http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.224.6131&rep=rep1&type=pdf

**Context**

1. Abstract
2. Introduction
   1. Project Outline
   2. Aims
3. Background literature
   1. Market Research
   2. Evaluation of competitors
   3. What is a flowchart?
   4. Web application development

3.4.1 Client Side Scripting

3.4.1.1 HTML

3.4.1.2 JavaScript

3.4.1.3 mxGraph

3.4.1.4 CSS

3.4.1.5 Ajax

3.4.1.6 Bootstrap

3.4.2 Server Side Scripting

3.4.2.1 Python

3.5 Database

3.5.1 SQLite

1. **Project Management**
   1. **Planning - Agile**
   2. **Management – Folder structure**
2. Requirements
   1. Requirement gathering
      1. Survey
      2. Observation and Personal Testing
   2. Requirement gathering analysis
      1. Survey results
      2. Observation and Personal Testing analysis

4.2 Functional Requirements

* 1. Non-functional Requirements
  2. Prototype
     1. Feedback

1. Design
   1. Use Case
   2. Database relation model
   3. Conceptual flow
2. Implementation
   1. Login, Register, Logout
   2. Profile page
   3. Homepage
   4. Functionalities
      1. File > Open
      2. File > Save
      3. File > Save as
      4. File > New
      5. Share
   5. Problems
   6. Solutions
3. Security
   1. Web application security
   2. Input Sanitisation
4. Performance
   1. Progressive rendering
   2. Response time
5. **Testing – what type???**
   1. **GUI Testing**
   2. **Backend testing**
   3. **Security testing**
   4. **Unit testing**
   5. **Evaluation**
6. Critical Evaluation
   1. Software Development
   2. Testing
   3. Additional features
7. Conclusion
8. Appendices
9. References

**Acknowledgements**

I am exceptionally thankful to my supervisor, Dr Paulo Oliva for his continuous support and guidance towards my project. His share of knowledge and advice has been outstanding. I would also like to show gratitude to my parents who have encouraged me to put in my best effort. Finally, I would like to bestow my appreciation to my friends for providing me with tips and tricks throughout this project.

**Abstract**

Cooking applications share similar features; the majority share functionalities that allow users to create and search for recipes. These recipes are poorly constructed, especially the structure of the methodology, which is too detailed and difficult to follow.

A survey was created to determine what resources people utilise in cooking a particular recipe. Analysis of the study displayed that most of the audience watched YouTube tutorials or read instructions from recipes online. However, there was an issue of waiting for a particular process to finish. Furthermore, respondents also indicated that they would rather have a diagram illustration compared to just text.

On this basis, I created a web project called Fastcook to tackle the issue of poorly constructed recipes. The projects aim is to solve these flaws by allowing users to interact with the system and constructing a flowchart that can better illustrate the current layout of recipes. This is designed and implemented using the Django web framework, python and a library written in JavaScript named mxGraph.

For my project, I have chosen a diagrammatic approach to resolve this issue because in general, information being processed via text is more difficult to remember than visual images. Research supports this as they have found that the human brain can handle images up to 60,000 times faster than words (Linkedin.com, 2018). Therefore, using graphs can support my stakeholders to comprehend the set of instructions more quickly compared to reading long paragraphs. Furthermore, a visual representation of the recipes can allow users to work 4taneously. This is because it provides an ideal representation of what the user can do overall. Whereas, when a recipe is written in steps, it makes the user follow the methods sequentially to ensure they have finished each step.

**Introduction**

The background context illustrates the research on the project; this will include the market research on current applications and what issues they face. This will then be followed up by the details on the requirements for my app, the designing and implementation phase including the knowledge gained within creating the project.

**Project outline**

Fastcook is a web application which allows users to interact with the system to create a graphical flowchart that illustrates the instructions of preparing a certain meal. This application will contain images that will represent the different events for those instructions. The images will be connected to each other by arrows to illustrate the recipe as a graph.

**Project aims**

The project aims to combat the poor layout of the instructions when following a particular recipe. Typically, the methods are portrayed in a way that shows the cook what to do per step. This consumes a lot of time as the cook would have to wait for every step to be completed. The user of Fastcook will be able to create a simplified flowchart that can demonstrate to the cook how to work simultaneously. This would be visually appealing to the user and will make cooking more efficient as they can see what to do while waiting for a particular step to finish.

**Motivation for work**

With the availability of recipes online, I could see that they faced a massive issue of portraying the preparation section. The problem I encountered was that when I finished completing one step in the recipe, I could see that the next instruction could have been accomplished within the previous step.

People who love to cook seem to have the issue of learning to cook efficiently. There are numerous amount of resources that can be utilised online to provide recipes for any meal. However, the problem with this is the way the information is presented. Most of the websites that I have visited such as BBC provides a simple layout, but with detailed sets of instructions, making it difficult to comprehend when learning to cook. As information is presented in steps, it is within human nature to follow each step independently. This disallows people to work simultaneously. My approach to tackling this issue is to allow users to create a flowchart representation of the recipes for users to be able to work simultaneously; hence, make their cooking more methodical.

# **Background Literature**

This section provides the background research I carried out to obtain the requirements for my Fastcook project.

**What is a flowchart?**

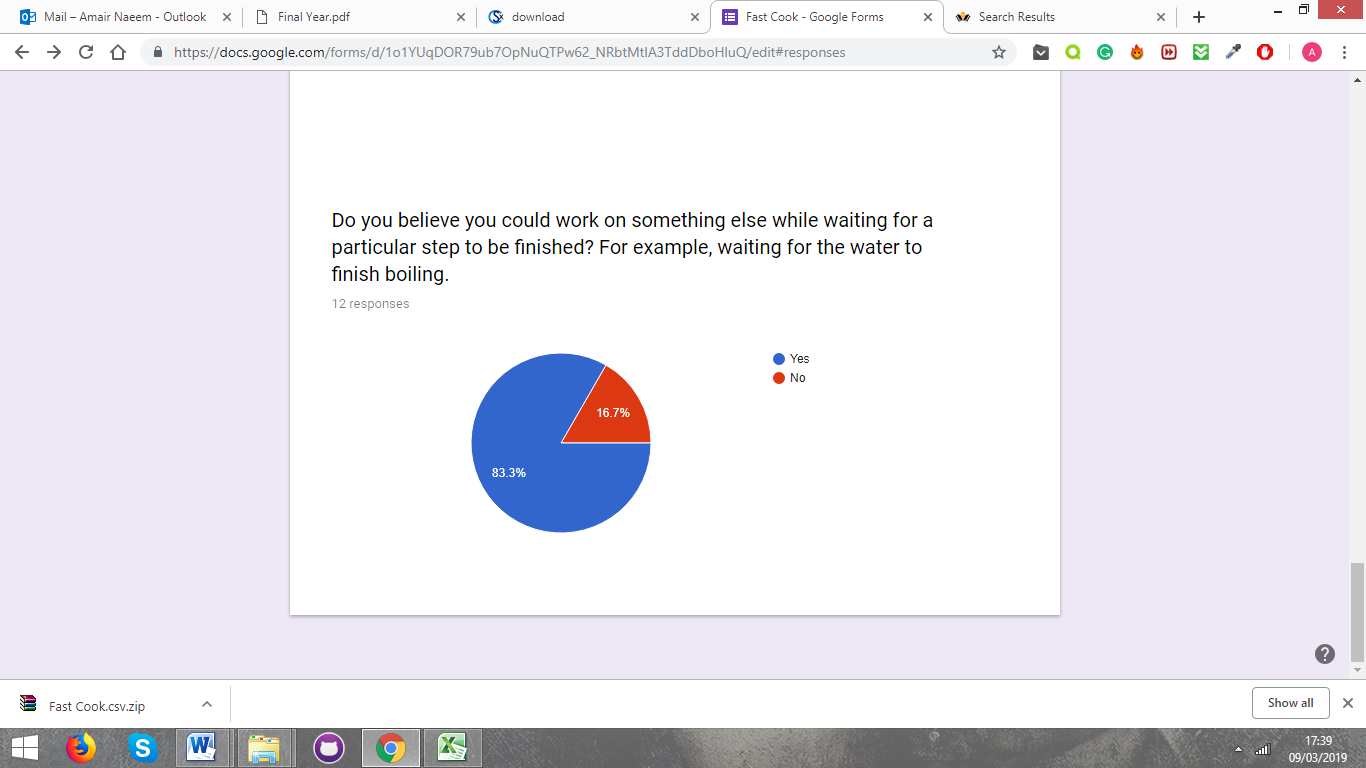
A flowchart is a diagram that describes sequential steps to be followed and perform an algorithm or process. Each step in the process is represented by a symbol that can have different definitions (Breezetree.com, 2018). Arrows connect the steps which illustrate a logical representation of information, allowing users to follow the process with ease.

For Fastcook, the ingredients will represent parts of the recipe and the arrows would indicate the next step in the recipe.

**Market research**

In order to conduct my market research, I created a questionnaire where I asked people who love to cook on the issues they are normally faced with, how they learnt to cook and resolutions of the problems they faced while following a recipe.

From my research, I discovered that most of my audience watched YouTube tutorials or read instructions from recipes online. However, they had the problem of waiting for a particular process to be finished. They believed that they could have used the time to prepare another part of the meal. Figure 1 illustrates that 83.3% of people could have easily accomplished another step while waiting for an action to be finished. Furthermore, the data also indicated that people familiar with cooking are more likely to have a visual representation, compared to a set of instructions written in plain text.

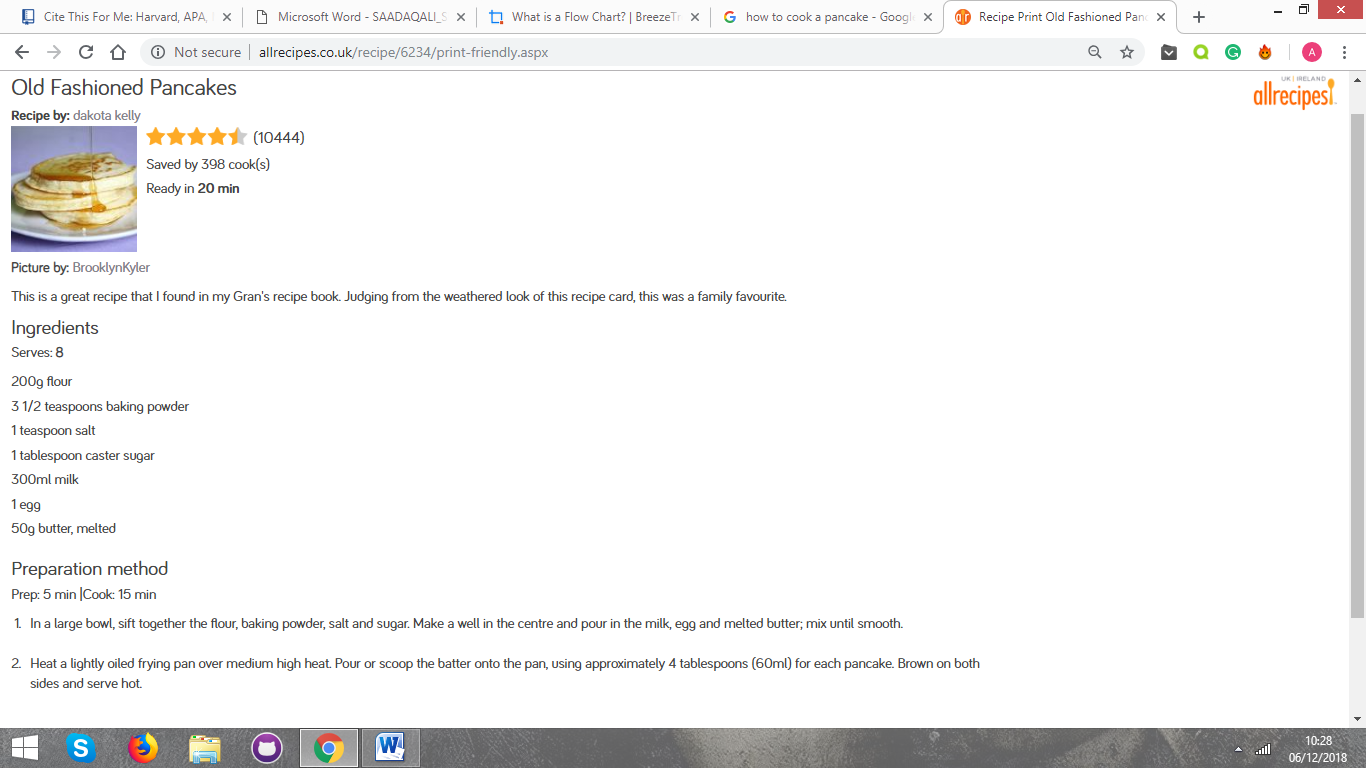


**Figure 1 – Survey result**

**Allrecipe**

Allrecipe is a community website application which allows users to upload their recipe by video or in text format. It contains a form where you can include the description, ingredients and directions on how to cook a particular meal. The layout of the web application is well presented as it has separated the paths, ingredients, additional notes and nutritional facts of the meal (see Figure 2). However, it still does not illustrate the sets of instructions via a diagram which disallows people to work effectively.

Title of recipe



Ingredients section

Preparation

Figure 2 – Allrecipes (Allrecipes.co.uk, 2018)

The layout is poorly constructed and the recipe as a whole does not look visually appealing. There is not any indication of colours used to display the method, which makes it difficult to follow. One appealing factor of the site was that it was easy to navigate from the different section of pages as it had a standard navigation bar located at the top of the page. Another feature that intrigued me was the way the user could browse through the recipes. From figure 2.1, we can see the different types of ways to browse through recipes such as ingredients, meal type, diet and health. This can narrow down the search for a recipe; hence make it easier for the user to get what they’re looking for.

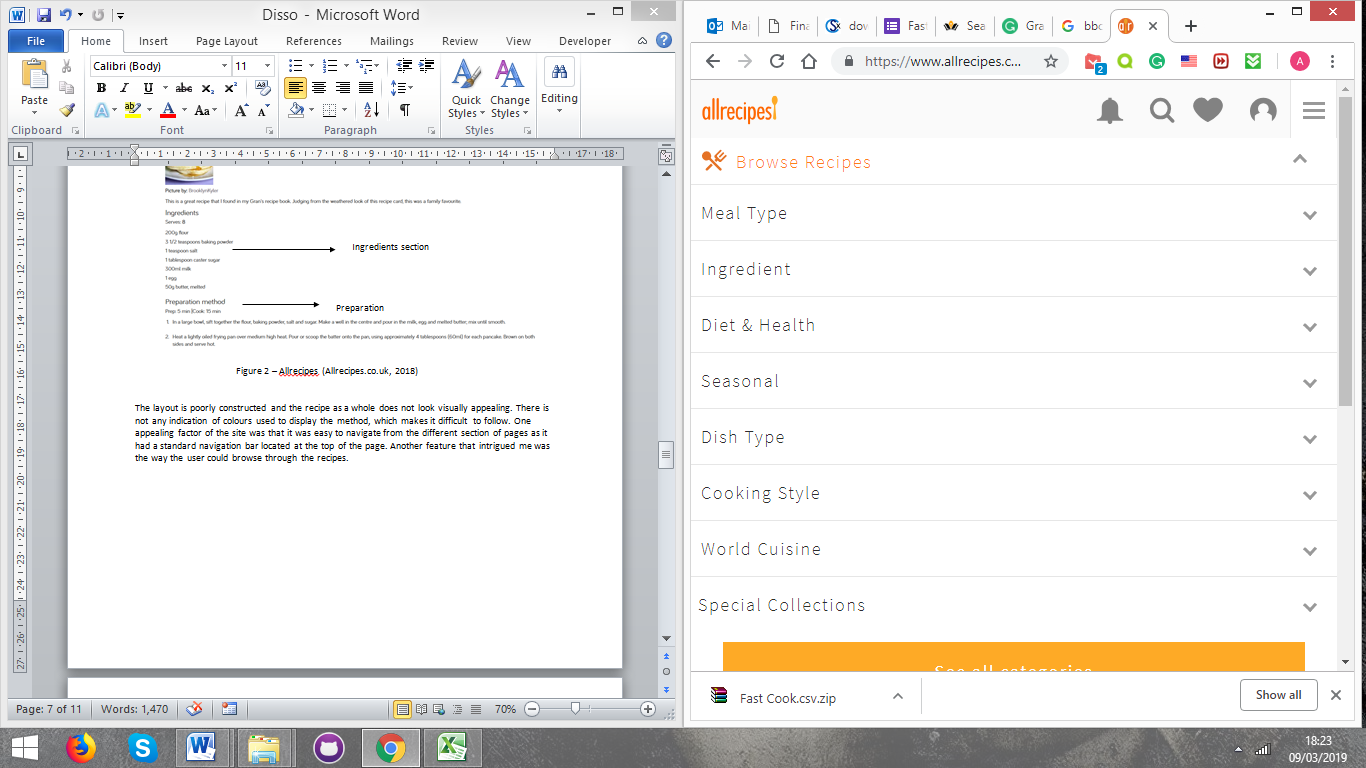


Figure 2.1 – allrecipes sections

**BBCGoodFood**

BBCGoodFood is another website that displays recipes online. The website provides recipes, but does not allow users to post their own. The recipe is tested by professionals and is then submitted for the public to see.

In comparison to allrecipe, the format and the way the sections are divided are presented more attractively. The website has additional features, such as the time it takes to cook and the difficulty level (highlighted in red in figure 3). This allows users to identify whether they are capable of cooking the particular meal. However, it still does not solve the issue of cooking efficiently as the methodology is written in a series of steps.

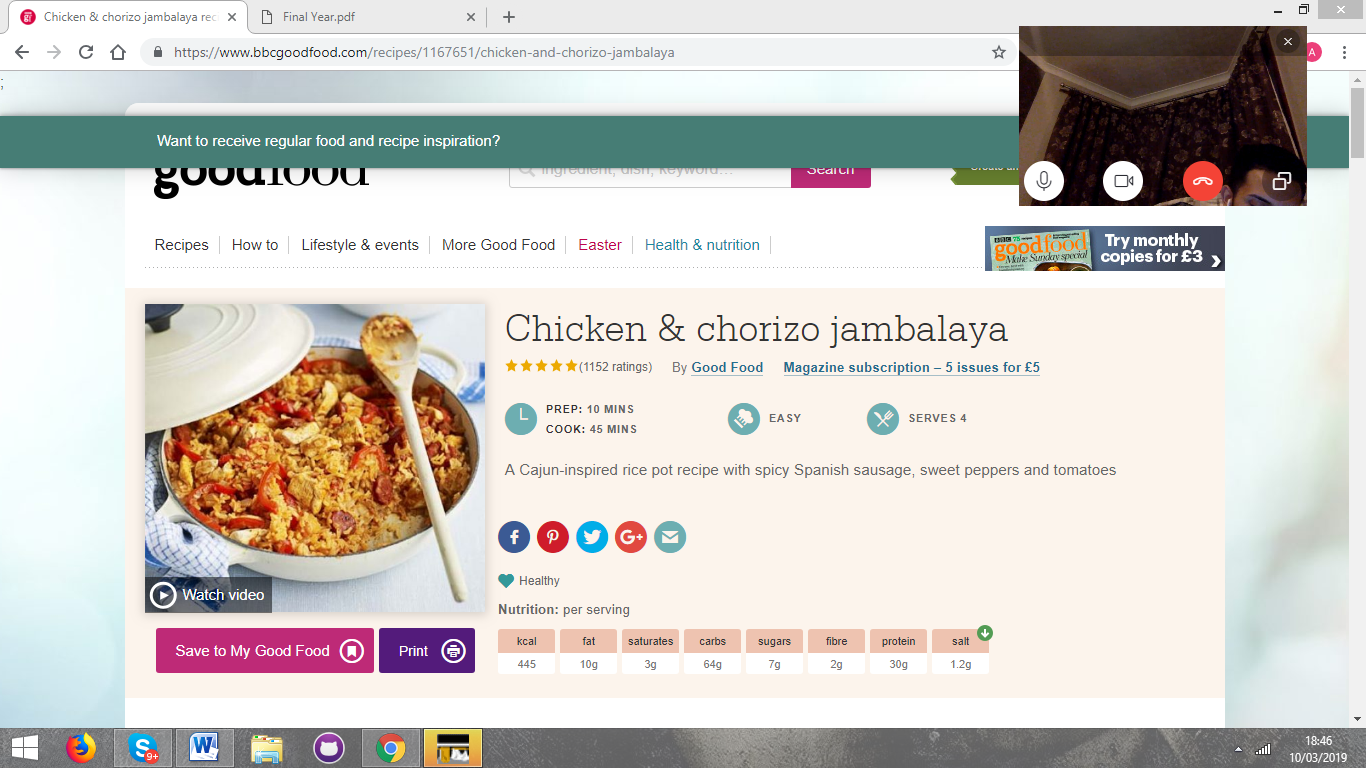


Figure 3 – BBCGoodFood recipe

|  |  |  |
| --- | --- | --- |
| Applications | Key Features | Lack of Features |
| Allrecipes | * Allows users to share their recipe online. * Easy to find the types of recipes as it is broken down into different sections. | * Poor format of the ingredients sections * Not visually appealing, looks really plain |
| BBCGoodFood | * Shows preparation and difficulty, which allows different types of users to know if they can cook the meal or not. * Better layout compared to allrecipes and is more colourful. | * Recipe is difficult to follow as it is too descriptive and is displayed in sequences. |

Figure 4 – Comparisons of apps

## **JavaScript and JQuery**

JavaScript is a programming language which is used on the web browser. It allows programmers to implement complex methods that would enable users to have a better experience. For example, it can provide the drag and drop methods for my web application. Furthermore, JavaScript is a simple language in terms of its syntax and so it is easy to implement.

JQuery is a JavaScript library that simplifies my code for faster web development. The usual tasks carried with JavaScript may require a lot of code for me to achieve. JQuery creates methods that I can use instead, which can be written in a few lines of code.

## **MxGraph**

The library chosen for this project is called mxGraph. It utilises JavaScript functions, HTML as well as scalable vector graphics (SVG) to render diagrams, allowing users to see how objects are related to each other (Jgraph.github.io, 2018). This library is sufficient to support my project as it contains useful functionalities to draw, interact with and associate context with a diagram.

The key advantage of mxGraph is that it is standardised, meaning my web application can be deployed on different browsers without further configuration from the client. Also, there are no plug-ins required to install mxGraph. It would merely be a matter of deploying the library on my web application (Jgraph.github.io, 2018).

## **aJax**

AJAX stands for asynchronous JavaScript and XML. XML is a markup language that provides the browser instructions to display a page. XMLs main functionality is to transfer data so that machines can easily communicate with each other.

AJAX is used to allow web pages to exchange information asynchronously. This implies that the web browser does not need to be refreshed when some part of the data on the page needs to be altered. The main reason why I chose AJAX is because it cuts down the network load hence improves the users’ experience. AJAX was used in order for users to be able to open the recipes they created, save the recipes or create a new recipe. This increases the performance of the browser and increases the speed to retrieve the information to the user.

https://dzone.com/articles/pros-and-cons-of-ajax

## **Django**

I have utilised Django to use on my web application. Django is a web framework written in python that inspires programmers to write clean code and includes packages which will make my application more efficient.

Django provides a great range of features that can help support my project. One of the key features it includes is the object relational mapper (ORM) system, where it would be easy to perform database operations. It would also be quicker for me to switch between different databases, for example, MySQL and SQLite. Another reason why I picked Django is that it has the “Don’t Repeat Yourself” dry pattern, which is also known as DRY. It means that my languages are separated in different files, and there should not be any repetition of code. This makes my functionalities efficient and illustrates cleaner code (Hackr Blog, 2018).

Django has a layer in its structure called views. The view is responsible for responding to the client, depending on what the user requests. It allows the HTML templates to access different data to display in the frontend. Django would mainly be used to enable users to register, login and save their diagram on the cloud so they can finish their work at a later stage. I will be separating each functionality by a separate view as different features will imply a different user request.

Django uses a Model-View-Controller architecture (MVC). The model indicates the logical data structure behind the application. This is represented by a database, Django support PostgreSQL, MySQL and Oracle. To configure the database, this is done inside the settings.py file. The view is the interface that is described on the front end. This is displayed to the users when they visit the website. HTML/JavaScript/CSS files generally represents these. The controller is what connects the two; it processes information from the model to the view (Medium, 2018).

## **HTML, CSS and Bootstrap**

HTML stands for HyperText Markup Language that is used to create web pages. Django utilises HTML as templates to render pages displayed on the browser.

Cascading style sheets (CSS) is a language that describes how the elements in HTML should be displayed. CSS is also used to describe the presentation of the web application which can include the colours, shapes and different font types (Hackr Blog, 2018).

Bootstrap is a HTML, CSS and JavaScript framework to help design webs much quicker. It will be used in my web application to create an attractive and a responsive interface.

## **Python**

Python is a server side and object oriented programming language mainly used for web development. It is simple and the language can be interpreted more easily compared to other languages. The language does not need to compile, which means that the code does not have to be translated to binary when it is run. Django utilises python to encourage quicker and cleaner development.

<https://www.pythonforbeginners.com/learn-python/what-is-python/>

<https://www.djangoproject.com/>

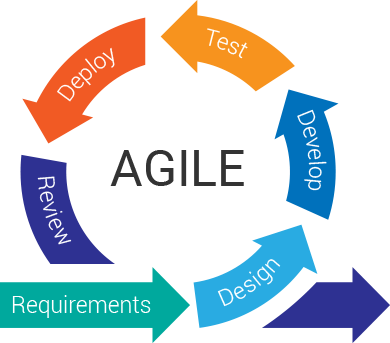
## **sQlite**

SQLite is the database management system that is utilised in Django. It is a popular deployed database engine and is widely used within web applications. SQLite provides rapid performance, reading and writing the files from this database is stated to be faster compared to reading and writing files from disk.

<https://www.sqlite.org/aff_short.html>

## **Project Management**

Planning



Agile development model.

Figure 5 – Comparisons of apps

<http://www.travancoreanalytics.com/why-agile-is-a-popular-choice/>

The development model chosen for this project was the agile development model. Agile development is an iterative process that focuses on ensuring the product is up to scratch.

The first part of the stage is the requirements process, where the aims and objectives had to be broken down into tasks for the development. From the background research, information was gathered to get the main requirements of the web application. The requirements were then set with deadlines to help organise the project.

As with large projects, there is always the possibility of issues occurring throughout the development process. Therefore, a risk assessment table was created try and detect any unforeseen risks that may potentially cause issues. The combination of the likelihood rating as well as the impact of the risk and how to prevent the risk was taken into consideration. (appendiz ?)

To visualise what the activities and milestones, a Gantts chart was created. The tasks within the Gantts chart was updated thoroughly, as some tasks were either finished before the deadline, or seemed too difficult to meet the deadline.

Management

Github is online clouds based service that allows users to store and track their data. A repository was created to keep backups of the reports and development process. This was done to ensure data was not lost and revert back to old code, if there were any errors in the current version.

# **Requirements**

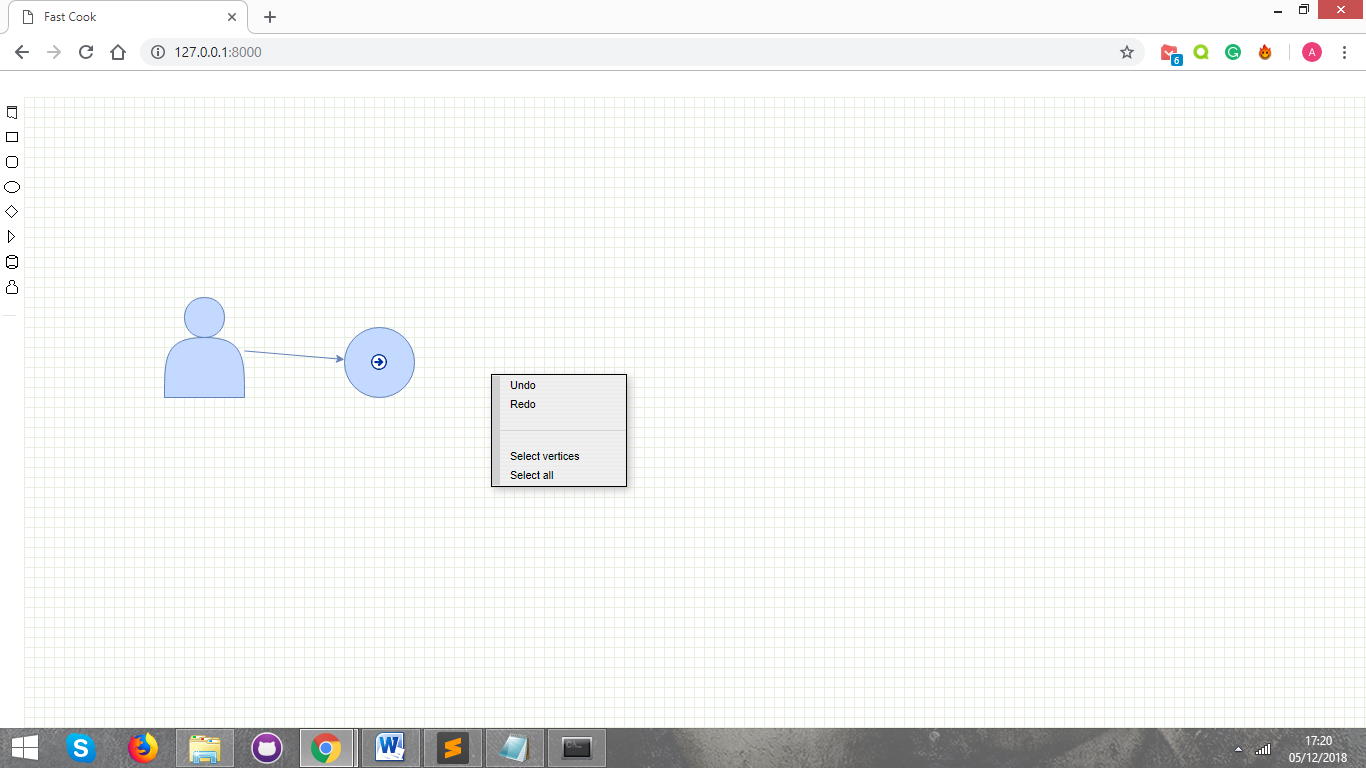
After the background research, a high-fidelity prototype of the project was created. High-fidelity is a representation of the actual implementation of the project. This was followed by the functional and non-functional requirements for the web application.

Prototype

The purpose of the prototype was to observe and get direct user feedback. Within this prototype, a simple toolbar, as well as a canvas was created. Users were able to drag and drop from the toolbar onto the canvas and connect elements using an arrow (as seen from figure 6.1).

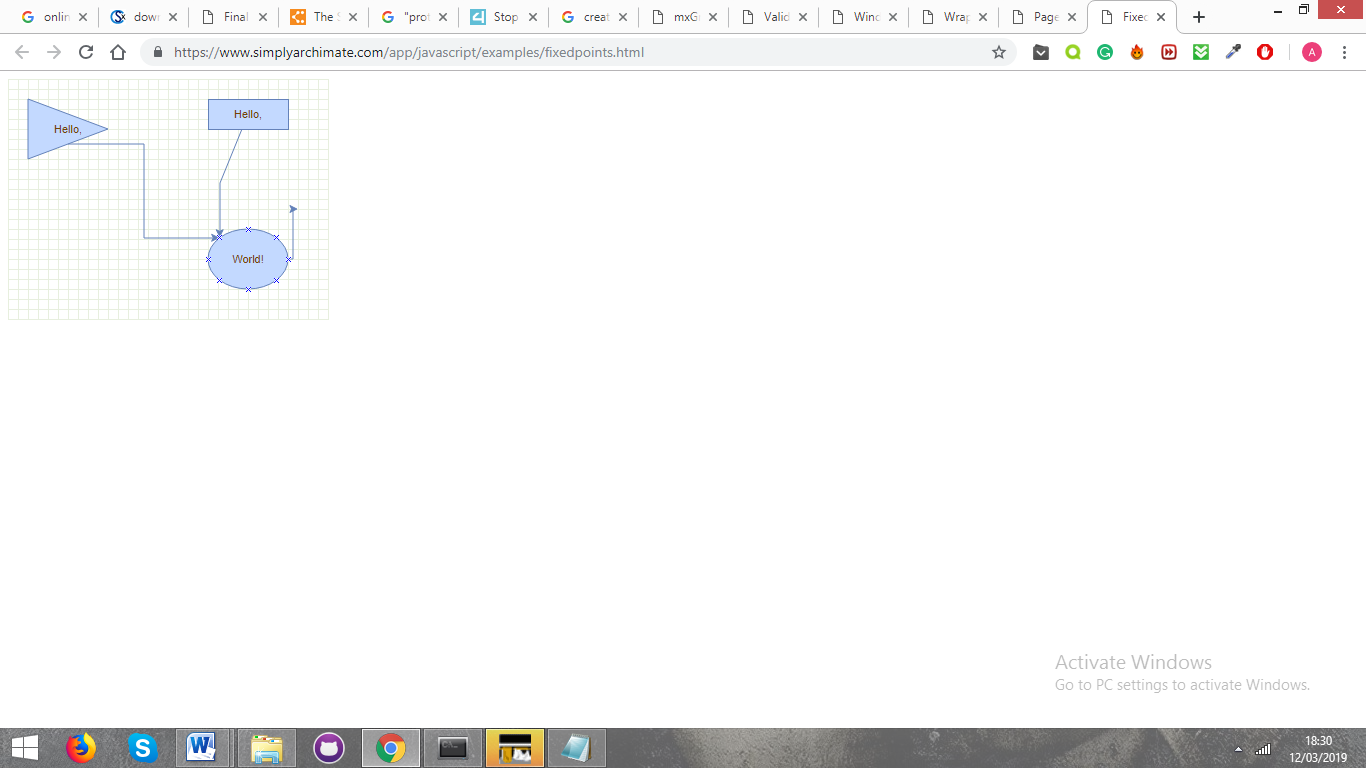
For users to draw the arrow, the user had to hover over the element that was dropped onto the canvas. Once hovered, the element would show crosses around the shape, in which the user would have to drag to create the arrow. The arrow can either be connected to another shape or be dangling on its own (see figure 6.2).

Toolbar – users are able to drag and drop elements to canvas



Allow users to connect elements

Figure 6.1 – Prototype



Crosses to draw arrows.

Figure 6.2 – Hovering to create arrow.

**Feedback**

Generally, users enjoyed the idea of the drag and drop functionality. It was described as simple and visually pleasing. Users stated that it was easy to figure out the purpose of the toolbar and believed that the overall process of design was user-friendly. However, they thought that the toolbar was relatively small and the icons were difficult to define.

Another issue was that some users were not aware of the “drawing the arrow” functionality. They found the crosses hard to see and did not realise that you could create an arrow from it. Furthermore, some users described the detached arrow as “pointless” as they believed that the arrow should always be connected to another element.

The feedback helped to identify the requirements section. This section is broken into functional and non-functional requirements. Functional requirements are used to describe what the system should do, whereas non-functional requirements specify how the system works (Hackr Blog, 2018).

## Functional Requirements

1. Users should be able to register on the web application as well as login.
2. The user should be able to have access to a toolbar, in which they can drag shapes and drop it onto a grid canvas.
3. The elements that are dropped onto the canvas should be resizable and be able to connect to other elements. The connection that is drawn should be easy for the user to identify.
4. The items dropped on the grid should allow users to input text so they can provide instructions or commands in what they need to do with the particular element. For example, “boil water” in a pot.
5. Users should be able to delete the elements on the grid.
6. The user should be able to undo or redo any action they committed on the grid.
7. The items stored on the canvas that are represented as JavaScript should be stored on the database, so that the users can finish off their flowchart from where they left in their previous session.

## Non-Functional Requirements

1. There will be server validation such as Cross-site request forgery (CSRF). These are tokens which will be used to enhance security and restrict attackers to retrieve data from the user.
2. Testing should be carried out throughout the web application as well as at the end of the application to ensure that different types of errors are handles correctly.
3. AJAX requests will be made to make the user experience more efficient such as when the user is able to save the graph.
4. The application should be accessible on most browsers.
5. Provide an attractive interface using Bootstrap and CSS.

Performance

* Minify
* CDN
* Scripts at the bottom
* CSS at the top

References

Linkedin.com. (2018). *5 Reasons Why Images Speak Louder Than Words*. [online] Available at: https://www.linkedin.com/pulse/5-reasons-why-images-speak-louder-than-words-gabe-arnold/ [Accessed 1 Dec. 2018].

Breezetree.com. (2018). *What is a Flow Chart? | BreezeTree*. [online] Available at: https://www.breezetree.com/articles/what-is-a-flow-chart/ [Accessed 6 Dec. 2018].

allrecipe