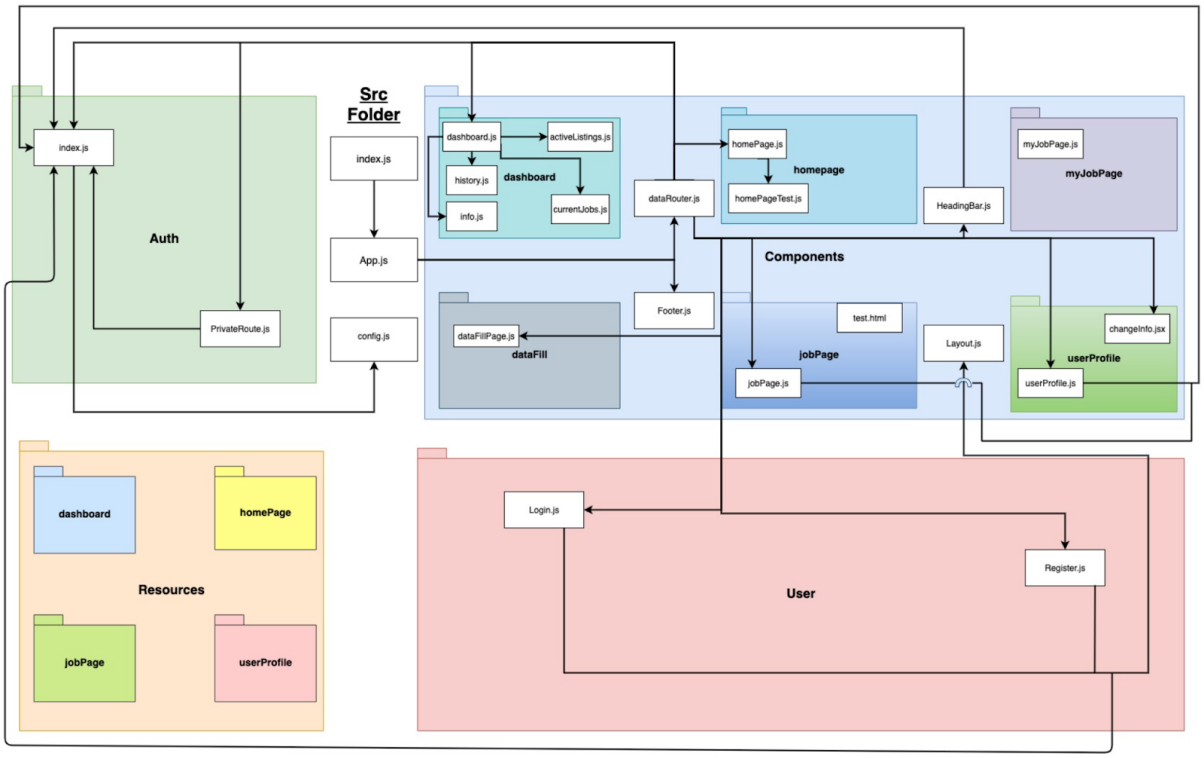
COMP4050 Group 2 Project Design and Overview

Group 2 Members:
Ameer Karas, 44948956
Chris Purkiss, 45258937
Joshua Vazzoler, 45228353
Katrina David, 45308748
Yefan (Tony) Li, 44768001

Package diagram:



Overview of the code (strengths and weaknesses): (200-300 words)

Following a review of the code base, our team has identified several issues. Firstly, the lack of a database; without a database the profile and dashboard section of the website is useless, resulting in errors and blocked access to pages. Therefore, we will need to implement a database.
Additionally, the code currently uses many different folders, most having no useful files in them. For example, the Resources file has four files in it; each of those files is a gitIgnore file. This results in a confusing layout of the files and how they interact.
Lastly, the overall spread of the files is confusing. For example, the index.js file is imported by 7 other files which makes it reduces the clarity of its purpose.
There are some strengths of the code that we have also identified. Firstly, of these is the use of many components and classes, rather than just one large file. This means is that the components and files can be given a clear purpose, for example, displaying the favours history or a user's profile information.
Secondly, the features of this project have their own folder/file. For example, the dashboard has its own file, as does the user profile. What this means is that debugging and altering these

features/components is much easier than it would otherwise be. These strengths and weaknesses will be considered during each sprint of the project.

Feature implementation: (500 words)

Map and Search - New Feature

This feature Involves guest users entering their address and choosing the favour they want. An embedded Google Map will also be on the home page and this works in conjunction with the search results. The search results will feature the individuals in the community offering that favour and an option to request that favour from the individual. The google map then reconfigures and illustrates which individuals are closest to the guest user based on the location they entered in the search phase of this feature. The google maps component will be generated by creating a new react component for the map which contains the Google Maps API key, then creating another component to hold the address on the map and finally embedding the map onto the home page.

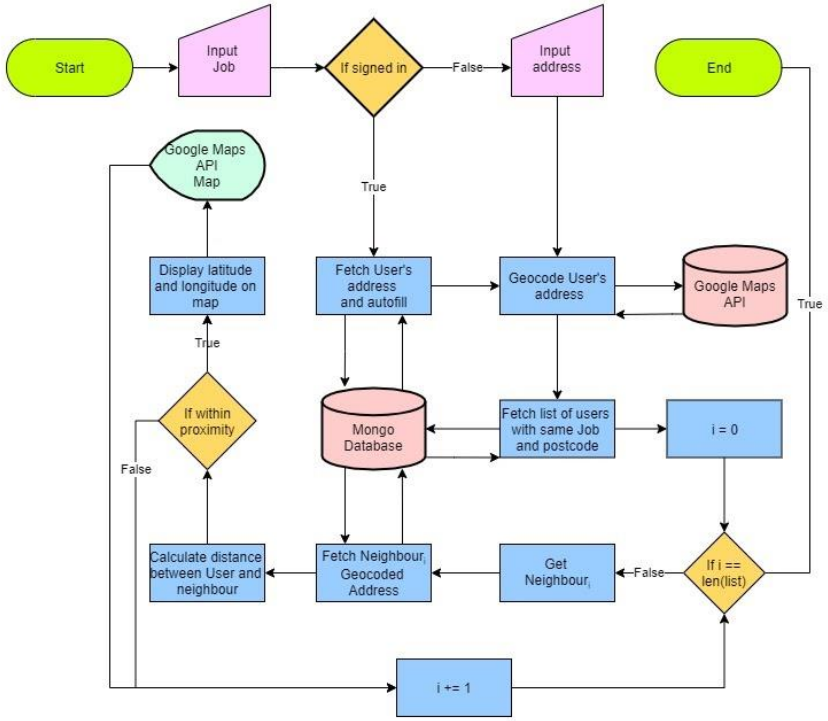


Figure 1: Flow diagram for 'Map and Search' feature.

Dashboard/User Profile

The dashboard and the user profile functionality are the central place that the users' personal information and the information about the various favours are stored. The dashboard contains will contain but not subjected to the following types of information:

- **Personal information** which takes the user into their personal profile page
- **Favour History** where the user can check the details about the jobs that they have already done.

- **Current Jobs** shows the current favours that the user is providing to the customer which includes details about the customer and their relevant information about the specific favour.
- **Active Listings** are the favours that a user is currently offering and also shows the relevant information about the favour and the person who is requesting the favour.
- **Bookmarked Favours** is a page reserved for favours that needs the users' attention or there are some complications that are associated with the favour which needs to be attended.

The User profile is a dedicated page where it shows the personal information that is linked to the **Personal Information** entry point in the dashboard and various other points throughout the application. This page should only be available to a user that is signed in, if a customer wants to learn more about that user, there will be an authentication mechanism to protect sensitive information.

MERN Stack

MERN is a full stack that follows the traditional 3-tier architectural model. The MERN acronym stands for MongoDB, Express, React, Node, which are the four technologies that the stack is comprised of. MongoDB is the database tier and is used to store any data of the application (for example, user profiles). React.js is used for building complex interfaces and dynamic client-side applications. Finally, Express.js is a server-side framework that runs inside a Node.js server, and is used for handling requests and responses over HTTP. This stack will allow us to construct a 3-tier architecture (front-end, back-end, database) using JavaScript and JSON.

Request System

The request system is an improvement to an existing feature where a user (customer) can request for a favour. This request system is integrated into the new Map and Search feature. This comes after a user (customer) searches for their desired favour and wants to book for that favour. The flow diagram of this system is as follows:

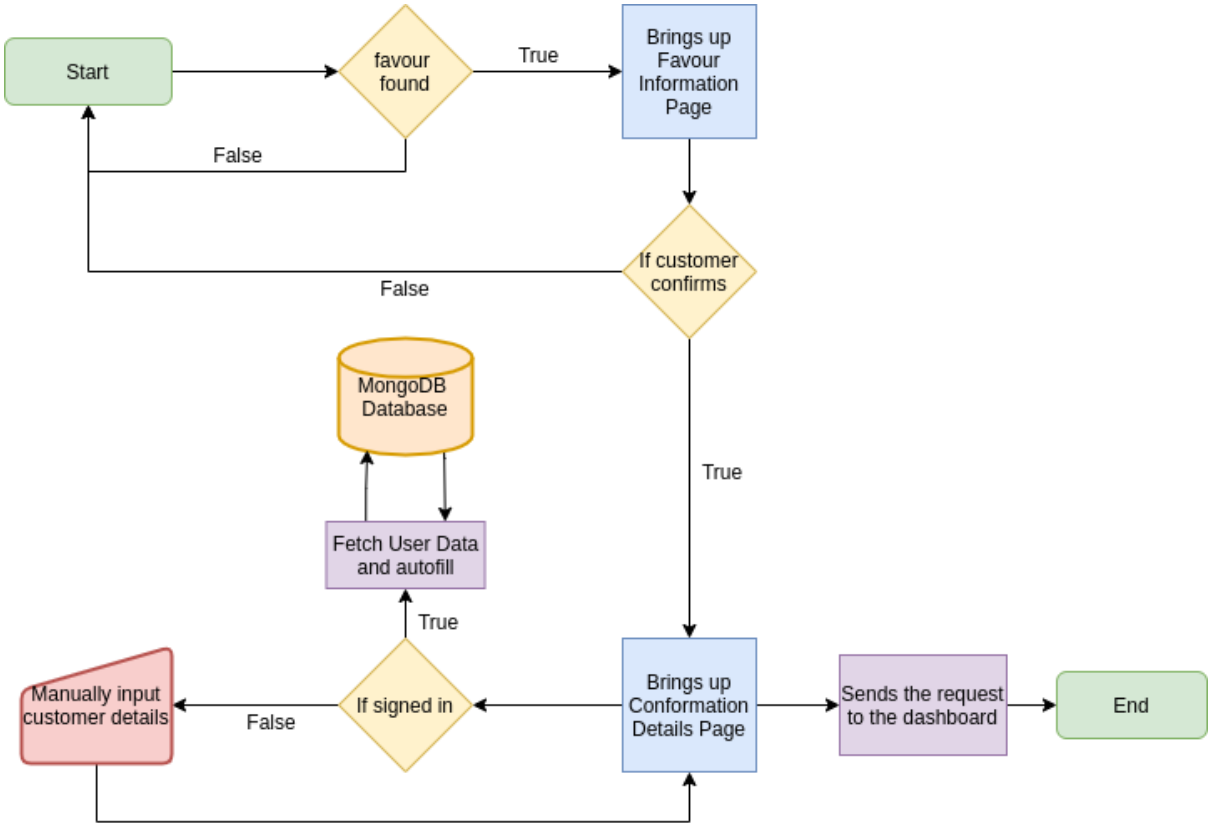


Figure 2: Flow diagram for ‘Request System’ feature.