Project Report

COMP9900: Information Technology Project 22T3

Group Name: 9900-F18P-theFuture

Submission Date: 16 November 2022

Group Members

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Email | Student ID | Role |
| Mingxuan Chen | mingxuan.chen@student.unsw.edu.au | z5330738 | Developer |
| Mohanteja Dharmavarapu | m.dharmavarapu@student.unsw.edu.au | z5242057 | Developer |
| Kamiyu Hijikata | k.hijikata@student.unsw.edu.au | z5353288 | Scrum master & Developer |
| Simon Keung | s.keung@student.unsw.edu.au | z3292483 | Developer |
| Tianchen Lan | tianchen.lan@unsw.edu.au | z5114773 | Developer |

Contents

[Contents 2](#_Toc119263340)

[Overview 4](#_Toc119263341)

[Background 4](#_Toc119263342)

[Existing Online Options 4](#_Toc119263343)

[Functionality Overview 4](#_Toc119263344)

[Architectural Overview 5](#_Toc119263345)

[Overview 5](#_Toc119263346)

[Frontend 6](#_Toc119263347)

[Backend 6](#_Toc119263348)

[Backend/Front End Integration 6](#_Toc119263349)

[Choice of Architecture 7](#_Toc119263350)

[Description of Functionalities 8](#_Toc119263351)

[Technical Challenges 12](#_Toc119263352)

[Recommendation of Similar Recipes 12](#_Toc119263353)

[Worked Example 13](#_Toc119263354)

[Price Estimate 13](#_Toc119263355)

[Ingredient Selection 14](#_Toc119263356)

[Loading Screen 15](#_Toc119263357)

[State Management 15](#_Toc119263358)

[Third Party Functionality 15](#_Toc119263359)

[Effect of Licensing 17](#_Toc119263360)

[User Documentation 18](#_Toc119263361)

[Server Setup and Installation (Estimated time: 20 minutes) 18](#_Toc119263362)

[Application Instructions 21](#_Toc119263363)

[Home Page 21](#_Toc119263364)

[Search & View Recipes 21](#_Toc119263365)

[Logging In 21](#_Toc119263366)

[Logging Out 21](#_Toc119263367)

[Creation of Explorer Account 21](#_Toc119263368)

[Upgrade of Account to Contributor Account 21](#_Toc119263369)

[Change Password 22](#_Toc119263370)

[Forgot Password 22](#_Toc119263371)

[Update Contributor Details 22](#_Toc119263372)

[Create a Recipe 22](#_Toc119263373)

[Edit a Recipe 22](#_Toc119263374)

[Delete a Recipe 23](#_Toc119263375)

[Like and unlike a Recipe 23](#_Toc119263376)

[Commenting a Recipe 23](#_Toc119263377)

[Support a Contributor 23](#_Toc119263378)

[Subscribe to a Contributor 23](#_Toc119263379)

[Similar Recipe Recommendation 23](#_Toc119263380)

[Submit Recipe Price Estimate 24](#_Toc119263381)

[References 25](#_Toc119263382)

# Overview

## Background

Recipes are documents that describe how to cook a particular dish. There are many ways in which recipes are shared; through online websites, through books, or even just by pen and paper passed between friends and family. In recent times however, the world wide web has caused massive growth in sharing and collaboration in practically any domain, and the online world is now a common place in which they are shared.

### Existing Online Options

There are numerous avenues which provide access to these recipes online; perhaps the most commonly used are websites that consist of user contributed recipes that are accessible as a database of information, such as Allrecipes and Kitchen Stories. While these provide vast amounts of knowledge, they provide little interaction between users beyond basic likes and comments. Other websites include those where the website creator/owner is also the sole content creator, such as Sally’s Baking Addiction (Sally's Baking Addiction, 2022). These often more social interaction from the poster by the way of more personal, blog-style recipe posts, but lack meaningful social interaction from the user end, and also lack the depth of information available on user-contributed sites. Finally, there also exists a multitude of forum-style websites such as Cooking Bites (CookingBites Cooking Forum, 2022) and Discuss Cooking (Discuss Cooking - Cooking Forums, 2022), as well as cooking-focused communities on content aggregation sites such as Reddit, where social interaction is aplenty. Unfortunately, their free flowing and less structured information makes it more difficult to ascertain information in a convenient manner, and often lacks an effective search function to retrieve older information.

## Functionality Overview

Having reviewed certain existing recipe websites and identified some limitations of those sites, we proposed to create a new online website for Contributors to share their recipes, which attempts to fill a niche that addresses a combination of needs not identically addressed by some other website.

To explain our website, we define two classes of users; Explorers and Contributors. An Explorer is someone who wants to find and view recipes and can either be a user with an account (and logged in), or it can be a guest with no Explorer. A Contributor is someone who wants to share their recipes and is an user with an account who has upgraded their account with additional profile information. A Contributor’s account also has all the functionality of an Explorer’s account.

In our website, everyday people can become a Contributor to casually write and publish recipes that is shared on the website. We say it is ‘casual’ as unlike some media, there is not high requirement to become a Contributor, nor is there any formal editorial process, the intent is to support free flowing of information.

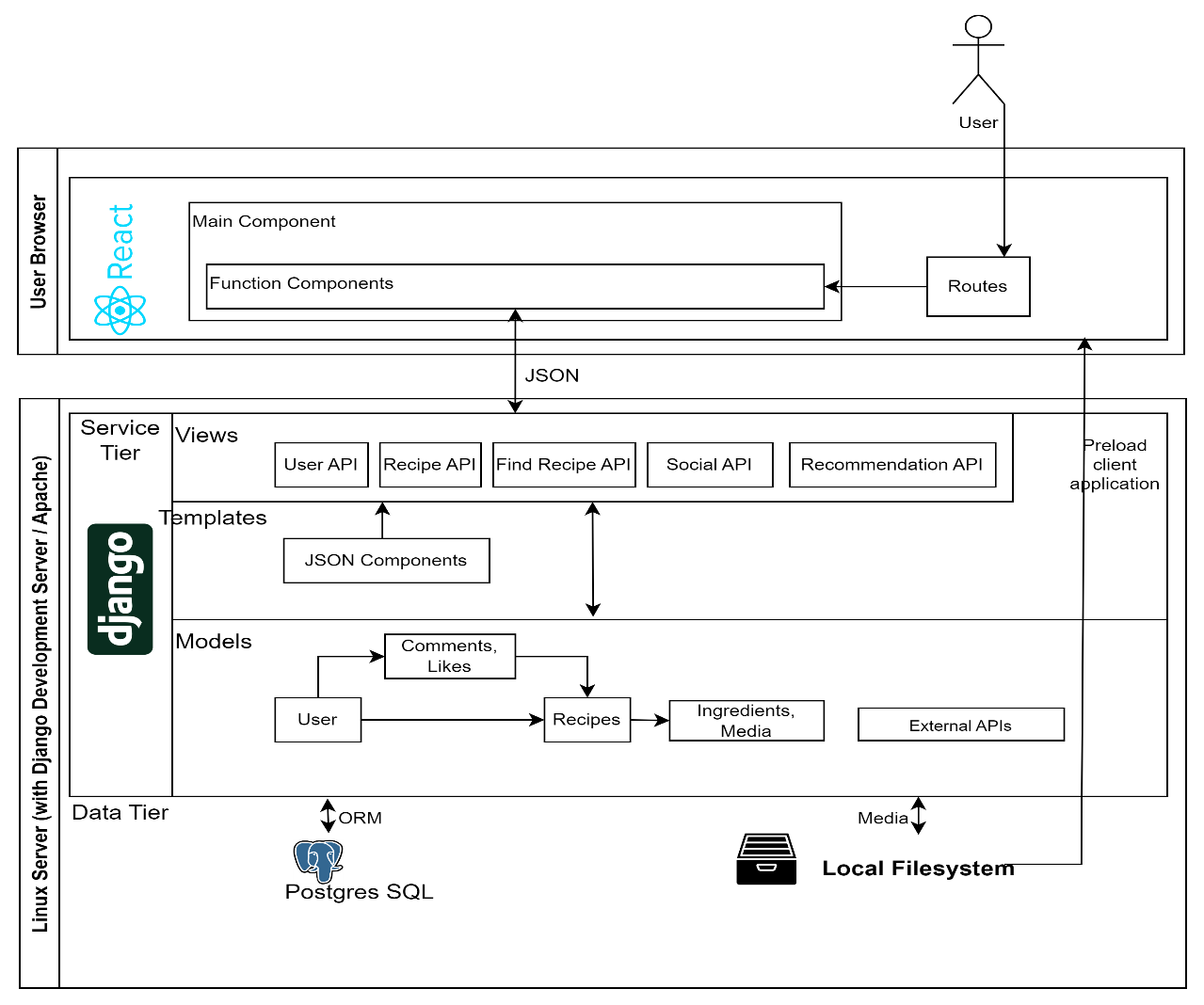
Once recipes are shared on the website, it is available for Explorers to search and discover. We placed particular emphasis on search and discovery. Explorers can search for recipes by a combination of keywords in the title, method, and selection of ingredients and meal type. This is very powerful as it allows Explorers to hone down on relevant recipes based on multiple criteria.

That is not all, one of our novel functionalities is it to provide price estimates of recipes based on its ingredients; we have provided functionality to search by an estimated price, as well as adjust price by ingredients that the Explorer already have. This means that Explorers can find recipes within their budget and also minimising wastage of ingredients when finding recipes.

We also introduce a high level of social functionality, where users can comment and like recipes, allowing Contributors to easily obtain feedback and further refine their recipes. Users can subscribe to Contributors they like to follow their new recipe releases they like, while contributors can grow their subscriber base which encourages them to make more contributions.

Lastly, our website has a second novel functionality, being tipping of Contributors. For a small fee charged to their credit card, Explorers can support Contributors and send them a virtual award. The Contributor will have a glyph displayed under their profile as a reward for their contribution to the community, and the Explorer will also be recognised as a supporter. This feature further enhances social engagement and allows Contributors to be rewarded for their effort.

## Architectural Overview



## Overview

The overall architecture can be succinctly summarized as a client single page application front end, a service layer consisting of a thin backend application, and a data layer consisting of a relational database and file storage. The web application and static assets are served to the client through either a Django development grade or Apache web server.

## Frontend

The frontend is a React application written in JavaScript, HTML and CSS.

Upon the user first accessing of the website, all HTML, JavaScript, and CSS assets will be served to the user and loaded in the initial HTTP request from the web server.

The React application consists of a main component which handles global features such as presenting the header/footer content and storing the authentication state.

The feature components can be accessed by different address paths, which are managed by the routing library which map requests to various paths to various components. The various feature components provide specific features to the web application.

Subsequent interactions with the application will not result in a page load, rather the client application will use preloaded assets and make asynchronous requests to various APIs in the backend.

## Backend

The backend is a Python application written using the Django framework. The backend application provides various web services to be consumed by the front end.

The Django framework uses a Model-View-Template architecture which is similar to the more common Model-View-Controller architecture (Django Software Foundation, 2022); the Model provides class objects, the View is similar to the Controller in MVC is the glue between the Model and Templates, and finally, the View renders the Templates which describe the format of the data responses. For APIs, we use Serializers, which are architecturally similar to Templates, but are more specialised for defining JSON request/response content rather than general HTML pages.

The Model is mapped to a PostgreSQL database through the Django Object Relational Mapper. Changes to the model are managed by Django Migrations, which update the corresponding database model to match updates to the Django model. Using the ORM, model objects can be automatically persisted in the database without writing any SQL queries, we only have to deal with the model object.

Storage of multimedia and binary data is in the file system rather than the relational database. This refers to user generated data like recipe pictures and profile pictures. In these instances, we have setup the ORM to store the filename in the database, but the file in the file system itself. This is because while databases can store binary data, it does not offer good performance (PostgreSQL, 2022).

## Backend/Front End Integration

Once the front end is loaded into the client, the front end will communicate with the back end for various operations to retrieve or send data to the server. Typically, the user initiates some action in the client, for example, clicking a link to view a recipe; the client does not yet contain all the data, so it must send a GET request to the backend API, and the API will respond with the relevant data to display the recipe. Not only that, the backend manages the system record for the whole website, so if the user wants to make some stateful change to the global website state, it front end sends a POST or PUT request, in conjunction with JSON or FormData, and the backend processes the request according to the business logic in the view which can result in success (and the website state is updated) or error.

## Choice of Architecture

We have chosen system components that have some relatively established and suited to our purpose. We also narrowed down our choice based on familiarity within our group, the importance of which cannot be understated compared to technical reasons for decisions, given the short time frame to deliver this project.

React – There are various JavaScript front end frameworks. Compared to native JavaScript, the React framework enables the application to be made up of Components which bind HTML against JavaScript variables, which improves code organisation and makes the application code more concise.

In comparison with other frameworks, React is currently the most popular front-end JavaScript framework (Stack Overflow, 2022), which suggestions there is a great supporting ecosystem.

Django – The benefit of Django is that out of the box, CRUD web applications can be built relatively quickly. There is in-built ORM, which means the developers only need to define the models, and it will automatically manage persistence of the models in the database. Using scaffolding tools, various APIs can be autogenerated for the models with little effort and will only need minor customisation. It is also the most popular Python framework (Stack Overflow, 2022), which means there is a great supporting community.

There are disadvantages with Django, the main one is probably the same one that makes Django suitable to rapid development, is that a lot comes out of the box, it is more rigid and monolithic, there is less flexibility to deviate from some of the technical choices it has made. However, our application requirement fits within the realm of common web applications, we do not see a need to deviate from it.

PostgreSQL – We have chosen this database as it is a common relational SQL database system. Overall, we are limited to one of the database systems Django support, PostgreSQL is one of the 5 that it supports (Django Software Foundation, 2022).

Relational database systems are less suited towards storing of large files. For this, we are using the file system instead. We rule out SQLite as the document suggests it is for applications that are “predominately read-only” (Django Software Foundation, 2022). We also rule out Oracle as it requires a commercial license). Finally, the documentation further states that “PostgreSQL is the most capable of all the databases… in terms of schema support” (Django Software Foundation, 2022). To minimise friction, we followed this recommendation in absence of other reasons.

Web Server – For development and demonstrative purposes the built in Django web server can easily launch a web server with minimal installation and configuration which makes it suitable for anyone to load the application.

However, for production purposes, the application can be deployed on Apache, being a production grade web server, it has more features such as security, scalability, and networking in mind.

# Description of Functionalities

**Functionality 1: “Contributors must be able to create and maintain a profile for themselves, which includes their username, contact details (email address), and a list of recipes that this contributor has published.”**

Users are able to create an account, for the purpose of our website their username is their email address in which they can login with. We also require them to validate their email address by clicking on a secret link sent to them at registration time. Users can login to their account which provides extra functionality. Once logged in they can update their details such as their password.

Logged in users can become a Contributor by submitting extra detail, being a Display Name, Profile Description, and Profile Photo. This information can also be updated within the Profile Page.

The Profile Page also shows a list of recipes the contributors has already contributed, with links to edit or delete those recipes.

**Functionality 2: “The system/platform must allow contributors to maintain a set of their own recipes that explorers can look through, with each recipe requiring a name, ingredients, method, meal-type(s) (breakfast, lunch, dinner, …), and a photo.”**

Logged in Contributors can click the Create Recipe button at any time, they populate a form including name, ingredients, method, meal-type(s) (breakfast, lunch, dinner, …), and a photo.

For each ingredient they also need to specify the quantity. Once they submit the form the recipe is published for all too see.

As described earlier, from the Profile page, logged in Contributors can see a list of existing recipes, including links to edit or delete.

Clicking the edit button takes the Contributor to the same form as the recipe creation form, which the existing recipes details prepopulated; the Contributor can’t edit the form fields at their discretion and submit the form, which will cause it to be updated immediately for all users to see.

Click the delete button deletes the recipe, and all data associated, including social interactions, are permanently removed.

**Functionality 3: “Explorers must also be able to convey their opinion for any recipe by either liking and/or commenting on it.”**

Logged in users, when viewing a recipe can click on a ‘Like’ button. This has the effect of incrementing a counter showing the number of users that have liked the same recipe, which gives an indication of recipe popularity. The user that has previously liked a recipe can also click on a ‘Unlike’ button within a recipe to revert it. Within the user’s profile page, they can also see a list of liked recipes, allowing them to access them again easily in the future.

Logged in users, when viewing a recipe, can also comment on the recipes on the page viewing the full recipe and give a review about that recipe which will help others to get a clear idea on how that recipe is.

This is done by entering their comment in a textbox and clicking the submission button.

The comment will be available for all other users to see (whether logged in or not) on that same recipe page. The comment includes the date of the comment as well as the display name of the commentor, if the user has a display name.

**Functionality 5: “The system/platform must also provide explorers with the ability to find recipes they are interested in based on any combination of the following: ingredients, method, meal-type(s), and recipe name. The resulting list should show a summary that includes the name and a photo thumbnail for each recipe.”**

On the home page, on the left hand side of the screen, whether the user is logged in or not, there is a form allowing users to apply these filters. Users populate one or more of these form fields and click on the search button. Fields which are populated will be used for filtering results, which are then displayed on the right hand side of the screen including the name and photo of the screen.

Details of the form fields are as follows:

* Text box for Recipe Title – which cause only recipes with the same text included in the Recipe Title to be included.
* Text box for Recipe Method – which cause only recipes with the same text included in the Recipe Method to be included.
* Checkboxes for Meal Type – which cause only recipes with one of the selected meal types of the displayed. If no meal types are selected, all meal types would be displayed.
* Multibox for Ingredients – which allow selection of multiple ingredients, which cause only recipes containing all of the ingredients to be applied.
* Estimated Price – described under Novel Functionality 2.

**Functionality 6: “Explorers must be able to navigate to the recipe details by clicking on the thumbnail. Details include: the recipe name, ingredients, method, meal type(s), a photo for the recipe, the number of likes for the recipe, and recipe comments.”**

All users, whether logged in or not, can click on a recipe thumbnail form the search screen. They will be taken to a page with a unique URL (which can be accessed directly from the browser). The page shows all the required details.

**Functionality 7: “Explorers must also be able to subscribe/unsubscribe to any contributors on the platform/system. When an explorer is subscribed to a given contributor, they should be able to see, in their personal recipe news feed, new recipes that are added or have been recently updated by that contributor.”**

Logged-in users, when viewing a recipe, they can see the Contributor that published the recipe on the right hand side of the screen. There is a button for users to Subscribe to that contributor (if they are not already Subscribed; if they are Subscribed that button will be to Unsubscribe). If the user clicks this button, they will be considered subscribed to this contributor. When the user visits their home page, if the user has at least one subscribed contributor, it will display clickable recipes and their thumbnails to most recent recipes published by the subscribed contributors.

**Functionality 8: “The system/platform must provide recommendations for recipes that are similar to a given reference recipe based on that reference recipe's ingredients, with recommendations to be ordered based on how close each result is to the reference recipe's ingredients. Design a metric to represent closeness between two sets of ingredients, and sort results from closest match to reference recipe’s ingredients to furthest match. Exclude any recipes from results that have no ingredients in common to the reference recipe.”**

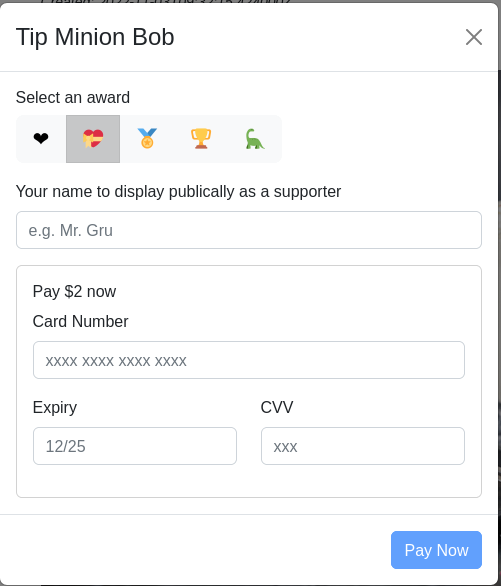
All users, whether logged in or not, when viewing a recipe, will see a number of recipes related to the current recipe that they are viewing. Those recipes are based on closeness of two sets of ingredients, with closest recipes displayed first. If the recipe has no ingredients in common with any other recipes, then no recommendations will be provided. Clicking on those recipes will take you to that recipe’s page. The closeness measure is described in the Technical Challenges section below.

**Novel Functionality 1: Support contributors by tipping**

As per our research, there are already many websites that offer recipes for free that most people would not expect to be required to pay for them. Like many other information resources, it cannot be dismissed that recipes take time and effort for contributors to write up and prepare. Recipes are not created without investment in ingredients and time to test and experiment. Thus, we want to provide a feature that allow Explorers to support contributors for their contributions, by allowing Explorers to make micropayments towards the contributors, it will give Contributors ‘kudos’ and encourage them to continue to develop and share better recipes. Explorers that tip will be recognised on the Contributor’s public profile page to provide a social reward towards tipping.

Rewarding of content creators is not a new concept. For example, Twitter has a Tip Jar feature (Twitter, 2022) that lets you tip people whose tweets you like; YouTube has a Super Thanks (Google, 2022) program allowing viewers to tip creators they like. However, the application of tipping to recipes is novel, our internet search of this has not found any result, and we believed this is an important niche to fill.

We have implemented this by allowing users to provide a virtual award.



Users can select an award, and upon entering of credit card details, the Contributor will receive a virtual award which is permanently displayed under the Contributor’s details on their published recipe pages.

**Novel Functionality 2: Recipe price estimate**

There are many factors in which people choose to cook a particular dish. While difficulty and culinary preferences are obvious reasons, across the world, people also consider the cost of ingredients. In the United States, the lowest income quintile of households spent 27% of their income on food in 2020 (U.S. Department of Agriculture, 2022). So, there is demand for users to be able to find budget conscious recipes and reduce their household expenditure.

Some recipe websites have a ‘budget conscious’ category or similar. However, these inevitably require the contributor to tag the recipe as such. Our novel functionality will estimate the price of recipes based on data. This means the recipe price estimate will be much more accurate and have a greater level of granularity. The price estimate can also update based on what’s already in the fridge.

From a technical perspective, this will require sourcing of data and application of this data to perform the estimate.

As prices of ingredients can vary in different localities, in our current release we only provide prices estimates for one geographic locality, with the ability to extend it to different localities.

The price estimates are displayed on each recipe’s page. There is a breakdown of each ingredient’s estimated price, the sum of which forms the total estimated price which is also displayed.

On the search filters, users are also able to filter based on the recipe’s total estimated price. Not only that, they can adjust the price, by selecting ingredients they already own, which helps users minimise food wastage as they can make use of those ingredients with minimal marginal cost.

The method we used to estimate prices is described in the Technical Challenges section below.

# Technical Challenges

## Recommendation of Similar Recipes

Based on the project specification, we were required to provide recommendations of recipes based on a measure of similarity based on the recipe’s ingredients.

A naïve measure for this would be simply to take the number of common ingredients of two recipes, the higher the count, the more similar they are.

However, we saw two limitations with this approach. Firstly, some recipes have a high number of ingredients, then their count will naturally be more likely to be higher. Secondly, some recipes have ingredients, for example, ‘salt’ or ‘sugar’ that are very common across the whole collection of recipes, commonality of these ingredients should take lower rate.

We solve this problem firstly be representing the recipe as a vector of ingredients not equal value but higher value based on measure used in search and retrieval known as *tf-idf* (term frequency – inverse document frequency). Term frequency refers to the frequency of a term (ingredient) in a document (recipe), inverse document frequency refers to whether a term (ingredient) is common or are across all documents (recipes) or not. For simplicity, we assume each ingredient can only appear once in a recipe.

Now let be an ingredient within a recipe, be a recipe in our collection of recipes. We then the define frequency of ingredient within its recipe as follows:

As you can see, the more ingredients in the recipe, the lower the weight of that term.

Next, we define the inverse document frequency of the ingredient in the collection as the log of number of recipes over number of recipes containing that ingredient:

In this formula, the higher the frequency of an ingredient across all recipes, the lower the number.

Finally, we define as to consider frequency of ingredients with a document with frequency in across recipes together. For any given recipe of with set of ingredients , we then represent it with a vector:

To compare the similarity of two recipes and , we take the cosine similarity of two vectors which measure their difference in direction (and normalising the magnitude):

### Worked Example

To illustrate this example, suppose we have three recipes in our collection, we want to know recipes similar to . Note that all other recipes have one other ingredient in common.

The IDF of terms are as follows:

|  |  |  |
| --- | --- | --- |
| Position | Frequency | idf |
| 1 | salt | 0.125 |
| 2 | potato | 0.301 |
| 3 | pepper | 0.602 |
| 4 | beef | 0.602 |
| 5 | chicken | 0.602 |

Then for all ingredients tf = 0.5 as there are 2 ingredients:

Finally,

As you can see is higher than and despite all have the same number of common ingredients as .

## Price Estimate

The challenge of estimating a recipe price is not simple, and we outline some of the considerations and problems we had to solve in order to come up with a recipe price.

Firstly, we consider what the drivers to a recipe price is; a simple view is that it is simply the sum of the price its ingredients. However, reality is more complex. When you purchase ingredients for a recipe, the package size may be different from the ingredient’s requirement, there are common ingredients you may already have where the marginal cost is minimal, the amount of the ingredient used is non-linear to the purchasing quantity and the amount actually used, and the actual amount that is cooked compared to the recipe.

The second problem we had was, how to get data for the price estimate. The obvious solution was to rely on some API or web scraping to collect the unit price of ingredients and apply based on the units used in the ingredient.

However, we decided to take a more novel approach. One reason for this approach is that one key aim of our website was to facilitate social interaction and free flowing of information, we decided to take a crowdsourcing approach to solve this problem. By collecting information from lived human experience, we can get genuine feedback of the cost of the recipe, which is superior to any modelled calculation.

For any given recipe, explorers are invited to submit a price for the ingredient in a recipe (if the explorer submits two prices for the same prices for the same ingredient, the latest one is taken).

Then for any recipe, the price of an ingredient is calculated as follows:

1. If there is at least one price submission of an ingredient for that recipe, then the average price submission of the ingredient for that recipe is the estimate.
2. If there are no price submissions of an ingredient for that recipe, then the average price submission of the same ingredient across all recipes is the estimate.
3. Finally, if the above does not apply, the average ingredient price submission of all ingredients is taken as the price estimate for the ingredient.

Finally, for a recipe we estimate the price of that recipe to be the total of estimated prices of ingredients.

We claim that where an ingredient falls under the first criteria, given sufficient submissions, a high-quality prediction is provided; the justification is that various users price submissions encompass realities of the sourcing and using of those ingredients in the context of the recipe, and the average of those submissions will provide a high accuracy reflection of that reflection.

We recognise some limitations of this approach, in particular, if there are no or limited submissions, we fall back to averaging across all recipes or all ingredients, which will be less accurate. Given more time, we would consider whether the crowdsourcing approach could be combined with a more mathematical estimate and perhaps also consume other data sources. We would also place more consideration into the localisation of the price estimate.

## Ingredient Selection

One problem we try to solve was how to maintain consistency in the ingredients within our recipe database.

In particular, if the same ingredients were entered slightly differently into the database, it will be hard to identify them as the same ingredient, it will affect the ingredient-based search, as well as the ingredient price estimate.

To solve this, we firstly use a curated source of ingredients. We have sourced our initial set of ingredients from FooDb. This project is a Canadian nationally funded project to provide a database of food constituents (FooDB, 2022).

When ingredients are entered, whether during creation of a recipe, or during searching of ingredients, we autosuggest ingredients from the ingredient list, so that users can select from the established list of ingredients. This way, it ensures consistency of ingredients. Of course, not every ingredient might be in the list, so the user can always enter a new ingredient, and if that ingredient is frequently entered, the ingredient list can be curated to include that ingredient.

Overall, this method improves consistency of ingredient names while giving people flexibility to enter other ingredients.

## Loading Screen

One issue we encountered was that during the waiting of fetching data of the backend, it could take a few seconds, particular when the backend is doing something that takes longer such as sending an email or processing a file upload, the front end needs to reflect that it is awaiting this response, otherwise the user may get confused about whether the request was submitted or even duplicate the request submission.

Therefore, we need to provide some user feedback about when we are awaiting the backend response and also prevent duplicate requests. We also wanted to implement this globally without having to specifically implement bespoke code for each specific request in our application.

To do this globally, we define an interceptor function; the interceptor function hooks into any request made through the axios library (which we use for asynchronous requests). Every time a request is made, we increment a counter, and the counter is decremented on resolution of that request (whether it was successful or not).

Whenever the counter is greater than 0, it means there is an outstanding request, and we place a overlay over the whole application, which prevents further user action until all requests are resolved.

Using this technique, we were able to improve the overall user experience.

## State Management

One common challenge within the React framework is how state is passed between components, from the state owner to child or grandchildren back to the state owner.

There are state management libraries such as Redux which can assist with this. However, in our case, we only had one main state object to maintain through the application, being the user state, the use of Redux did not fall into the recommended use cases (Redux, 2022).

In the end, we settled with using the context provider provided by the React Router which we use; we can easily pass down a *state* object and corresponding *setState* function to children, which the children can access using the useOutletContext function (Remix Software, Inc., 2022). The children can also update the state of the parent, by using the *setState* function that is passed down.

# Third Party Functionality

|  |  |  |
| --- | --- | --- |
| Third Party Functionality | Purpose | License |
| Django | Django is the key backend Python framework that we use. The framework provides out-of-the box routing & ORM, and provides some structure (Model-View-Template) to our application.  Using this framework allowed to focus on writing business logic with established patterns that support rapid development and overall good system design, rather than reinventing the wheel. | 3-clause BSD |
| Djoser | Djsoer is an REST authentication package that works with Django. This package provides functionality to create and manage accounts as well as manage authentication.  It is good practice to use established authentication libraries as there are many security considerations, that creating an authentication library is considered a specialised task which should only be done if that is the task at hand. | MIT License |
| Django REST Framework | Django REST Framework is a package that works with Django to provide REST APIs. Out of the box, Django was mainly designed for HTML responses, and while it can also produce JSON responses, the Django REST Frameworks provides functionality to do this in a more organised and convenient way. The key feature is Serializers, which allow declarative definition of mapping between JSON and Django models. | 3-clause BSD |
| React | React is the key front end library that we use. React provides two key functionality: firstly, it allows us to break down different parts of a front end application as components which gives our application better structure; secondly, it provides syntactic sugar to place JavaScript variables within HTML elements, which automatically re-render upon changes, allowing us to program the components more declaratively. | MIT License |
| React-Bootstrap | Bootstrap is a CSS framework which provides a simple grid layout as well as sensible styling defaults and utilities which allow us to develop a reasonable looking frontend design quickly. The React-Bootstrap is a React Component library based off Bootstrap written as pure React components meaning it does not depend and won’t conflict with JQuery. | MIT License |
| React-Router | React-router provides client side routing, meaning that different pages of the application will be bound to different states and/or components of the front end application. It also means that users can navigate to different addresses within the application, without requiring a server side request. | MIT License |
| React-Router-Bootstrap | This library provides a simple way to declare links in React-Bootstrap styling which are compatible with React-Router. | Apache License 2.0 |
| Axios | Axios is a library that is used to perform HTTP requests asynchronously using a promise-based syntax. While there is native fetch functionality, axios provides benefits such as backward compatibility, XSRF protection, automatic transformation of JSON to objects and vice versa. | MIT License |
| Card-Validator | This library determines the validity of credit card details entered. We use this in the tipping functionality to check whether the entered details are valid or not. | MIT License |

## Effect of Licensing

The packages we use are licensed under one of 3-clause BSD, MIT License, and Apache License 2.0. The licenses largely have the same intent and have the following in common:

* They are permissive license which has few restrictions on use of the library.
* If the code or parts of the code under this license is modified, the full text of the license must be included in the distribution.
* The above does extend to the application that links to it, the license only applies to the library code.

Overall, there is limited risk to our application to using these packages, they can be used in a commercial context, and they do not cause our application to have a requirement to become open source. Only changes to the libraries itself must become open source if they are distributed, however, we have not made any modifications to any of the libraries used. Once a package has been downloaded, it is licensed to us in perpetuity under that license at least for that version of that package, there is no risk of the license being retracted to stop our application from operating.

# User Documentation

## Server Setup and Installation (Estimated time: 20 minutes)

1. **VMWare Workstation 16 Player** and the provided Lubuntu image should already be setup. If not, please find the link below:

<https://webcms3.cse.unsw.edu.au/COMP9900/22T3/resources/80157>

2. Start the virtual machine andlogin using the Virtual Machine Credentials:

Username: **lubuntu**   
Password: **lubuntu**

Using Firefox browser, download the project.zip file from WebCMS or Email and save it in /home/lubuntu.

3. Start a new terminal (Start Menu, System Tools, QTerminal).

Unzip the file

|  |
| --- |
| unzip project.zip |

4. Install PostgreSQL

Become root.

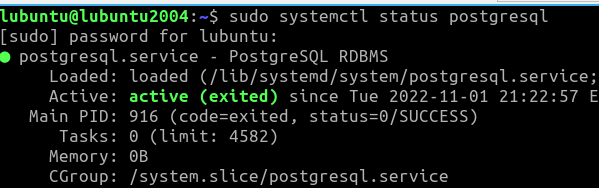
|  |
| --- |
| sudo su |

Enter password **lubuntu**

|  |
| --- |
| sudo apt update  sudo apt install postgresql |

5. Use this command to check your postgresql is active:

|  |
| --- |
| sudo systemctl status postgresql  sudo pg\_isready |



6. Now create database in postgresql:

|  |
| --- |
| sudo su – postgres |

The terminal will turn to:

|  |
| --- |
| postgres@lubuntu......:~$ |

Now type:

|  |
| --- |
| psql |

the terminal will turn to:

|  |
| --- |
| postgres=# |

Now input the commands:

|  |
| --- |
| CREATE USER postgres\_user WITH PASSWORD 'postgres';  CREATE DATABASE foodies\_database;  GRANT ALL PRIVILEGES ON DATABASE foodies\_database to postgres\_user;  \q  exit |

 7. Go to the folder foodies\_backend/ using the cd command

|  |
| --- |
| cd project/foodies\_backend/  sudo apt install python3-pip  pip install django djoser djangorestframework psycopg2-binary  python3 manage.py migrate  python3 manage.py loaddata seed.json  python3 manage.py runserver |

Now backend has setup and is running

8. In a new terminal window, install Node.js (do not use apt-get as it is out of date).

Download *node-v16.18.1-linux-x64.tar.xz* and install it.

|  |
| --- |
| wget <https://nodejs.org/dist/v16.18.1/node-v16.18.1-linux-x64.tar.xz>  sudo mkdir -p /usr/local/lib/nodejs  sudo tar -xJvf node-v16.18.1-linux-x64.tar.xz -C /usr/local/lib/nodejs |

9. Confirm Installation

|  |
| --- |
| export PATH=/usr/local/lib/nodejs/node-v16.18.1-linux-x64/bin:$PATH  . ~/.profile  node -v  npm version |

10. Install yarn:

|  |
| --- |
| sudo su  export PATH=/usr/local/lib/nodejs/node-v16.18.1-linux-x64/bin:$PATH  . ~/.profile  corepack enable |

Test installation using

|  |
| --- |
| yarn -v |

Go to the foodies\_frontend/ directory and run this command to install required packages

|  |
| --- |
| cd project/foodies\_frontend  yarn install |

Now frontend has setup

Then run this command

|  |
| --- |
| yarn start |

This will start a development web server.

Once you see the message ‘webpack compiled…’, you can open Firefox browser and navigate to <http://localhost:3000>.

The database has been seeded with recipes and 3 users. The credentials of those users are below.

|  |  |  |
| --- | --- | --- |
| Email | Password | Type |
| [a@a.com](mailto:a@a.com) | P@ssword1 | Contributor |
| [b@b.com](mailto:b@b.com) | P@ssword1 | Contributor |
| [c@c.com](mailto:c@c.com) | P@ssword1 | Explorer |

## Application Instructions

### Home Page

1. When you first access the website, you will see latest recipes.
2. If you are logged in, you will also see subscriber’s latest recipes.
3. When you reached the end of the page, you can show more recipes by clicking ‘Show More’.
4. Click on any recipe takes you to view the recipe.

### Search & View Recipes

Prerequisites: None.

1. Click on the Home link at the top menu bar
2. Apply one or more of the following search filters:
   1. Enter keywords in the title textbox to filter on the recipe title
   2. Enter keywords in the method textbox to filter on the recipe method text.
   3. Select ingredients in the ingredient multibox to filter on the recipe ingredients. All selected ingredients must be in the recipe to be returned.
   4. Select one or more meal type. Recipes in any of the selected meal type will be returned.
   5. Select an estimated price. The estimated price can be reduced by the estimated value of ingredients you select in the multibox.
3. Click search.
4. The matching recipes will be displayed, if you scroll to the end you can click ‘Show More’ to display more matching recipes.
5. Click on relevant recipes to view it.

### Logging In

Prerequisites: you already have an account. If you do not have an account, follow the instructions to Creation of Explorer Account.

1. Click Login Link at the top menu bar.
2. Populate Email, Password.
3. Click the Login Button.

### Logging Out

Prerequisites: you are already logged in

1. Click the Logout Link at the top menu bar.

### Creation of Explorer Account

Prerequisites: not logged in. If you are already logged in, follow the instructions to Logging Out.

1. Click Login.
2. Click the Register button.
3. Populate Email, Password, and Password Confirmation.

### Upgrade of Account to Contributor Account

Prerequisites: you are logged in.

1. Click on the Profile link at the top menu bar.
2. Populate Display Name, Profile Description and select a photo.
3. Click Update.

### Change Password

Prerequisites: you are logged in.

1. Click on the Profile link at the top menu bar.
2. Click Change Password.
3. Enter Email Address and click on Reset Password Request.
4. Click on the link that is sent to your email address.
5. Enter a new password and click on the submission button.

### Forgot Password

Prerequisites: you are not logged in.

1. Click on the Login link at the top menu bar.
2. Click Forgot Password.
3. Enter Email Address and click on Reset Password Request.
4. Click on the link that is sent to your email address.
5. Enter a new password and click on the submission button.

### Update Contributor Details

Prerequisites: you are logged in and you are a contributor.

1. Click on the Profile link at the top menu bar
2. Click on the Update Profile tab.
3. Edit Display Name, Profile Description and select a photo if desired.
4. Click Update

### Create a Recipe

Prerequisites: you are logged in and you are a contributor.

1. Click on the Profile link at the top menu bar.
2. Click ‘Create New Recipe’
3. Populate all fields, in particular:
   1. Select a photo by clicking the Browse button
   2. To add ingredients, in the first box, select an ingredient, enter the quantity of that ingredient in the adjacent textbox, and click Add. Multiple ingredients can be added this way.
4. Once completed, click Submit

### Edit a Recipe

Prerequisites: you are logged in and you are a contributor.

1. Click on the Profile link at the top menu bar.
2. The recipes you have previously created are displayed under the My Recipes tab, click the Edit link on the recipe you want to edit.
3. Edit any fields, the form is the same as the one for recipe creation.
4. Once completed, click Submit

### Delete a Recipe

Prerequisites: you are logged in and you are a contributor.

1. Click on the Profile link at the top menu bar.
2. The recipes you have previously created are displayed under the My Recipes tab, click the Delete link to delete that recipe.

### Like and unlike a Recipe

Prerequisites: you are logged in and you are currently viewing a recipe.

1. If you have not already liked the recipe, you can click a button on the top right to Like the recipe.
2. If you have already liked the recipe, you can click a button on the top right to Unlike the recipe.
3. The effect of Liking the recipe is that when you click on the Profile link at top menu bar then click on the Liked Recipes tabbed, all recipes that you have liked will be present there.
4. Liking a recipe also increments the counter showing how many people have liked a recipe, which indicates to others how popular a recipe is.

### Commenting a Recipe

Prerequisites: you are logged in and you are currently viewing a recipe.

1. At the bottom of the page, you can enter a comment, and click the button to submit the comment.
2. The comment will immediately be available for all other users that visit the same recipe page to see.

### Support a Contributor

Prerequisites: you are logged in and you are currently viewing a recipe.

1. On the left hand side of the page, you can click Tip.
2. Select one of the 5 glyphs which will be an award displayed on the contributor’s profile.
3. Enter your name to be recognised publically as a supporter. It does not have to be your real name.
4. Enter your credit card details and click Pay Now.
5. If your card is a valid number (e.g. 4242 4242 4242 4242) and it is successfully processed, your chose glyph with the supporter’s name will be added to the bottom of the contributor’s details when viewing recipes for all to see.

### Subscribe to a Contributor

Prerequisites: you are logged in and you are currently viewing a recipe.

1. On the left hand side of the page, you can click on the Subscribe button.
2. You can unsubscribe from a contributor by clicking the same button again.
3. While you are subscribed to a contributor, on the home page, the latest recipes from your subscribed contributors will be displayed.

### Similar Recipe Recommendation

Prerequisites: you are currently viewing a recipe.

1. At the bottom of the page, there are a number of recipes that will be recommended to you, that are similar to the current recipe you are viewing (based on the ingredients within the recipe).
2. You can click on the recipe which will take you to that recipe.

### Submit Recipe Price Estimate

Prerequisites: you are logged in and you are currently viewing a recipe.

1. On the same row of each ingredient there is a textbox, you can enter the price of the ingredient for the recipe, based on your experience.
2. Your contribution will be aggregated with other user’s submissions and contribute to the estimated price of the ingredient and recipe.

# References

CookingBites Cooking Forum, 2022. *CookingBites Cooking Forum.* [Online]   
Available at: https://www.cookingbites.com/  
[Accessed 03 October 2022].

Discuss Cooking - Cooking Forums, 2022. *Discuss Cooking - Cooking Forums.* [Online]   
Available at: https://www.discusscooking.com/  
[Accessed 03 October 2022].

Django Software Foundation, 2022. *Databases.* [Online]   
Available at: https://docs.djangoproject.com/en/4.1/ref/databases/  
[Accessed 03 October 2022].

Django Software Foundation, 2022. *FAQ: General.* [Online]   
Available at: https://docs.djangoproject.com/en/4.1/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-how-come-you-don-t-use-the-standard-names  
[Accessed 03 October 2022].

Django Software Foundation, 2022. *Migrations.* [Online]   
Available at: https://docs.djangoproject.com/en/dev/topics/migrations/#backend-support  
[Accessed 03 October 2022].

FooDB, 2022. *About.* [Online]   
Available at: https://foodb.ca/about  
[Accessed 12 November 2022].

Google, 2022. *Turn on and manage Super Thanks.* [Online]   
Available at: https://support.google.com/youtube/answer/10878910?hl=en  
[Accessed 03 October 2022].

PostgreSQL, 2022. *Binary Files in DB.* [Online]   
Available at: https://wiki.postgresql.org/wiki/BinaryFilesInDB#Storing\_Meta\_data\_and\_symbolic\_link\_in\_the\_database\_to\_where\_a\_Binary\_file\_is\_located  
[Accessed 12 November 2022].

Redux, 2022. *When should I use Redux.* [Online]   
Available at: https://redux.js.org/faq/general#when-should-i-use-redux  
[Accessed 12 November 2022].

Remix Software, Inc., 2022. *React Router - useOutletContext.* [Online]   
Available at: https://reactrouter.com/en/main/hooks/use-outlet-context#useoutletcontext  
[Accessed 12 November 2022].

Sally's Baking Addiction, 2022. *Trusted Recipes from a Self-Taught Baker.* [Online]   
Available at: https://sallysbakingaddiction.com/  
[Accessed 03 October 2022].

Stack Overflow, 2022. *Stack Overflow.* [Online]   
Available at: https://survey.stackoverflow.co/2022/#most-popular-technologies-language  
[Accessed 03 October 2022].

Twitter, 2022. *About tips and tipping.* [Online]   
Available at: https://help.twitter.com/en/using-twitter/tips  
[Accessed 03 October 2022].

U.S. Department of Agriculture, 2022. *Food spending as a share of income declines as income rises.* [Online]   
Available at: https://www.ers.usda.gov/data-products/chart-gallery/gallery/chart-detail/?chartId=58372  
[Accessed 03 October 2022].