D) [2], as seen in Table 1. Despite this investment, the sector is ranked 5th for net sales growth effectively illustrating the nature of the Pharmaceutical industry, see Table 2. The market is unstable, and there is intense competition between companies across geographies. The Pharmaceutical industries market follows a differentiated oligopolistic structure [3]. Notably, the products are heterogeneous and, therefore, vary in many ways. Due to this Pharmaceutical companies rely on brand loyalty to make headway in the market.

Table 1: Table showing the percentage of R&D intensity in the top 5 sectors [2].

Sector	*Global R&D intensity (%)*	*EU-608 R&D intensity (%)*	*US-829 R&D intensity (%)*	*Japan-360 R&D intensity (%)*	*RoW- 703 R&D intensity (%)*
Pharmaceuticals & Biotechnology	14.4	13.3	17.1	13.3	12.0

Software & Computer Services	10.1	10.6	13.2	2.1	6.8
Technology Hardware & Equipment	8.0	15.1	9.9	5.2	4.2
Leisure Goods	5.8	3.0	5.8	5.8	6.2
Aerospace & Defence	4.5	6.0	3.2	1.4	6.0

Table 2: Table showing the changes in sales in the top 5 sectors over the course of a year [2].

Sector	*Overall sales change (%)*	*EU-608 Sales change (%)*	*US-829 Sales change (%)*	*Japan-360 Sales change (%)*	*RoW-703 Sales change (%)*
Health Care Equipment & Services	17.5	10.7	20.5	7.9	11.9
Software & Computer Services	9.5	6.5	6.6	12.4	22.4
Automobiles & Parts	5.9	5.6	1.2	7.9	8.6
Technology Hardware & Equipment	4.8	-6.5	6.2	4.7	6.2
Pharmaceuticals & Biotechnology	4.6	3.4	6.5	-2.4	6.5

Business Intelligence (BI) could be defined as the goal to “retrieve, transform, and monitor an organizations data to gain business intelligence” [4] whereas Competitive Intelligence(CI) could be defined as "process of gathering, analysing and interpreting internal and external intelligence on competitors and the competitive environment to guide a company's strategy, planning and tactical decision making in its commercial operations” [5]. Within this paper the term used is Competitive Intelligence, hence known as CI, to refer to a combination of these things; including gathering, analysing, interpreting and monitoring of an organisations data, and the data of is competitors', to gain business intelligence and guide strategy, planning and tactical decision making. Companies acquire CI from two types of sources; primary and secondary. Primary Intelligence refers to publicly unavailable knowledge whereas Secondary Intelligence refers to knowledge that is publicly available such as desk research, hard/online resources. By following the CI cycle, depicted in Figure 1, it is possible for organisations to gather intelligence that informs their KIT/KIQ’s (key intelligence topics/key intelligence questions).

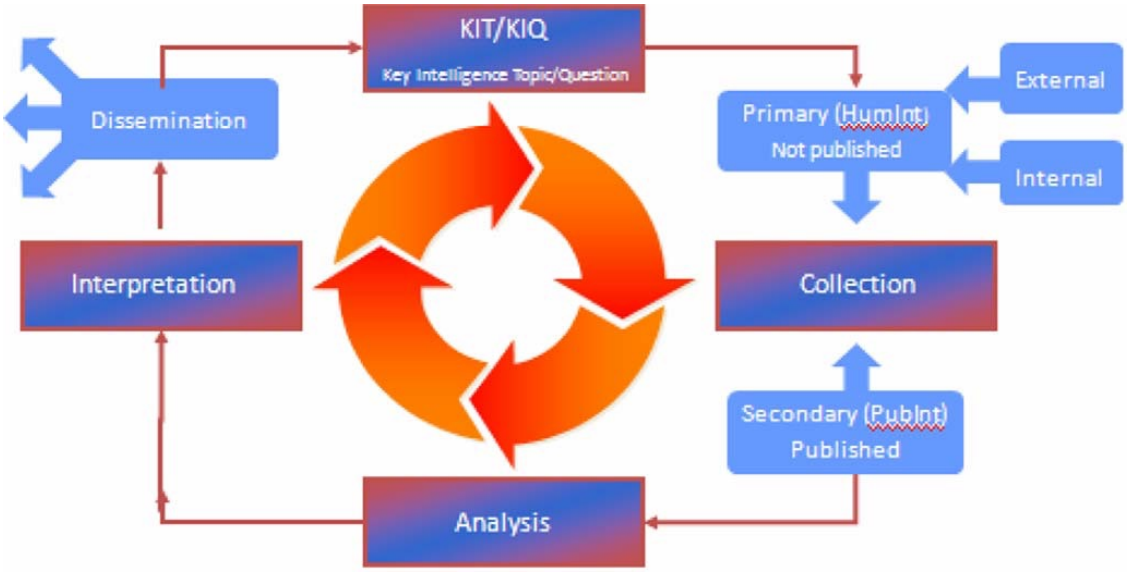


Figure 1: The Competitive Intelligence Cycle [5]

Secondary Intelligence, due to the internet especially, is much more readily available to organisations. The practice of gathering open source data has been in practice for decades [6]. Open Source intelligence, henceforth known as OSINT, is the scanning, finding, gathering, exploitation, validation, analysis and sharing with intelligence seeking clients of publicly available print and digital/electronic data from unclassified, non-secret and ‘grey’

literature [7]. Despite the possibilities OSINT offers organisations, structural problems associated with OSINT data present their own challenges. In order to make use of OSINT organisations have to be able to deal with the form of the data and extracting it in relevant portions, the internet's volatility, source validation as well as the sheer amount of data there would be to process. Furthermore, organisations spend more time gathering this data rather than analysing it.

1.2 Motivation

Pharmaceutical organisations conduct business intelligence gathering regularly and rely heavily on the information gathered to make critical decisions. Previous exposure to the pharmaceutical industry during a year's industrial placement highlighted the dependency on this type of information that is gathered from open source data and compared against company confidential information (CCI). This experience also showed that the process of gathering is not as efficient as it could be; resources are being dedicated to the manual gathering and processing of this data when they could be reallocated to other work if this was automated. Currently, Regulatory Affairs teams are dedicating human resources to the gathering, storing and reporting of competitive intelligence. During the time spent observing a Regulatory Affairs team in a large Pharma, the department kept track of several things. These included international legislation regarding the type of drugs their company is working on, ensuring the marketing and advertising of those drugs follow the requirements and restrictions posed by governing bodies and also keeping track of drug performance within the market to name a few. In this instance "drug performance" means how quickly a drug is approved for use once it has passed the final stages of testing. To track this, currently, an employee would be required to manually search for these dates on websites like the FDA (US Food and Drug Administration) and TGA (Australian Therapeutic Good Administration). They search for specific drugs that they are monitoring, find the date of approval and log this date so that comparisons can be made against their internal data to see how quickly they got to market compared to their competitors across geographies and therapeutic groups. Knowing whether their drug was approved before a competitor's, how much longer that approval took, as well as other factors were of value to the teams as they needed to report their findings to the CEO. This information was relevant in making several

organisation-wide decisions such as where to fund marketing, which drugs needed to be marketed better or even whether a particular drug was worth the marketing investment if a competitor's drug's performance was too far ahead of their own.

This experience raised questions about the efficiency of the current process and how the department could re-distribute its resources and update its processes to improve this. This project has identified the gathering, storing and reporting of competitive intelligence as a process that could be made more efficient and, through automation, could free up resources to perform other important work.

1.3 Aims and Objectives

There are three key elements to this project; data gathering, data classification, and data visualisation. These are the three main milestones for the project, and a fourth would be linking these aspects together to form one comprehensive program. Finally, an interface is needed so that users can interact with the application as the three aspects would be created separately initially. Assessing these milestones generates a set of requirements and related outcomes. This section includes a high-level breakdown of the aforementioned milestones.

1. Gather open source data regarding drug inclusion dates.
    a. Web scraper visits official websites and searches for specific drug information.
    b. Extract data from page sources.
    c. Find drug inclusion/approval date.
    d. Store data.
    e. Classify data into therapeutic groups.
    f. Automatic searches once a month.
        i. An appropriate time for the target website – based on time zone and work week.
2. Classify the gathered data.
    a. Chose an approach.

    b. Define a test set.
    c. Create the model.
    d. Store the results.
3. Store the gathered data for use by the application.
    a. Store data from the data gathering portion of the application.
        i. Store company, drug information, and inclusion date data.
    b. Store user data.
        i. Allow users to log into the application.
        ii. Store company affiliations.
        iii. Affiliations with other users e.g. in the same company.
4. Create a dashboard that visualises the data.
    a. Use charts and graphs to allow users to compare their company competitiveness in a certain market.
        i. Geographic/therapeutic.
    b. Users can interact with the graphics to include/exclude data from the various charts and graphs.
5. Link the three components of the application.
    a. Web scraper can store data.
    b. The dashboard can pull from the stored data.
6. GUI – for user interaction with the application.
    a. Users can view the dashboard and filter information they are seeing.
    b. Users can log in to their account.
    c. Users can get more information about the project.

This list of objectives is used to judge the success of the project and to create the project plan.

Chapter 2 Literature Review

This chapter explores the current research and solutions relevant to this project. Looking at other data gathering, and data analytics solutions provide insight into the market space that this project targets, as well as the approaches used to tackle similar problems. To assign therapeutic areas to each drug in the dataset there is a need for the program to understand the purpose of that drug and what it treats. This information is represented as a string, so the second portion of this chapter explores approaches to natural language processing. As this project is looking to automate real-time data collection for business use, there is a need to understand the impact of automation in the workplace as well as the usefulness of access to real-time data.

2.1 Current Solutions and Applications in HealthCare.

Several data analytics solutions are being offered currently, and many are operating in the healthcare industry. Two examples of this are Splunk and SAS whose software is utilised in various ways.

Splunk software allows the collection and analysis of big data to give insights into operational performance and business results [8]. Currently, Splunk is used in several industries, including healthcare, for which the company provides case studies on how the solution has been implemented. There are three healthcare case studies describing how Splunk has been applied to industry problems regarding data gathering and analytics.

Cerner Corporation is a healthcare software IT company that enables the validation of patient's insurance information and was looking to quickly detect and correct errors to reduce the number of resubmissions and denied claims which delay healthcare delivery and impact the revenue cycle. Through the application Splunk products patient eligibility is monitored in near real-time and engineers can view data streams on Splunk dashboards. The use of configurable dashboards allows pattern detection and performance analysis. [9]

Molina Healthcare is a Fortune 500, multistate health care organisation arranging service delivery and offering information management solutions to those receiving care through

government plans. The Splunk solutions are utilised by Molina to enable fast IT issue resolution through visibility and correlation across IT incidents. Splunk also provides operational intelligence targeting Molina's rapid growth and enabling automation. Through Splunk, Molina employs data mining techniques to better understand why people join and leave their plans, as well as identify trends of sepsis and other issues. [10]

The final Splunk case study is a pharmaceutical company, Recursion Pharmaceuticals, which set a goal to discover treatments for 100 genetic diseases by 2025. Recursion faced difficulties in handling large amounts of time-series data. Recursions machine learning system was integrated with Splunk which monitors and logs operational data to gain insight into laboratory processes to understand correlations as they are happening. Recursion works on several molecules in parallel, using genetic disease models and deep learning algorithms to decide which drugs to pursue further study. Splunk solutions catch anomalies in the automated operations with dashboards displaying quality overtime to the users. [11]

SAS is a world leading data analytics solution employing data mining, statistical analysis, text analytics and optimisation to allow users to access, prepare and model data to improve the return on analytics investment [12]. SAS is also used in the healthcare industry to provide insights into data.

Universitair Medisch Centrum (UMC) Utrecht, a Dutch hospital, uses SAS solutions to proactively treat/prevent infections in premature babies. Premature babies are monitored through a connection to several healthcare devices, and UMC had collected 10 years' worth of patient data that they wished to use to develop models in SAS to answer the question of whether it was possible to proactively treat these premature babies using data analytics. A SAS statistical model was created to support or deny the suspicion of infections, such as sepsis. This model could also prevent unnecessary use of antibiotics, which is costly, as it was shown that 60% of babies were given antibiotics unnecessarily. The SAS model produced was 90% accurate, significantly higher than doctor predictions based on examination and symptoms which is approximately 40%. [13]

SAS was also employed by SMS-oncology to give insights into ongoing clinical trials to enable proactive intervention. Clinical trials are growing in size, complexity, duration and cost; in response, many Pharmaceutical Companies are outsourcing trial to contract research organisations (CROs) like SMS-oncology. Clinical trial data can span tables across hundreds of pages, and therefore it is difficult to glean meaningful insights from the results. Integrating SAS Visual Analytics enabled analyses to be performed and to generate visualisation during the trial, not just at the end, allowing the identifications of trends. Furthermore, SAS made these discoveries much quicker than the manual process that was being used. [14]

2.2 Natural Language Processing

Natural Language Processing (NLP) can be used to solve several types of problems such as:

1. Natural Language Inference
2. Question Answering
3. Semantic Similarity
4. Text Classification

Natural Language Inference (NLI) aims to determine if a given statement semantically entails another given statement [15] inferring whether a problem instance equals another in meaning, if not in the words used. There are 3 types of inferences that can be made; entailment, neutral and contradictory. Figure 2 shows an example where a premise is classified into the three categories of hypothesis mentioned earlier. [16]

Premise
A woman selling bamboo sticks talking to two men on a loading dock.

Hypotheses
Entailment: *There are at least three people on a loading dock.*
Neutral: *A woman is selling bamboo sticks to help provide for her family.*
Contradiction: *A woman is not taking money for any of her sticks.*

Figure 2 NLI example from Bowman's SNLI dataset.

Question answering consists of models predicting the best answer to a question given a context passage which contains the answer [17]. This is done by the model studying the passage and question and evaluating their contextual relationship in order to give the best answer. This model is commonly seen today in the forms of virtual assistants such as Siri and Alexa. [16]

Semantic Similarity can be defined as the measure of conceptual distance between two objects, based on the correspondence of their meaning [18]. There are many applications of semantic similarity such as information retrieval on the internet as well as categorisation and summarization of text. However, these are domain specific and therefore require different algorithms despite the central concept, semantic similarity, being the same. [16]

Finally, text classification is a simpler task compared to the other three, with a goal of automatically classifying text into defined categories [19]. Text classification usually includes preparing the dataset through pre-processing and cleaning and feature engineering which, on a high level, transforms the raw data into feature vectors, counts them and scores them depending on the number of times they appear. Traditionally, many models can be applied to this task. [16]

There are also numerous ways to approach NLP tasks as the unstructured nature of natural language, from both syntactic and semantic points of view, makes it ambiguous and therefore, difficult to work with. One approach is the use of Fuzzy Logic (FL) which can be defined as “a system of reasoning and computation in which the objects of reasoning and computation are classes with unsharp (fuzzy) boundaries.” [20] and can be used to approximate reasoning. FL is a mechanism for associating imprecise values and imprecise propositions, allowing modelling of qualitative reasoning employed by humans, to interpret natural language [21]. Possibilistic Relational Universal Fuzzy (PRUF) can be used to infer meaning in a specific set of natural language premises. It is possible to implement PRUF as a question answer system where the training set is not complete by computing a possibility distribution of the data set. For each instance of natural language in the set the most possible inference is assigned based on the PRUF possibility distribution.

Another approach is to use a model to evaluate a test set and apply the “learned” knowledge to other data sets in order to classify the data. There are 4 main categories of NLP representations [22]:

1. Vector space models
2. Dimensionality reduction techniques
3. Clusters based on distributional similarity
4. Language models

A Vector Space Model (VSM) represents objects, such as text documents, in the form of an algebraic model as vectors of identifiers. These are vectors of words and similar words of multiple degrees of similarity and are used to perform NLP. This can be achieved through using neural networks to learn distributed representations of words [23].

Dimensionality refers to the number of features associated with the text string that is being classified. Dimensionality reduction can be achieved through feature selection; by selecting a subset of features, feature extraction; combining original features into one new feature and term grouping; correlating occurrences of words across a group of data sets [24]. Principal Component Analysis can be used to perform feature extraction whereas the k-means algorithm can be applied to achieve term grouping.

Clustering based on distributional similarity is a method for classifying words based on context. This approach models senses “as probabilistic concepts or clusters c with corresponding cluster membership probabilities p(c|w) for each word w.” By creating a dataset of frequently occurring pairs, x and y, it is possible to classify x according to their distributions as objects of y. Clustering will be used to model element associations. [25] Finally, there are various language models that can be general or task specific all making use of different architectures and algorithms in order to perform NLP. Some make use of neural networks whereas others use classification algorithms such as Naïve Bayes or Linear Regression. An example of this would be the application of a Bayesian model to perform probabilistic language modelling where a string is categorised in terms of a “ probability distribution over all possible strings in the domain” [26].

2.3 Business Impact

According to Melchert et al there are three IT trends that can combine the concepts of Business Intelligence (BI), Business Process Modelling (BPM) and Enterprise Application Integration (EAI) and contribute to Corporate Performance Management (CPM);

- Business Process Automation (BPA)
- Real-time analytics
- Process Performance Management (PPM)

BPA automates business process implementation through the execution of workflows involving multiple applications. Real-time analytics combines BI and EAI, whilst reducing the time factors in decision support allowing a closer link between analytics and business operations. PPM works to improve process by comparing BI for old processes, new processes and future processes to identify potential improvements. [27]

Figure 3 shows the relation of these IT techniques to the business aspects they support as well as CPM. EAI establishes a common integration infrastructure to allow multiple applications to be accessed through a single interface which is meant to ease use where there are business processes that are supported by several applications. However, EAI typically takes place on an “IT Level” and the need for close alignment with BPM has shaped BPA allowing the integration of applications based on business processes. [27]

A company’s competitiveness relies on its ability to react to business events rapidly. Being able to access data in real-time would allow a company to react to the latest news as soon as possible. EAI solutions can integrate the data to subscribing applications in nearly real-time however, EAI does not process the data in any way. BI provides the analysis aspect, producing information that can be used in order to make decisions around courses of action

however, BI cannot do this in real-time. Combining these aspects led to Real-time Analysis functionality. [27]

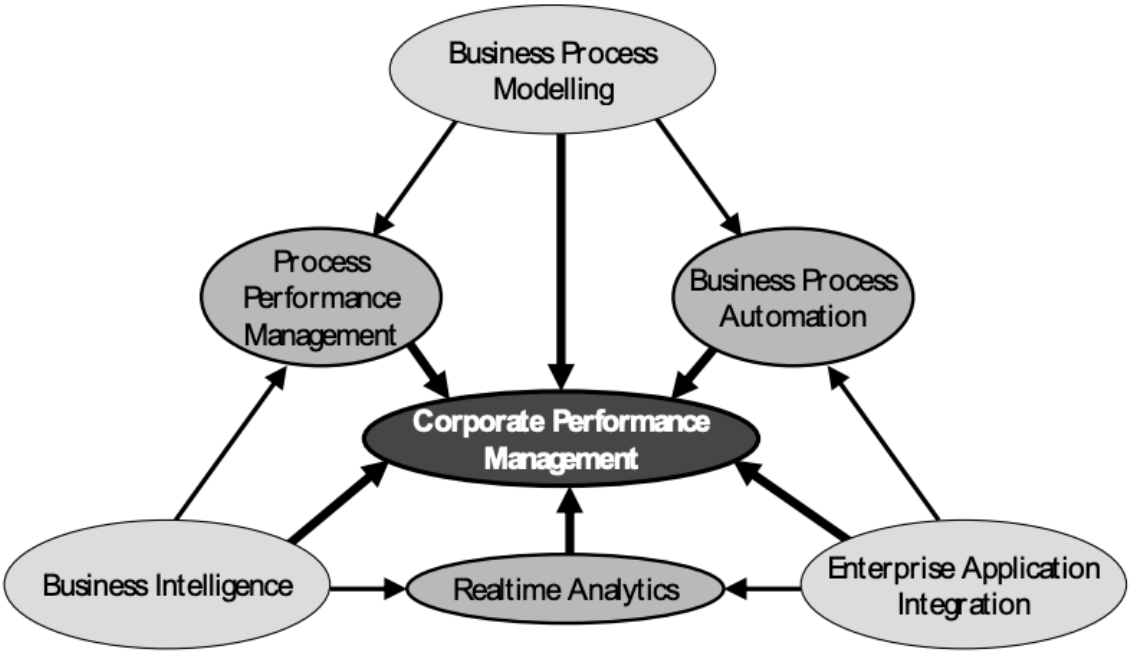


Figure 3 Converging technologies for CPM. [27]

Automation itself usually "refers to the use of IT to assist or replace employees in the performance of business process" [28] and also demonstrates that there are levels of automation, some automated systems still require an aspect of human involvement. BPA improve business activity performance, enabling enterprise wide monitoring and coordination as it is cheaper, executes in less time and provides better results however the development of such systems can be costly [29].

Real Time Business Intelligence [RTBI] comprises of real-time information delivery, real-time data modelling, real-time data analysis and real-time action based on provided insights. RTBI should facilitate the transition of data into information into action. Current BI fails in this area as both transitions suffer from bottle necks; data into information is delayed to the lack of experts capable of configuring and running analytical software whereas information into action is hindered by the fact that current BI output does not feed back into business processes. [30]

2.4 Project Implications

From the research that was conducted several conclusions were drawn in relation to the proposed system. Firstly, despite there being many data analytics software available they are not currently being used for the purpose that this system is targeting. They are concentrating on IT operations, healthcare data and drug trial data rather than looking at competitive intelligence and business process.

Where the system would assign therapeutic groups to drugs based on their use could be considered an NLP text classification problem. In order to develop this aspect of the proposed system FL string comparison could be utilised. As it gives an indication of the probability of an unstructured object, such as a string, belonging to a certain class.

Finally, the research conducted into the impact of automation and real-time analytics on business processes suggested that automation improves business activity performance as it can reduce costs and execution time whilst providing better results. Real-time analysis would be a benefit to businesses by reducing the bottlenecks in current processes where the analysis does not feedback into businesses processes.

Chapter 3 Methodology

This chapter outlines the possible approaches that could have been taken to the various aspects of this project including which programming language to use, possible testing approaches. The best options for the project's implementation were found by considering how each area can be executed.

3.1 Programming Language

When designing a new application, the choice around which programming language to use is key. Different languages are designed to make certain tasks easier and the developers' preferences and prior experiences should also be taken into consideration.

Table 3: Programming Language comparison

Language	*Pros*	*Cons*
Python	• Using python currently. • Favourite of up and coming coders. • Recommended for beginners – top introductory coding language in American University programs. • Mostly used for web applications and information security. • Popular for data analysis. • Python-based web development frameworks growing in popularity [35]. • Fast development speed. • Useful open source libraries.	• Unstable - updates to libraries and frameworks can affect your project.
Java	• Previous experience using this language. • Most widely used language in the world [36]. • Stable.	• More code required to do the same thing than Python. • Quite hard to become proficient in. • Slow to develop in.
HTML & CSS	• Web development. • Easy to learn. • Easy to change and update. • Supported by all browsers.	• No prior knowledge. • Consider browser differences – webpage may display slightly differently.

JavaScript	• Adds functionality to web-pages. • Very fast. • Very popular.	• No prior knowledge. • Need to consider client-side security [37].
C	• Previous experience using this language. • More compact than C++ and faster. • Grandfather of Java, JavaScript and Python.	• Didn't enjoy using C. • Out of practice.
C++	• Previous experience using this language. • Object Orientated. • Well suited to large projects.	• Didn't enjoy using C++. • Out of practice.

The programming language chosen was Python as it has a number of advantages, outlined above, and has functionality relevant to the project such as its popularity in the data science community and readily available web development framework options.

3.2 Hosting and Web Frameworks

Web hosting services can be considered the intermediaries between a service provider and the customers. That is, a web hosting service provider can facilitate hosting web sites, comprising web servers, FTP and SSH access, storage space as well as software capabilities [31]. There are three main aspects of hosting or cloud services available; infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS) as shown in Table 4.

Table 4: Comparison of cloud service options [32]

IaaS	**PaaS**	**SaaS**
• Vendor provides computing resources; servers, storage, networking. • Service users use their own platforms and applications within the service	• Cloud environment where users can develop, manage and deliver applications. • Includes storage, computing resource and a suite of development tools.	• Access vendor's cloud-based software. • No need to install applications on local devices. • Service users do not need to manage, install or upgrade

providers infrastructure. • Pay for what is used. • Scalable. • Data stored in the cloud – no single point of failure.	• Possible to test, develop and host applications in the same environment.	the software – the service provider handles this. • Access from anywhere with an internet connection.

On the other hand, a web framework can be considered to be a collection of packages or modules that allow developers to write web applications without having to handle low level details such as interpreting requests, producing responses or storing data persistently [33].

For the python programming language specifically, there are several web frameworks available, such as Django, Flask or Pyramid. Though their main uses are the same, each framework has their own unique offerings and should be considered with the project aims in mind, see Table 5.

Table 5: Comparison of 3 Python web frameworks [34]

Django	**Flask**	**Pyramid**
• This is the most popular python web framework. • ORM out of the box. • Can be rather clunky on small scale, better suited to large projects. • Used by Bitbucket, Pinterest and Instagram.	• Youngest of the three web frameworks. • Simple – good for small projects. • Flexible. • Fast, easy way to make small, one-off tools or simple web-interfaces. • Micro-Framework; better suited to small applications with simple requirements.	• Most flexible of the three frameworks. • Suited to small applications or big projects. • So many options – can be intimidating to unfamiliar developers.

Flask was chosen as it was best suited the scope of the project.

3.3 Source/Version Control

Version control is the practice of keeping different versions/drafts of work and managing them in order to differentiate between them. Working in this way has many resulting benefits, as shown in Table 6.

Table 6: Version Control Benefits [38]

Benefit	**Description**
Traceability	Identifying the development of the document by retaining drafts and details of changes.
Identifiability	Possible to link documents to decisions and timestamps.
Clarity	Multiple versions are distinguishable, and the latest version is easily identifiable.
Reduced Duplication	Old or potentially out of date copies can be destroyed.
Reduced Errors	Drafting work reduces the likelihood of work being lost or the chance being unable to rollback to old, working versions.

For software/system projects version control also extends to the code being written. There are several version control systems that handle the management of versions for the developer. These systems use code repositories to store the latest version of code and keep it separate from the version that is being worked on. When the developer is happy with the updates that have been made that code is pushed and merged with the code in the repository and the changes made are noted. This provides a record of all changes made throughout the project lifecycle [39]. This project made use of GitHub in order to do this throughout the project.

3.4 Testing Approach

Testing is a key part of any project, especially software/ information system development. Software testing is the process of executing a program with the goal of finding errors [40]. Furthermore, most projects have a test plan where the approach taken consists of checking if a program for specified input gives correct and expected outputs. In summary, testing evaluates software quality.

There are 2 main testing approaches; white box testing and black box testing, shown in Table 7. White box testing relies on tests being conducted with no knowledge of the internal structure of the system and is based on output requirements. On the other hand, black box testing requires that insight into the internal workings of the system being tested, is highly effective in detecting and resolving problems as it is a strategy for debugging and can also be considered as an approach for security testing. More recently, a third method, a hybrid of the two; gray box testing, has become more popular. This approach requires some knowledge of the systems code and logic processes. Gray box testing is non-intrusive and unbiased, it is used effectively during the integration of two modules.

Table 7: Examples of White Box and Black Box testing methods.

White Box Testing	Black Box Testing
• Basis path testing. • Loop testing. • Control structure testing.	• Equivalent partitioning. • Boundary value analysis. • Cause-effect graphing techniques. • Comparison testing. • Fuzzy testing. • Model based testing.

White and black box testing can be considered the foundations of most testing approaches. Popular considerations that are now made when approaching software testing are test driven development (TDD), continuous improvement and user acceptance testing (UAT).

Test driven development is more than an approach to testing and should be considered throughout the software lifecycle as it can affect analysis, design and programming decisions. TDD consists of writing automated tests prior to developing functional code in small, rapid iterations [41]. By comparison, continuous improvement considers data on lifecycle stage performance and helps to make decisions going forward and eliminate waste. However, as it is a reactive approach the improvements made can be limited [42]. The Capability Maturity Model (CMM) is a reference model for appraising software process maturity and level 5, optimising, is continuous process improvement [43].The CMM emphasises that continuous improvements are something that should be aimed to be included.

Finally, user acceptance testing must be taken into consideration as it can have detrimental effects on a project if not conducted effectively. Studies see high failure rates in new information systems: of 6700 projects across 500 enterprises 24% were cancelled and 17% saw cost overruns [44]. This was amplified in more complex projects; in projects with 100,000+ functions it was observed that 65% were cancelled and 35% saw cost overruns [44]. A major cause of this was attributed to poor management of user requirements. Further study was conducted with 8000 projects and the three main causes of projects being late, over-budget, or simply non-functional were [44];

1. Lack of user input
2. Incomplete requirements
3. Changing requirements

From this it is possible to see that there are two main categories of software defects which highlight the importance of conducting regular UAT;

1. Defects in implementing specified user requirements due to design or coding errors.
2. Defects in the correctness of requirements due to discrepancies between specified user requirements and true user requirements.

In order to test the solution a combination of component, integration and UAT testing will be carried out.

3.5 Classification Models

There are several learning models/algorithms that can be used in order to classify gathered data, each one with its own advantages and disadvantages which make it appropriate for different tasks.

Decision trees split data according to specific criteria, repeatedly, maximising the separation of the data and creating a tree-like structure. This is done using a greedy approach, where

each optimal split-point is selected at each step, with no look-ahead considering possible combinations which may lead to better results. However, decision trees can be expressed as rules easily which makes them appropriate for use in diagnostic tasks. [45]

The Naïve Bayes algorithm takes a probabilistic approach, utilising a labelled training set to estimate parameters of a generative model. New data is classified through selecting the class in which the new data most likely belongs to. This classifier assumes all attributes of examples are independent therefore, each attribute is learned separately which simplifies the learning, especially when the dataset is large. [46]

The Perceptron algorithm mimics neurological behaviours by training a linear classifier using an update rule. With each pass through the training set associated weights are updated based on whether the last weight used resulted in a correct classification. For linearly separable problems the algorithm tends to converge to a weight value which classifies the whole training set correctly. [47]

Support Vector Machines (SVMs) approach classification as an optimisation problem. Presented with two groups of data and SVM predicts a classification by asking, for an unlabelled example, which group includes the labelled example. The classifier produced is non-linear and generalises by maximizing the margin between the two groups. SVMs do not assume the problem is linearly separable, representing the data through a non-linear mapping function with a linear boundary between groups. [48]

Logistic Regression models (LRMs) belongs to the class of generalised linear models [49]. LRMs work well where the outcome variable is discrete as it describes the relationship between an outcome and a test set, aiming to find the best fit. The outcome variables in an LRM are binary or dichotomous (branched into two parts). Furthermore, the analysis is based on a binomial statistical distribution rather than a normal distribution. [50]

The choice of classification model will be based on the accuracy when applied to the dataset.

3.6 Database

Regarding data storage solutions there are several options available. Relational databases are a leading model however NoSQL technologies are increasing in popularity due to the increasing needs for scalability and performance to handle big data systems [51]. Therefore, examples of both will be considered going forward.

MongoDB is an example of a NoSQL database which stores data in BSON (a binary equivalent of JSON) allowing a schema-less model, requiring only an ID. Scalability is implemented through sharding, a method for distributing data across multiple machines [52], and replication is supported through an asynchronous master-slave model so only the master node processes writes and both slave and master can process reads. MongoDB provides different frameworks for data manipulation and changes made are atomic.

PostrgreSQL is an open source relational database which implements a server/client model and supports ANSI:SQL2011, the standard database language. Jung, Youn, Bae and Choi have compared PostrgeSQL and MongDB performance in a big data environment and see that relational databases struggle with the varied data structure associated with big data whereas NoSQL solution, such as MongoDB, handle various formats and large quantities of data with more ease. Their study compared insert, select, update and delete operations and saw that MongoDB was faster and MongoDB's performance improved when working with unstructured rather than relational data. They concluded that a relational model, such as PostgreSQL, performs better when the environment requires precision and structure. [53]. As this is true of this projects solution, PostgreSQL was chosen as the database solution that would be utilised.

Chapter 4 Quality Plan

In order to plan for quality, this section will identify the critical project processes and process quality standards that will be used to evaluate them [54]. Development processes will also be defined to ensure a consistent quality throughout the project lifecycle.

4.1 Process Description

The application development will follow an agile methodology inspired by SCRUM and structured following the V model, but with a few modifications, as it is a single-developer project rather than a team. This will allow for the development to follow a feature sprint dev, test, release cycle with continuous testing which will suit the modular nature of the application. Additional steps like gathering user feedback can be added to the testing stage of the sprint when necessary, another benefit of developing in an agile way.

4.1.1 Development Process

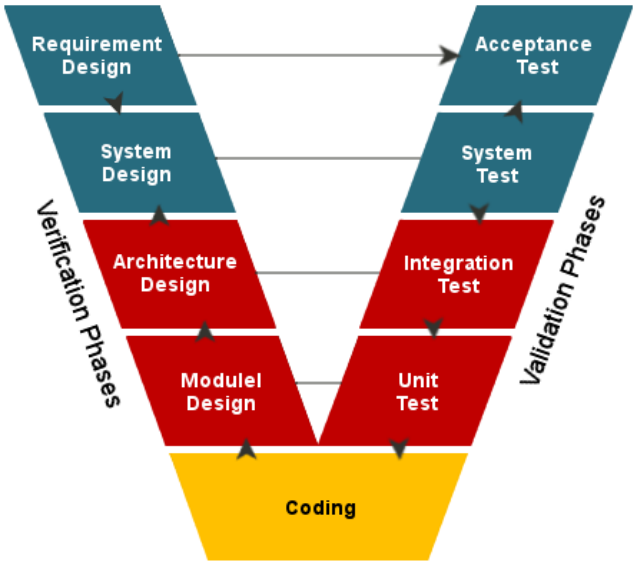


Figure 4: V Model diagram

The V model in Figure 4 accurately represents the development process of this project. At each stage of the project, there will be testing to ensure a high level of quality. The verification side works towards the analysis of the different aspects of the project and the

validation side tests the project. Following this model throughout the project is key as "verification and validation is one of the software-engineering disciplines that help build quality into software" [55]. Evaluating the software during each phase using static testing ensures that requirements from the previous stage are met. Similarly, dynamic testing at each stage ensures that the product is meeting the requirements and is also producing the desired outcomes. As the model indicates, verification and validation will be present throughout the development process and due to this being a single person project these stages can be integrated seamlessly during each stage.

4.1.2 Component Lifecycle

This project will be managed in an agile way and the component lifecycle defined below will reflect this. Using a continuous delivery software strategy will allow the project to progress as efficiently as possible. Continuous delivery aims to "create a repeatable, reliable and incrementally improving process for taking software from constant to customer" [56]. The component lifecycle pipeline, Figure 5, used in this project will include development, integration, and review stages. These will be done in a sprint-like manner inspired by SCRUM [57] and adapted for this single developer project.

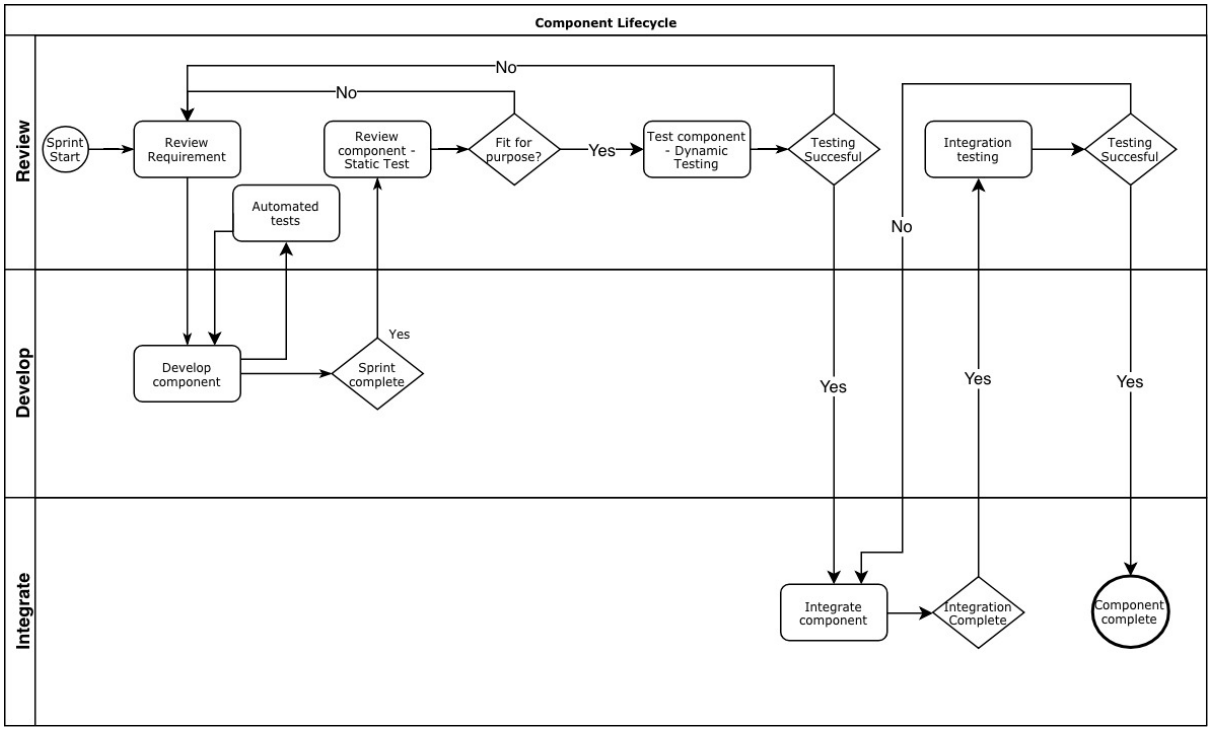


Figure 5: Component Lifecycle flow diagram

The process outlined in Figure 5 will be followed during the development lifecycle of each component until they come together to form a key element of the project. Figure 5 shows a high-level overview of the component's lifecycle including the three swim lanes that the necessary tasks fall under; review, develop and integrate. Once all key elements are completed the process will be followed for them treating them as components of the main application.

The first stage of the cycle would be to review the requirements. To ensure the component passes a quality review it is essential that the true purpose of the component is understood and the outcome for the sprint is outlined. This step should reduce the risk of unnecessary components being developed as well as needing to re-visit components later in the development cycle to re-work them to be fit for purpose. Developing each component in this way, following a SCRUM-like approach, will allow the program to be modular which will increase the efficiency in the review stages. By developing and reviewing the program in chunks it should allow for a more effective review process as we can review the component itself and then the main program once the new component has been integrated in order to ensure the process is going smoothly.

During the development stage, the component will be built over the course of the sprint. Once the sprint has ended the product will be reviewed against the initial requirements. If the component was completed and aligns with the requirement it can move on to be tested. Sometimes a feature is not completed in its designated sprint as there were unforeseen complications. When this happens, the feature will rollover to the next sprint which is reflected in the loop between feature review and requirements review. At this stage, a testing recursive loop could be added to ensure that continuous testing is undertaken whilst developing the component. This will be explored further in the v-model section of this quality plan.

If the development of the feature was completed during the designated sprint, then the component can be moved on to the review. The initial review of the component will be against the requirements to make sure the component is fit for purpose and is producing the desired outcomes. Once this static testing has been completed the feature can move

along to dynamic testing at a component level. Once the component passes these tests successfully it will be integrated into the main application and undergo integration testing to ensure that the new component works well within the application itself as well as regression testing to ensure that the new code has not had a negative effect on older components. Once this testing has been successful the component lifecycle is complete.

4.1.3 Version Control

Throughout the project, GIT will be used to version control the code that is produced. Version control will "record changes to a file or set of files over time so that you can recall specific versions later" [58]. Developing components in this way should aid with integration. There will be a master branch that all working, integrated and tested components will be pushed to. A dev branch will be used for component development and integration testing to ensure that the master will always have the latest, working copy. Merging to develop suggests the component has passed its unit testing and is ready for integration tests. As this is a single person project there will not be a need for separate branches for the development of each component as there is no risk of developers overwriting each other's work here. The dev branch will be sufficient in this project.

4.1.4 Continual Improvement

To ensure the project has a high level of quality a continual improvement process will be implemented. Evaluating the efficiency, effectiveness, and flexibility of project processes will allow for improvement. This will ensure that the project is meeting its quality goals.

4.2 Quality Goals

This section aims to quantify the "quality of application as well as quality of testing" [59]. ISO/IEC 25010 [60] identifies 8 areas that a product can be evaluated in order to determine its quality. There are two parts to ISO/IEC 25010; the first applies to the product quality and the 8 characteristics it defines, the second is the quality in use and the characteristics that are needed for a product to meet this.

Table 8: Product Quality - ISO/IEC

Characteristics	Sub-Characteristics	Definition
Function Stability	Functional Completeness	The degree to which the set of functions covers all the specified tasks and user objectives.
	Functional Correctness	The degree to which the functions provides the correct results with the needed degree of precision.
	Functional Appropriateness	The degree to which the functions facilitate the accomplishment of specified tasks and objectives.
Performance Efficiency	Time-behaviour	The degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.
	Resource Utilization	The degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements.
	Capacity	The degree to which the maximum limits of the product or system, parameter meet requirements.
Compatibility	Co-existence	The degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.
	Interoperability	The degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.
Usability	Appropriateness recognisability	The degree to which users can recognize whether a product or system is appropriate for their needs.
	Learnability	The degree to which a product or system enables the user to learn how to use it with effectiveness, efficiency in emergency situations.
	Operability	The degree to which a product or system is easy to operate, control and appropriate to use.
	User error protection	The degree to which a product or system protects users against making errors.
	User interface aesthetics	The degree to which a user interface enables pleasing and satisfying interaction for the user.
	Accessibility	The degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.

Reliability	Maturity	The degree to which a system, product or component meets needs for reliability under normal operation.
	Availability	The degree to which a product or system is operational and accessible when required for use.
	Fault tolerance	The degree to which a system, product or component operates as intended despite the presence of hardware or software faults.
	Recoverability	The degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.
Security	Confidentiality	The degree to which the prototype ensures that data are accessible only to those authorized to have access.
	Integrity	The degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.
	Non-repudiation	The degree to which actions or events can be proven to have taken place so that the events or actions cannot be repudiated later.
	Accountability	The degree to which the actions of an entity can be traced uniquely to the entity.
	Authenticity	The degree to which the identity of a subject or resource can be proved to be the one claimed.
Maintainability	Modularity	The degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.
	Reusability	The degree to which an asset can be used in more than one system, or in building other assets.
	Analysability	The degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified.
	Modifiability	The degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality.
	Testability	The degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be

		performed to determine whether those criteria have been met.
Portability	Adaptability	The degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments.
	Installability	The degree of effectiveness and efficiency in which a product or system can be successfully installed and/or uninstalled in a specified environment.
	Replaceability	The degree to which a product can replace another specified software product for the same purpose in the same environment.

Table 8 outlines, in detail, the characteristics that will be used to determine the quality of the application created during the project. The application will meet the requirements and deliver the desired outcome in order to be deemed functionally suitable. Timing will be key to this applications performance efficiency to ensure it does not become a problem for the website it is gathering data from, this will also affect the applications compatibility. Usability is also an important characteristic. The ease of use for the user will determine whether the system is an adequate replacement for their current processes. Due to the nature of the application, it's quality will be directly linked to security as accessibility and integrity are key. The development process has been designed to increase the maintainability of the application. Finally, the portability is key as adapting to different operational/usage environments is important for this programme. Functional testing will allow the whole system to be verified against these quality characteristics.

4.2.1 Non-Functional Requirements

4.2.1.1 Usability – User Interface & Accessibility

The application should be easy for users to interact with and achieve their desired result. The interactions needed should be self-explanatory so that the user can navigate the application un-aided. User stories will be created to outline how specific users would utilise the system.

4.2.1.2 Reliability

The application needs to be available and up to date when the user's login.

4.2.1.3 Security

The web application has to have high security as the data that users can input will be company confidential information (CCI). Due to this, it is essential that users cannot view other user's data. The security for the open source data does not have to be as extreme as anyone can find this data online.

4.2.1.4 Maintainability

The application will be developed as components and therefore will be modular by design. Designing the application in this way will increase how reusable the separate components are as they will work independently of the main application. The modifiability of the application will be high as it could be applied to several use cases by changing the data you are looking to gather e.g. crime rates, sports scores. The application will be developed in a way that includes continuous testing from a component to the system level, this should ensure high testability.

4.3 Risk Management

"Risk management means risk containment and mitigation" [66]. Table 9 identifies risks associated with the project and create plans to mitigate the issues. Classifying the risks will help to prioritise them.

Table 9: Identifying quality risks of the project

Risk	**Details**	**Mitigation**	**Impact**	**Probability**
Functionality	Unused features are developed/features that do not meet the requirements.	Static testing on the requirements for the project and detailed design created to limit the chance of this occurring.	2	2

Performance	Application run-time exceeds user expectation whilst searching for new data.	Use algorithms with best time case possible applicable to the problem.	5	3
Compatibility	The application consists of the 3 main components which could run as separate applications – these need to co-exist in order for the project to work.	Integration testing to ensure that the 3 components mesh harmoniously.	6	2
Usability	Users need to be able to filter data to be able to get the comparisons they need to make informed decisions.	Ensure necessary functionality is built into the systems so that users can do what they need to do as easily as possible.	6	1
Reliability	Data needs to be up to date and readily available for users when they use the system.	Ensure that data scraping occurs during times where the website load would be at its lowest to ensure that the scraping is done quickly and efficiently. Also, flag data that is current and allow users to force an update on a specific drug and country if needed.	8	4
Security	The authenticity of the data needs to be high so the sources the scraper targets need to be secure sources. Confidentiality is key as users will be able to input CCI and therefore it is essential that other users cannot access each other's data.	Some sort of logic will have to be built into the web scraper to verify the data sources. Use secure logins and permissions to ensure that people only see their own data.	9	6

Chapter 5 Project Plan

In order to better understand how the project will be structured, a Gantt chart was created based on the list of objectives defined in 1.3 Aims and Objectives as shown in Figure 6. The project due date is the 29th of April 2019 therefore, in order to thoroughly plan the project all tasks that needed to be completed were laid out. The project milestones were considered and finalised as research, design, develop, test and write report highlighting the main deliverables of the project. The develop milestone had “mini-milestones” within it, which were relevant to the major features; data gathering, data storage and data visualisation as well as linking these features into a single, working system. The milestones within the development milestone can be considered as the key deliverables of the product. For each individual task the time it would take to complete was considered and used to create a Gantt chart which visualises how to project will come together within the allocated time.

Contingency time was allocated at the end to re-draft the report and the development was planned to finish a couple of weeks before the final deadline in case any unplanned issues arose. As this is a one-person project, contingency time was a very important aspect to consider in order to ensure that if anything effected the developer and put the project behind schedule there would be time to recover and get back on track.

Throughout the project, check-ins were scheduled with the project supervisor to get feedback and advise on how the problem was being approached and on whether the developer was meeting deliverable deadlines. This helped keep the project in scope and on target.

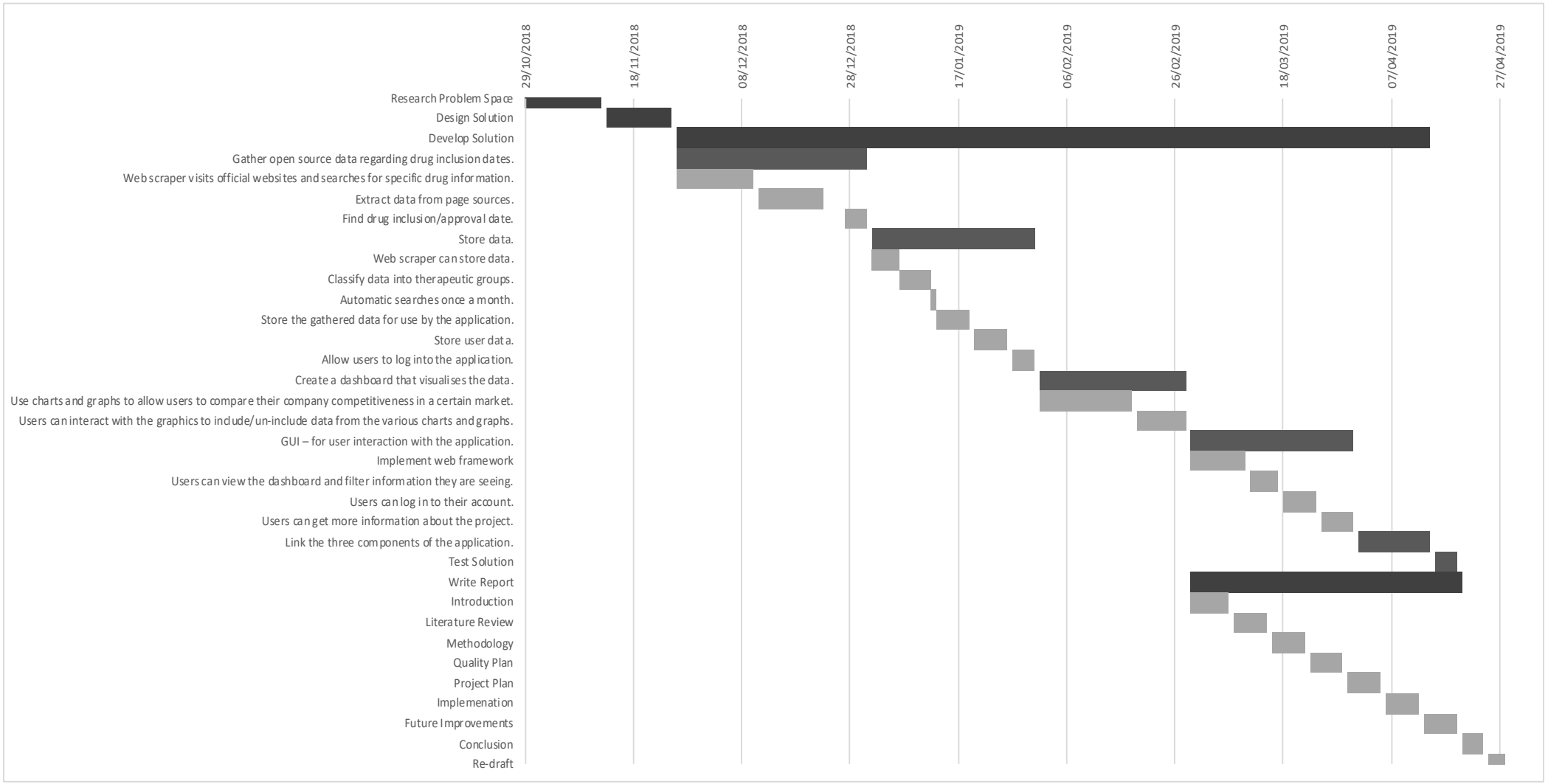


Figure 6 Gantt Chart.

Chapter 6 Implementation

This chapter looks at each stage of implementation: design, development and testing. The design section explores user stories, front-end design choices as well as back-end configurations to aid in the next section, development. The second section looks at how the system was built, and the methods chosen. The final section of this chapter contains the testing; the tests carried out, the expected results, the actual results and any necessary modification that were made to achieve the expected result.

6.1 Design

This section will look at the design of the project. User personas and relevant stories will be considered to fully understand the end users stakes in the product. Wireframes will be used to visualise the front-end and a Unified Modelling Language (UML) model will be used to describe the back-end design. An Entity Relationship (ER) diagram shows how the PostgreSQL database should be implemented in order to store the data gathered.

6.1.1 Personas

The following personas were created to gain insight into the systems target audience members. They are based on personal experience of members of a regulatory affairs team in a large pharmaceutical company and their first-hand accounts of the delivery and importance of this task to users higher in the company hierarchy. This exercise was completed in order to better understand user expectations, understand how they would use the system as well as uncover any missed features or functionality [67].

The following personas represent several aspects of potential users; their name, age, occupation, keywords that would describe them, their job-related goals and frustration, technological literacy, personality traits, a quote about the current way of working and a short bio relating to their involvement with the process the proposed system hopes to replace. Their technological literacy is measured through a traffic light system for 4 categories just to give a high-level indication of the “users” comfort levels with different types of technologies.

Karen

Age	29
Occupation	Regulatory Affairs
Keywords: Keen, Passionate, enjoys her job.	

Bio:
Karen works in the regulatory affairs team in a local office of a large pharmaceutical company.
A part of her job is to keep track of competitor drugs and to do this, every day, she visits websites like the FDA to see if there have been any updates for drugs on her watchlist and to see if any new drugs have been put in for approval. She then logs any changes in her spreadsheet which she gives to her manager once a month.

Goals:

- Have more responsibility at work.
- Do her job well.
- Get a promotion by the time she is 30.

Frustrations:

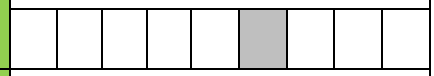
- The task takes forever.
- The task is a bit boring.
- Feels there are better things she could be doing with her time.

Technology:

Social Media	
IT and Internet	
Mobile Apps	
Specialist Software	

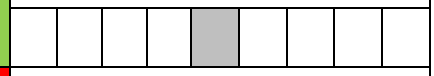
Personality:

Introvert — Extrovert



Imaginative — Unimaginative

Fearless — Timid



"I feel like I am not being fully utilised, anyone can google and fill in a spreadsheet, right?"

Robert

Age	43
Occupation	Regulatory Affairs - Manager
Keywords: Tired, Responsible, Stressed.	

Bio:
Robert manages a regulatory affairs team locally for a large pharmaceutical company.
Once a month he checks a spreadsheet put together by members of his team which contains information on competitor drugs. He's looking for any signs that this information needs to be passed on earlier than the usual once a quarter update, but this never happens.
Once a quarter, as a team, they make some graphs from the spread sheet and pass them on to someone at a higher pay grade than him.
He is extremely busy.

Goals:

- Delegate as much work as possible.
- Not to make any mistakes or miss anything important.
- Impress his boss.
- Keep his team happy.

Frustrations:

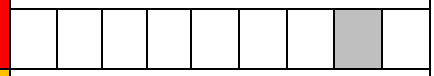
- He feels members of his team are constantly occupied with the competitor info gathering task when he needs them

Technology:

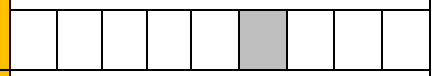
Social Media	
IT and Internet	
Mobile Apps	

Personality:

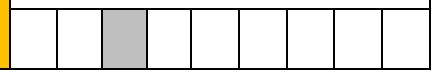
Introvert — Extrovert



Imaginative — Unimaginative



Fearless — Timid



	elsewhere and he's not sure how to help them.	Specialist Software		"Keep the team motivated and keep the bosses happy!"

Amy	
Age	38
Occupation	VP Regulatory Affairs
Keywords: Busy, Focused, Professional.	

Bio:
Amy is the Regulatory Affairs Vice President (VP) of a large pharmaceutical company.
Part of her job is to present competitive intelligence research to the CEO once a quarter. She needs to know the facts and figures so that she can answer any questions and present them in a clear and concise way with visual representations. This information is very important as it can affect decisions regarding the company's strategies.

Goals:

- Work hard and become one of the few female CEOs.
- Be known for always having the answer.
- Always be prepared.

Frustrations:

- The quarterly meeting sometimes feels like it happens too late, they're being reactive instead of proactive.
- The graphics she's provided are sometimes too general and sometimes to detailed to get an overall picture.

Technology:		**Personality:**
Social Media		Introvert — Extrovert
IT and Internet		Imaginative — Unimaginative
Mobile Apps		Fearless — Timid
Specialist Software		"There must be a better way we can approach these quarterly meetings so that we get the best information!"

Ian	
Age	38
Occupation	CEO
Keywords: Busy, Analytical, Passionate.	

Bio:
Ian is a CEO of a large Pharmaceutical company.
Once a quarter he meets with his VPs and during this he is given information from regulatory affairs regarding competitors. This information gives insight whether his company is being competitive across therapeutic and geographical markets. Now he needs to think about whether to invest more or less in specific drugs in order to compete against other Pharmaceutical companies.

Goals:

Frustrations:

Technology:		**Personality:**
		Introvert — Extrovert

<table>
<tr>
<td rowspan="4">• Make important decisions based on the best information.
• Before speaking on a topic be well informed.</td>
<td rowspan="4">• The regulatory affairs portion, though presented well, sometimes seems a bit late in the day or requires closer examination later as the data isn't presented effectively.</td>
<td>Social Media</td>
<td></td>
<td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td>
</tr>
<tr>
<td rowspan="2">IT and Internet</td>
<td rowspan="2"></td>
<td colspan="9">Imaginative Unimaginative</td>
</tr>
<tr>
<td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td>
</tr>
<tr>
<td rowspan="2">Mobile Apps</td>
<td rowspan="2"></td>
<td colspan="9">Fearless Timid</td>
</tr>
<tr>
<td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td>
</tr>
<tr>
<td>Specialist Software</td>
<td></td>
<td colspan="9">"Do the local teams understand how important this is? Can I get something high level on a more regular basis?"</td>
</tr>
</table>

6.1.2 User Stories

From the descriptions in 6.1.1 Personas, it is possible to construct user stories to better understand how the different types of users would use the system.

Karen represents a local regulatory affairs team member and is a user who would interact with the system on a regular basis. Karen can log in to the system, see the dashboard and filter what the graphics are displaying to match the specific thing she is looking for e.g the number of oncology drugs that were approved by the FDA last year. Karen wants to do this so that she can see an updated view of competitor drug approvals in different therapeutic and geographical areas so that she can report to her boss if there has been a significant change that needs to be passed on. Incorporating this system into her work benefits Karen as she no longer has to spend hours of her week searching for this information on her competitors, nor does she have to keep track of changes in a spreadsheet as it is all done for her. Furthermore, producing reports is easier as the dashboard itself contains graphics that other team members, her boss and even her boss's boss can see whenever they login. The most she would have to do is summarise or export them, if requested to do so giving her more time to work on other important tasks.

Robert represents a manager of a regulatory affairs team, possibly Karen's manager. Robert is often shown the dashboard by his team members and is glad that it has freed up their time to work on other tasks. He doesn't log into the system very often himself as his IT literacy isn't great, however he has a fair idea of how to do so if he needed to or he could

ask a member of his team to open it up for him. He loves the visibility of the online dashboard as it takes a bit of pressure off him regarding the monthly check-ins to see if anything needs to be flagged, as his boss can also see the dashboard and flag any updates that may need further inspection. Overall, it has made the team more efficient as a laborious manual task has been taken off their hands.

Amy represents a regulatory affairs VP. Amy is still provided a summary of competitive intelligence gathered by the organisation's various regulatory affairs teams in preparation for the quarterly review meeting with the CEO, but she often logs into the dashboard to keep an eye on things and loves that she can do that. The graphics are eye-catching and easy to filter so that she can easily see what is going on with the data without needing an expert analyst to take time to process the data. Even though it isn't her job to monitor this she does like that she can check in from time to time. It also puts her mind at rest as she has access to up-to-date information whenever she needs it so, if anything last minute comes up, she can prepare quickly rather than needing to wait on several teams to compile data for her.

Ian is the CEO of a pharmaceutical company. Ian often worried that he wasn't being presented with information within a time-framed that allowed for action. This new dashboard that regulatory affairs presented is a great idea as it updates itself and shows the data in a way that is easy to interpret. Furthermore, anyone who could need access to this data has a login and they can go in, view it and filter it however they like to get what they need. This means he can access competitive intelligence information when he needs it, not just once a quarter, which should hopefully lead to more timely strategic decisions.

6.1.3 Wireframes

In this section, a template for the front-end design is provided. This acted as a guideline during the development portion of the project. The user interface (UI) required two wireframes, one to serve as a template for the "pre-login" portion which will contain information around the project itself, and the other for the "post-login" portion where users can see the graphs generated from the open source data that has been gathered.

Figure 7 shows the design for the “pre-login” homepage. This would be where users will be able to login or register as well as navigate to other pages with more information regarding the project. The wireframes for these are in the appendix; there is a page describing the three main elements of the project shown in Figure 14, a page detailing the project background shown in Figure 15 as well as a contact us page shown in Figure 16. The “pre-login” design was inspired by small scale web applications that use their website to market the solution as well as implement it such as canny.io [68], branch.io [69] or stellar.io [70].

Once the user logs in they are presented with a dashboard displaying graphics based on the data gathered by other parts of the system as shown in Figure 8. Appropriate charts were used for the data to ensure it is as readable as possible for the user. These charts will be interactive to ease filtering and engage users. They’re size and the page layout was dependent on the readability of the chart.

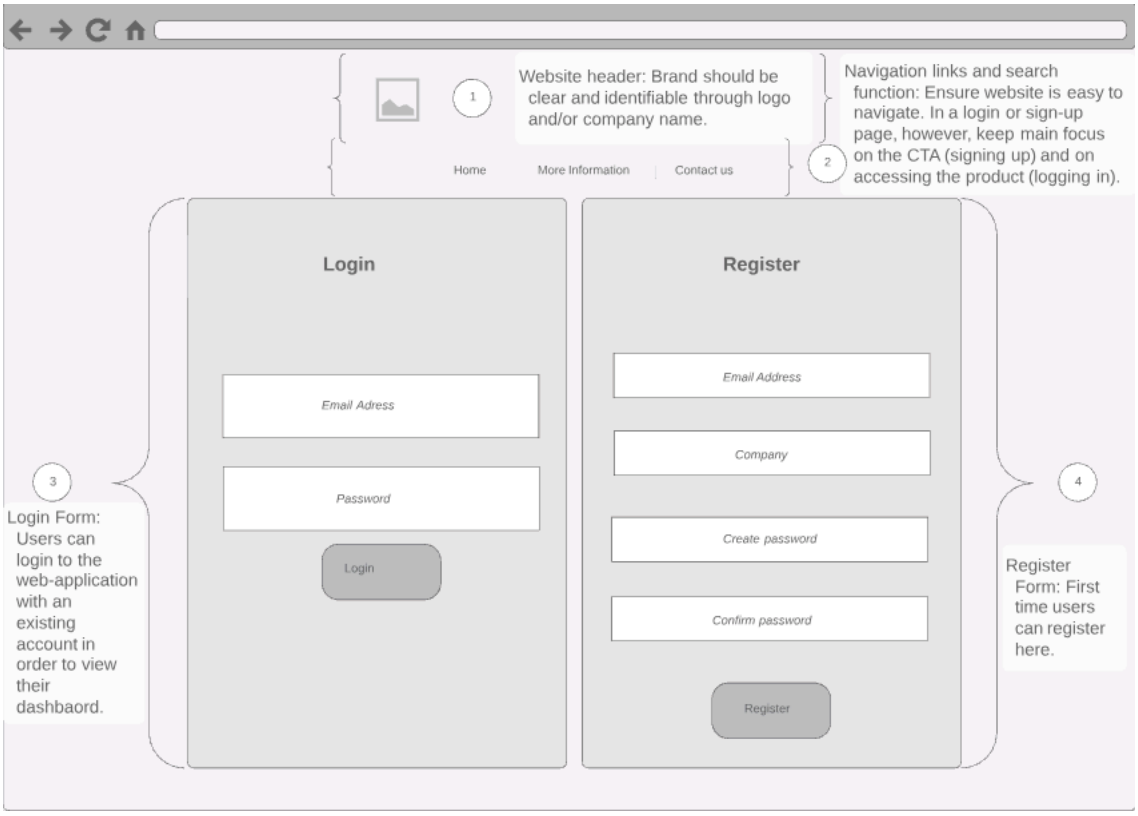


Figure 7 Home page wireframe.

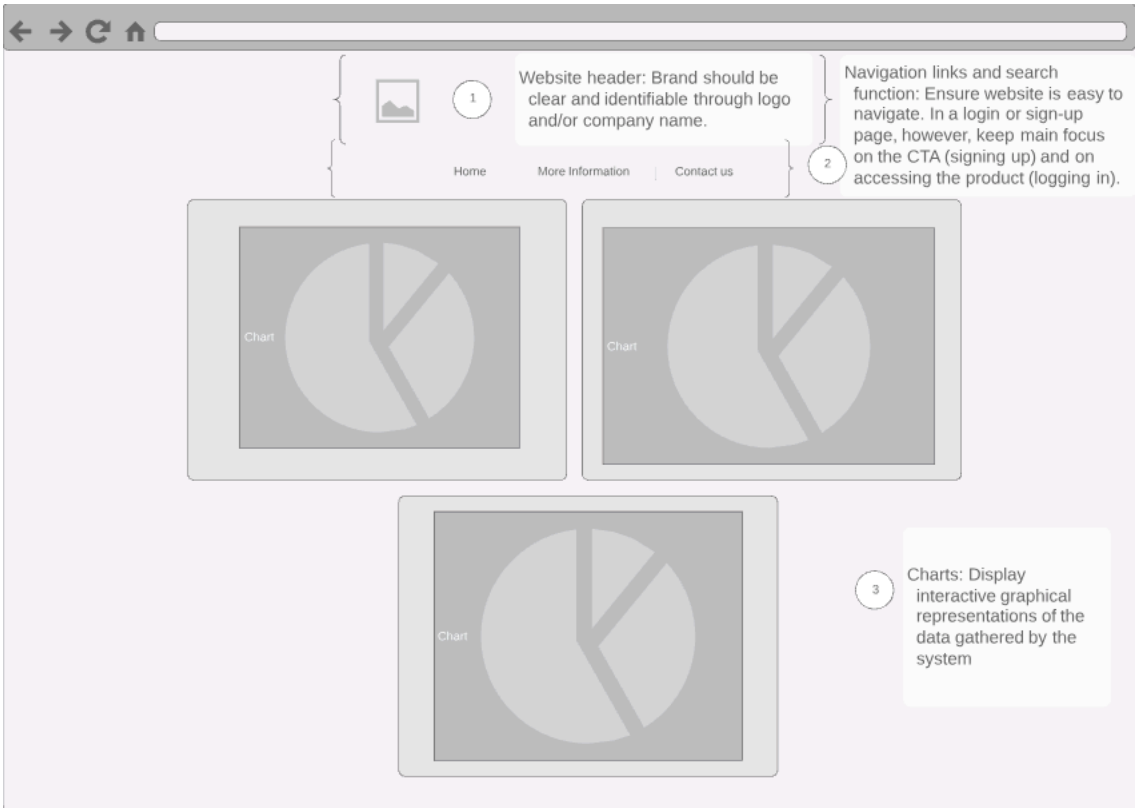


Figure 8 Dashboard page wireframe.

6.1.4 Logo

Figure 9 shows the solutions logo. The system has been named “DataSource” as it is a source of competitive intelligence data for pharmaceutical companies. The tagline is “The drug approval date dashboard” highlighting the project’s focus. The image has a two-fold meaning; on one the one hand it represents a graph, linking to the visualisation side of the system, on the other hand it represents a drug molecule to emphasise that the data being handled is in relation to the pharmaceutical industry.

Figure 9 The solutions logo.

6.1.5 Technical Stack Diagram

A technology stack gives a high-level overview of the services, frameworks and programming languages used in a single application [71]. The full stack shows all the technologies that will go into the system, whereas the front-end and back-end stacks split the technologies based on where they contribute. This system's proposed technology stack is depicted in Figure 10.

In order to develop the front-end a combination of programming languages was used. HTML was used to structure the web application, whereas CSS was used to style it. Bootstrap libraries were used to make use of open source styling and functionality to reduce development time. JavaScript was used to add needed functionality to the web application that cannot be achieved through a combination of HTML and CSS. Finally, chart.js, an open source data visualisation library, was utilised to create the interactive graphics on the dashboard.

From a back-end perspective there are several aspects to be considered. The web framework that implemented was Flask as it coincided with the programming language chosen and works well with the scale of the project. This aspect of the system was developed in Python and makes particular use of the Pandas and BeautifulSoup libraries. Pandas aided in the data analytics and BeautifulSoup aided in the web-scraping aspects of the system.

The final aspect of the back-end is the database. The chosen database for the system was PostrgreSQL as the system would benefit from the structure of a relational model. The database was designed to handle user login details as well as to provide a structure to store the open-source data. The design of the database will be discussed further in section 6.1.6 Database.

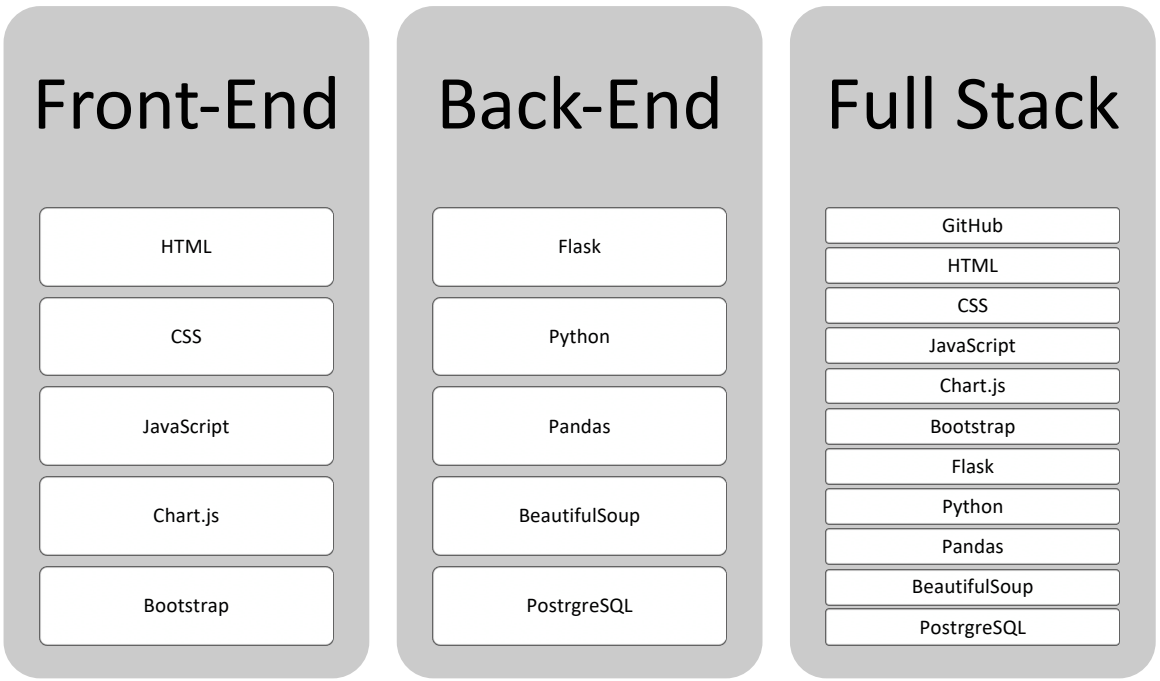


Figure 10 Technical Stack Diagram.

6.1.6 Database

The PostgreSQL relational database for the system was conceptualised using an entity relationship (ER) model as shown in Figure 11. An ER model can be considered a blueprint for the database describing necessary tables and how they are related to each other [72]. The database design consisted of 3 tables; one to store user data, one to store the open source data that has been gathered and one for company information.

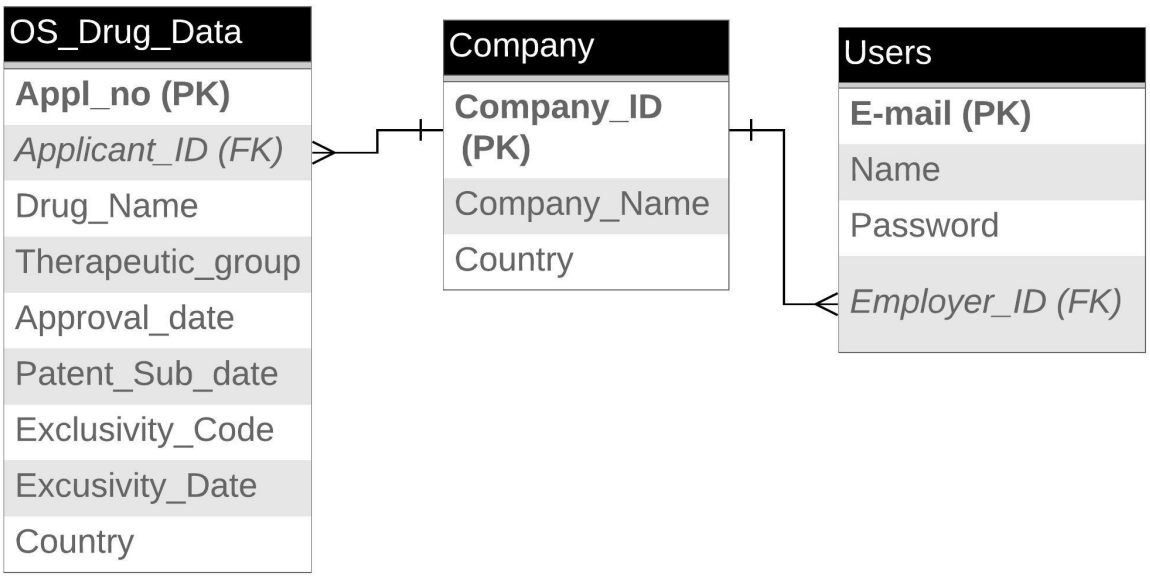


Figure 11 ER Model.

The OS_Drug_Data table was designed to store the open source data that was scraped from websites, like the FDA, and collate it for manipulation later on. The main data points of interest were the company that submitted the drug for approval, the therapeutic group, approval data, patent submission date, exclusivity date (if available) and the country. Each drug's unique identifier (primary key) was design to be its application number. The company table was designed to hold information about Pharmaceutical companies, mainly their trade name and country of origin, and an assigned unique ID number.

The final table was designed to be the user's table with the purpose of providing login/registration functionality to the web application. The user's email was assigned to act as a primary key as it is assumed that company email addresses are unique. The user table was also designed to store which Pharmaceutical company the user works for to keep track of which companies are utilising the system. Additional functionality could be added to the system later regarding company information which will be discussed further in Chapter 7 Social, Legal and Ethical Considerations

This section explores the social, legal and ethical considerations related to this project and the solution that was produced. Currently this is a proof of concept and therefore these considerations are based on if this solution were used by real customers.

7.1 Social

As the system handles open source data and enables users' access to data, there should not be any concerns regarding the systems purpose, however there are some mixed feelings towards automating processes as some systems can make human employees unnecessary. This is not the case with this system as it exists to enable the employees in regulatory affairs so that they can be more efficient with their time. This system could change perceptions of the importance of data analytics as it shows how having access to data can enable business decisions.

7.2 Legal

Legally, as this system is handling open source data there is no issues regarding GDPR. Regarding storing user data in the future, this will need to be stored securely in compliance with the Data Protection Act.

7.3 Ethical

The project did not require ethical approval as all data handled was open source. Furthermore, there were no questionnaires that were used that handled personal data.

Chapter 8 Next Steps and Future Improvements.

The relationships between entities are as follows;

- One-to-many relationship between Company and OS_Drug_Data as one company can have many applications for drug approval.
    - The foreign key in the OS_Drug_Data table will be Applicant_ID referring to the company that submitted the application.
- One-to-many relationship between Company and Users as one company can have many users utilising the system.
    - The foreign key in the Users table will be Employer_ID referring to the company that employs them.

The database design was normalised in order to avoid information redundancy and update anomalies [73]. There are three types of anomalies that the process of normalisation should hinder; update anomalies where not all instances of duplicate data have been updated, deletion anomalies where attributes are lost due to the deletion of other attributes and, finally, insertion anomalies where attributes cannot be inserted to the database without other attributes present. The database design underwent three stages of normalisation, so the final ER diagram is in 3NF. In order to be in 1NF any repeating attributes were moved to their own tables, 2NF introduced primary keys and contained only columns dependent on that key. Finally, 3NF ensures no columns are dependent on anything but the primary key. If this was the case any attributes dependent on an attribute that wasn't the primary key was moved to a separate table. [74]

A final consideration around the database design was that the users table proposes to store user passwords and this needed to be done in a secure way. The Open Web Application Security Project (OWASP) gives guidance on how to store passwords properly, some of their suggestions are; to hash the passwords as one of many steps, use a strong fixed length cryptographically strong random value (SALT) and to leverage adaptive one-way functions such as Argon2, PBKDF2, Scrypt or Bcrypt. [75]

As Figure 10 in 6.1.5 Technical Stack Diagram suggests, the system was designed to use the Flask Web Framework, which has methods for secure password storage. Flask has a Bcrypt extension that hashes passwords. Using a setter method would encrypt the plaintext password before it would ever be stored in the database. The Flask framework also gives methods for authentication and forgotten password reminders. [76]

6.1.7 Class Diagram

In order to design the structure of the system a class diagram has been constructed, showing the attributes, methods and relationships between objects that were needed for the system to work. The model created is high-level and conceptual with the aim to give insight into how the separate classes interact as part of the finished system. The system was intended to be modular, with each module handling different sources such as the FDA or TGA and following a similar structure to that shown in Figure 12 with minor modifications where necessary as each site may differ. This was done so that the system will be easily scalable, and changes made to any sites will only affect the specific module and not the entire system. Furthermore, this design partitioned the back-end and front-end meaning one can be updated without affecting the other.

For the detailed description of the diagram it was considered as if it was designed for the FDA website. The Get_Files and Get_Therapy classes were used to find the information; they both scraped websites to find the specific information they were looking for and Get_files returned unzipped csv files whereas Get_Therapy wrote the scraped data to a csv. The FDA website stores its drug approvals in a zip file which is updated monthly – the Get_Files class targeted these files specifically. However, this data does not contain what the drugs treat as, in the FDA's case, this is stored on separate web pages.

The Get_Uses class was designed with this in mind as it took the already gathered data by Get_Files and scraped the separate pages to append the use cases and their codes for each individual drug. Mirroring this process was the Get_Therapy and Get_Therapy_Area classes which found descriptions of drugs in different therapy areas.

The output of Get_Uses and Get_Therapy_Area fed into the Fuzzy_String_Comparison which was designed to compare the descriptions from the separate results and assign a therapy area to each use case through FL. The purpose of the Merge_Use_Therapy was to add these newly assigned therapy areas to the data returned by Get_Uses which created a complete data set.

The final aspect of the module was the Manipulate_Data which took the completed data set and fed it to different methods which returned subsets that were fed to the front-end and displayed to the user. These manipulations/subsets were designed separately in 6.1.9 Planned Data Modification as it was necessary to consider what would be useful to the target users.

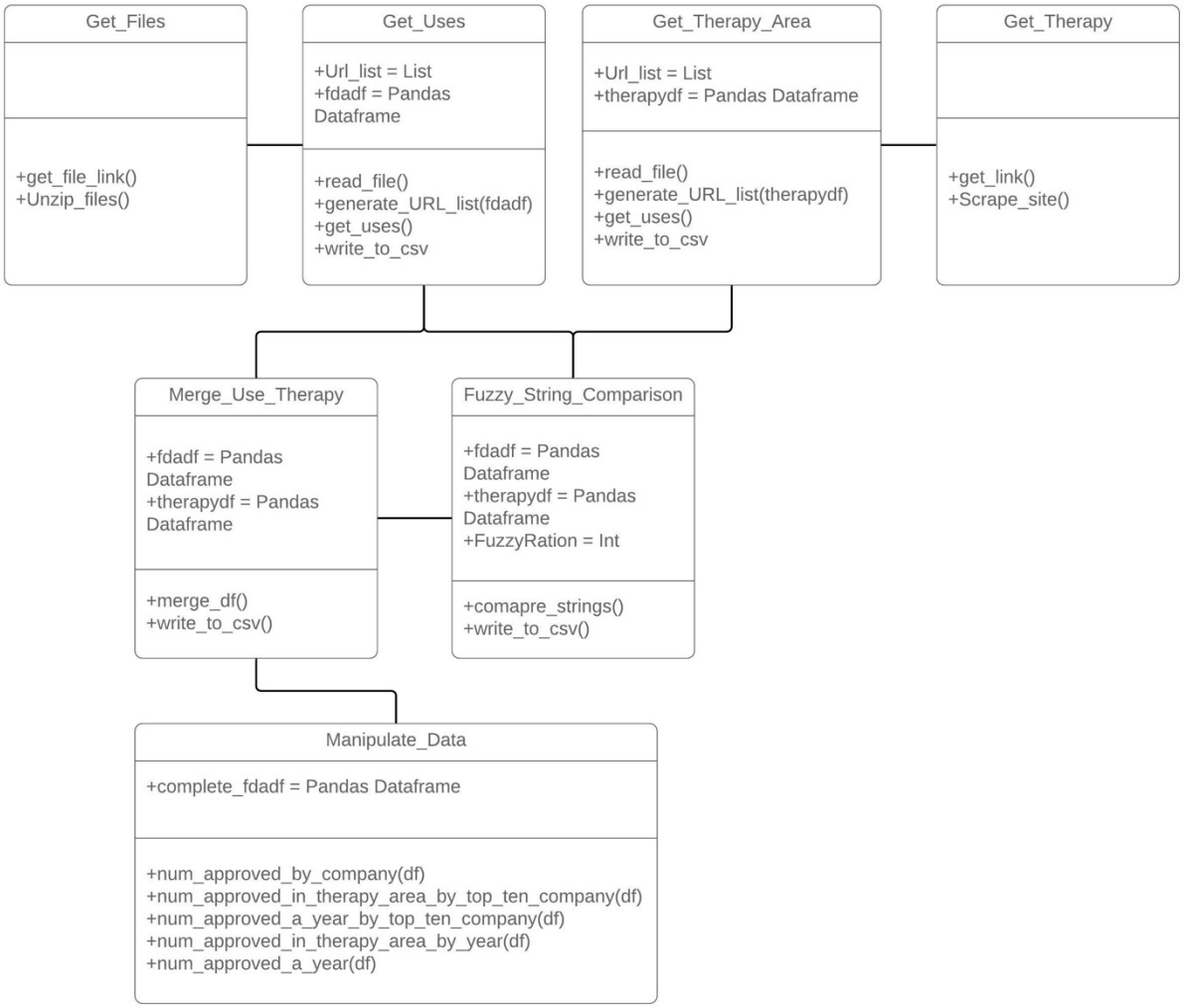


Figure 12 UML Class Diagram.

6.1.8 Sequence Diagram

The purpose of the sequence diagram in Figure 13 was to give a high-level idea of how the system will run, building on the class diagram from the previous section. This sequence diagram is related to the back-end processes, showing the necessary order for the data gathering, string comparison and data manipulation, which was fed back to the front-end where the Actor is viewing the site. The back-end needed to run in a particular order to ensure all the data for dependent methods has been gathered before they are called, this happened once a month when the target website updated its data. The Actor was not responsible for running the system, just able to access the front-end which was fed the results of the data manipulation once all the pre-requisites had been completed.

The first steps were to gather the drug data and therapy area data which then fed into the Get_Uses and Get_Therapy_Area classes which stored that data separately. The next stage was comparing the strings then merging and storing the results as a complete dataset that contained all the necessary data. The final stage was to perform pre-defined manipulations on the data and feed the results to the user interface (UI) so that the Actor can interact with the graphs.

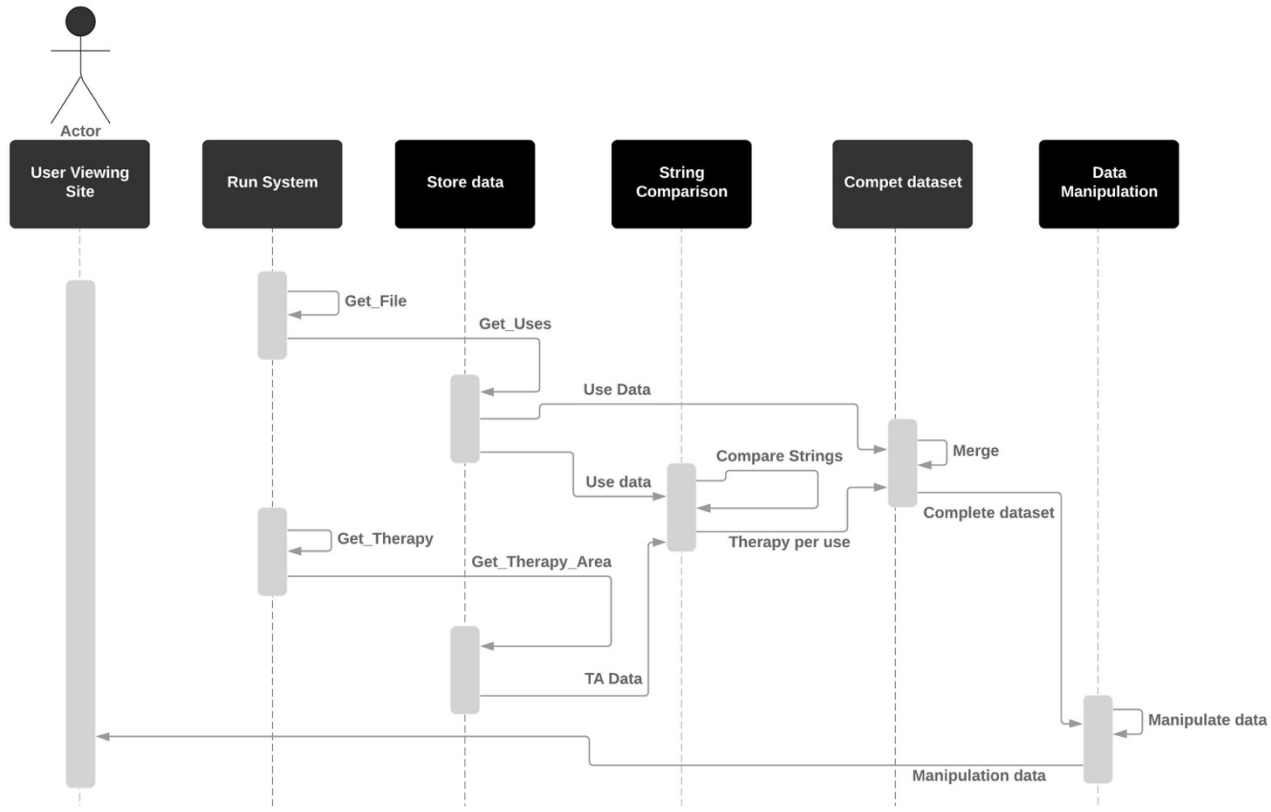


Figure 13 UML Sequence Diagram.

6.1.9 Planned Data Modification

In order to display data that is useful to the end user on the dashboard the following data sub sets were created and used as the basis for the graphs and charts that the user was able to filter interactively through the UI.

From the personas developed earlier in the design process, 6.1.1 Personas, the most useful information to each user was considered. For the CEO and VP personas overall trends comparing their company's number of approvals in geographic and therapeutic areas would be useful so that they can track performance and the affect investments have had on this. These metrics could also highlight areas that need further investment for them to be further ahead than their competitors. For the manager and team member seeing specific therapeutic area metrics would allow them to keep track of recent changes and approvals. Being able to see recent competitor approvals by therapy area would also help them flag the need for a strategic response to others in the organisation.

Therefore, the number of drugs that have been approved for each company will be useful too as it would highlight where their company ranks overall. Along this vein the number of drugs approved by year could also be useful, especially grouped by company or therapeutic group, as they would be able to see trends in the data over time, which the personas exercise highlighted as a particular area of interest. Other useful subsets may become apparent during the development stage.

6.2 Development

This section looks at the development of the system itself, how the design specified was implemented successfully and any necessary modifications that were made along the way. Firstly, the Proof of Concept (PoC) was considered to understand what the baseline of success would be. Then how each method chosen was implemented during development has been described along with successes and failures as well as deviations made from how the development was expected to happen along with the implications for the project. The technology stack from 6.1.5 Technical Stack Diagram will be referenced in terms of how each one was used.

6.2.1 Proof of Concept

It was decided early on in the development of the system that, as the design was modular, the PoC would be to create a module for a specific source and the UI. Therefore, the FDA was chosen as the first module to develop the data gathering solution for. This module would demonstrate the web scraping, NLP and data manipulation aspects of the project which can be considered the backbone as it can be replicated with minor modification for other countries. The UI would also need to be implemented for the system to reach PoC, as this aspect would be displaying the modified data and allow user interaction with the graphs and charts. For the PoC, the essential features are the data gathering and visualisation. The login/registration functionality as well as the rest of the website are peripheral to the true purpose of the project and therefore were not considered vital features in reaching PoC. In terms of MoSCoW prioritisation [77], the login/registration features are “could haves” as the system only handles open source data so there is no data-sensitive reason to implement this for the PoC. In Chapter 7 Social, Legal and Ethical Considerations

This section explores the social, legal and ethical considerations related to this project and the solution that was produced. Currently this is a proof of concept and therefore these considerations are based on if this solution were used by real customers.

7.1 Social

As the system handles open source data and enables users' access to data, there should not be any concerns regarding the systems purpose, however there are some mixed feelings towards automating processes as some systems can make human employees unnecessary. This is not the case with this system as it exists to enable the employees in regulatory affairs so that they can be more efficient with their time. This system could change perceptions of the importance of data analytics as it shows how having access to data can enable business decisions.

7.2 Legal

Legally, as this system is handling open source data there is no issues regarding GDPR. Regarding storing user data in the future, this will need to be stored securely in compliance with the Data Protection Act.

7.3 Ethical

The project did not require ethical approval as all data handled was open source. Furthermore, there were no questionnaires that were used that handled personal data.

Chapter 8 Next Steps and Future Improvements, the priority of these features may change.

6.2.2 Python

The programming language used for the back-end development was Python due to its suitability to the tasks being performed, as discussed in 3.1 Programming Language, namely web scraping and data analytics. Furthermore, previous experience programming in Python was a further advantage for the developer.

For the data gathering portion of the project there were to main aspects;

1. Gathering the data from website through web-scraping.
2. Working with the gathered data to create a complete dataset for the visualisation aspect of the system.

Python has libraries that are particularly suited to these tasks; BeautifulSoup for web-scraping and Pandas for data handling. Both of these libraries were utilised which allowed time to be managed more efficiently as it was possible to avoid having to develop this code from scratch.

The BeautifulSoup library [78] is used to extract data from HTML and XML files. It does this by reading the HTML or XML as a tree of python objects, henceforth know as Soup as shown in Code Snippet 1. It was then possible to parse the soup to find the tags containing the data that was needed. In this case, the FDA's approved drug zip file was the target so that it could be imported to the project, unzipped and used to gather additional data. This is a simple example of how BeautifulSoup was utilised during development. It is possible to gather data based on the class used in tags, or by a partial string contained in the text as shown in Code Snippet 2 as this functionality was used to find all use cases of each drug approved by the FDA.

Additionally, BeautifulSoup methods such as find_next_sibling() helped to generalise the code so that it worked for webpages with slight structural differences as shown in Code

Snippet 3. When looking for summaries of what a drug treats and its therapeutic area, the number of drugs changed per page. This method meant that the same code could be used in the case of every webpage shortening development time and the amount of code needed.

The second library heavily utilised in the development of this system was the Python Pandas library [79] which provides high-performance data structures and analysis tools which are easy to use. All the data handled throughout the development of this project was stored in a pandas DataFrame (DF) at some point. A DF is a 2D, mutable tabular structure that allows multiple column types which made it ideal for handling the variable open source data being collected by the system. Accessing data in a DF was made simple by functions such as iterrows [80], shown in Code Snippet 4, which made iterating through the rows of a DF object easy. In this case iterrows was used to compile a list of URLs from the data in a DF to be passed on to a web-scraper method to collect drug use cases.

DFs were easily created from CSV files as shown in Code Snippet 5. Here the three files scraped from FDA website containing drug, patent and exclusivity data were read into DFs. The method was easily implemented, specifying the delimiter used in the files (normally “,” is the default delimiter) and datatypes for specific columns using a Python dictionary [80]. This method of reading files was utilised throughout the project and the data-type dictionary grew with the number of attributes in a single data set. The final datatype dictionary is shown in Code Snippet 6.

Pandas allowed easy merging [81] and slicing of DFs, which eased the handling of datasets and their various subsets. Code Snippet 7 shows the three FDA DFs, defined in Code Snippet 5, being merged in order to create the initial FDA dataset which was the basis for all future datasets created during development. Conversely, Code Snippet 8 shows how this dataset was sliced to only contain application number, product number, application type and patent use code which was then fed to another code module which created a list of FDA URLs to be scraped for drug use cases.

In order to manipulate the complete FDA dataset, once all the scraping and merging of data had been done, Pandas functionality was once again utilised to create the subsets described

in 6.1.9 Planned Data Modification. Pandas groupby [80] and pivot [81] were used to create subsets of data to be passed to the front-end. The groupby function was used to subset the data whereas the pivot function helped reorganise the data into a format that was more easily interpreted by the visualisation methods on the front-end, as shown in Code Snippet 9. The resulting pivot table, Table 14, is the subset created that was then utilised by the front-end to visualise this particular view of the data.

Python as a language also had useful features that not only enabled but eased development of the system. The concise nature of the language significantly minimised the number of lines that were written and made basic programming paradigms such as loops and if statements quicker to implement. Features like this, were taken advantage of throughout the development of the system.

6.2.3 Therapeutic Group Assignment

Initially inspired by Fuzzy Logic (FL) applications in NLP, as mentioned in 2.2 Natural Language Processing, this aspect of the system was implemented as a string comparator as Python has a FuzzyWuzzy library [85] which performs string comparison using FL to measure the difference between two strings. The FuzzyWuzzy library returns different ratios indicating the similarity of the strings; ratio which returns a value indicating how exact the match is, partial ratio indicating whether there is a partial match, token sort ratio indicating whether the words in the strings match (matches out of order strings), token set ratio which is the same as sort but considers duplicate words as one and, finally, WRatio applies a weight to attempt to calculate the best score.

The approach taken looped through the use cases dataset and therapeutic group datasets, as DFs, scraped from websites in previous code modules comparing the description of each FDA use code against the description of each drug treatment in all therapeutic groups. For each comparison the different FuzzyWuzzy ratios were calculated, and for each use code if the combination of ratios was the best results it had gotten that therapy group was assigned to it, as shown in Code Snippet 10. This approach seemed to work initially, however when it came to manipulate the data it seemed that there were significantly more cardiovascular

drugs than any other. Revisiting the string comparator revealed that the majority of drugs were being misclassified as cardiovascular as it was the first therapy group each use case description was compared against. Furthermore, as many string's ratios were the same for more than one therapy group description they were still being miss-classified.

This realisation led to the development of a classifier module to replace the string comparator in order to correctly classify the use cases into therapeutic groups utilising the Python Scikit_Learn library as described by Susan Li [86]. This module also used the use cases and therapeutic group datasets but applies a Logistic Regression classification algorithm, as described in 3.5 Classification Models. The descriptions of treatments and the associated therapy group (class) was used as the basis for the training set. Each therapy group was given an ID and the treatment descriptions were vectorised to get the term frequency and inverse document frequency to create features of each therapy group, as shown in Table 15 Classifier feature set. This produced the two most correlated unigrams and bigrams associated with each therapy group that were used to train the model. The model then applied this "knowledge" to the use cases descriptions and assigned each one the therapeutic group with the best fit.

The accuracy of each model was calculated, as shown in Table 1, which suggested the Multinomial NB would perform the best. Despite this, the model used was the Logistic Regression, which supposedly was the second most accurate but upon a read through of the output seemed more accurate than the Multinomial NB model.

Table 10 Classification Model Accuracy.

Model	**Accuracy**
LinearSVC	0.213568
LogisticRegression	0.223501
MultinomialNB	0.239739
RandomForestClassifier	0.195605

6.2.4 Data Manipulation

Using the methods described in 6.2.2 Python, the designed data sets outlined in 6.1.9 Planned Data Modification were created and stored as subsets of the complete data set to be passed to the UI to be displayed as interactive graphs and charts that the users will be able to filter further based on their needs.

A dedicated code module was created to perform data manipulations by reading in the complete data set and passed it to different methods which were responsible for producing a specific view of the data. Code Snippet 11 shows how two of these methods took the DF, grouped the data based on what was under scrutiny, formatted it into a pivot table and then stored it in dedicated files that would be read by the web framework solution and passed on to the front-end which will be explained further in sections 6.2.5 Flask and 6.2.6 UI.

6.2.5 Flask

Flask is the Python web framework that was utilised to build the project which acts as a server on the local machine. A main.py file was written which includes all of the routes for web page navigation. Code Snippet 12 shows the route and method for serving the different webpages making the path the various html file names. The app was configured to run on port 80 and was run in debug mode as that allowed developer updates to the application to be shown after a page refresh rather than having to restart the application. The debug mode also gave insight into any errors encountered when attempting to run the application. Running the application in this way enabled quicker development times as changes made could be seen reflected in the application nearly immediately and it also helped when testing the application as the debug mode errors were used as a starting point when fixing issues.

Serving the webpages in this way meant that html pages in the web contents folder could be served through the Flask framework. Flask was also used to handle all GET and POST methods to and from the web application. This included registering a user, as shown in Code Snippet 14, which took input from the registration form and created a user object to be

stored. It also handled user login, as shown in Code Snippet 15, which requested the login information from the login form and then redirected the route to either the welcome page or failed login page depending on whether the credentials were correct. Flask was also used in this way to pass data to the front-end, as shown in Code Snippet 16, which shows the data for the number of approvals a year graphic being read from a file and passed to the web application.

6.2.6 UI

The UI was developed using a combination of HTML, CSS and JavaScript whilst also utilising open source libraries, bootstrap [87] and chart.js [88]. The UI was created to serve two purposes; give insight into the project and to visualise the data gathered by the rest of the system as described in 6.1.3 Wireframes.

The Bootstrap library was utilised throughout the development of the front-end in order to speed and ease the process. Code Snippet 17 shows how simple it was to import the Boostrap library into each web page. Furthermore, the Bootstrap library made adding basic features to the web application much simpler as it wasn't necessary to build them from scratch. The navigation bar was made using the Bootstrap navbar class as shown in Code Snippet 18. This navigation bar was used throughout the UI and was chosen as it was simple and fit well with the look of the UI, see Figure 17 which shows the UI homepage with the navbar across the top. Bootstrap classes also handled the issues of dynamic resizing as shown in Figure 18 where the navigation bar has re-configured itself dynamically to fit the new size of the window. Bootstrap classes such as cards were also utilised to make the layout of the web application more interesting. The features page of the website was developed using cards to highlight the three main features of the system and a user can choose to read more about them by expanding the card as shown in Figure 19 and Figure 20.

The UI was developed in two parts; pre-login and post-login. The development of the pre-login and post-login portions of the UI utilised different technologies and features in order to serve their intended purposes.

As mentioned, the pre-login site was intended to allow users to access the data visualisation dashboard by logging in and to give the user information about the system that had been developed. Earlier in this section the use of Bootstrap to present that information in a more creative way was described. Google Maps API was also utilised when developing the contact page to provide an interactive map of where the project was developed as shown in Figure 21.

The login/registration form was developed using Bootstrap classes, the developer's own CSS and JavaScript. It was also linked to the Flask web framework to pass the form data submitted by the user to register them or to authenticate them through the action and method designated to each form. The action corresponded with a specific Flask route which triggered that code, whilst the method described whether the form would POST or GET information. JavaScript functions were used to show the registration portion of the form when the user clicked register as shown in Figure 22, and then validate some information inputted to that form. The registration form was developed to only be seen if "Register" was clicked by the user, indicating they wanted to sign up. This was achieved using classes and a JavaScript function which shows/hides a class on a specific action, which in this case was clicking register as shown in Code Snippet 19.

The registration forms validation was controlled mostly through HTML by setting required lengths and types, as shown in Code Snippet 20. Setting specific types and length requirements enabled feedback to the user if the data they had entered was not valid as shown in Figure 23, Figure 24 and Figure 25. The password type hid the string as it was being typed and was also subject to length requirements, however a JavaScript function was written to add additional validation in this case – that the input to the "Password" and "Confirm Password" fields matched. As the user typed, the function was triggered as each key was released. The function compared the two strings that had been entered by the user and disabled the submit button until they were equal so that a user could not register unless the passwords they had entered were the same as shown in Code Snippet 21. In order to keep the user informed of the reason they were encountering trouble; a pop-up

box was implemented which explained that the passwords did not match as shown in Figure 26.

The post-login portion of the site was developed to function as the dashboard where users would be able to see the charts and graphs based on data from the back-end. The Chart.js library was imported, see Code Snippet 22, which was used to create interactive data visuals. JavaScript was used to create HTTP requests to get the information from Flask, the Flask side is shown in Code Snippet 16 and the front-end equivalent is shown in Code Snippet 23.

Chart.js takes data input and creates interactive graphs that can be displayed on websites. This library was utilised to ease this portion of development and to create consistent, good quality, interactive visualisations of the data collected by the system. Each chart displayed on the website was created using this library by specifying a chart type e.g line or bar, then the relevant data was inputted, and additional visual aspects were adjusted such as bar colour and line tension. Figure 27 shows an example of a graph that was displayed on the website and Figure 28 shows the same graph after it had been filtered through the UI.

6.3 Testing

The testing of the solution has been split into four parts; back-end testing, front-end testing, integration and User Acceptance Testing (UAT). The Back-end and front-end testing sections will concentrate on component testing to ensure that the individual pieces are working as they should, the integration testing will concentrate on how those components fit together and whether the entire system works as expected. Finally, UAT will look at what users think of the system.

6.3.1 Back-end Testing

Overall the back-end tests were successful, outlined in Table 11. Modifications had to be made to the loop gathering the use cases as it didn't handle multiple use cases for one drug initially but that was rectified. The biggest issue that was discovered when testing the back-

end was that the string comparator had miss-classified the majority of the drugs and therefore this module of code had to be replaced with the classifier module.

Table 11 Back-End tests.

Test ID	Test	Test Description	Acceptance Criteria	Outcome	Pass/Fail	Comments
BE1	Download drug files.	Getting the files which have the data about drugs approved, patents and exclusivity.	3 files should be downloaded, imported to the project and unzipped for later use.	A zip file was downloaded and unzipped containing exclusivity.txt, patents.txt and products.txt.	Pass	Figure 29
BE2	URL list – uses.	Creating a list of URLs to scrape based on the data from the drug files.	Same number of URLs as drugs in products file. URLs consist of a drugs appl_no, product_no and appl_type.	All URLs were created were valid and one was made for every application.	Pass	Table 16
BE3	Get Use descriptions.	For each URL scrape the page for how the drug is used.	Collect all uses of each drug.	Collected first use.	Fail - Table 17	A loop was added so that all use cases would be reordered. - Table 18
BE4	URL list – therapy areas.	Create a list of URLs to scrape for treatment descriptions of each drug in each therapy area.	Same number of URLs as therapy areas.	All URLs created were valid and there was one for each therapy area.	Pass	Table 19
BE5	Get treatment descriptions.	Get Information for each	Description of each drug	Each drug in each therapy area's	Pass	Table 20

		drug in each therapy group which contains treatment.	in each therapy area.	information was gathered.		
BE6	String Comparator.	Compare each use against each treatment and assign a therapeutic area to it based on similarity.	Uses are assigned therapy areas that usually deal with the illnesses treated by the drug.	Majority of drugs miss-classified e.g Plaque Psoriasis was classified as a cardio-vascular drug rather than dermatology.	Fail	Had to create the classifier to replace this module.
BE7	Classifier	Compare each use against each treatment and assign a therapeutic area to it based on similarity.	Uses are assigned therapy areas that usually deal with the illnesses treated by the drug.	Majority of drugs correctly classified e.g Plaque Psoriasis was classified as dermatology.	Pass	Table 21
BE8	Merge data	Create a complete dataset including all the data gathered.	Original data files are appended with the URL, use code, Use and Therapeutic area.	Complete dataset contains all necessary data.	Pass	
BE9	Data Manipulation	Data is grouped and formatted to be passed to the front-end.			Pass	

6.3.2 Front-end Testing

The front-end tests, detailed in Table 12, were mostly successful. An area that needed improving was the registration form validation as, initially, when comparing passwords, if they did not match, it was still possible to submit the form. To resolve this the submit button was disabled until the passwords matched.

Table 12 Front-end tests.

Test ID	Test	Test Description	Acceptance Criteria	Outcome	Pass/Fail	Comments
FE1	Registration Validation.	Cannot register unless all input is valid	Names are between 2 and 20 characters. Email is in email format. Password and Password confirmation match. Company is between 2 and 20 characters.	Couldn't submit form unless all criteria were met.	Fail	Initially the password matching function did not disable the submit button, this was added to resolve this issue.
FE2	Login Authentication.	User credentials are validated before accessing the dashboard.	User cannot access the system without valid credentials.	Invalid credentials – asked to try again. Valid Credentials – access the dashboard.	Pass	Figure 30
FE3	Google API	Interactive google map on contact us page.	Google map showing Reading university.	Map is fully interactive and links to cached google profile, zoomed in on Reading University initially.	Pass	Figure 21
FE4	Site Navigation	The user can navigate the site with the navigation bar.	All intended pages are reach from any page on the site.	The links all redirect to the correct webpage.	Pass	

FE5	Interactive Charts	Charts can be filtered on the page.	Data can be excluded or included from view by the user.	Filtering was possible on all graphs with a legend.	Pass	Figure 27, Figure 28
FE6	Logout	User can logout of the dashboard.	User is logged out and needs to resubmit their credentials to see the dashboard.	User redirected to the homepage.	Pass	

6.3.3 Integration Testing

The main thing that needed to be tested regarding the integration of the back-end and front-end was that the front-end could access the data needed to visualise what had been gathered, as shown in Table 13. To test that the method was working the data passed to the front-end for one of the charts was printed to the console so upon inspection of the page the data and corresponding labels was printed there.

Table 13 Integration Tests.

Test ID	**Test**	**Test Description**	**Acceptance Criteria**	**Outcome**	**Pass/Fail**	**Comments**
I1	Receiving Data	Front-end is receiving data through flask.	Data for a graph is printed to the console to show it was passed.	Data and labels seen in page inspection console.	Pass	Figure 31

6.3.4 UAT

Early on in the projects development it was presented to technology consultants at Splunk, a global data analytics leader as mentioned in 2.1 Current Solutions and Applications in HealthCare. They're feedback was extremely positive, saying they were impressed with what had been achieved as well as comparing the solution to Splunk and wondering why they hadn't been targeting this aspect of the healthcare industry with their own product.

Matthew Bevan, a manager at Splunk commented that the “project that resonates so intrinsically with a production ready solution offering from an organisation. Her presentation showed a grasp of some of the data challenges facing organisations today and how companies like Splunk attempt to address those”. His full letter can be seen in Figure 32. This support shows that experts of other data analytics solutions see the value in this project and its aim, validating the solution and what has been achieved.

Chapter 7 Social, Legal and Ethical Considerations

This section explores the social, legal and ethical considerations related to this project and the solution that was produced. Currently this is a proof of concept and therefore these considerations are based on if this solution were used by real customers.

7.1 Social

As the system handles open source data and enables users' access to data, there should not be any concerns regarding the systems purpose, however there are some mixed feelings towards automating processes as some systems can make human employees unnecessary. This is not the case with this system as it exists to enable the employees in regulatory affairs so that they can be more efficient with their time. This system could change perceptions of the importance of data analytics as it shows how having access to data can enable business decisions.

7.2 Legal

Legally, as this system is handling open source data there is no issues regarding GDPR. Regarding storing user data in the future, this will need to be stored securely in compliance with the Data Protection Act.

7.3 Ethical

The project did not require ethical approval as all data handled was open source. Furthermore, there were no questionnaires that were used that handled personal data.

Chapter 8 Next Steps and Future Improvements

The next step for this project would be to scale it up, as whilst it currently handles all therapeutic areas, new modules need to be added to handle multiple geographies by replicating the FDA PoC. The backbone has been created so with additional time modules handling other websites like the TGA could be added to build up a complete dataset of therapeutic and geographic drug approvals. As the solution is a partitioned system distinct data gathering modules will be necessary to interact with data from other geographies. A one-time change to the front-end will be necessary to add charts related to geographical comparison will be necessary which will then be used for any future modules added as it will just display the additional geographic data with each addition.

A future consideration would be to improve the look of the web application. Apart from the graphs and charts themselves the website is designed simply. An improvement would be for the design of the website to reflect the complexity and importance of the work done by the back-end of the system.

Section 6.1.6 Database alluded to future possibilities for the system regarding the use of company data. All Pharmaceutical companies have CCI which they measure open source data against to further inform them of their current status in the market. The solution in this current state is gathering and displaying the open source data, saving regulatory affairs time and making competitive intelligence analysis easier. However, as the system does not handle CCI and analyse this compared to the open source data it is only semi-automating the full task. Adding the capability to handle CCI and filter the dashboard to compare the open source data against the company of whoever is logged in will fully automate the entire process. In order to do this the database will need additional tables and the system's security will need to seriously be considered. Security is not really a concern of the system in its current state as the data is all readily available online. This would change drastically if the capability to handle CCI was introduced, as alluded to in 6.2.1 Proof of Concept .

Continuing on from this, notification functionality would also make the system more tailored to the user; if they have a drug that they are particularly interested in they it would

be useful for them to receive email notifications if an update is detected where that drug is mentioned.

Chapter 9 Conclusion and Reflection

9.1 Conclusion

The overall system that has been created throughout the course of this project is a success, as it fulfils its aim to automate the gathering, storage and reporting of open source drug approval data for the use of Regulatory Affairs teams in the Pharmaceutical industry for competitive intelligence analysis. Having a specific target audience and problem to solve served as motivation throughout the project to create something of a high standard that would truly be of use.

The end system has a complex back-end, automating the process of gathering large amounts of data, classifying it and manipulating it into simple views that can be passed to a web application to be displayed interactively to users to benefit their insight into the current drug market. The system was designed to a much larger scope than the PoC created so that that it can be developed further outside of this project, as it is true belief of the developer that this system has identified a gap in the market where current data analytics companies are not operating and the user experiencing the problem don't understand how they could implement something like this.

The skills used in the development of DataSource were learned across a number of years from a variety of modules including Python and Data Science Application, AI and Software, Quality and Testing as well as invaluable experience gained from a placement year in the Pharmaceutical industry. This, combined with the further research done into this problem and possible solutions, served to solidify the developers resolve the create a system that combined data gathering, classification and visualisation in an automated way to create DataSource: The Drug Approval Date Dashboard.

9.2 Reflection

This project was the result of experiences from my placement year at a large pharmaceutical industry where I observed the inefficient process regulatory affairs was following. Having spent time understanding the task I felt that there must be a way this task

could benefit from the implementation of some sort of system that could aid the team, if not automate the process completely. This motivated me to conduct research into possible solutions to the problem and led to the idea to create DataSource. Initially the problem seemed quite simple; gather data from websites such as the FDA, store it, create graphs to display the data. However, there were unforeseen complication that affected the project on more than one occasion.

Throughout this project I was increasingly challenged as I had been rather ambitious regarding what I wanted to achieve. I was passionate about solving a real-world problem with my project and, at first, expected the outcome to be an out-of-the-box solution that would automate the targeted business process. The scope soon become a proof of concept as I realised within the time there was no way a solution covering all geographies would be possible. I was very disappointed in myself as I wanted to include as much as I could, but I also realised that trying to do it all would result in an end product that did not reflect the possibilities of a system like this and therefore, a proof of concept was the best course of action. There were times where this project increased my faith in my own abilities as a programmer and computer scientist as I managed to create modules that successfully achieved the intended purpose. There were also times that I was extremely frustrated and was not sure that my aim was achievable as I did not believe that I was capable of creating this system. These moments were usually a result of something not working as expected or data differing from what was expected.

There were several good aspects of this project; as time went on skills that I was developing in other modules became increasing applicable to my chosen problem. My python module was invaluable in teaching me programming skills and approaches that shaped the way that data was handled by the system and research conducted for my AI module inspired the classification approach taken to assigning therapeutic groups to drugs. The research portion of this project was enjoyable, and I learnt so much as a result of doing my literature review and methodology.

As mentioned, there were challenges that arose that altered the course of the project at times. The data that was gathered proposed several issues as it did not contain what was

needed to be useful to the target users; the therapeutic areas. The initial data set did not even include descriptions of how the drugs would be used. When I realised this, it initially seemed like the end of the project as I wasn't sure how to handle it. Getting the use cases turned out to be rather simple as I just employed the same scraping method to that I used to get the initial data on each drug individually. However, assigning the therapeutic groups was the real challenge as this data was not actually available on the FDA website. I found a source which contained therapy areas and descriptions of drugs in those therapy areas and I realised I could use this to compare the use cases I already had against and assign each to a group this way. I employed fuzzy logic to compare strings and I thought this aspect of the project was complete but when I came to analyse the data, I realised that the majority of drugs had been miss-classified. This led to me being inspired by my AI coursework and treating it as a text classification problem from natural language processing and employing a logistic regression model to assign drugs to a therapy area which was much more accurate.

I'm not sure that there was more that I could have done as I am extremely proud of what I have achieved during the course of this project, but I do recognise that there are obvious next steps for this project, if it were to continue. As this was just a proof of concept the next steps would be to add the additional modules of code to handle other geographies which would add that additional utility to the system as regulatory affairs is interested in competition across geographical and therapeutic markets. In terms of the front-end, I tried my best but am not a strong front-end developer, and just wanted to create a viable web-application to demonstrate the solution. I would, ideally, like the front-end to reflect the back-end in terms of sophistication.

If I had to approach a project like this again, I would work more systematically and try and finish sooner. There were several factors outside of the project which seriously affected how much time I spent on this throughout the academic year, prioritising the closest deadline rather than looking at the bigger picture. I would also try to have more faith in my own abilities as I caused myself unnecessary stress and worry thinking I would not be able to produce something of a high enough quality to be successful and if I could stop those moments I would have been able to work much more efficiently.

Works Cited

[1] E. M. S. W. Ahmad Badr, “The contribution of CI to the strategic decision making process: Empirical study of the European pharmaceutical industry.,” *Journal of Competitive Intelligence & Management,* vol. 3, no. 4, pp. pp.15-35, 2006.

[2] EU, “The 2015 EU Industrial R&D Investment Scoreboard,” EU, 2015. [Online]. Available: http://iri.jrc.ec.europa.eu/scoreboard15.html. [Accessed 25 Jan 2019].

[3] M. M. Ann-Marie Craig, “Market structure and conduct in the pharmaceutical industry,” *Pharmacology & Therapeutics,* vol. 66, no. 2, pp. pp.301-337, 1995.

[4] P. N. S. T. a. V. G. Labhansh Atriwal, “Business Intelligence Tools for Big Data,” *Journal of Basic and Applied Engineering Research,* vol. 3, no. 6, pp. pp. 505-509, 2016.

[5] Y. Aspinall, “Competitive intelligence in the biopharmaceutical industry: The key elements,” *Business Information Review,* vol. 28, no. 2, pp. pp. 101-104, 2011.

[6] R. D. Steele, The New Craft of Intelligence: Personal, Public & Political, Oakton, VA: OSS International Press, 2002.

[7] C. S. Fleisher, “Using open source data in developing competitive and marketing intelligence,” *uropean Journal of Marketing,* vol. 42, no. 7/8, pp. pp. 852-866, 2008.

[8] Splunk, “Splunk,” Splunk, [Online]. Available: https://www.splunk.com/. [Accessed 14 Apr 2019].

[9] Splunk, “Cerner Corporation Achieves Real-Time Operational Visibility Into Complex Healthcare Transactions,” Splunk, [Online]. Available: https://www.splunk.com/pdfs/customer-success-stories/splunk-at-cerner.pdf. [Accessed 14 Apr 2019].

[10] Splunk, “Molina Healthcare Gains Healthy Advantage With Splunk Enterprise and Splunk ITSIv,” Splunk, [Online]. Available: https://www.splunk.com/pdfs/customer-success-stories/splunk-at-molina-healthcare.pdf. [Accessed 14 Apr 2019].

[11] Splunk, “Recursion Pharma Targets 100 Genetic Diseases With Splunk and Machine Learning,” [Online]. Available: https://www.splunk.com/pdfs/customer-success-stories/splunk-at-recursion-pharmaceuticals.pdf. [Accessed 14 Apr 2019].

[12] SAS, “The SAS® Platform,” [Online]. Available: https://www.sas.com/en_gb/software/platform.html. [Accessed 14 Apr 2019].

[13] SAS, “UMC Utrecht uses data analytics to proactively treat or even prevent infections in premature babies,” SAS. [Online]. Available: https://www.sas.com/en_gb/customers/umc-utrecht-data-analytics.html. [Accessed 14 Apr 2019].

[14] SAS, “Powerful analytics enables rich insights into ongoing clinical trials, enabling proactive intervention,” SAS, [Online]. Available: https://www.sas.com/en_gb/customers/sms-oncology.html. [Accessed 14 Apr 2019].

[15] How Natural Language Inference Models “Game” the Task of Learning, “How Natural Language Inference Models “Game” the Task of Learning,” Medium, 19 Apr 2018. [Online]. Available: https://medium.com/center-for-data-science/how-natural-language-inference-models-game-the-task-of-learning-61d2f744955c. [Accessed 07 Apr 2019].

[16] R. Taylor, “Project 4: OpenAI, Improving Language Understanding with Unsupervised Learning,” Own Work, Reading, 2019.

[17] S. Kumar, “AI Outperforms Humans in Question Answering,” Medium, 27 Mar 2018. [Online]. Available: https://medium.com/the-new-nlp/ai-outperforms-humans-in-question-answering-70554f51136b. [Accessed 07 Apr 2019].

[18] V. M. Atish Pawar, “Challenging the Boundaries of Unsupervised Learning for Semantic Similarity,” *IEEE Access,* vol. 7, pp. pp. 16291 - 16308, 2019.

[19] S. Bandal, “A Comprehensive Guide to Understand and Implement Text Classification in Python,” Analytics Vidhya, 23 Apr 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/. [Accessed 7 Apr 2019].

[20] L. A. Zadeh, “Fuzzylogic—apersonalperspective,” *Fuzzy Sets and Systems,* vol. 281, no. 0165-0114, pp. pp. 4-20, 2015.

[21] L. Zadeh, “PRUF—a meaning representation language for natural languages,” *International Journal of Man-Machine Studies,* vol. 10, no. 4, pp. pp. 395-460, 1978.

[22] F. Huang, “Learning Representations for Weakly Supervised Natural Language Processing Tasks,” *Computational linguistics - Association for Computational Linguistic,* vol. 40, no. 1, p. pp. 185, 2014.

[23] T. Mikolov, “Efficient Estimation of Word Representations in Vector Space,” 16 Jan 2013. [Online]. Available: https://arxiv.org/abs/1301.3781. [Accessed 13 Apr 2019].

[24] Y. H. L. A. K. Jain, “Classification of Text Documents,” *The Computer Journal,* vol. 41, no. 8, pp. pp. 537-546, 1998.

[25] F. Pereira, “Distributional clustering of English words,” 22 Aug 1994. [Online]. Available: https://arxiv.org/pdf/cmp-lg/9408011.pdf. [Accessed 13 Apr 2019].

[26] A. Stolcke, “Bayesian Learning of Probabilistic Language Models,” [Online]. Available: http://ftp.icsi.berkeley.edu/ftp/pub/ai/stolcke/thesis-abstract.pdf. [Accessed 13 Apr 2019].

[27] F. Melchert, “Aligning Process Automation and Business Intelligence to Support Corporate Performance Management,” 2004. [Online]. Available: https://www.alexandria.unisg.ch/66514/1/2004%2520-%2520Melchert,%2520Winter,%2520Klesse%2520-%2520Aligning%2520Process%2520Automation%2520and%2520Business%2520Intelligence%2520to%2520support%2520CPM.pdf. [Accessed 14 Apr 2019].

[28] P. Harmon, Business Process Change: A Business Process Management Guide for Managers and Process Professionals, San Francisco: Elsevier Science & Technology, 2014.

[29] P. Trkman, “The critical success factors of business process management,” *International Journal of Information Management,* vol. 30, no. 2, pp. pp. 125-134, 20120.

[30] Z. C. a. D. D. N. B Azvine, “Towards real-time business intelligence,” *BT Technology Journal,* vol. 23, no. 3, pp. pp. 214-225, 2005.

[31] S. O. Radu Prodan, “A Survey and Taxonomy of Infrastructure as a Service and Web Hosting Cloud Providers,” Institute of Computer Science, University of Innsbruck Technikerstraße 21a, Innsbruck, Austria, 2009.

[32] IBM, “IaaS, PaaS and SaaS – IBM Cloud service models,” IBM, [Online]. Available: https://www.ibm.com/cloud/learn/iaas-paas-saas. [Accessed 20 Feb 2019].

[33] Python, “Web Frameworks for Python,” Python, 12 Feb 2019. [Online]. Available: https://wiki.python.org/moin/WebFrameworks. [Accessed 20 Feb 2019].

[34] R. Brown, “Django vs Flask vs Pyramid: Choosing a Python Web Framework,” AirPair, [Online]. Available: https://www.airpair.com/python/posts/django-flask-pyramid#2-about-the-frameworks. [Accessed 20 Feb 2019].

[35] A. Petkov, “Here are the best programming languages to learn in 2018,” freeCodeCamp, 06 Jan 2018. [Online]. Available: https://medium.freecodecamp.org/best-programming-languages-to-learn-in-2018-ultimate-guide-bfc93e615b35. [Accessed 18 Jan 2019].

[36] Randstad, “decoding coding languages: comparing 11 popular programming languages,” Randstad, 04 Jan 2017. [Online]. Available: https://www.randstad.ca/job-seeker/job-tips/archives/comparing-11-popular-coding-languages_530/. [Accessed 18 Feb 2019].

[37] FreeCodeCamp, “Advantages and Disadvantages of JavaScript,” freeCodeCamp, [Online]. Available: https://guide.freecodecamp.org/javascript/advantages-and-disadvantages-of-javascript/. [Accessed 18 Feb 2019].

[38] University of Leicester, “Version control,” University of Leicester, [Online]. Available: https://www2.le.ac.uk/services/research-data/organise-data/version-control. [Accessed 05 Mar 2019].

[39] D. Spinellis, “Version Control Systems,” *IEEE Software,* vol. 22, no. 05, pp. pp. 108-109, 2005.

[40] I. Jovanović, “Software Testing Methods and Techniques,” *The IPSI BgD Transactions on Internet Research,* vol. 5, no. 1, pp. pp.30 - 41, 2009.

[41] H. S. David Janzen, “Test-Driven Development: Concepts, Taxonomy, and Future Direction,” IEEE Computer Society, Sep 2005. [Online]. Available: https://digitalcommons.calpoly.edu/cgi/viewcontent.cgi?referer=https://scholar.google.co.uk/&httpsredir=1&article=1034&context=csse_fac. [Accessed 20 Feb 2019].

[42] K.-J. S. Brian Fitzgerald, “Continuous Software Engineering and Beyond: Trends and Challenges,” Lero—The Irish Software Engineering Research Centre, Limerick, Ireland, 2014.

[43] D. Z. D. G. W. H. M. P. James Herbsleb, “Software Quality and the Capability Maturity Model,” *Communications of the ACM,* vol. 40, no. 6, pp. pp.30-40, 1997.

[44] V. V. Fred D. Davis, “Toward Preprototype User Acceptance Testing of New Information Systems: Implications for Software Project Management,” *IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT,* vol. 51, no. 1, pp. pp. 31-46, 2004.

[45] S. Dreiseitl, “Logistic regression and artificial neural network classification models: a methodology review,” *Journal of Biomedical Informatics ,* vol. 35, no. 5-6, pp. pp. 352-359, 2002.

[46] A. McCallum, “A Comparison of Event Models for Naive Bayes Text Classification,” 1994. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.65.9324&rep=rep1&type=pdf. [Accessed 15 Apr 2019].

[47] J. Brank, “Interaction of Feature Selection Methods and Linear Classification Models,” 2002. [Online]. Available: https://pdfs.semanticscholar.org/7c71/4989e5c3b05794dd3c394b582906a67dc2b4.pdf. [Accessed 15 Apr 2019].

[48] M. Khashei, “A novel hybrid classification model of artificial neural networks and multiple linear regression models,” *Expert Systems with Applications,* vol. 39, no. 3, pp. pp. 2606-2620, 2012.

[49] D. Pregibon, “Logistic Regression Diagnostics,” *The Annals of Statistics,* vol. 9, no. 4, pp. pp. 705-724, 1981.

[50] D. W. Hosmer, “Introduction to the Logisitc Regression Model,” in *Applied Logisitc Regression*, New Jersey, John Wiley & Sons, 2000, pp. pp. 1-56.

[51] L. e. al., “Choosing the right NoSQL database for the job: a quality attribute evaluation,” *Journal of Big Data,* vol. 18, no. 2, pp. pp. 1-26, 2015.

[52] MongoDB, “Sharding,” MongoDB, [Online]. Available: https://docs.mongodb.com/manual/sharding/. [Accessed 15 Apr 2019].

[53] M.-G. Jung, S.-A. Youn, J. Bae and Y.-L. Choi, “A Study on Data Input and Output Performance Comparison of MongoDB and PostgreSQL in the Big Data Environment,” in *2015 8th International Conference on Database Theory and Application (DTA)*, Jeju, 2015.

[54] “Stage 3: Plan the Project,” University of Wisconsin, 1 February 2006. [Online]. Available: https://pma.doit.wisc.edu/plan/3-2/print.html. [Accessed 15 October 2018].

[55] D. R. Wallace and R. U. Fujii, “Software Verification and Validation: An Overview,” *IEEE Software,* vol. 6, no. 3, pp. 10-17, May/Jun 1989.

[56] contributor, “The Continuous Delivery Pipeline — What it is and Why it’s so Important in Developing Software,” DevOps, 29 July 2014. [Online]. Available: https://devops.com/continuous-delivery-pipeline/. [Accessed 17 October 2018].

[57] K. S. &. J. Sutherland, “What is SCRUM?,” SCRUM.org, 2018. [Online]. Available: https://www.scrum.org/resources/what-is-scrum. [Accessed 16 October 2018].

[58] S. C. a. B. Straub, “1.1 Getting Started - About Version Control,” Git, 2014. [Online]. Available: https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control. [Accessed 20 October 2018].

[59] Rutvi, “Quality Goals: Define and Measure Quality of your application,” Infostretch, 22 November 2009. [Online]. Available: https://www.infostretch.com/blog/quality-goalsdefine-and-measure-quality-of-your-application/. [Accessed 21 October 2018].

[60] ISO/IEC, “ISO/IEC 25010:2011,” ISO, 2017 (Last Review). [Online]. Available: https://www.iso.org/standard/35733.html. [Accessed 21 October 2018].

[61] S. A. Lowe, “9 metrics that can make a difference to today’s software development teams,” TechBeacon, 6 June 2016. [Online]. Available: https://techbeacon.com/9-metrics-can-make-difference-todays-software-development-teams. [Accessed 22 October 2018].

[62] ProjectCodeMeter, “Weighted Micro Function Points,” Project Code Meter, [Online]. Available: http://www.projectcodemeter.com/cost_estimation/help/GL_wmfp.htm. [Accessed 22 October 2018].

[63] IBM, “Halstead Metrics,” IBM, [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SSSHUF_8.0.2/com.ibm.rational.testrt.studio.doc/topics/csmhalstead.htm. [Accessed 22 October 2018].

[64] IBM, “V(g) or Cyclomatic Number,” IBM, [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SSSHUF_8.0.2/com.ibm.rational.testrt.studio.doc/topics/csmcyclomatic.htm. [Accessed 22 October 2018].

[65] Sealights, “Code Coverage Metrics,” Sealights, [Online]. Available: https://www.sealights.io/test-metrics/code-coverage-metrics/. [Accessed 22 October 2018].

[66] Casr, “Understanding Risk Management in Software Development,” Cast, [Online]. Available: https://www.castsoftware.com/research-labs/risk-management-in-software-development-and-software-engineering-projects. [Accessed 22 October 2018].

[67] U.S. Department of Agriculture's (USDA) Economic Research Service (ERS)., “Personas,” Usability.gov, [Online]. Available: https://www.usability.gov/how-to-and-tools/methods/personas.html. [Accessed 16 Apr 2019].

[68] [Online]. Available: https://canny.io/.

[69] [Online]. Available: https://branch.io/.

[70] [Online]. Available: https://stellar.io/.

[71] Mixpanel, “What is a technology stack,” Mixpanel, [Online]. Available: https://mixpanel.com/topics/what-is-a-technology-stack/. [Accessed 16 Apr 2019].

[72] C. SINGH, “Entity Relationship Diagram – ER Diagram in DBMS,” Beginners Book, 2015. [Online]. Available: https://beginnersbook.com/2015/04/e-r-model-in-dbms/. [Accessed 16 Apr 2019].

[73] SQA, “The Process of Normalisation,” SQA, 2007. [Online]. Available: https://www.sqa.org.uk/e-learning/MDBS01CD/page_20.htm. [Accessed 17 Aor 2019].

[74] SQA, “Summary of Normalisation Rules,” SQA, 2007. [Online]. Available: https://www.sqa.org.uk/e-learning/MDBS01CD/page_34.htm. [Accessed 17 Apr 2019].

[75] J. Steven, “Password_Storage_Cheat_Sheet.md,” OWASP, [Online]. Available: https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Password_Storage_Cheat_Sheet.md. [Accessed 17 Apr 2019].

[76] R. Picard, “Patterns for handling users,” Flask, 2014. [Online]. Available: http://exploreflask.com/en/latest/users.html. [Accessed 2017 Apr 2019].

[77] Agile Business Consortium, “MoSCoW Prioritisation,” Agile Business Consortium, 2014 Onwards. [Online]. Available: https://www.agilebusiness.org/content/moscow-prioritisation. [Accessed 17 Apr 2019].

[78] BeautifulSoup, “BeautifulSoup Documentation,” BeautifulSoup, [Online]. Available: https://www.crummy.com/software/BeautifulSoup/bs4/doc/. [Accessed 17 Apr 2019].

[79] Pydata, “10 Minutes to Pandas,” Pydata, [Online]. Available: https://pandas.pydata.org/pandas-docs/version/0.22/10min.html. [Accessed 17 Apr 2019].

[80] [Online]. Available: https://pandas.pydata.org/pandas-docs/version/0.22/basics.html#iterrows.

[81] “5.5 Dictionaries,” [Online]. Available: https://docs.python.org/3/tutorial/datastructures.html#dictionaries.

[82] [Online]. Available: https://pandas.pydata.org/pandas-docs/version/0.22/merging.html.

[83] [Online]. Available: https://pandas.pydata.org/pandas-docs/version/0.22/groupby.html#groupby-sorting.

[84] [Online]. Available: https://pandas.pydata.org/pandas-docs/version/0.22/reshaping.html.

[85] GeeksforGeeks, “FuzzyWuzzy Python library,” GeeksforGeeks, [Online]. Available: https://www.geeksforgeeks.org/fuzzywuzzy-python-library/. [Accessed 10 Apr 2019].

[86] S. Li, “Multi-Class Text Classification with Scikit-Learn,” Towards Data Science, 19 Feb 2019. [Online]. Available: https://towardsdatascience.com/multi-class-text-classification-with-scikit-learn-12f1e60e0a9f. [Accessed 11 Apr 2019].

[87] [Online]. Available: https://getbootstrap.com/docs/4.3/getting-started/introduction/.

[88] [Online]. Available: https://www.chartjs.org/.

[89] “V Model,” Professional QA, 06 September 2016. [Online]. Available: http://www.professionalqa.com/v-model. [Accessed 17 October 2018].

[90] ISO/ISE, “Product Quality - ISO/IEC 25010,” ISO, 2017(Last Reviewed). [Online]. Available: https://edisciplinas.usp.br/pluginfile.php/294901/mod_resource/content/1/ISO%2025010%20-%20Quality%20Model.pdf. [Accessed 21 October 2018].

Table of Figures

Table of Tables

Table of Code Snippets

Appendix

IndexCases

Wireframes

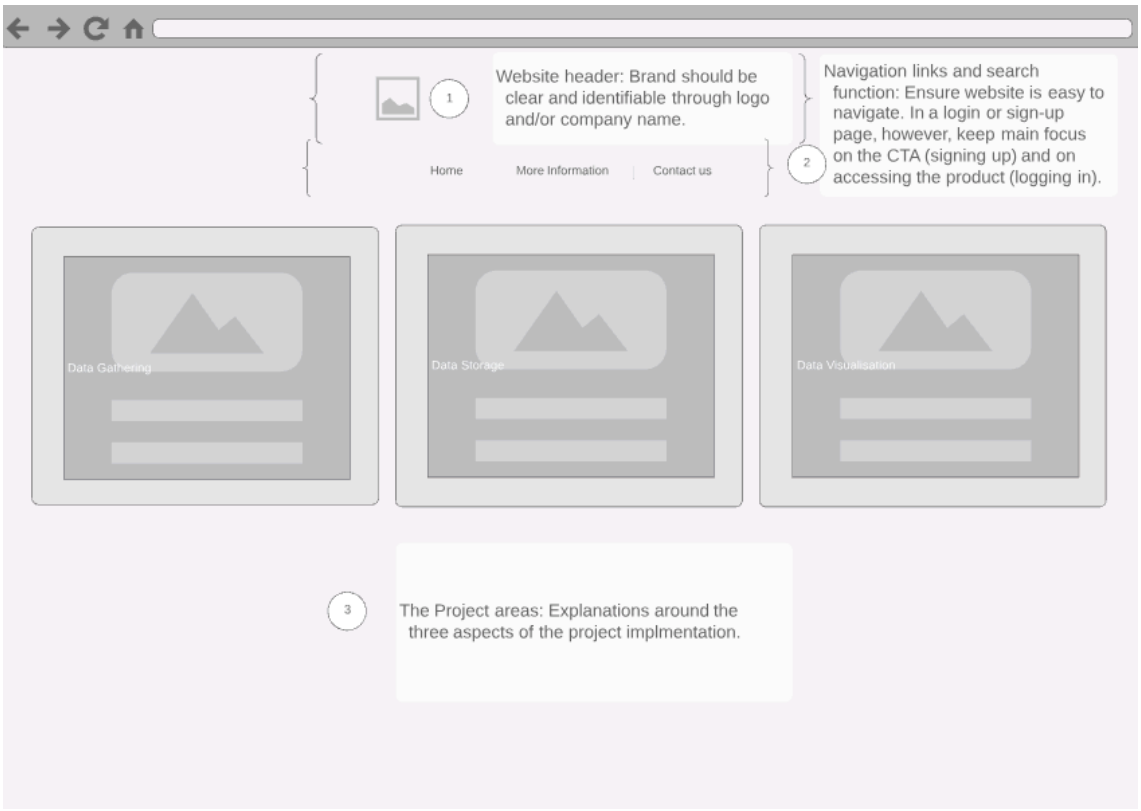


Figure 14 Project Breakdown page Wireframe.

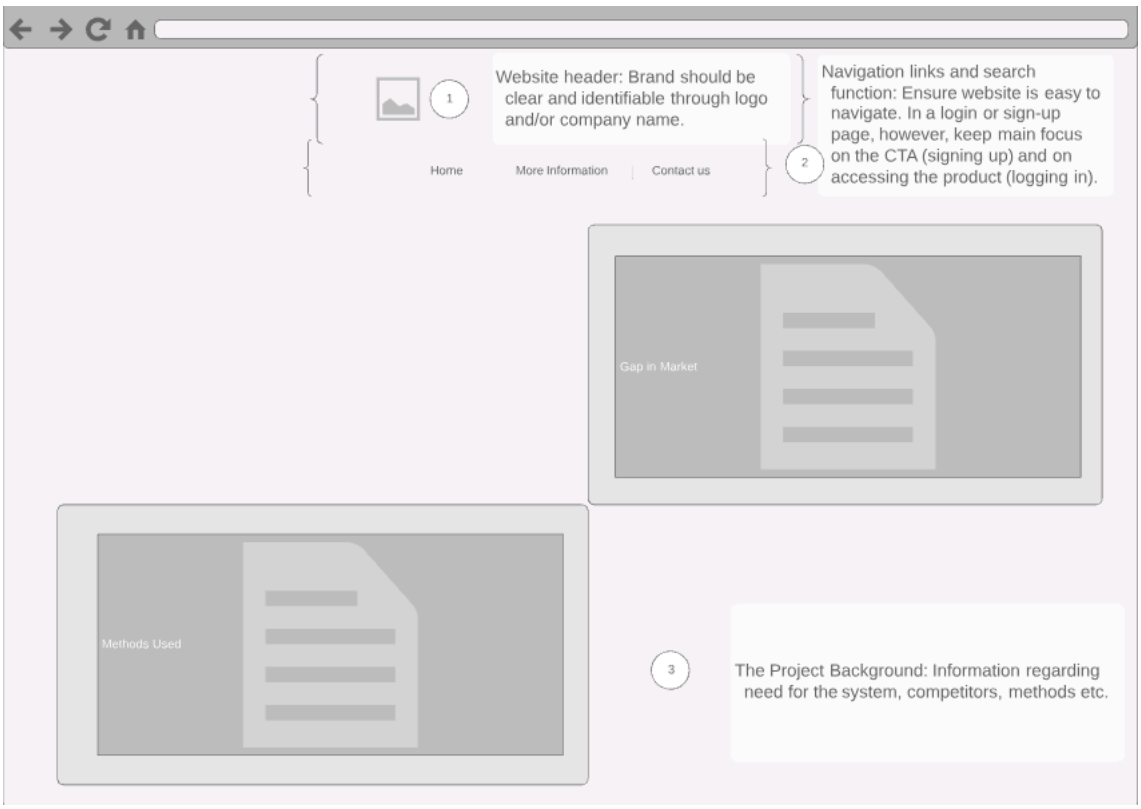


Figure 15 Project Background information page wireframe.

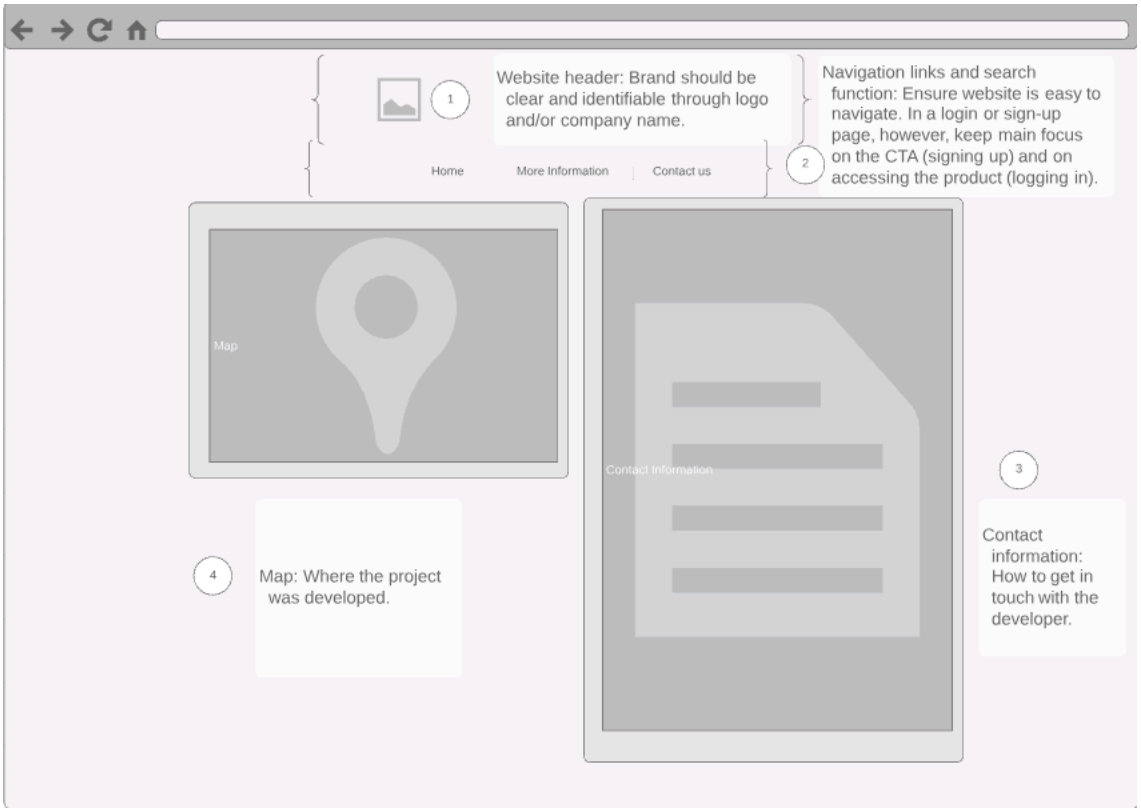


Figure 16 Project Contact Us page wireframe.

Code & Output

```
fdaSite = 'https://www.fda.gov/Drugs/InformationOnDrugs/ucm129662.htm'

page_html = urllib.request.urlopen(fdaSite)
soup = BeautifulSoup(page_html, 'html.parser')
```

Code Snippet 1 Reading HTML into a soup object.

```
use = soup.find_all(class_="tooltiptext")
useCodes= soup.find_all(text=re.compile('U-'))
numUseCodes= len(soup.find_all(text=re.compile('U-')))
```

Code Snippet 2 Gathering Use Codes from FDA web pages with BeautifulSoup.

```
for section in soup.find_all('h3'):
                nextNode = section
                while True:
                    nextNode = nextNode.find_next_sibling()
                    if nextNode and nextNode.name == 'p':
                        print(nextNode.text)
                        use_writer.writerow([urlList[urls],nextNode.text])
                    else:
                        break
```

Code Snippet 3 Using BeautifulSoup's find_next_sibling_method() to gather treatment summaries and therapeutic groups for each drug approved by the FDA.

```
def createURL(df):
    constructedURL = []

    with open('FDA_Module/urls.csv', mode = 'w') as url_file:
        use_writer = csv.writer(url_file, delimiter=',')
        use_writer.writerow(['Product_No','Appl_No','Appl_Type','URL'])
        for index, row in df.iterrows():
            prodNo = row['Product_No']
            appNo = row['Appl_No']
            appType = row['Appl_Type']
            constructURL =
'https://www.accessdata.fda.gov/scripts/cder/ob/patent_info.cfm?Product_No={}&Appl_
No={}&Appl_type={}'
            constructedURL.append(constructURL.format(prodNo,appNo,appType))
            print(index)
            use_writer.writerow([prodNo,appNo,appType,constructedURL[index]])

        return constructedURL
```

Code Snippet 4 Method utilising iterrows to construct a list of URLs from the data in a DF to be passed to a web-scraping method.

```
dtype_dic= {'Appl_No': str,
            'Product_No' : str}

drugsFDAdf = pd.read_csv("FDA_Module/tmpfdazip/products.txt", delimiter='~', dtype
= dtype_dic)
patentFDAdf = pd.read_csv("FDA_Module/tmpfdazip/patent.txt", delimiter='~', dtype =
dtype_dic)
exclusivityFDAdf = pd.read_csv("FDA_Module/tmpfdazip/exclusivity.txt",
delimiter='~', dtype = dtype_dic)
```

Code Snippet 5 Reading CSVs imported from the FDA website into Panda DFs specifying column data types with a Python Dictionary.

```
dtype_dic = {' Use Code': str,
             'Use': str,
             'Appl_No': str,
             'Product_No': str,
             'Type' : str,
             'Patent_No' : str,
             'Therapeutic_Area' : str
             }
```

Code Snippet 6 Full Data Type Dictionary for Complete Dataset.

```
def createFDA_df():

    patentFDAdfCleaned = patentFDAdf[['Appl_No', 'Product_No',
'Patent_No','Patent_Use_Code','Submission_Date', 'Appl_Type']]
    exclusivityFDAdfCleaned = exclusivityFDAdf[['Appl_No',
'Exclusivity_Code','Exclusivity_Date']]
    drugsFDAdfCleaned = drugsFDAdf[['Appl_No', 'Trade_Name', 'Ingredient',
'Applicant', 'Approval_Date', 'Type']]

    FDAdf = pd.merge(patentFDAdfCleaned, exclusivityFDAdfCleaned, on='Appl_No')
    FDAdf1 = pd.merge(drugsFDAdfCleaned, FDAdf,  on='Appl_No')

    return FDAdf1
```

Code Snippet 7 Method for creating the initial FDA dataset using merge().

```
def useSearch():
    patentFDAdfCleaned = patentFDAdf[[ 'Product_No','Appl_No',
'Appl_Type','Patent_Use_Code']]

    return patentFDAdfCleaned
```

Code Snippet 8 Method creating slice of initial FDA dataset for use in another code module.

```
def num_approved_in_therapy_area_by_year(df):
    appDate_therapy =
df.groupby(['Therapeutic_Area','Approval_Year'])['Appl_No'].count()
    appDate_therapy = appDate_therapy.to_frame(name = 'count')
    pivot = appDate_therapy.pivot_table(index='Therapeutic_Area',
columns='Approval_Year', values='count')
    pivot[np.isnan(pivot)] = 0

    (pivot).to_csv('FDA_Module/Data_Manipulation/approved_a_year_by_therapy.csv')


    return None
```

Code Snippet 9 Method for extracting a subset of data from the complete data set, using Pandas groupby and pivot functions, that will be passed to the front-end to be visualised as a chart.

Table 14 Resulting pivot table showing the number of drugs approved a year in each therapy area.

Therapeutic_Area	**1999**	**2001**	**2002**	**2003**	**2004**	**2005**	**2006**	**2007**	**2008**	**2009**	**2010**	**2011**	**2012**	**2013**	**2014**	**2015**	**2016**	**2017**	**2018**
['CARDIOLOGY/ VASCULAR DISEASES (122) ']	0	0	0	1	0	0	0	0	1	1	0	0	1	2	1	4	2	1	1
['DERMATOLOGY (109) ']	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3	1	3	3
['ENDOCRINOLOGY (229) ']	0	0	0	0	0	0	0	0	0	1	2	0	1	2	3	2	3	8	4
['FAMILY MEDICINE (624) ']	2	1	1	0	0	3	2	2	1	2	3	5	7	7	10	13	11	10	24
['GASTROENTEROLOGY (105) ']	0	0	0	0	0	0	0	0	1	2	0	0	0	0	1	0	1	2	0
['HEMATOLOGY (135) ']	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	3
['HEPATOLOGY (LIVER, PANCREATIC, GALL BLADDER) (54) ']	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1	3
['IMMUNOLOGY (216) ']	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	2	2	0	1
['INFECTIONS AND INFECTIOUS DISEASES (211) ']	0	0	0	1	0	0	0	1	0	0	1	1	3	2	6	2	5	5	3
['MUSCULOSKELETAL (109) ']	0	0	0	0	0	0	0	0	0	0	1	0	0	2	0	0	2	1	0
['NEPHROLOGY (88) ']	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	2	1
['NEUROLOGY (189) ']	0	0	0	1	0	0	0	0	3	2	4	2	0	1	2	4	7	6	6
['OBSTETRICS/ GYNECOLOGY (WOMEN'S HEALTH) (120) ']	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	2	2	0	0
['ONCOLOGY (263) ']	0	0	1	0	2	1	2	1	2	0	4	4	7	5	5	8	4	12	9
['OPHTHALMOLOGY (56) ']	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	3	0
['OTOLARYNGOLOGY (EAR, NOSE, THROAT) (23) ']	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
['PEDIATRICS/ NEONATOLOGY (124) ']	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

['PHARMACOLOGY/ TOXICOLOGY (32) ']	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
['PSYCHIATRY/ PSYCHOLOGY (73) ']	0	0	1	0	1	0	3	0	0	3	0	2	0	4	1	7	0	2	2
['PULMONARY/ RESPIRATORY DISEASES (135) ']	0	0	0	0	0	0	0	0	0	0	0	0	1	2	4	2	2	0	4
['UROLOGY (54) ']	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

```
for indexB,row in use_codes.iterrows():
        useSTR = row.Use
        for indexA, row in treatments.iterrows():
            treatmentSTR = row.treatment
            WRatio = fuzz.WRatio(treatmentSTR, useSTR)
            PRatio = fuzz.partial_ratio(treatmentSTR, useSTR)
            TSortRatio = fuzz.token_sort_ratio(treatmentSTR, useSTR)
            TSetRatio = fuzz.token_set_ratio(treatmentSTR, useSTR)
            Ratio = fuzz.ratio(treatmentSTR, useSTR)
            if Ratio >= highest_Ratio and WRatio >= highest_WRatio and PRatio >=
highest_PRatio and TSetRatio >= highest_TSetRatio and TSortRatio >=
highest_TSortRatio :
                highest_WRatio = WRatio
                highest_PRatio = PRatio
                highest_TSortRatio = TSortRatio
                highest_TSetRatio = TSetRatio
                highest_Ratio = Ratio
                useCode = use_codes[' Use Code'][indexB]
                tarea = treatments['Therapy_area'][indexA]
        use_writer.writerow([useCode, useSTR, treatmentSTR, tarea, highest_WRatio,
highest_PRatio, highest_TSortRatio,highest_TSetRatio,highest_Ratio])

        print( useCode, useSTR,treatmentSTR,  tarea,  highest_PRatio,
highest_WRatio, highest_Ratio, highest_TSetRatio, highest_TSortRatio)
```

Code Snippet 10 String Comparator code.

Table 15 Classifier feature set.

Therapy Area	**Most Correlated**	
	Unigrams	**Bigrams**
CARDIOLOGY/VASCULAR DISEASES	heart hypertension	treatment high treatment hypertension
DENTAL AND ORAL HEALTH	patch adult	canker sores treatment canker
DERMATOLOGY	acne rosacea	plaque psoriasis treatment acne
DEVICES	pulmonary obstructive	chronic obstructive obstructive pulmonary
ENDOCRINOLOGY	ii diabetes	ii diabetes treatment type
FAMILY MEDICINE	migraine diabetes	treatment hypertension treatment migraine
GASTROENTEROLOGY	ulcerative constipation	idiopathic constipation treatment heartburn
GENETIC DISEASE	epilepsy seizures	cystic fibrosis treatment hemophilia
HEALTHY VOLUNTEERS	grass subtypes	rhinitis conjunctivitis immunization influenza
HEMATOLOGY	lymphoma hemophilia	treatment thrombocytopenia treatment hemophilia
HEPATOLOGY (LIVER, PANCREATIC, GALL BLADDER)	hcv hepatitis	treatment hepatitis chronic hepatitis
IMMUNOLOGY	infection hiv	hiv infection treatment hiv
INFECTIONS AND INFECTIOUS DISEASES	infection hiv	hiv infection treatment hiv
INTERNAL MEDICINE	amyloidosis	amyloidosis adults

	hyperuricemia	polyneuropathy hereditary
MUSCULOSKELETAL	osteoporosis arthritis	multiple sclerosis relapsing multiple
NEPHROLOGY	overactive kidney	overactive bladder treatment overactive
NEUROLOGY	parkinson migraine	treatment migraine parkinson disease
NUTRITION AND WEIGHT LOSS	obesity weight	weight management chronic weight
OBSTETRICS/GYNECOLOGY (WOMEN'S HEALTH)	pregnancy contraception	vasomotor symptoms breast cancer
ONCOLOGY	leukemia cancer	breast cancer lung cancer
OPHTHALMOLOGY	inflammation ocular	ocular hypertension glaucoma ocular
ORTHOPEDICS/ORTHOPEDIC SURGERY	painful bone	osteoporosis bone adults active
OTOLARYNGOLOGY (EAR, NOSE, THROAT)	media otitis	allergic rhinitis otitis media
PEDIATRICS/NEONATOLOGY	pediatrics children	treatment attention attention deficit
PHARMACOLOGY/TOXICOLOGY	induced reversal	chemotherapy induced induced nausea
PODIATRY	warts perianal	perianal warts treatment external
PSYCHIATRY/PSYCHOLOGY	depression schizophrenia	treatment depression treatment schizophrenia

PULMONARY/RESPIRATORY DISEASES	pulmonary asthma	obstructive pulmonary treatment asthma
RARE DISEASES AND DISORDERS	intraepithelial epileptic	epileptic seizures treatment epileptic
RHEUMATOLOGY	rheumatoid arthritis	treatment rheumatoid rheumatoid arthritis
SLEEP	2014 insomnia	january 2014 treatment insomnia
TRAUMA (EMERGENCY, INJURY, SURGERY)	operative analgesia	treatment neuropathic neuropathic pain
UROLOGY	impotence prostate	treatment prostate prostate cancer
VACCINES	zoster immunization	disease caused active immunization

```
def num_approved_in_therapy_area_by_top_ten_company(df):
    appDate_therapy =
df.groupby(['Applicant','Therapeutic_Area'])['Appl_No'].count()
    appDate_therapy = appDate_therapy.to_frame(name = 'count')

appDate_therapy=appDate_therapy.sort_values(by='count',ascending=False).head(11)
    pivot2 = appDate_therapy.pivot_table(index='Applicant',
columns='Therapeutic_Area', values='count')
    pivot2.dropna()
    pivot2= pivot2.head(10)
    pivot2[np.isnan(pivot2)] = 0

(pivot2).to_csv('FDA_Module/Data_Manipulation/num_approved_in_therapy_area_by_top_t
en_company.csv')
    return None

def num_approved_a_year_by_top_ten_company(df):
    appDate_therapy = df.groupby(['Applicant','Approval_Year'])['Appl_No'].count()
    appDate_therapy = appDate_therapy.to_frame(name = 'count')

appDate_therapy=appDate_therapy.sort_values(by='count',ascending=False).head(11)
    pivot2 = appDate_therapy.pivot_table(index='Applicant',
columns='Approval_Year', values='count')
    pivot2.dropna()
    pivot2[np.isnan(pivot2)] = 0

(pivot2).to_csv('FDA_Module/Data_Manipulation/num_approved_a_year_by_top_ten_compan
y.csv')
    return None
```

Code Snippet 11 Two methods from the data manipulation code module.

```
@app.route("/<path:path>")
def serve_webpage(path):
    return flask.send_from_directory('./web_contents/', path)
```

Code Snippet 12 Main route and method for serving the web pages with FLASK.

```
app.run(debug=True, port=80)
```

Code Snippet 13 App Run Configuration

```
@app.route("/register", methods=["POST"])
def register_post():

    first_name = flask.request.form['f_name']
    last_name = flask.request.form['l_name']
    password = flask.request.form['password']
    c_password = flask.request.form['c-password']
    email = flask.request.form['email']
    company = flask.request.form['company']
    print(first_name + last_name )
    try:
        user=User(
            f_name = first_name,
            l_name = last_name,
            password = password,
            email = email,
            company = company
        )
        db.session.add(user)
        db.session.commit()
        return '{},{},{},{},{},{}'.format(first_name, last_name, password,
c_password, email, company)
    except Exception as e:
        return(str(e))
```

Code Snippet 14 Flask Registration POST method.

```
@app.route("/login", methods=["POST", "GET"])
def login():
        #populate_OS_db()
        email = flask.request.form['email-login']
        password = flask.request.form['password-login']
        if(email == 'test@gmail.com' and password == 'test1'):
            return flask.redirect('welcome.html')
        else:
            flask.flash('Login Unsuccessful')
            return flask.redirect('failed-login.html')
```

Code Snippet 15 Flask Login User POST GET method.

```
@app.route("/data", methods=["GET"])
def getdata():
    with open('./FDA_Module/Data_Manipulation/approval_a_year_data.csv', 'r') as
file:
        data = file.read().replace('\n', ',')
    return data

@app.route("/label", methods=["GET"])
def getlabel():
    with open('./FDA_Module/Data_Manipulation/approval_a_year_label.csv', 'r') as
file:
            label = file.read().replace('\n', ',')
    return label
```

Code Snippet 16 Flask methods for passing data to the web application.

```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.0/css/bootstrap.min.css"
integrity="sha384-PDle/QlgIONtM1aqA2Qemk5gPOE7wFq8+Em+G/hmo5Iq0CCmYZLv3fVRDJ4MMwEA"
        crossorigin="anonymous">
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
        crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
        crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.0/js/bootstrap.min.js"
integrity="sha384-7aThvCh9TypR7fIc2HV4O/nFMVCBwyIUKL8XCtKE+8xgCgl/PQGuFsvShjr74PBp"
        crossorigin="anonymous"></script>
    <link href="https://fonts.googleapis.com/css?family=Oxygen|Poppins"
rel="stylesheet">
    <link href="practice.css" rel="stylesheet" type="text/css">
```

Code Snippet 17 Front-end imports.

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
                    <a class="navbar-brand" href="#"></a>
                    <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarNavAltMarkup"
                        aria-controls="navbarNavAltMarkup" aria-expanded="false"
aria-label="Toggle navigation">
                        <span class="navbar-toggler-icon"></span>
                    </button>
                    <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
                        <div class="navbar-nav">
                            <a class="nav-item nav-link active"
href="practice.html">Home <span class="sr-only">(current)</span></a>
                            <a class="nav-item nav-link"
href="practicefeatures.html">Features</a>
                            <a class="nav-item nav-link"
href="practiceinformation.html">Information</a>
                            <a class="nav-item nav-link"
href="practicecontacts.html">Contact</a>
                        </div>
                    </div>
                </nav>
```

Code Snippet 18 Navigation Bar using Bootstrap.

```
<script>
         function showRegister(){
            $("#register-form").removeClass("collapse").addClass("expand");

        }

    </script>

<script>
        function showLogin(){
           $("#login-form").removeClass("collapse").addClass("expand");
           $("#register-form").removeClass("expand").addClass("collapse");
       }

   </script>
```

Code Snippet 19 JavaScript function showing the Registration form.

```
<div class="col-md-6 register-form collapse" id="register-form" >
                    <h2>Register</h2>
                    <form action="/register" method="POST">
                        <input type="varchar" class="form-control form-group"
name="f_name" placeholder="First Name" required minlength="2" maxlength="20"
value="" />

                        <input type="varchar" class="form-control form-group"
name="l_name" placeholder="Last Name" required minlength="2" maxlength="20"
value="" />

                        <input type="email" class="form-control form-group"
name="email" placeholder="Email" required value=""/>

                        <input type="password" class="form-control form-group"
name="password" id="pass1" onkeyup="checkPasswordMatch();" placeholder="Password"
required minlength="6" maxlength="12"value=""/>

                        <input type="password" class="form-control form-group"
name="c-password" id="pass2" onkeyup="checkPasswordMatch();" placeholder="Confirm
Password" required minlength="6" maxlength="12"value="" />

                        <div class="collapse form-group"
id="divCheckPasswordMatch"><p class="alert alert-danger">Passwords do not
match</p></div>

                        <input type="text" class="form-control form-group"
name="company" placeholder="Company" required minlength="2" maxlength="20"
value=""/>

                        <button class="form-group btn btn-sm my-button2 "
id="register-button" >Register</button>

                        <p class="register form-group">Already Registered? <a
href="#" onclick="showLogin()"> Login </a></p>

                    </form>
                </div>
```

Code Snippet 20 Registration form.

```
<script>
          function checkPasswordMatch() {
      var password = $("#pass1").val();
      var confirmPassword = $("#pass2").val();

      if (password != confirmPassword || confirmPassword != password){
          $("#divCheckPasswordMatch").removeClass("collapse").addClass("expand");
          $("#register-button").prop('disabled',true);
      }
      else{
          $("#divCheckPasswordMatch").removeClass("expand").addClass("collapse");
          $("#register-button").prop('disabled',false);
      }

  }

  </script>
```

Code Snippet 21 Check Passwords JavaScript function.

```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.7.3/Chart.bundle.min.js"></s
cript>
```

Code Snippet 22 Importing Chart.js

```
const HttpData = new XMLHttpRequest();
                    var data1 = "";
                    var begStr = "["
                    var endStr = "]"
                    const urldata = 'http://localhost:80/data';
                    HttpData.open("GET", urldata);
                    HttpData.send();
                    HttpData.onreadystatechange = (e) => {
                        data1 = HttpData.responseText;
                        data1 = data1.replace(/\\/g, "");
                        //dara1=data1.replace(/\\/g,",");
                        //data1 = JSON.parse(data1)
                        data1 = data1.substring(0, data1.length - 1);
                        data1 = begStr + data1 + endStr;
                        console.log(data1);
```

Code Snippet 23 HTTP request to get the data from Flask.

```
var ctx = document.getElementById('approved_a_year_chart');
                    var myChart = new Chart(ctx, {
                        type: 'line',
                        data: {
                            labels: [1999, 2000, 2001, 2002, 2003, 2004, 2005,
2006, 2007, 2008, 2009, 2010,2011,2012,2013, 2014, 2015, 2016, 2017, 2018],
                            datasets: [{
                                label: '# of Approvals', // name the series
                                data: [3, 0, 1, 3, 3, 3, 4, 7, 4, 10, 11, 15, 17,
23, 30, 36, 52, 46,59,64],
                                lineTension: 0.1,
                                pointStyle: 'cross',
                                borderWidth: 1,
                                borderColor: '#423538',
                            }]
                        },
                        options: {
                            layout: {
                                padding: {
                                    left: 5,
                                    right: 5,
                                    top: 10,
                                    bottom: 10
                                }
                            },
                            legend: {
                                display: false,
                            },

                            title: {
                                display: true,
                                text: 'Number of drugs approved by the FDA per
year',
                                fontSize: 16,
                                fontFamily: "'Oxygen', sans-serif",
                                fontColor: '#423538',

                            },
                            scales: {
                                yAxes: [{
                                    ticks: {
                                        beginAtZero: true,
                                    }
                                }]
                            }
                        }
                    });
                </script>
            </div>
```

Code Snippet 24 Implementation of a Chart.js chart.

UI

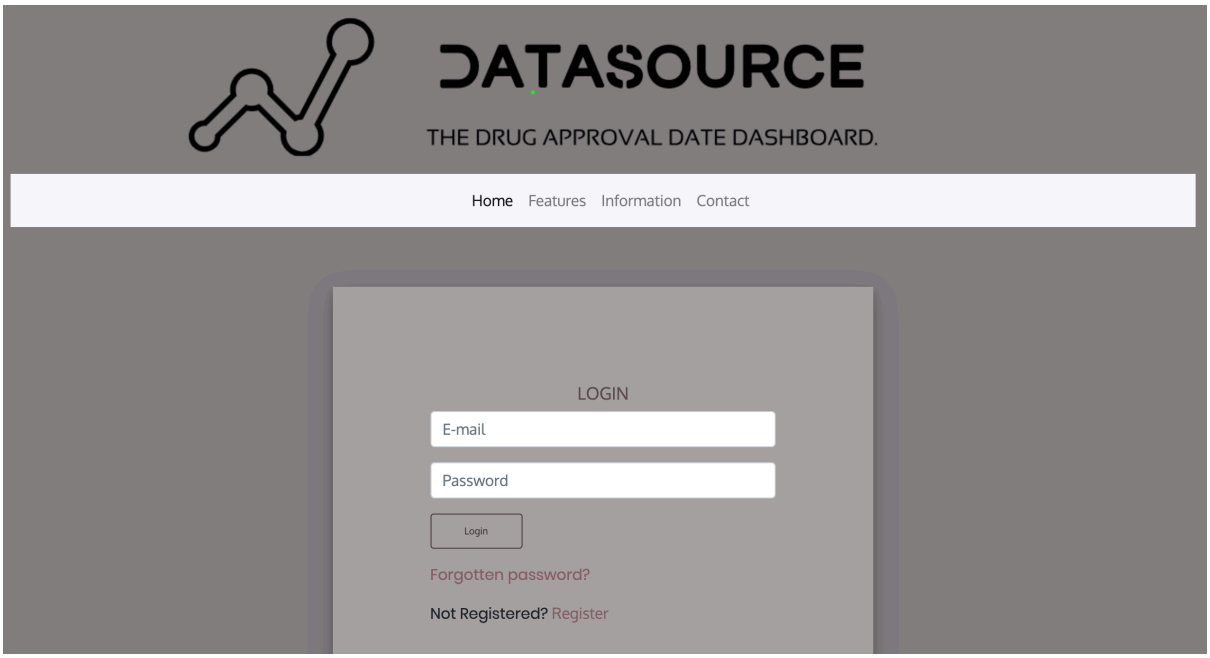


Figure 17 Home Page.

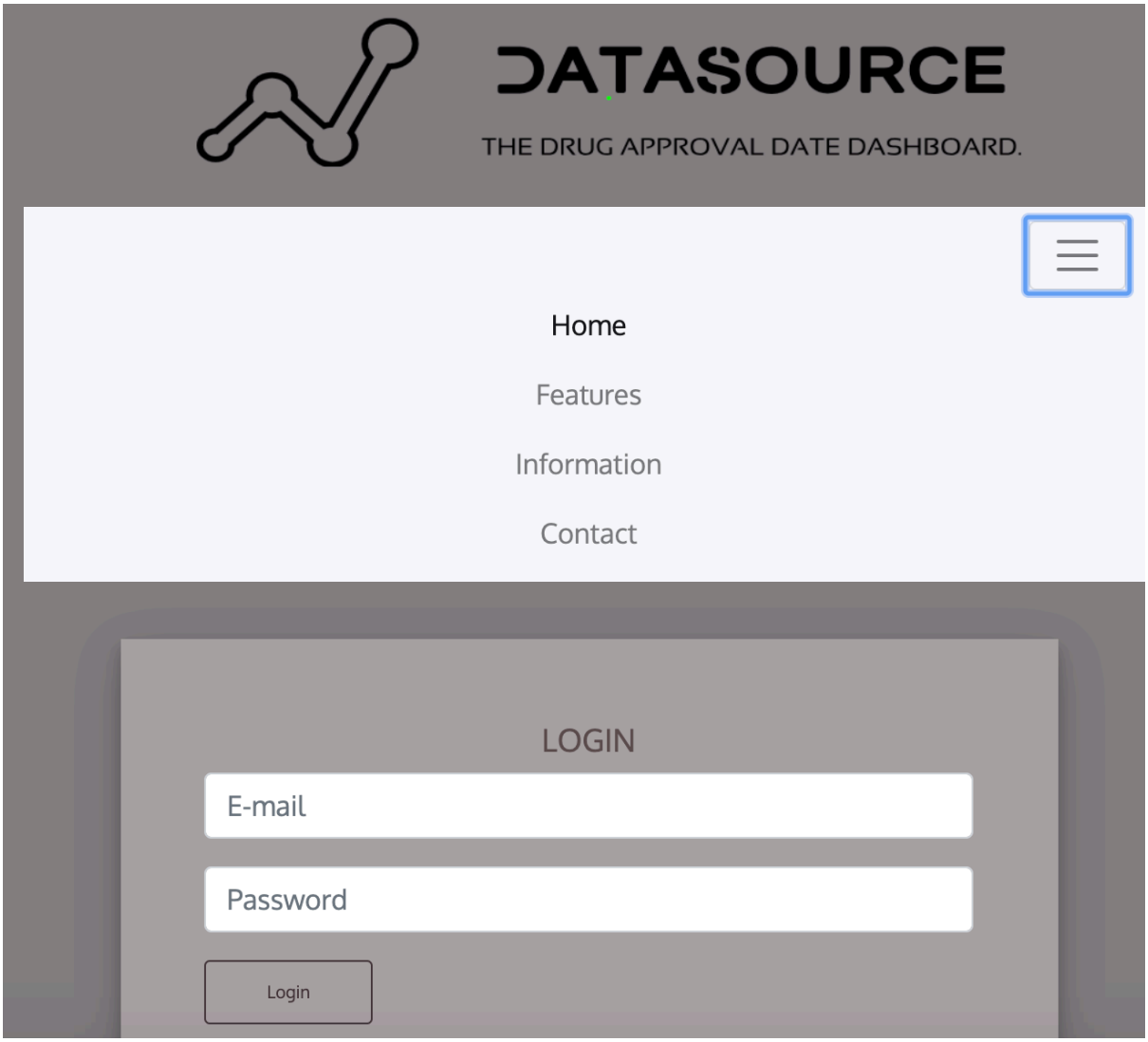


Figure 18 Home Page dynamic resizing.

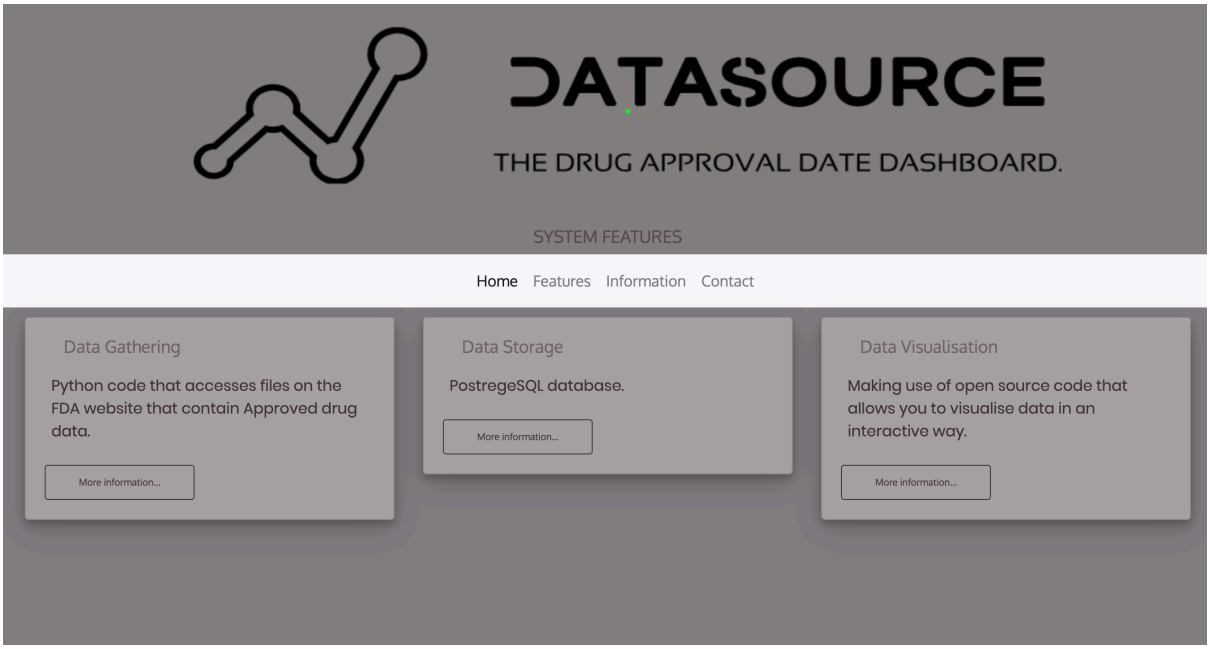


Figure 19 Feature Page with collapsed cards.

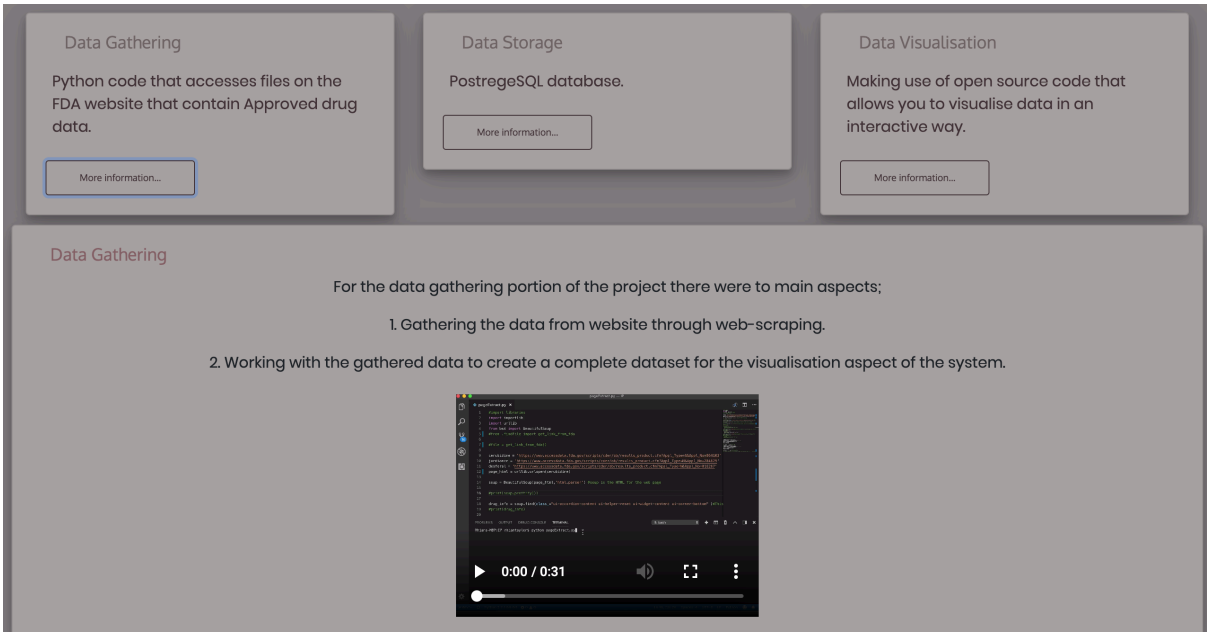


Figure 20 Feature Page with Data Gathering Card expanded.

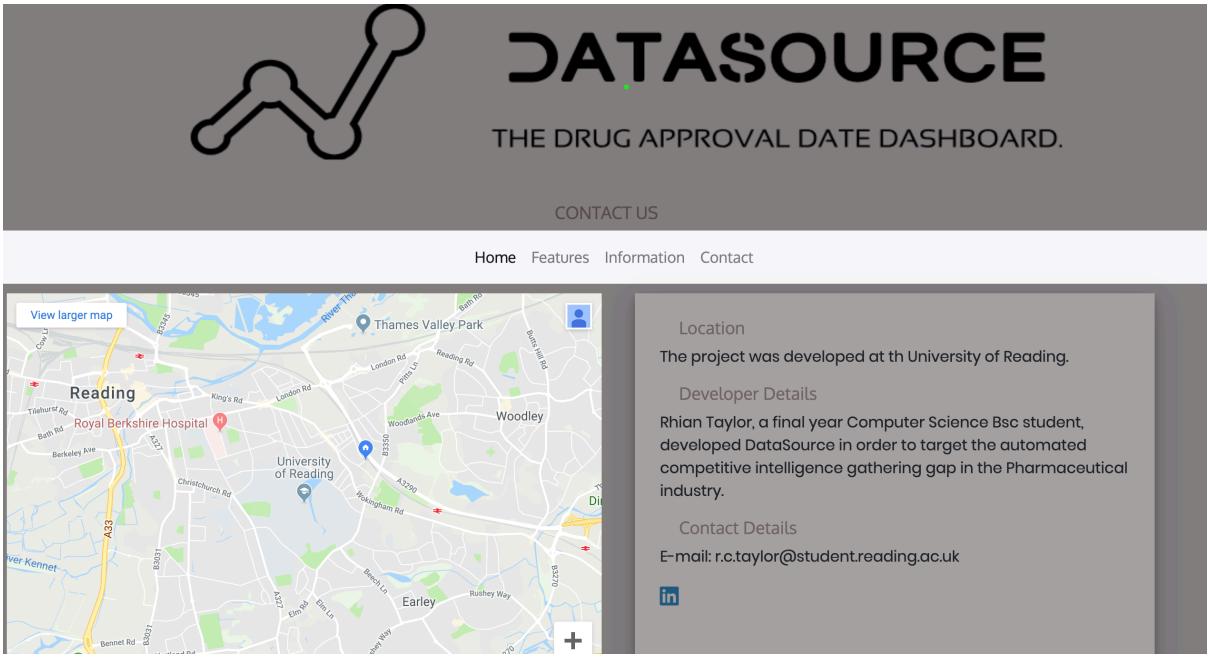


Figure 21 Contact Page.

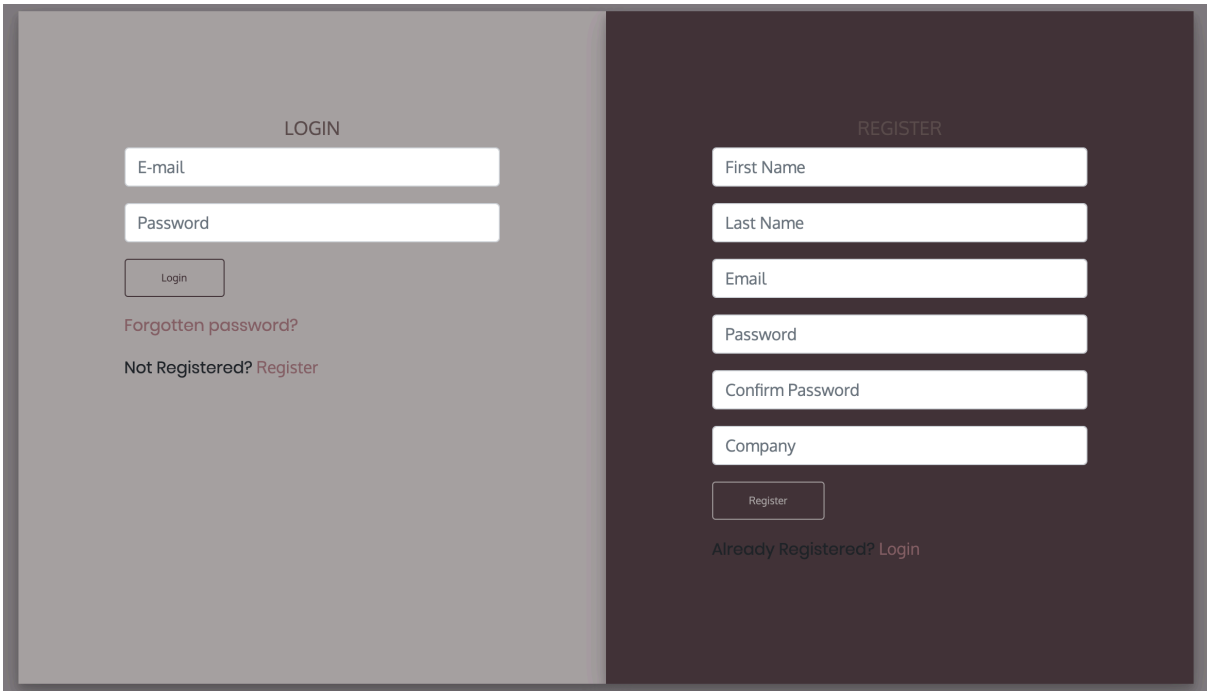


Figure 22 Home page with Registration form.

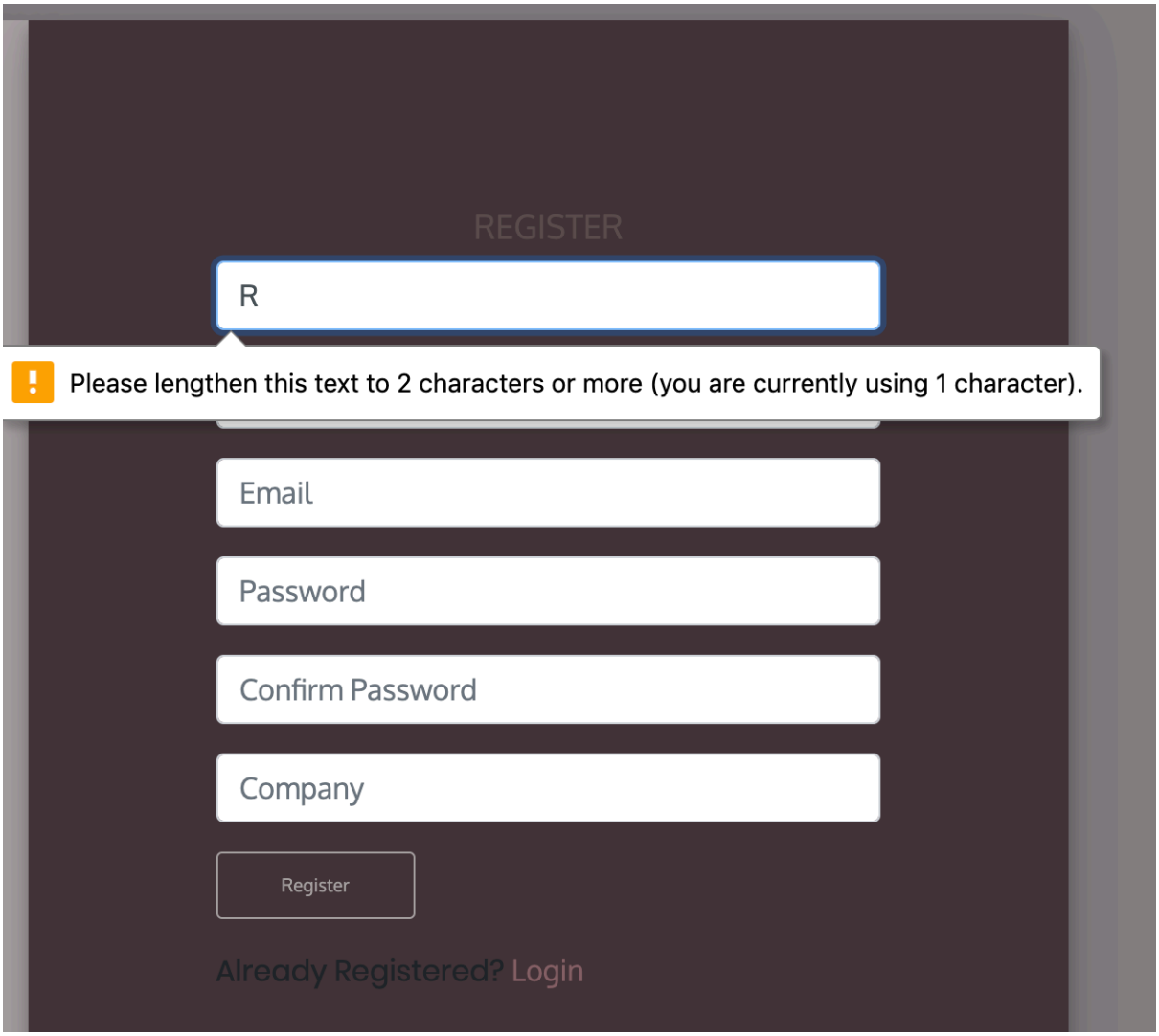


Figure 23 Result of not meeting the min length requirement.

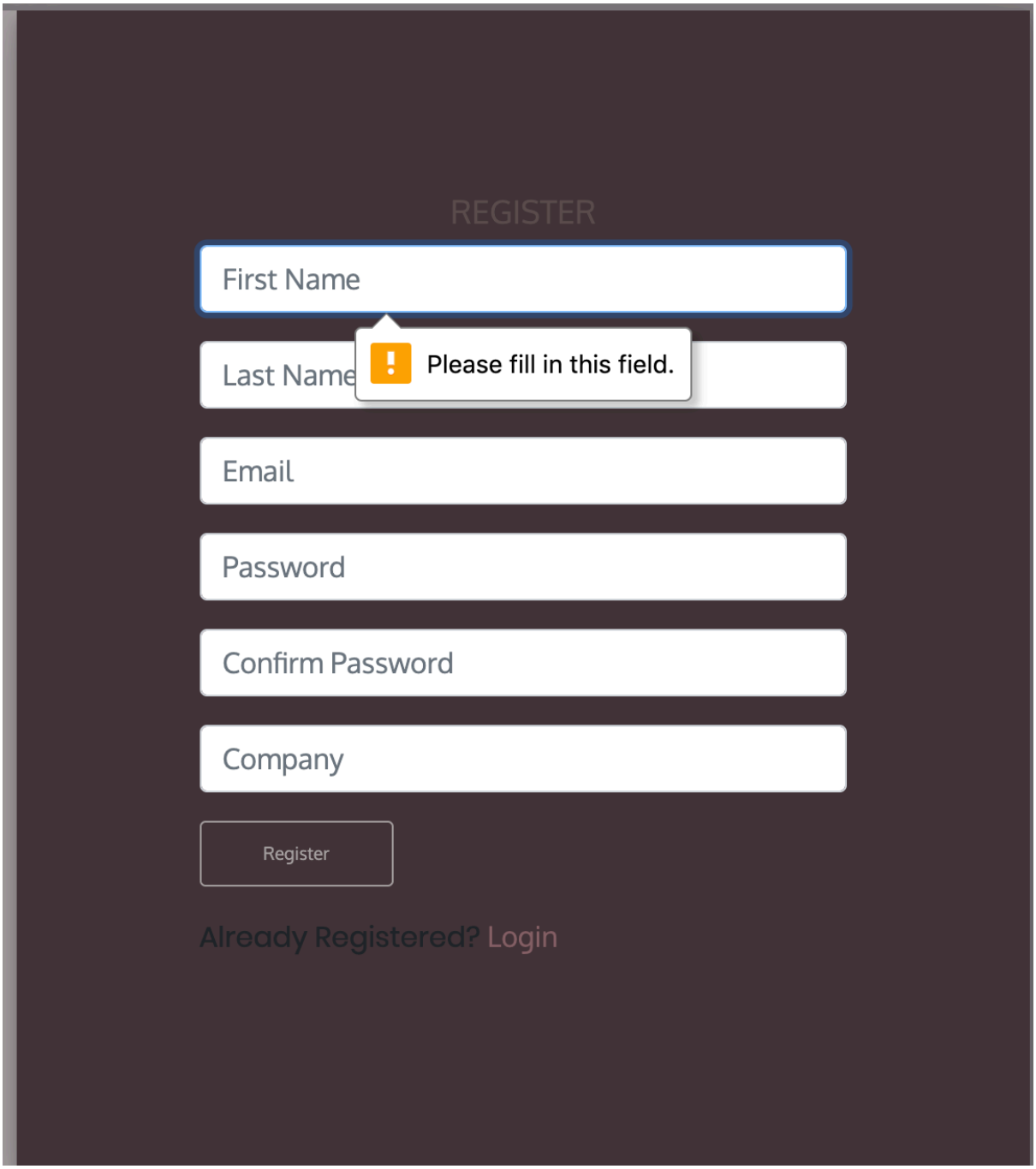


Figure 24 Result of leaving a field blank.

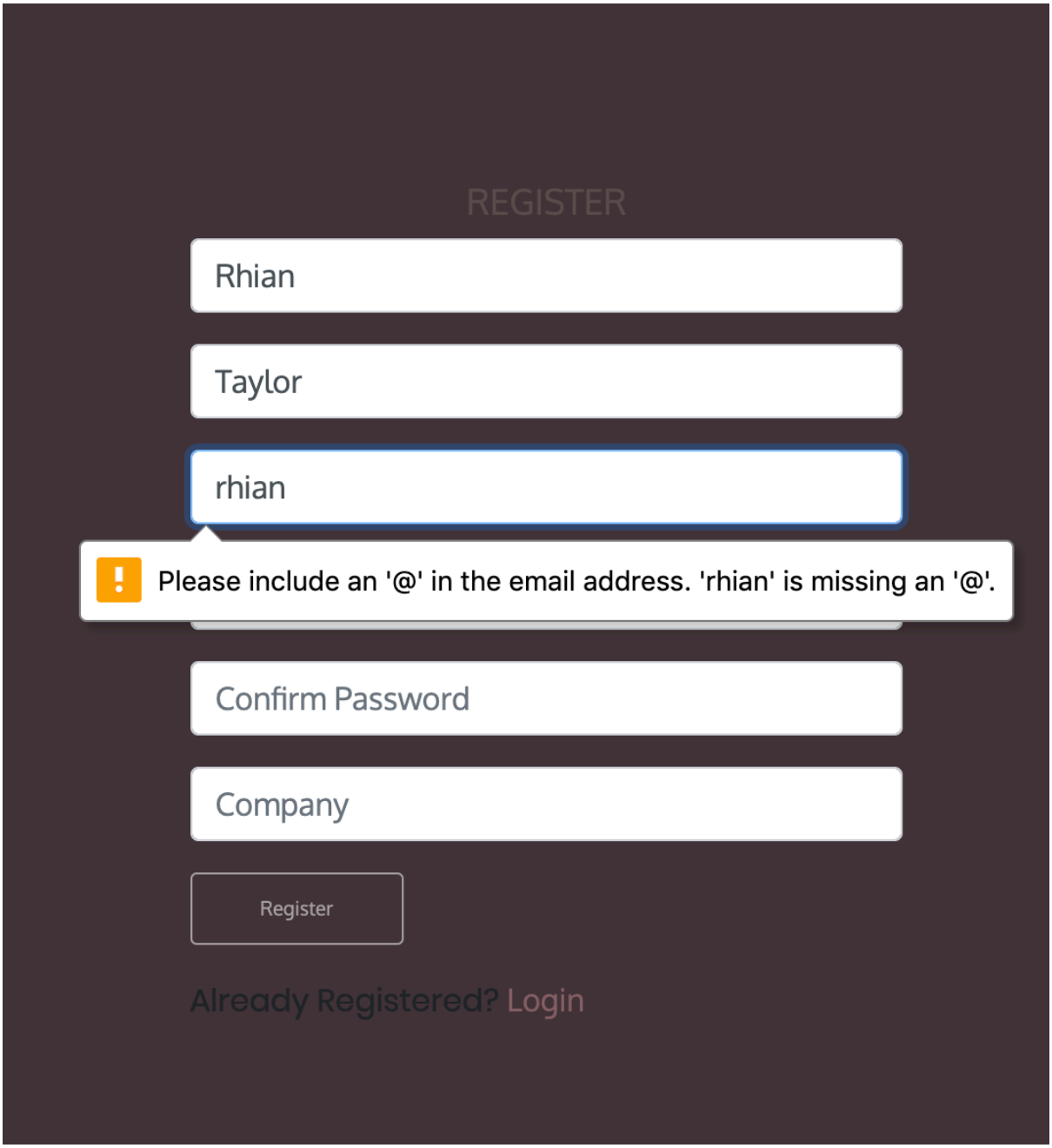


Figure 25 Result of not inputting a correctly formatted email address.

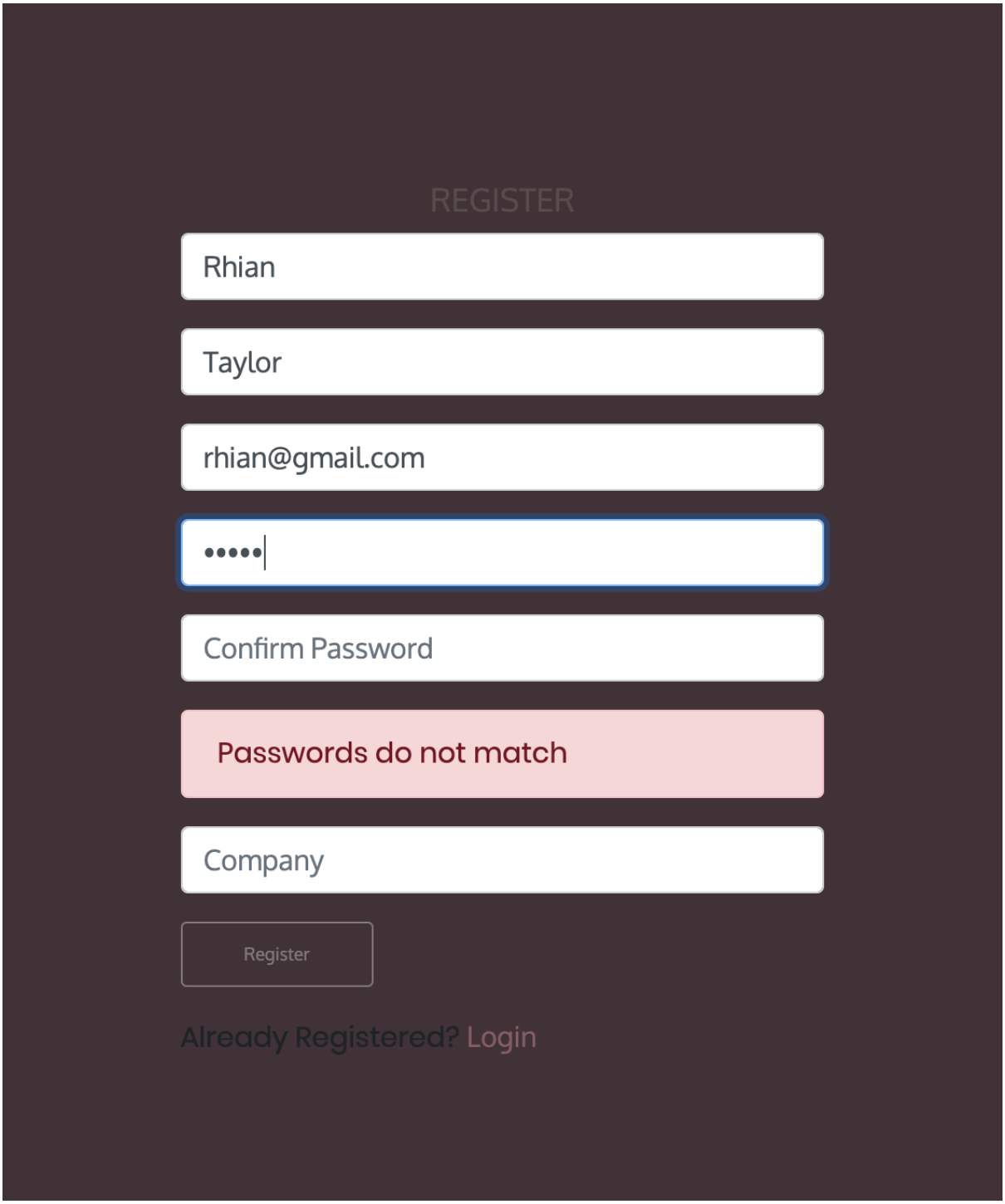


Figure 26 Result of entering passwords that do not match

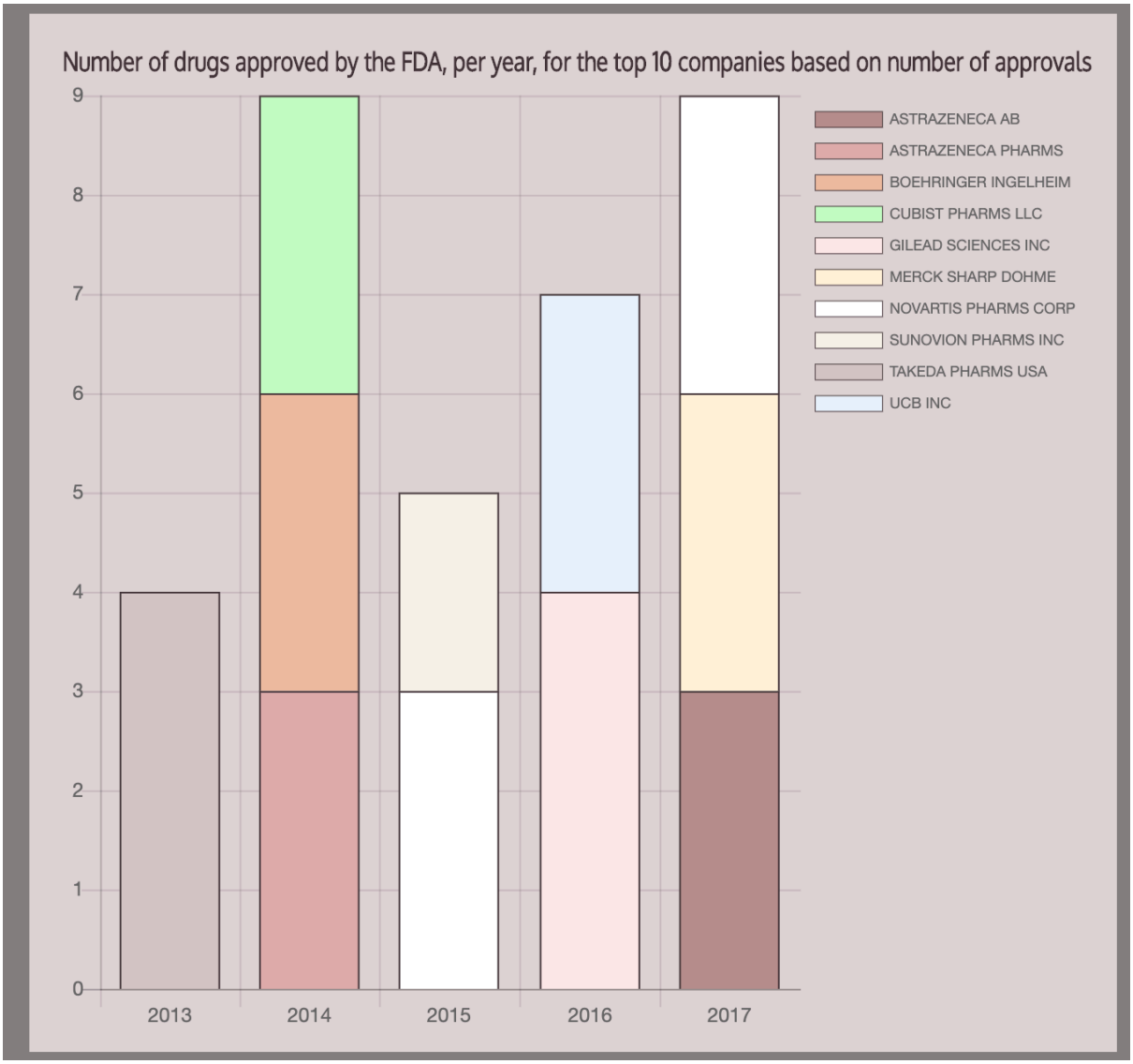


Figure 27 Interactive chart produced with system data and chart.js

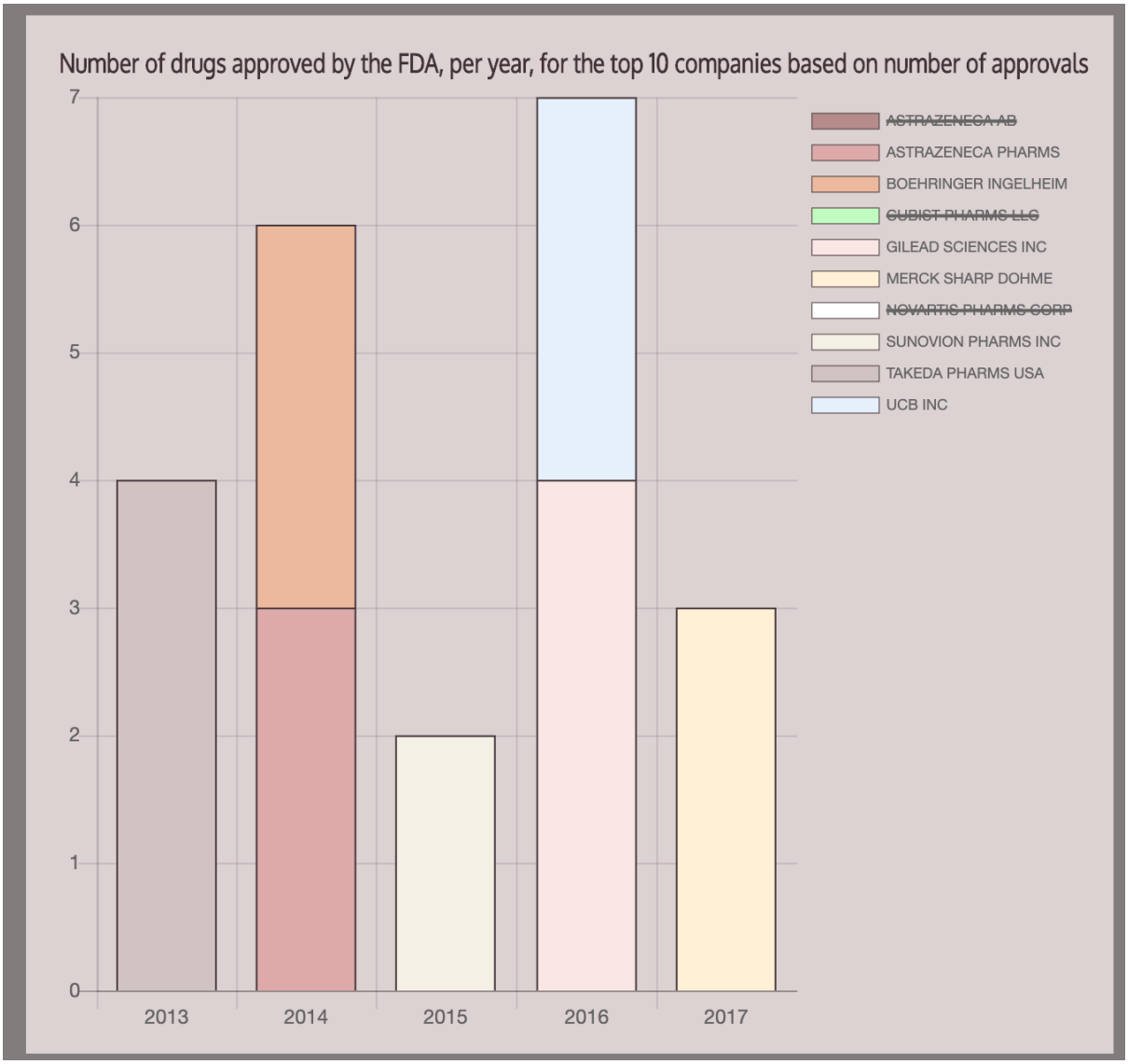


Figure 28 Chart on site, having been filtered by user.

Testing

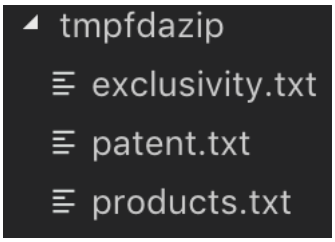


Figure 29 BE1 Result.

Table 16 BE2 Result example.

Product Number	Application Number	Application Type	URL
001	020571	N	https://www.accessdata.fda.gov/scripts/cder/ob/patent_info.cfm?Product_No=001&Appl_No=020571&Appl_type=N

Table 17 BE3 Before - Fail.

URL	Use Code	Use
https://www.accessdata.fda.gov/scripts/cder/ob/patent_info.cfm?Product_No=001&Appl_No=020571&Appl_type=N	U-449	USE IN COMBINATION WITH 5-FLUOROURACIL AND LEUCOVORIN FOR THE TREATMENT OF METASTATIC COLORECTAL CANCER WHERE THE DOSE OF LEUCOVORIN IS AT LEAST 200MG PER SQUARE METER

Table 18 BE3 After - Pass.

URL	Use Code	Use
https://www.accessdata.fda.gov/scripts/cder/ob/patent_info.cfm?Product_No=001&Appl_No=020571&Appl_type=N	U-449	USE IN COMBINATION WITH 5-FLUOROURACIL AND LEUCOVORIN FOR THE TREATMENT OF METASTATIC COLORECTAL CANCER WHERE THE DOSE OF LEUCOVORIN IS AT LEAST 200MG PER SQUARE METER
	U-606	USE OF IRINOTECAN IN COMBINATION WITH 5-FLUOROURACIL AND LEUCOVORIN FOR THE TREATMENT OF METASTATIC COLORECTAL CANCER

Table 19 BE4 Results.

Therapy_area	**URL**
Cardiology/Vascular Diseases (122)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/1/cardiology-vascular-diseases
Dental and Oral Health (7)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/2/dental-and-oral-health
Dermatology (109)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/3/dermatology
Devices (1)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/22/devices
Endocrinology (229)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/4/endocrinology
Family Medicine (624)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/23/family-medicine
Gastroenterology (105)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/5/gastroenterology
Genetic Disease (61)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/34/genetic-disease
Healthy Volunteers (2)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/21/healthy-volunteers

Hematology (135)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/6/hematology
Hepatology (Liver, Pancreatic, Gall Bladder) (54)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/24/hepatology-liver-pancreatic-gall-bladder
Immunology (216)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/7/immunology
Infections and Infectious Diseases (211)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/25/infections-and-infectious-diseases
Internal Medicine (3)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/26/internal-medicine
Musculoskeletal (109)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/8/musculoskeletal
Nephrology (88)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/9/nephrology
Neurology (189)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/10/neurology
Nutrition and Weight Loss (12)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/27/nutrition-and-weight-loss
Obstetrics/Gynecology (Women,Äôs Health) (120)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/11/obstetrics-gynecology-womens-health
Oncology (263)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/12/oncology
Ophthalmology (56)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/13/ophthalmology
Orthopedics/Orthopedic Surgery (6)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/28/orthopedics-orthopedic-surgery
Otolaryngology (Ear, Nose, Throat) (23)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/14/otolaryngology-ear-nose-throat
Pediatrics/Neonatology (124)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/15/pediatrics-neonatology
Pharmacology/Toxicology (32)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/16/pharmacology-toxicology
Podiatry (8)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/30/podiatry
Psychiatry/Psychology (73)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/17/psychiatry-psychology
Pulmonary/Respiratory Diseases (135)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/18/pulmonary-respiratory-diseases

Rare Diseases and Disorders (2)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/35/rare-diseases-and-disorders
Rheumatology (52)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/19/rheumatology
Sleep (3)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/31/sleep
Trauma (Emergency, Injury, Surgery) (8)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/20/trauma-emergency-injury-surgery
Urology (54)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/32/urology
Vaccines (31)	https://www.centerwatch.com/drug-information/fda-approved-drugs/therapeutic-area/33/vaccines

Table 20 BE5 Result Example.

Therapy Group	**Treatment**
cardiology-vascular-diseases	To reduce the risk of major cardiovascular (CV) events in people with chronic coronary or peripheral artery disease
cardiology-vascular-diseases	For the the prophylaxis of venous thromboembolism
cardiology-vascular-diseases	For the treatment of hypertension
cardiology-vascular-diseases	For the prevention of cardiovascular and cerebrovascular events
cardiology-vascular-diseases	For the treatment of chronic heart failure
cardiology-vascular-diseases	For the treatment of Lysosomal Acid Lipase (LAL) deficiency

Table 21 BE7 Example of classifier results.

Use Code	**Use**	**Therapeutica Area**
U-1761	plaque psoriasis	DERMATOLOGY
U-1796	topical treatment inflammatory papules pustules mild moderate rosacea	DERMATOLOGY
U-1841	use long-term, maintenance treatment airflow obstruction patients chronic obstructive pulmonary disease (copd)	PULMONARY/RESPIRATORY DISEASES

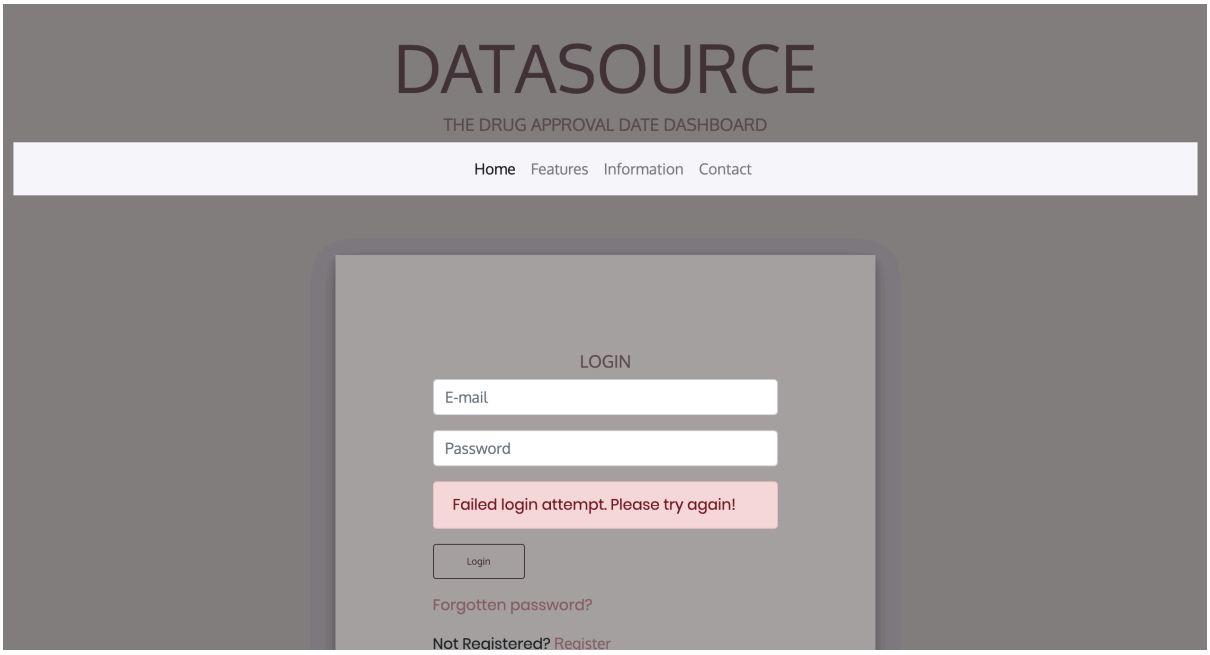


Figure 30 FE2 Result of failed login attempt.

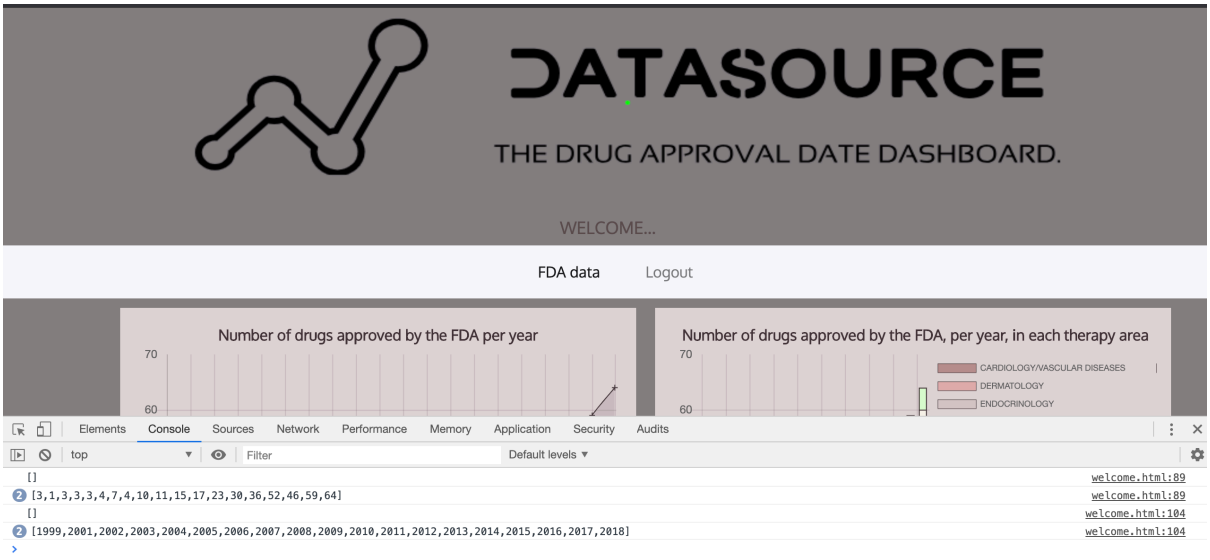


Figure 31 FE3 Console Log.

Rhian attended an interview panel with myself and some of my team in November of last year as part of our graduate recruitment programme. Whilst it was not possible to offer her a position this was due in no way to her showing during that process, and entirely due to internal headcount challenges.

As part of the interview, Rhian provided a presentation and overview of her final year project, a drug approval date dashboard which gathers data automatically from a variety of data sources and use it as a basis for Regulatory affairs within the pharmaceutical industry.

As someone who has been involved in the hiring of graduates into the IT industry over a number of years, it is rare to find an individual with a project that resonates so intrinsically with a production ready solution offering from an organisation. Her presentation showed a grasp of some of the data challenges facing organisations today and how companies like Splunk attempt to address those.

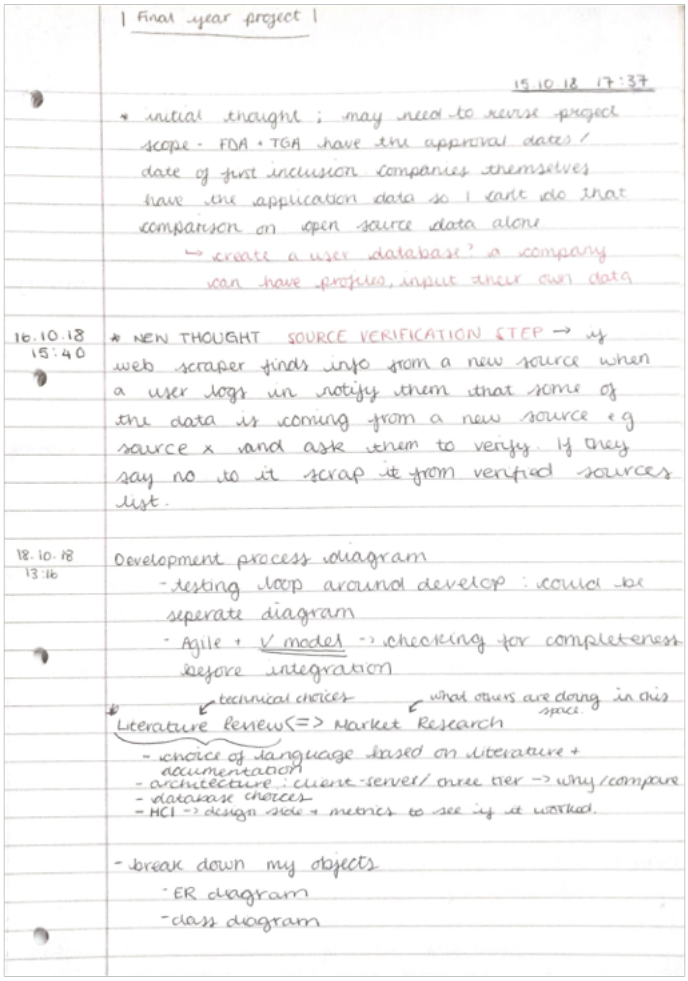
It was a pleasure to have met Rhian as part of the interview process and I wish her all the best for her future career and am of little doubt that she will be an asset to the IT industry as a whole.

Regards

Matthew Bevan
Professional Services Manager – EMEA
Splunk Inc.

Figure 32 Letter from Matthew Bevan, Splunk, commenting on the project.

Logbook



| Final year project |

15.10.18 17:37

* initial thought ; may need to revise project scope - FDA + TGA have the approval dates / date of first inclusion companies themselves have the application data so I can't do that comparison on open source data alone
  ↪ create a user database? a company can have profiles, input their own data

16.10.18 15:40

* NEW THOUGHT SOURCE VERIFICATION STEP → if web scraper finds info from a new source when a user logs in notify them that some of the data is coming from a new source e.g source x and ask them to verify. If they say no to it scrap it from verified sources list.

18.10.18 13:16

Development process diagram
- testing loop around develop : could be seperate diagram
- Agile + V model -> checking for completeness before integration

technical choices ↙ ↘ what others are doing in this space

Literature Review <=> Market Research

- choice of language based on literature + documentation
- architecture : client-server/ three tier -> why/compare
- database choices
- HCI -> design side + metrics to see if it worked.

- break down my objects
  - ER diagram
  - class diagram

DIAGRAMS ;

- ER
- ARCHITECTURE
- Timing diagram
- CLASS
- Login UML
- data flow

* Web Scraper → weekly search (not all at once), based on time zone? do it in the night over weekend so data is ready for Monday morning

29/10/18 11:25

* Principles of Modeling and simulation: A multidisciplinary approach, Nikel D. Petty
  chapter 6: Verification and Validation

** Encyclopedia of software engineering John J Marciniak
  - Verification and Validation Dolores R Wallace, Roger U Fujii

* System Analysis, design, and development: Concepts, Principles, and Practices Charles S. Wasson

24/10/18

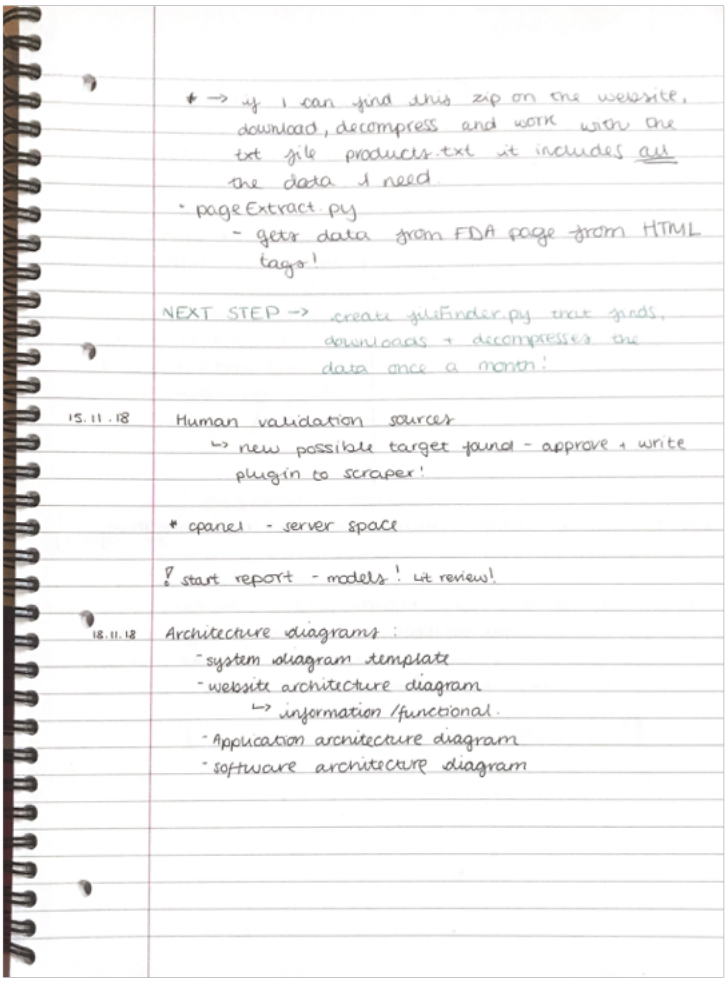
Quality plan
- Scrum structured around V
- Process checking quality
- Goal around quality

29/10/18

Web application homepage
- login / registration form front end

13/11/18

Web scraper development ;
- PDF extractor (scraper.py)
  - cair import extract
  - found pdf list of all drugs on FDA website
  * orange book data files compressed !



* → if I can find this zip on the website, download, decompress and work with the txt file products.txt it includes all the data I need.
- pageExtract.py
  - gets data from FDA page from HTML tags!

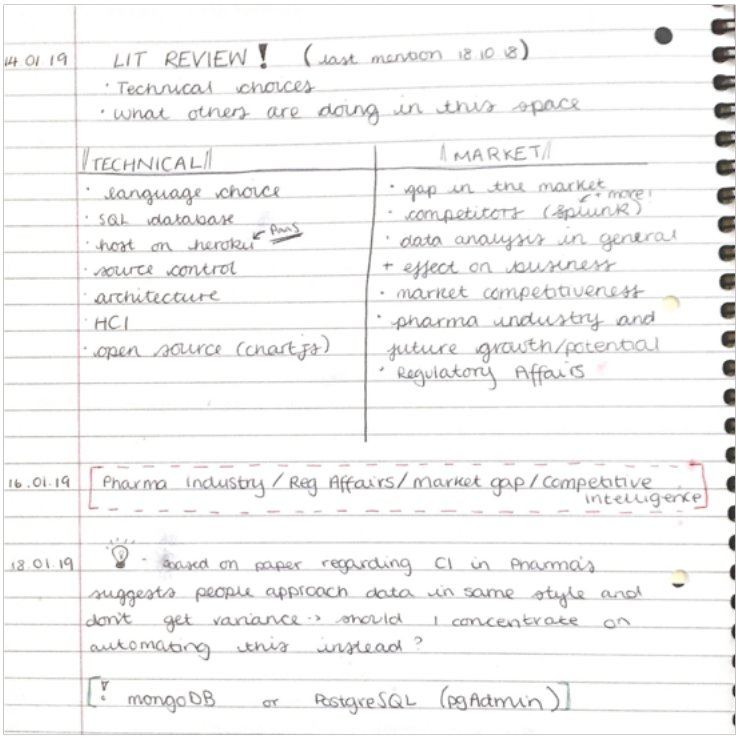
NEXT STEP → create gitFinder.py that finds, downloads + decompresses the data once a month!

15.11.18 Human validation sources
↳ new possible target found - approve + write plugin to scraper!

* cpanel - server space

! start report - models! Lit review!

18.11.18 Architecture diagrams:
- system diagram template
- website architecture diagram
  ↳ information / functional.
- Application architecture diagram
- software architecture diagram



14.01.19 LIT REVIEW ! (last meeting 18.10.18)

- Technical choices
- what others are doing in this space

TECHNICAL	MARKET
· language choice	· gap in the market + more!
· SQL database	· competitors (SplunkR)
· host on heroku (Paas)	· data analysis in general + effect on business
· source control	· market competitiveness
· architecture	· pharma industry and future growth/potential
· HCI	· Regulatory Affairs
· open source (chartjs)	

16.01.19 [Pharma Industry / Reg Affairs / market gap / Competitive intelligence]

18.01.19 - based on paper regarding CI in Pharma's suggests people approach data in same style and don't get variance -> should I concentrate on automating this instead ?

[! mongoDB or PostgreSQL (pgAdmin)]

FOCUS: Automate data gathering, store + display data!
↳ do I want to analyse it?

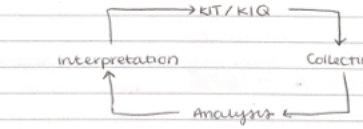
[WHAT HAVE I DONE SO FAR?]
- HTML home page, locally hosted.
- found + downloaded drug file from FDA website
  ↳ contains all drugs, updated monthly
- Read articles on Pharma + competitive intelligence.
- written a quality report.

21.1.19 [Pharma Industry / Competitive Intelligence / market]

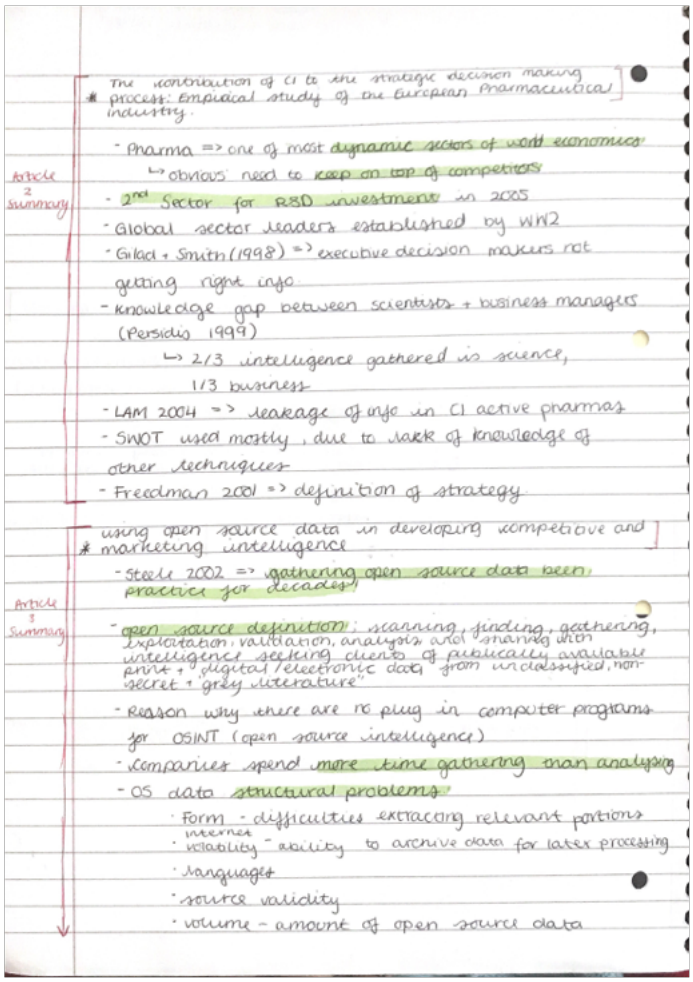
Article 1 summary

* Competitive intelligence in the biopharmaceutical industry: the key elements (Yulia Aspinall)
  - Liebowitz, 2008 definition of CI
  - KIT & KIQ (key intelligence topics & key intelligence questions)
    ↳ 1. Strategic decisions + actions 2. early warnings 3. descriptions of key players in your market place.

! Competitive intelligence cycle:



- Primary & Secondary intelligence
  publically unavailable knowledge → publically available intelligence - desk research, hard/online resources.
- Popular analysis methods: SWOT, gap analysis ...



Article 2 Summary

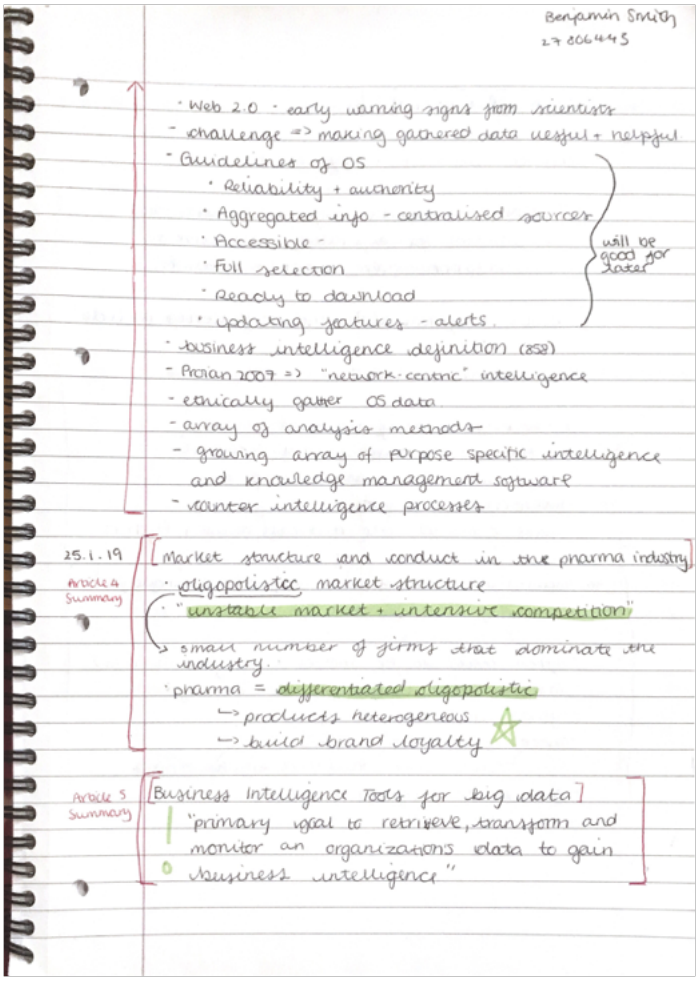
* The contribution of CI to the strategic decision making process: Empirical study of the European Pharmaceutical industry.

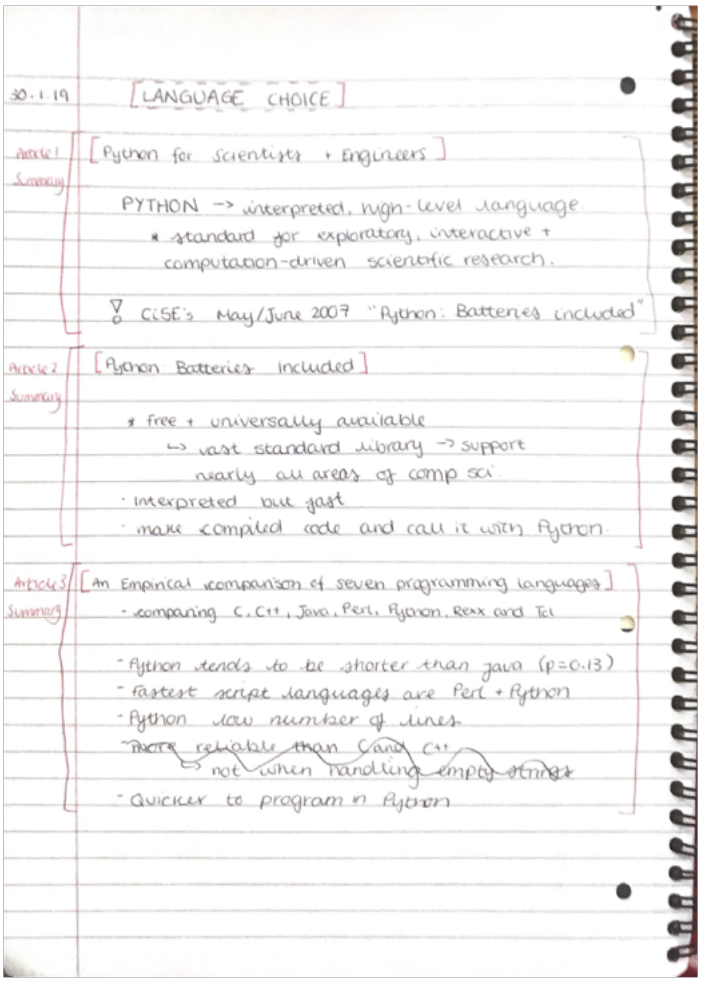
- Pharma => one of most dynamic sectors of world economics
  - ↳ obvious need to keep on top of competitors
- 2nd Sector for R&D investments in 2005
- Global sector leaders established by WW2
- Gilad + Smith (1998) => executive decision makers not getting right info
- Knowledge gap between scientists + business managers (Persidis 1999)
  - ↳ 2/3 intelligence gathered is science, 1/3 business
- LAM 2004 => leakage of info in CI active pharmas
- SWOT used mostly, due to lack of knowledge of other techniques
- Freedman 2001 => definition of strategy

Article 3 Summary

* using open source data in developing competitive and marketing intelligence

- Steele 2002 => gathering open source data been practice for decades
- open source definition: scanning, finding, gathering, exploitation, validation, analysis and sharing with intelligence seeking clients of publically available print + digital / electronic data from unclassified, non-secret + "grey literature"
- Reason why where are no plug in computer programs for OSINT (open source intelligence)
- Companies spend more time gathering than analysing
- OS data structural problems
  - Form - difficulties extracting relevant portions
  - Interface
  - Volatility - ability to archive data for later processing
  - Language
  - Source validity
  - Volume - amount of open source data


Benjamin Smith
27006445
· Web 2.0 - early warning signs from scientists
- challenge => making gathered data useful + helpful
- Guidelines of OS
· Reliability + authority
· Aggregated info - centralised sources
· Accessible
· Full selection
· Ready to download
· Updating features - alerts
will be good for later
- business intelligence definition (BSI)
- Pirian 2007 => "network-centric" intelligence
- ethically gather OS data
- array of analysis methods
- growing array of purpose specific intelligence and knowledge management software
- counter intelligence processes
25.1.19
Article 4 Summary
[Market structure and conduct in the pharma industry]
· oligopolistic market structure
· "unstable market + intensive competition"
↳ small number of firms that dominate the industry.
· pharma = differentiated oligopolistic
↳ products heterogeneous
↳ build brand loyalty
Article 5 Summary
[Business Intelligence Tools for big data]
"primary goal to retrieve, transform and monitor an organizations data to gain business intelligence"



30.1.19

[LANGUAGE CHOICE]

Article 1 Summary

[Python for Scientists + Engineers]

PYTHON -> interpreted, high-level language
* standard for exploratory, interactive + computation-driven scientific research.

∇ CiSE's May/June 2007 "Python: Batteries included"

Article 2 Summary

[Python Batteries Included]

* free + universally available
↳ vast standard library -> support nearly all areas of comp sci
- interpreted but fast
- make compiled code and call it with Python.

Article 3 Summary

[An Empirical comparison of seven programming languages]
- comparing C, C++, Java, Perl, Python, Rexx and Tcl

- Python tends to be shorter than java ($p=0.13$)
- fastest script languages are Perl + Python
- Python low number of lines
- ~~more reliable than~~ C and C++ ↳ not when handling ~~empty strings~~
- Quicker to program in Python

5.2.19

DATABASE : postgresql in command line atm

Table users :-

user_id (pk) sequence — made this

username varchar unique not null

password varchar not null

email varchar unique not null

created_on timestamp not null

last_login timestamp

company varchar not null

* PGCRYPTO extension to store user passwords

6.2.19

- Restful API → flask + python

  python + postgresql database

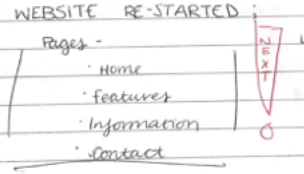
-- trying to allow users to register via website

12.2.19 DDOS Prevention ; not letting people refresh getting data more than once a day !

ARCHITECTURE ; model with modules to handle different sources

USER STORIES ! ; - export
- site itself
- Reg. Affairs / CEO
- email update on specific flagged companies

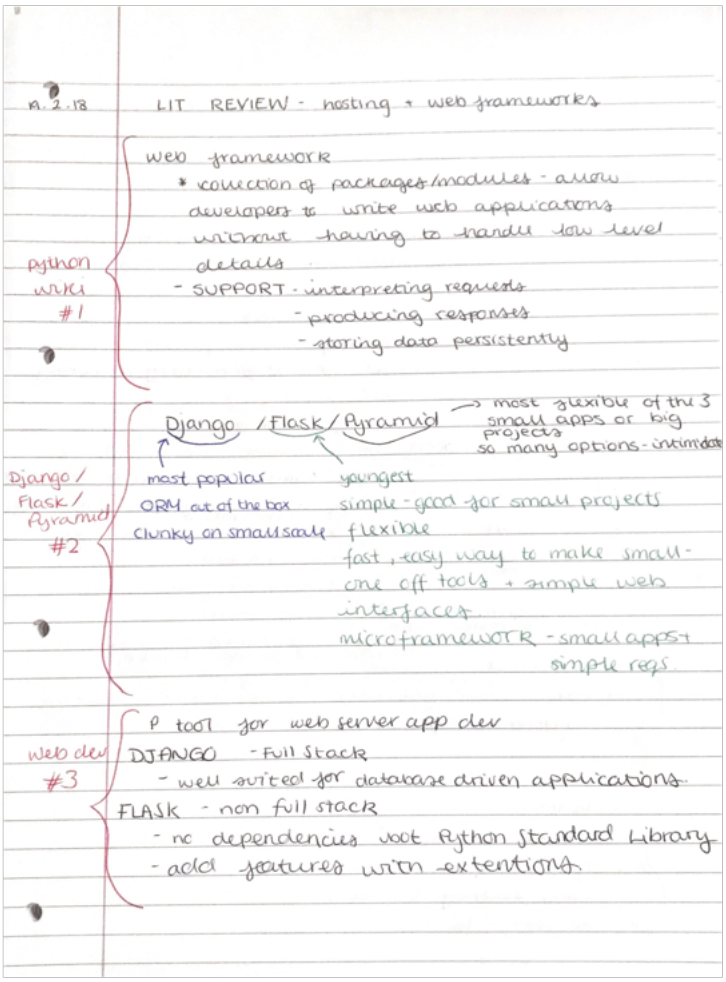
16.2.19 WEBSITE RE-STARTED ;

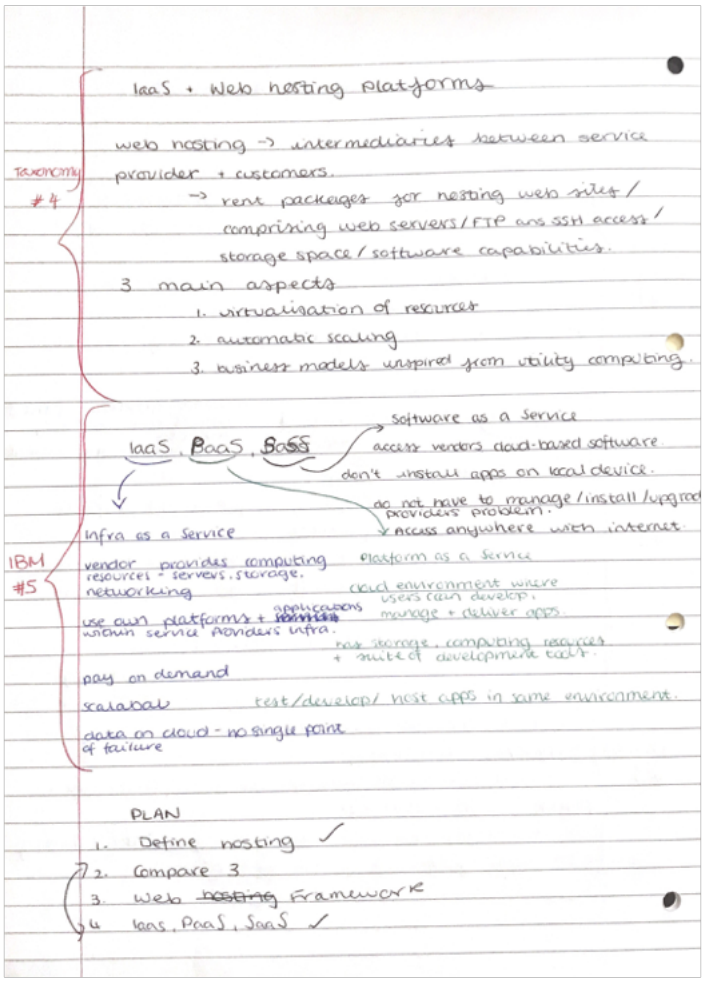


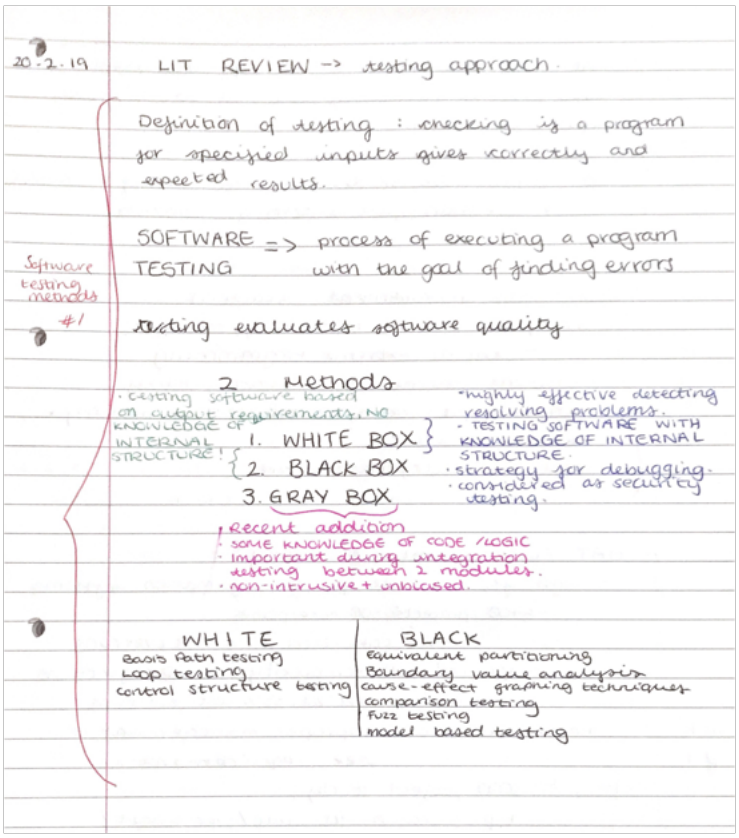
LOGGED IN ? => can create page now !

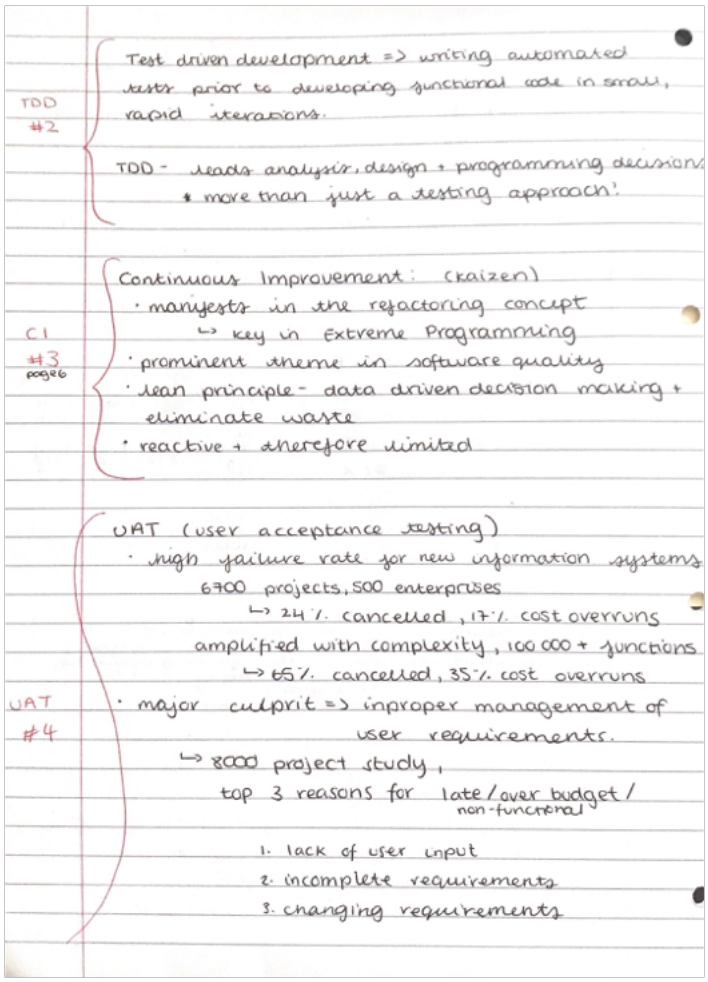
18.2.19 LIT REVIEW
- programming language choice
  → compare Python, Java, C++, C, HTML+CSS and JavaScript !

NEXT
- Next look at Python Web frameworks (Django / flask) + compare with hosting platforms !


n. 2.18
LIT REVIEW - hosting + web frameworks
python wiki #1
web framework
* collection of packages/modules - allow developers to write web applications without having to handle low level details
- SUPPORT - interpreting requests
- producing responses
- storing data persistently
Django / Flask / Pyramid
most flexible of the 3 small apps or big projects so many options - intimidat
Django / Flask / Pyramid #2
most popular
ORM out of the box
clunky on small scale
youngest
simple - good for small projects
flexible
fast, easy way to make small-one off tools + simple web interfaces
microframework - small apps + simple reqs
Web dev #3
a tool for web server app dev
DJANGO - Full Stack
- well suited for database driven applications
FLASK - non full stack
- no dependencies out Python Standard Library
- add features with extentions


IaaS + Web hosting platforms
web hosting → intermediaries between service provider + customers.
Taxonomy #4
→ rent packages for hosting web sites / comprising web servers / FTP and SSH access / storage space / software capabilities.
3 main aspects
1. virtualisation of resources
2. automatic scaling
3. business models inspired from utility computing.
IaaS . PaaS . SaaS
software as a service
access vendors cloud-based software.
don't install apps on local device.
do not have to manage / install / upgrade providers problem.
access anywhere with internet
Infra as a Service
vendor provides computing resources - servers, storage, networking
use own platforms + applications within service Providers infra.
pay on demand
scalabal
data on cloud - no single point of failure
IBM #5
Platform as a Service
cloud environment where users can develop, manage + deliver apps.
has storage, computing resources + suite of development tools.
test/develop/ host apps in same environment.
PLAN
1. Define hosting ✓
2. Compare 3
3. Web hosting Framework
4. IaaS, PaaS, SaaS ✓


20.2.19
LIT REVIEW -> testing approach
Definition of testing : checking is a program for specified inputs gives correctly and expected results.
Software testing methods #1
SOFTWARE TESTING => process of executing a program with the goal of finding errors
testing evaluates software quality
2 Methods
· testing software based on output requirements, NO KNOWLEDGE OF INTERNAL STRUCTURE!
1. WHITE BOX
2. BLACK BOX
3. GRAY BOX
· highly effective detecting resolving problems.
· TESTING SOFTWARE WITH KNOWLEDGE OF INTERNAL STRUCTURE.
· strategy for debugging.
· considered as security testing.
Recent addition
· some KNOWLEDGE OF CODE / LOGIC
· important during integration testing between 2 modules.
· non-intrusive + unbiased.
WHITE
Basis Path testing
Loop testing
control structure testing
BLACK
Equivalent partitioning
Boundary value analysis
cause-effect graphing techniques
comparison testing
Fuzz testing
model based testing



TDD #2

Test driven development => writing automated tests prior to developing functional code in small, rapid iterations.

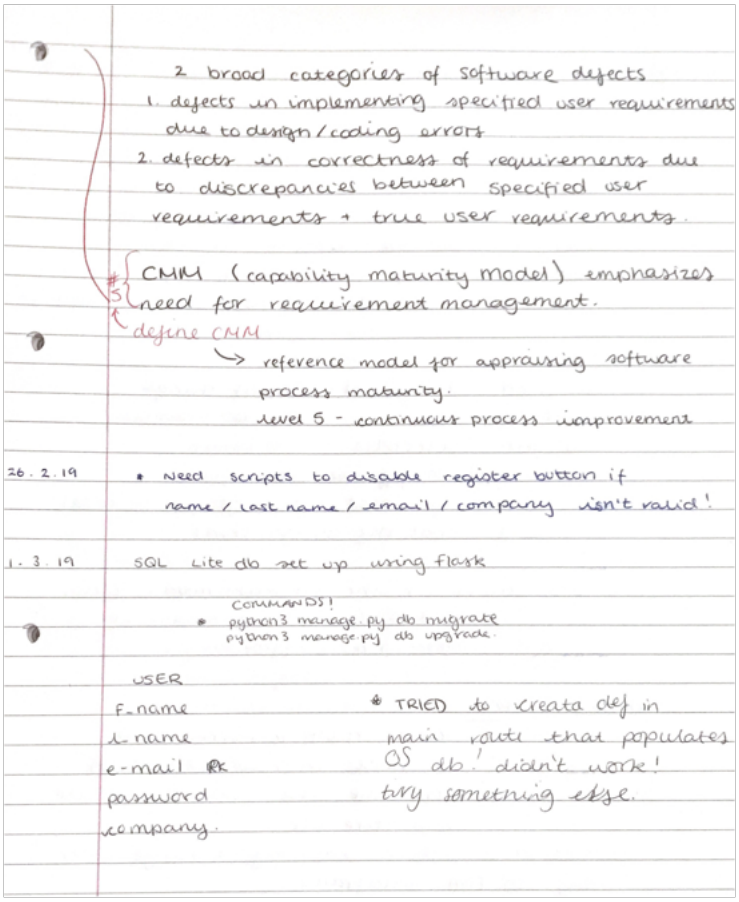
TDD - leads analysis, design + programming decisions
* more than just a testing approach!

CI #3 page6

Continuous Improvement: (kaizen)
- manifests in the refactoring concept
  - ↳ key in Extreme Programming
- prominent theme in software quality
- lean principle - data driven decision making + eliminate waste
- reactive + therefore limited

UAT #4

UAT (user acceptance ~~testing~~)
- high failure rate for new information systems
  - 6700 projects, 500 enterprises
    - ↳ 24% cancelled, 17% cost overruns
  - amplified with complexity, 100 000 + functions
    - ↳ 65% cancelled, 35% cost overruns
- major culprit => inproper management of user requirements.
  - ↳ 8000 project study,
    top 3 reasons for late / over budget / non-functional
    1. lack of user input
    2. incomplete requirements
    3. changing requirements



2 broad categories of software defects
1. defects in implementing specified user requirements due to design / coding errors
2. defects in correctness of requirements due to discrepancies between specified user requirements + true user requirements.

#3 CMM (capability maturity model) emphasizes need for requirement management.

define CMM
→ reference model for appraising software process maturity.
level 5 - continuous process improvement

26.2.19 • Need scripts to disable register button if name / last name / email / company isn't valid!

1.3.19 SQL Lite db set up using flask

COMMANDS!
• python3 manage.py db migrate
python3 manage.py db upgrade

USER
F_name
L_name
e-mail PK
password
company.

* TRIED to create def in main route that populates DB db. didn't work! try something else.

4.3.19 GET THERAPY AREA:

- currently therapy area not included in FDA files
- Patent-Code in data refers to what the drug treats
- on each drugs individual web page if you hover over the use code the thing it treats shows up. * some have multiple
  ↳ URL includes appl-no, product-no, appl-type

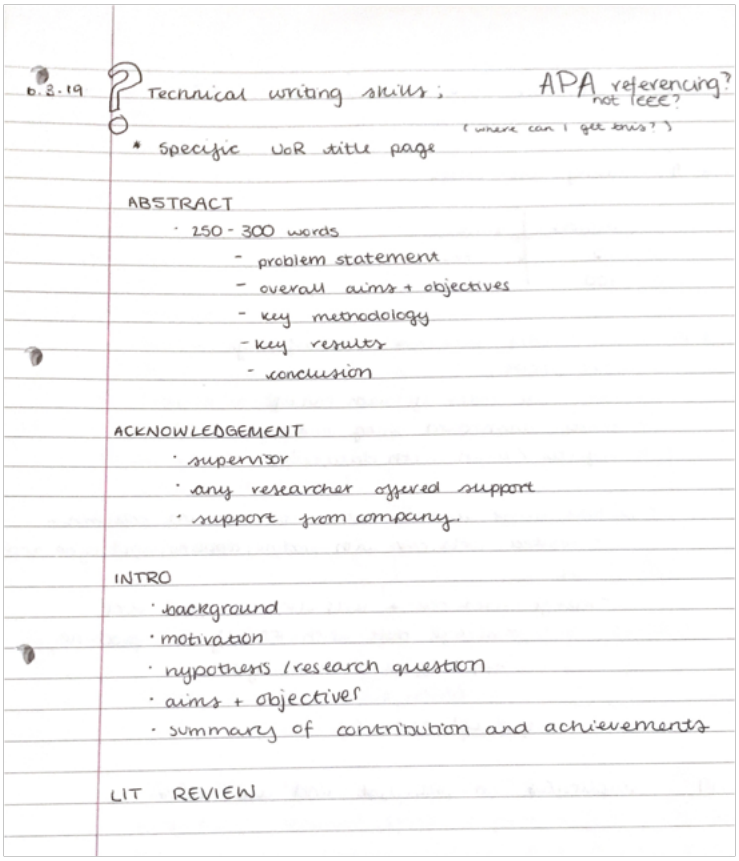
PLAN

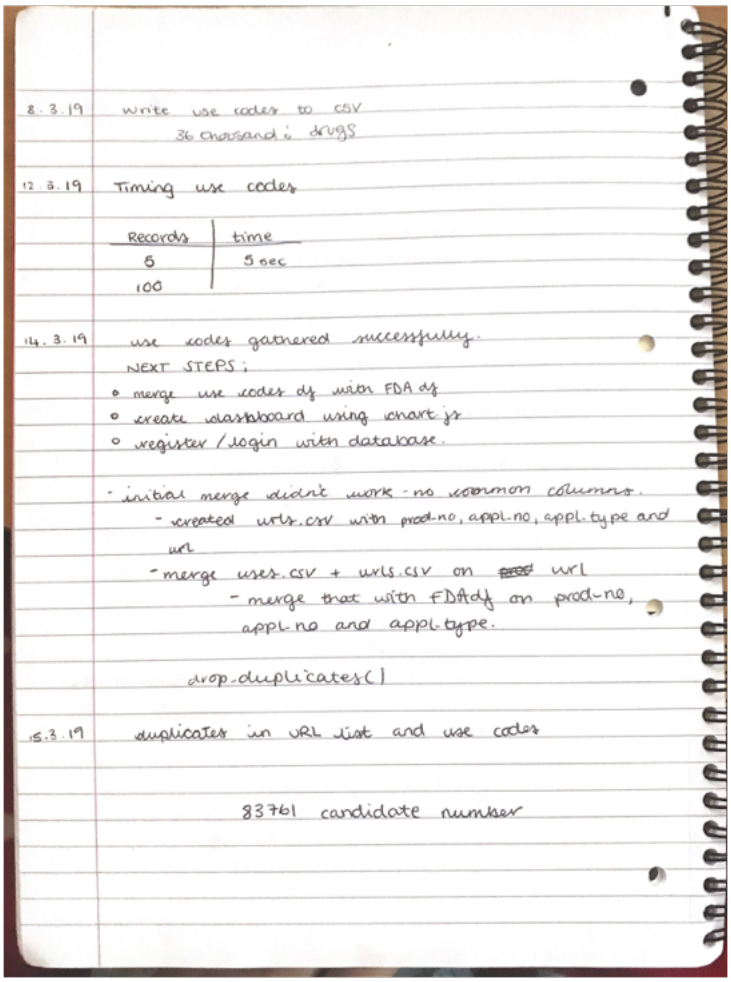
- new read file method → returns things needed for URL
- new file - usecodes
  - method to generate urls
  - method to scrape each page for use codes and what the drugs treat.

* some drugs in the FDA files have use codes, others have NaNs (seems like only have multiple codes on their pages.

NEXT STEPS

- compare scrape results to current use code => if same in new use column store use else update useCode and store use.
- method to return this df → merge with my OS-FDA dataframe.
- store data in dB!


6.2.19
Technical writing skills;
APA referencing?
not IEEE?
(where can I get this?)
* Specific UoR title page
ABSTRACT
· 250 - 300 words
- problem statement
- overall aims + objectives
- key methodology
- key results
- conclusion
ACKNOWLEDGEMENT
· supervisor
· any researcher offered support
· support from company.
INTRO
· background
· motivation
· hypothesis / research question
· aims + objectives
· summary of contribution and achievements
LIT REVIEW



8.3.19 write use codes to csv
36 thousand drugs

12.3.19 Timing use codes

Records	time
5	5 sec
100	

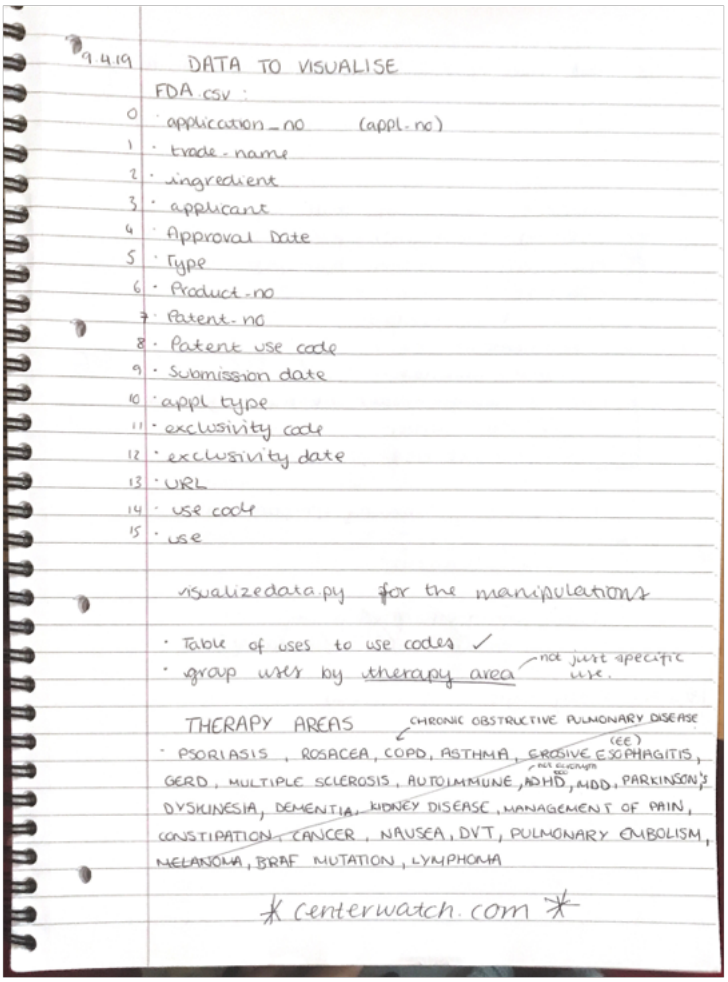
14.3.19 use codes gathered successfully.
NEXT STEPS;
- merge use codes df with FDA df
- create dashboard using chart.js
- register / login with database.

- initial merge didn't work - no common columns.
  - created urls.csv with prod_no, appl_no, appl_type and url
  - merge uses.csv + urls.csv on ~~prod~~ url
    - merge that with FDAdf on prod_no, appl_no and appl_type.

drop_duplicates()

15.3.19 duplicates in URL list and use codes

83761 candidate number



9.4.19

DATA TO VISUALISE

FDA csv :

0 · application_no (appl_no)
1 · trade_name
2 · ingredient
3 · applicant
4 · Approval Date
5 · Type
6 · Product_no
7 · Patent_no
8 · Patent use code
9 · Submission date
10 · appl type
11 · exclusivity code
12 · exclusivity date
13 · URL
14 · use code
15 · use

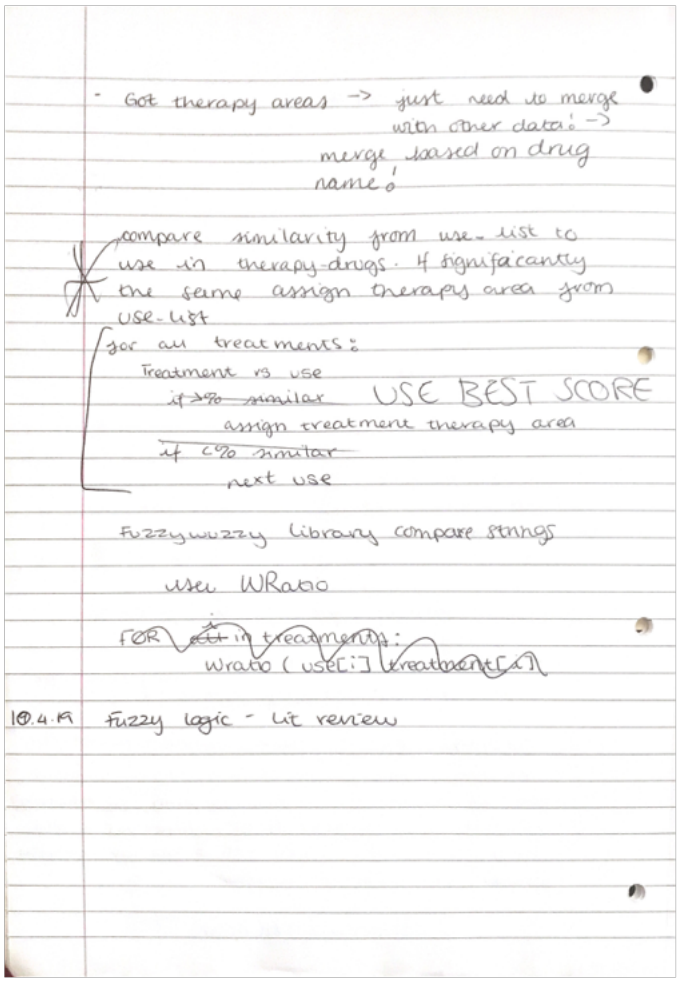
visualizedata.py for the manipulations

· Table of uses to use codes ✓
· grap uses by therapy area — not just specific use.

THERAPY AREAS

· PSORIASIS, ROSACEA, COPD (CHRONIC OBSTRUCTIVE PULMONARY DISEASE), ASTHMA, EROSIVE ESOPHAGITIS (EE), GERD, MULTIPLE SCLEROSIS, AUTOIMMUNE, ADHD, MDD, PARKINSON'S, DYSKINESIA, DEMENTIA, KIDNEY DISEASE, MANAGEMENT OF PAIN, CONSTIPATION, CANCER, NAUSEA, DVT, PULMONARY EMBOLISM, MELANOMA, BRAF MUTATION, LYMPHOMA

* centerwatch.com *



- Got therapy areas -> just need to merge with other data -> merge based on drug name!

compare similarity from use-list to use in therapy-drugs. If significantly the same assign therapys area from use-list

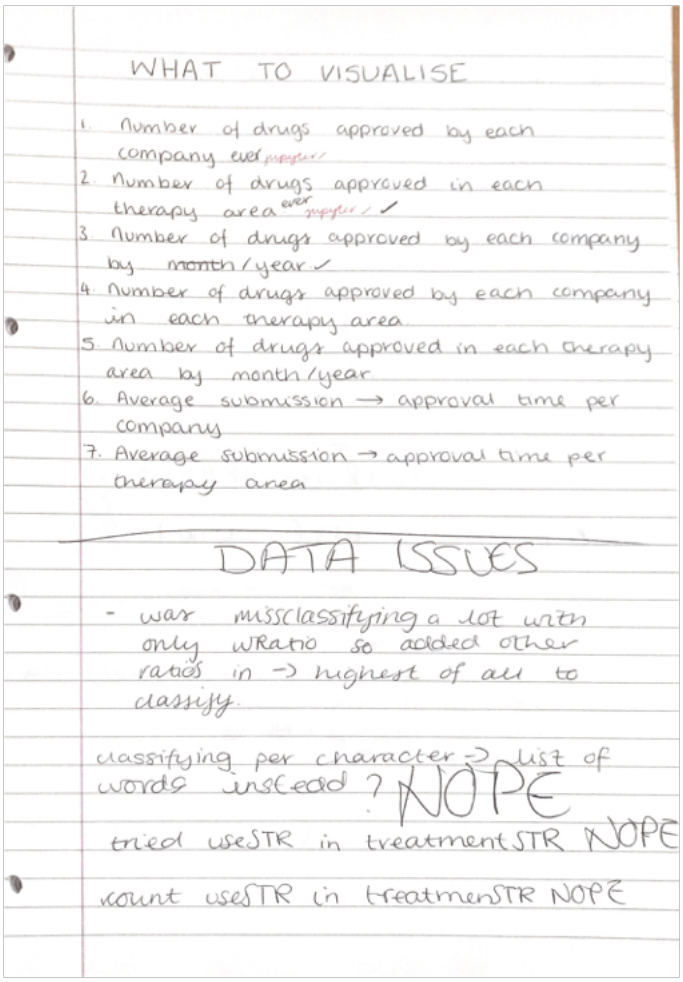
for all treatments:
  Treatment vs use
    ~~if >% similar~~ USE BEST SCORE
      assign treatment therapy area
    ~~if c% similar~~
      next use

Fuzzywuzzy library compare strings

use WRatio

FOR ~~each~~ in treatments:
  Wratio (use[:] treatment[:])

10.4.19 Fuzzy logic - Lit review



WHAT TO VISUALISE

1. Number of drugs approved by each company ever
2. Number of drugs approved in each therapy area ever ✓
3. Number of drugs approved by each company by ~~month~~ / year ✓
4. Number of drugs approved by each company in each therapy area
5. Number of drugs approved in each therapy area by month / year
6. Average submission → approval time per company
7. Average submission → approval time per therapy area

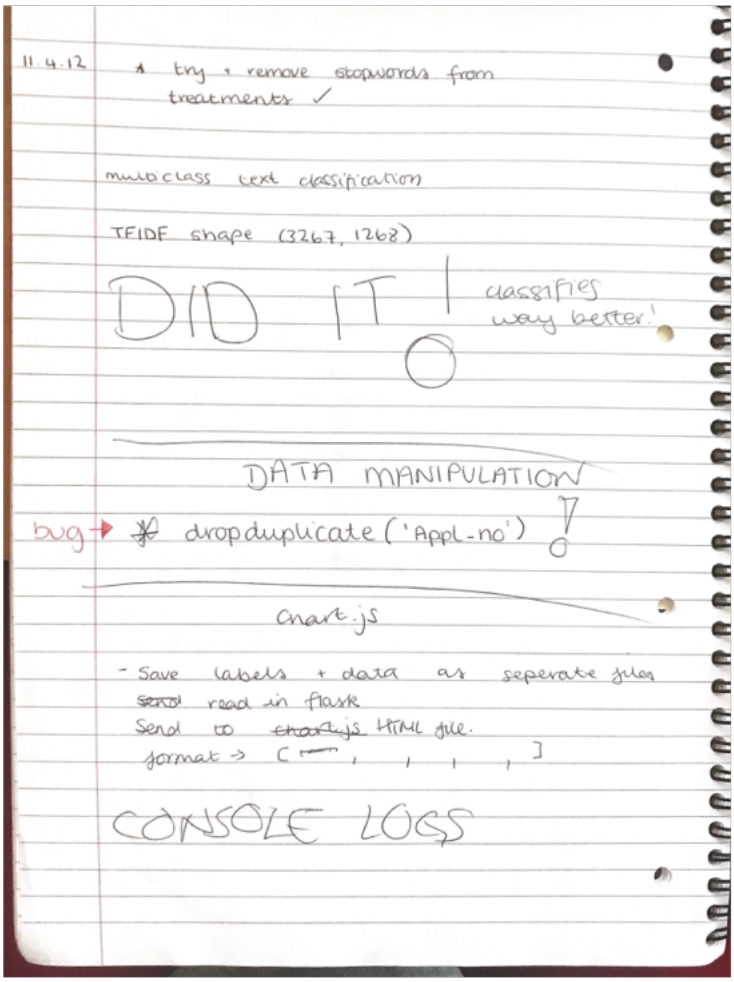
DATA ISSUES

- war misclassifying a lot with only wRatio so added other ratios in -> highest of all to classify

classifying per character -> list of words instead ? NOPE

tried useSTR in treatmentSTR NOPE

count useSTR in treatmenSTR NOPE


11.4.12
* try + remove stopwords from treatments ✓
multiclass text classification
TFIDF shape (3267, 1268)
DID IT ! classifies way better!
DATA MANIPULATION
bug → # dropduplicate('Appl-no')
Chart.js
- Save labels + data as seperate files
read in flask
Send to HTML file.
format → [, , , ,]
CONSOLE LOGS

LIT REVIEW:

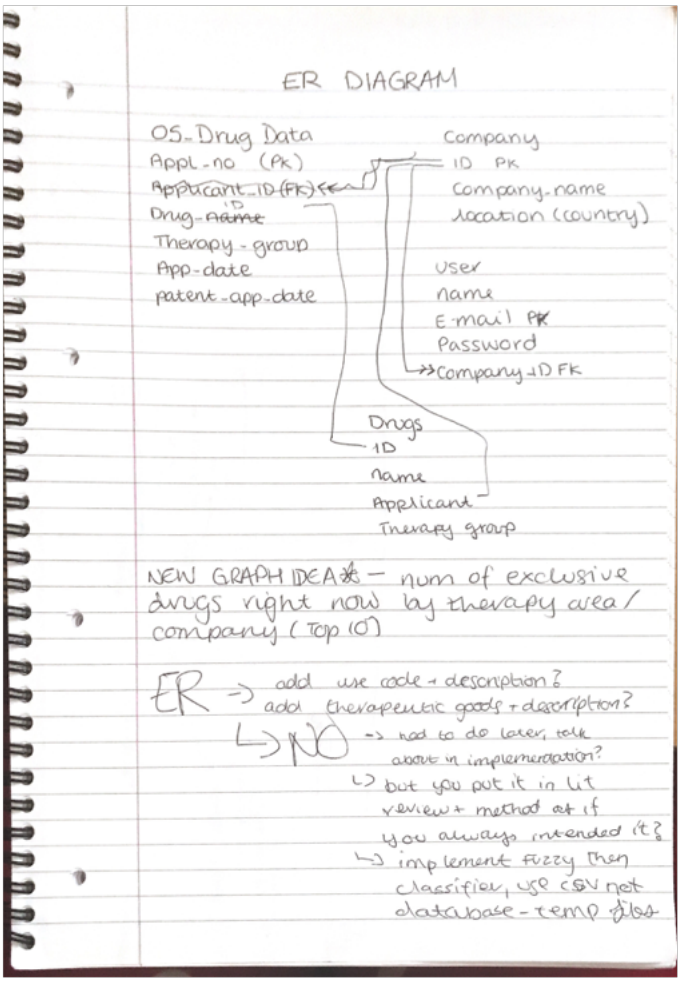
- NLP
- Different types eg NLI, QA, SS, TC
  - used part of AI coursework → need to cite it as work I've written.
- Fuzzy logic
- Models.

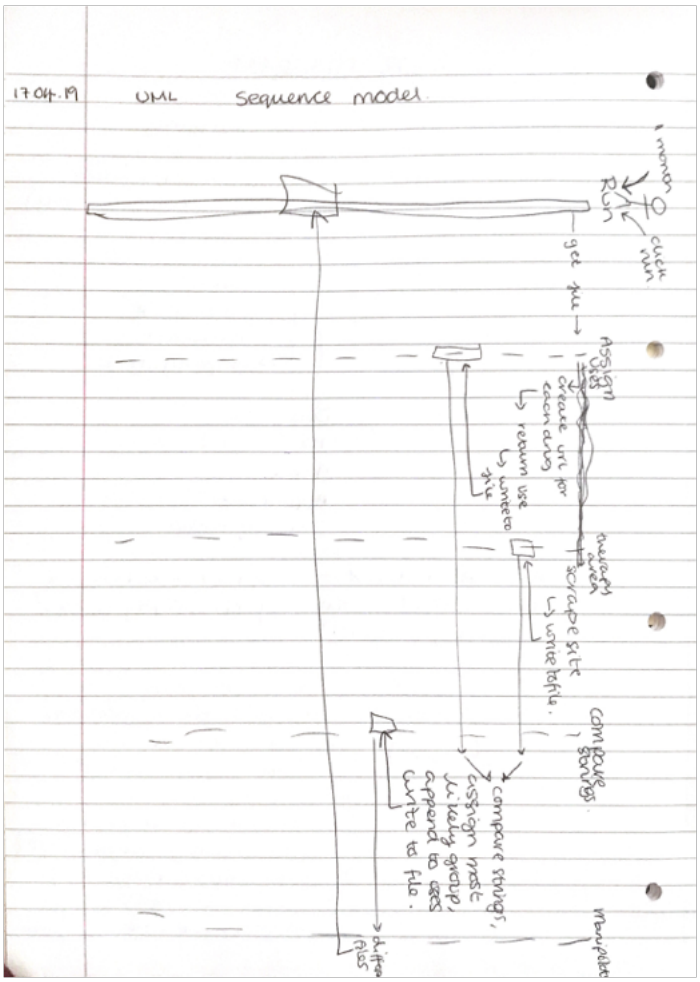
- Data gathering Analytics Solutions.
- Splunk
- SAS

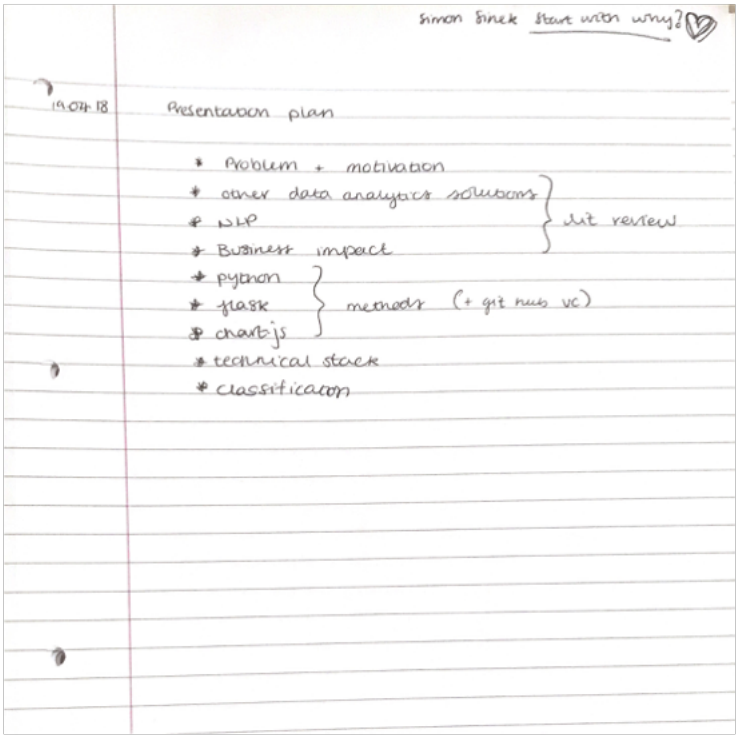
16.4.19 USER STORIES:

Personas:

Karen Davies. 29 Reg affairs
finds drug approval data.
Frustrations: takes forever.
boring.


ER DIAGRAM
05_Drug Data
Appl_no (PK)
Applicant_ID (FK)
Drug_ID
Therapy_group
App_date
patent_app_date
Company
ID PK
company_name
location (country)
User
name
E-mail PK
Password
company_ID FK
Drugs
ID
name
Applicant
Therapy group
NEW GRAPH IDEA - num of exclusive drugs right now by therapy area / company (Top 10)
ER -> add use code + description?
add therapeutic goods + description?
NO -> not to do later, talk about in implementation?
but you put it in Lit review + method at if you always intended it?
implement fuzzy then classifier, use CSV not database - temp files


17.04.19
UML
Sequence model
RUN
get file
compare strings
Manipulate


Simon Sinek Start with why?
19.07.18
Presentation plan
* Problem + motivation
* other data analytics solutions
* NLP
* Business impact
lit review
* python
* flask
* chartjs
methods (+ git hub vc)
* technical stack
* classification