



**UNSW**  
SYDNEY

# Project Report - SocialSavour

COMP3900 Computer Science Project

3900F18AMeanCodingMachine

Submission Date: 4 August, 2023

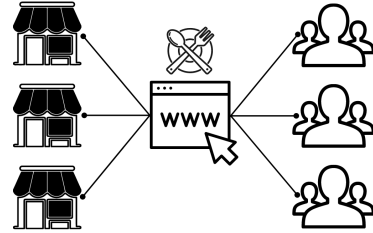
Name	zID	Email	Role
Ahmed Al Abassy	z5312664	z5312664@ad.unsw.edu.au	Backend Development
Hannah Van Roy	z5437861	z5437861@ad.unsw.edu.au	Frontend Development
Cameron White	z5320168	z5320168@ad.unsw.edu.au	Frontend Development
Allan Sugianto	z5289607	a.sugianto@student.unsw.edu.au	Backend Development
Jiseop Lim	z5210599	z5210599@ad.unsw.edu.au	Backend Development

# Table of Contents:

<b>Overview:</b>	<b>3</b>
<b>Functionalities and project objectives:</b>	<b>6</b>
<b>Third-party functionalities:</b>	<b>16</b>
<b>Implementation challenges:</b>	<b>18</b>
<b>Installation Manual:</b>	<b>19</b>
<b>Reference list:</b>	<b>21</b>

# Overview:

SocialSavour is an application that aims to connect eateries with their customers by blending the themes found in conventional eatery applications such as Uber Eats with those found in social applications such as Instagram. Such a platform helps restaurants socialize and market their business to their consumers using a medium that purely serves



their interests as active participants in the dining industry. Additionally, the platform also benefits consumers by equipping them with the ability to find eateries tailored to their needs while giving them the opportunity to save money through vouchers. The final application divides its logic into two parts, the frontend and the backend. In the backend, Social Savour employs Express.js to create and route its server to a specific port. Additionally, the backend is also where the application generates a pooled connection to the MySQL database. This connection is incorporated by a variety of service functions that directly access the database. The controller functions used in each express route utilize these functions to receive and process the requested data to answer a request made from the frontend. The frontend has been made using the react framework and the MUI react component library to generate a simple and easy-to-understand UI that can be used to explore the application's features. Using the Axios module to make requests to the backend, the frontend can communicate with the server and the database to obtain important information while the user interacts with the application's interface.

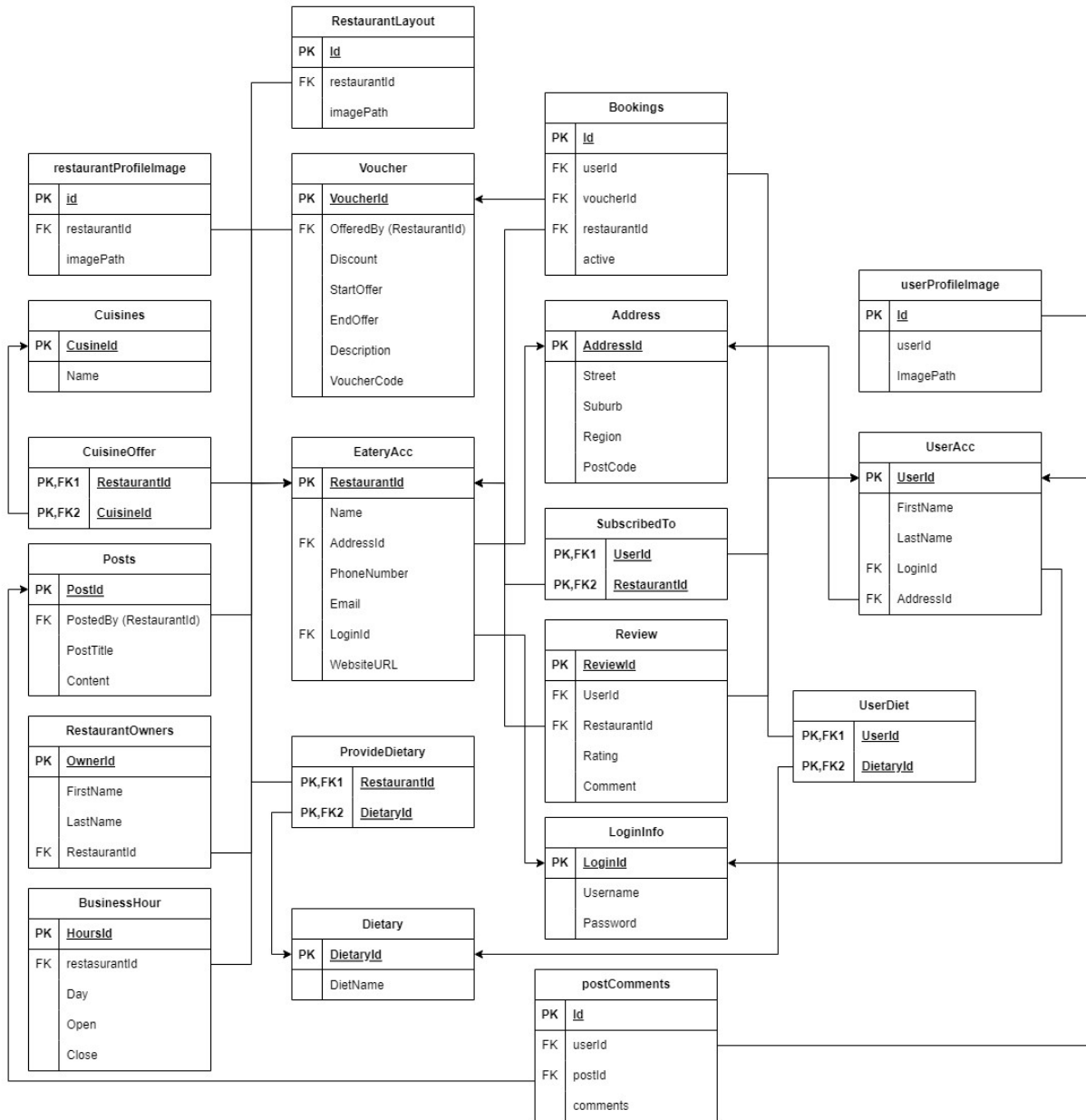
# Database Architecture

The Figure below is the database ER diagram used to store the data for our eatery management application. The diagram is similar to what is proposed in the previous proposal report with some modification which includes:

- Tables storing the profile image path of the user and the restaurant (restaurantProfileImage, userProfileImage) in the backend as well as restaurant layout where the user can see and book a table (restaurantLayout).
- Active column in Bookings table which indicates whether the voucher has been verified and redeemed.
- Likes column in the Posts table which counts on how many different users liked the post made by the eatery.
- Added Table where user comments on the post posted by the eateries are stored (postComments).

Link for the image:

[https://drive.google.com/drive/folders/1WpcdDANOlvLrC387IIUOvr9QQDpGVH-t?usp=drive\\_link](https://drive.google.com/drive/folders/1WpcdDANOlvLrC387IIUOvr9QQDpGVH-t?usp=drive_link)



# Functionalities and project objectives:

The following is a list of the core objectives created during the project proposal:

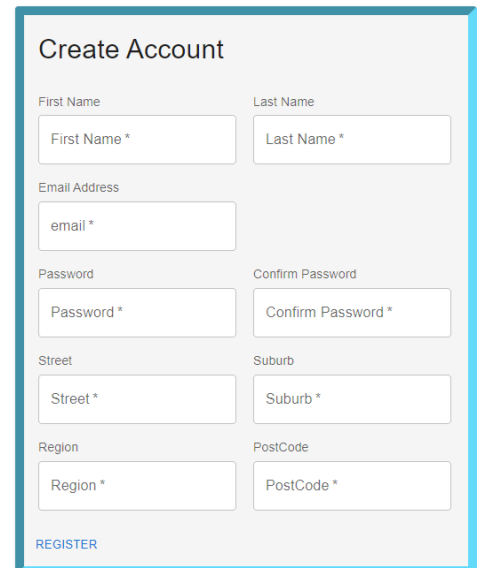
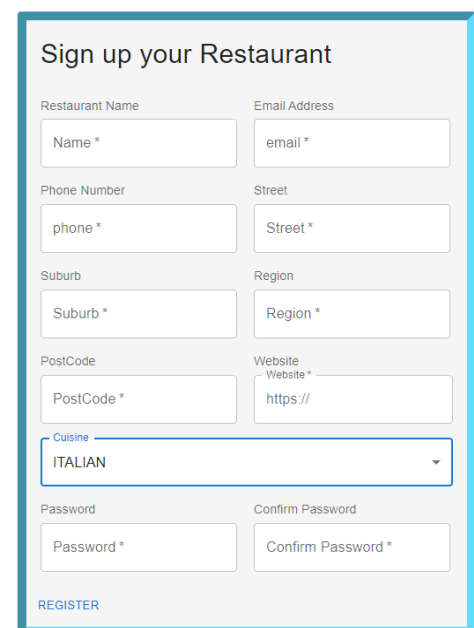
- **Objective 1:** To make sure that the users are updated with new events that bring benefits to them, and construct a secure link between eateries and customers
- **Objective 2:** To ensure that certain users are not excluded from the benefit of finding great eateries due to personal preferences or circumstances.
- **Objective 3:** To assist those who have to follow strict dietaries due to their medical or health related issues. People who have to eat specific types of foods should put less effort into searching for suitable foods.
- **Objective 4:** To provide opportunities for users to speak out their opinions and feedback regarding their experiences. This is a very useful feature for both restaurants and users, because by taking feedback into account, they learn their mistakes to improve.
- **Objective 5:** To Serve customers with cheaper food prices to **encourage participation** in using the eatery management system more. This also encourages the users to try foods in the restaurants that they have not ordered before.
- **Objective 6:** To ensure that users are not overwhelmed with useless or incompatible eateries to avoid clutter within the user interface and facilitate a simple and convenient experience connecting with invaluable eateries.
- **Objective 7 (Novel):** To provide users with a randomized way of obtaining coupons in a fun and entertaining way
- **Objective 8 (Novel):** To provide a random eatery feature for people who have trouble deciding on what they are going to have for their meal or which eatery to visit.
- **Objective 9:** To allow users to generate additional random options if they are unhappy with the provided outcome.

## Functionality - Registration

For an application that aims to connect eateries to users, it is important to be able to distinguish the two parties and more specifically the different users and eateries using the platform. For this reason, account making is an essential component of SavourySaver as well as all contemporary websites. Once an unregistered user visits SavourySaver, they can choose to create either a user or eatery account to make use of the application. Using the respective banner, an unregistered user can create a user account by clicking on 'REGISTER' and an eatery account by clicking on 'NEW RESTAURANT'.



If a user decides to create a User account they will be presented with the page on the right, which requires their details including their full name, email, desired password and address. Similarly, if they decide to create an eatery account they will be represented with a similar form which has the same details with the addition of a phone number, website, and a preselected cuisine. Both registration pages check important details to ensure that they are valid and secure while ensuring that an email can't be reused. If successfully created, the account is stored on the database with an encrypted password.

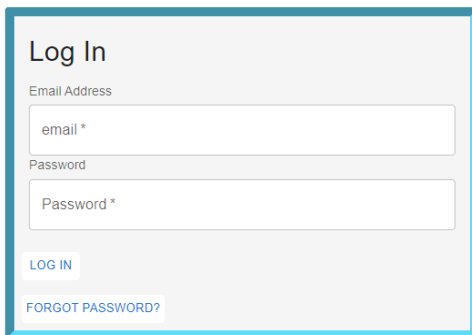
A registration form titled 'Create Account'. It contains input fields for First Name, Last Name, Email Address, Password, Confirm Password, Street, Suburb, Region, and PostCode. Each field has an asterisk indicating it is required. At the bottom left, there is a blue 'REGISTER' link.A registration form titled 'Sign up your Restaurant'. It contains input fields for Restaurant Name, Email Address, Phone Number, Street, Suburb, Region, PostCode, Website, Password, and Confirm Password. The Restaurant Name field has an asterisk. The Website field has a pre-filled value 'https://'. There is a dropdown menu for 'Cuisine' with 'ITALIAN' selected. At the bottom left, there is a blue 'REGISTER' link.

Objectives addressed:

As a result of this functionality, we are able to target all our objectives as it allows us to make a distinction between users and eateries on the application. This means that we

can deliver objectives 1 to 9 by using the account registration feature to distinguish users and eateries making it an essential feature to have.

### Functionality - Login/Logout

A login form titled "Log In" with a light gray background. It contains two input fields: "Email Address" with a placeholder "email \*" and "Password" with a placeholder "Password \*". Below the fields are two links: "LOG IN" and "FORGOT PASSWORD?". The form is outlined with a blue border.

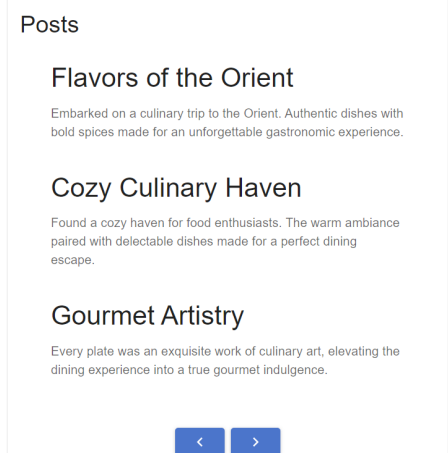
Based on the existence of the registration feature, it is necessary to have a complementary login feature which allows users to make use of their accounts and the application. This functionality asks the user for their email and password. If the correct account information is supplied, the user will be logged into their unique account and be given a token stored as a cookie which

will be used as a form of authentication when they attempt to access features of the application. Once logged in, a user will have the option to logout. This option will blacklist the token initially supplied and remove the token cookie from the browser, preventing it from being used again.

Objectives addressed: As with the previous functionality, this conventional feature is also essential for both eateries and registered users to be able to access the application's features. Additionally it allows each user to own a secured account that is tailored to them and the details they have provided. For this reason, the ability to login and logout is a necessity for all 9 objectives.

### Functionality - Posts

SocialSaviour has functions to create new posts about special events to keep the users updated. It's also

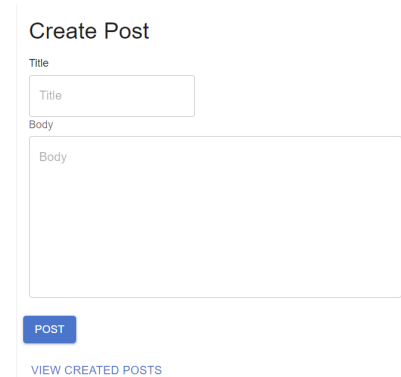




possible to search for certain posts based on the identification numbers of eateries that have posted. When a registered user clicks an eatery banner, they are able to view posts that are created by that eatery. The website also has arrows that allow users to scroll through previous posts. The image on the right shows posts that are seen by a user, and another one at bottom-right shows the page where restaurants create posts. Restaurants can choose a title and a body (description) of their posts, and they are also able to view previously created posts.

### Objectives:

By utilizing posts, the users would not miss updates of eateries and they will enjoy new meals and beneficial events. This links the users and restaurants together, so it is closely related to objective 1. Additionally, posts can act as an advertisement to attract more users to try their foods, therefore it also satisfies objective 5.



The image shows a 'Create Post' form. It has a title input field and a larger body input field. Below the body field is a blue 'POST' button. At the bottom of the form, there is a link that says 'VIEW CREATED POSTS'.

### Functionality - Searching Restaurants by names

Allow a variety of users to effectively find suitable eateries based on an array of factors which will be used to narrow the scope of eateries yielded upon a search. This method is important because the users do not need to spend too much time finding the restaurants they're looking for.

SocialSaviour provides a searching function to help users look for the eateries that they want. The users input the name of restaurants, and SocialSaviour will return the result that they wanted. If no results are found then the site displays 'No Results' to alert users.

## Objectives:

This method makes sure that the users are not excluded from the benefit of finding eateries that they desire. It satisfies objective 2, 5, 6 and possibly objective 3, because even the users who prefer specific meals can also search restaurants that they already know.

## User Home


[BROWSE](#)

### Functionality - Searching Restaurants by dietaries and cuisines

Allow the user to narrow their search depending on their dietary needs.

Scrolling down the list of eateries until the users find their preferred meals is great, but it could take too much time and work, and it may make the users feel discouraged to use the application. SocialSaviour is implemented with various searching methods for users to choose their preferences more easily with less time and work consumed. For

Cuisine ▼

MM/DD/YYYY 

S

Dietary ▲

lactose free

gluten free

vegetarian

example, if someone wants to have an Italian meal, they could search Italian cuisines in the search box. SocialSaviour provides 3 dietary options; lactose free, gluten free, and vegetarian.

Objectives:

Users who need to follow strict dietary rules due to their medical related issues can find their preferred meals. People who had to consume specific types of foods would require less effort to find what they want, so it is closely linked to satisfaction of objective 2, 3 and 6.

### Functionality - Reviews

Allow users with accounts to write reviews about the restaurants they visited.

Reviews written by users of the application are a great way for eateries to receive valuable feedback. Restaurants and eateries try their best to receive positive reviews from the users, because those reviews lead to circumstances where people find their restaurants more and more. Not only the users can write reviews, but it's also possible to look for reviews that they want to go over. When a registered user clicks an eatery banner, they can view the reviews of that eatery. The reviews, just like posts, have arrows to scroll through previous reviews. The reviews also display who it was written by.

Objectives:

The Review system satisfies objective 4. With the system implemented, the users are able to express their thoughts on their meals, and this provides the eateries to improve

#### Reviews

Delicious food, friendly staff, and cozy ambiance. A perfect place to enjoy a delightful dining experience.

Written by: John Smith

Excellent service and mouthwatering dishes. The flavors danced on my palate. I'll definitely be back for more!

Written by: Emily Johnson

Top-notch restaurant with a diverse menu. The presentation was artful, and the taste was simply divine.

Written by: Michael Williams



themselves. It also partially satisfies objective 1, since reviews are also one of the ways to construct a bond between eateries and users.

### Functionality - Recommendation

Displays and recommends restaurants that are offering discounts or events to all users. SocialSaviour can recommend eateries to users, as well as showing new restaurants. This feature is great for new users who want to have a meal but do not have a definite preference on what to eat. The banners also display restaurants' names, location, and options to share and subscribe.

#### New Restaurants



asdf

not added | St. Ives

[SHARE](#) [LEARN MORE](#) [UNSUB](#)



olio

not added | some suburb

[SHARE](#) [LEARN MORE](#) [SUB](#)



Real Man

not added | Mac

[SHARE](#) [LEARN MORE](#) [SUB](#)

#### Suggestions For You



asdf

not added | St. Ives

[SHARE](#) [LEARN MORE](#) [UNSUB](#)



olio

not added | some suburb

[SHARE](#) [LEARN MORE](#) [SUB](#)



Real Man

not added | Mac

[SHARE](#) [LEARN MORE](#) [SUB](#)

### Objectives:

To encourage users to try different restaurants that they have never used before. This leads to more participation to use SocialSaviour. This feature is tightly related to

satisfying objective 5, and objective 1 as well, since more participation from users means they get to interact with eateries even more.

#### Functionality - Searching restaurants by location / distance

Finds and displays restaurants found near the user's inputted location, while factoring in preferred distance.

There are some users who are not very familiar with places around their home.

This leads them to a chance of missing great eateries with quality foods and services nearby. To prevent this,

SocialSaviour is implemented with a function where the user inputs their location and their preferred distance of the area they want to search from that location. The system calculates all the restaurants that are within the given distance, starting from a given location, and displays them.

#### Objectives:

Users can access nearby eateries / restaurants easily, and facilitate a simple and convenient experience connecting with eateries that are far. At the same time, it helps them to avoid restaurants that are out of their preferences, so it is linked closely to objective 6. And just like other searching methods, this feature narrows down the scope so that the users can choose their meal more easily. Additionally by utilizing this search method, the users are more willing to try eateries that they have not visited before. Therefore this also satisfies objective 2, 3, and 5.

## Functionality - Vouchers

Restaurants can offer vouchers to some users. For example, if a user visits a certain restaurant often, then they could give the user a voucher as a token of appreciation. A user may use vouchers to earn discounts so they can buy meals with cheaper prices. In SocialSaviour, a registered restaurant can set the percentage of discount, number of vouchers, and expiry dates. A voucher code is a mixture of randomly generated alphabet characters. They can also view previously created vouchers.

### Vouchers

Percentage *	Number of Vouchers *
Start Date MM/DD/YYYY	End Date MM/DD/YYYY
<input checked="" type="checkbox"/> Reoccurring	
<button>CREATE VOUCHER</button>	
<a href="#">VIEW CREATED VOUCHERS</a>	

### Objectives:

This feature lets users enjoy meals with discounted prices. There are several ways to obtain vouchers, which may depend on restaurants. Users that have used SocialSaviour a lot would have more chances to get a voucher. This naturally brings objective 5 to be satisfied. It also fulfills objective 1 because vouchers are closely related to events, and this encourages more interaction between the users and restaurants.

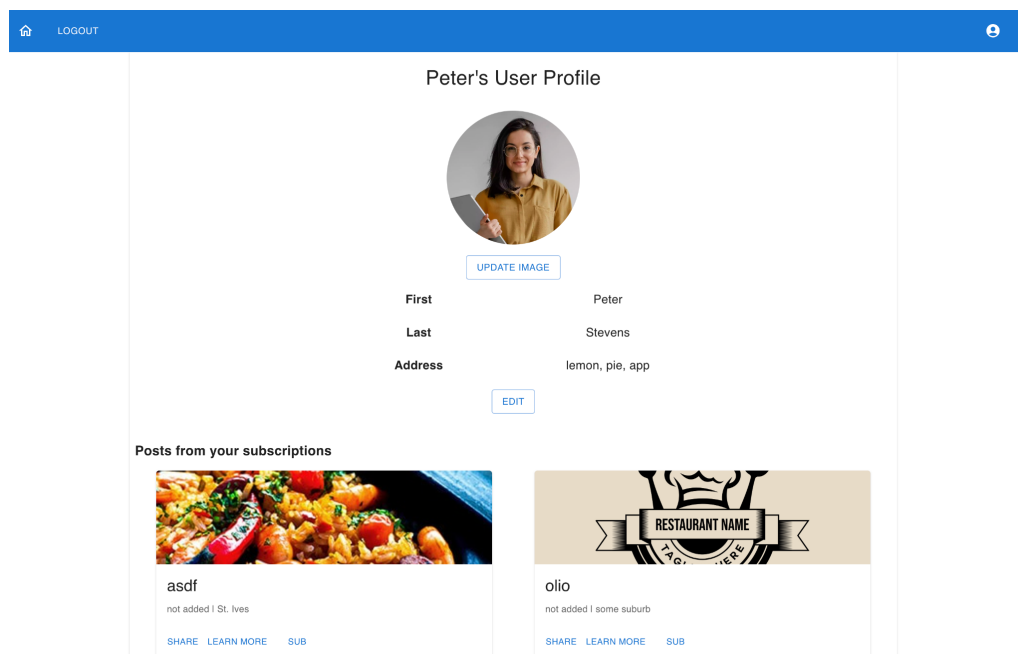
- 1) Idea is to implement a 'random' factor for people to try their luck, like a slot machine. All users have a chance of receiving discount vouchers daily. Discount vouchers are usually provided to newcomers or by regular events, but this random coupon provider is a feature that is run every day.
- 2) The system suggests a random place in their area to get food based on many factors like the user's location and preferences.
- 3) Allow the user to choose to either regenerate or confirm a random suggestion

## Functionality - User Profile

Users have a profile that includes all relevant personal information with their account including a profile picture. The profile displays the restaurant posts from all the eateries they subscribe to. Each restaurant post includes the option to share, learn more or unsubscribe to the eatery. The personal information including the profile picture is editable to keep the information up to date. The user profile is reachable from the profile icon on the far right of the banner from any location in the website.

### Objectives:

This feature lets users view the posts from all the eateries they are interested in. The page enables them to manage which posts they want to see by allowing them to unsubscribe from eateries and keep track of all their subscriptions in one location. Eateries are encouraged to improve their experience to encourage re-engagement from their users and more subscriptions so that their posts will continue to reach their customers.



## Third-party functionalities:

### Amazon Relational Database System (RDS)

Amazon RDS is a cloud management system which is managed by Amazon Web Services (AWS). It supports various database engines such as PostgreSQL as well MySQL which is what we used for our application. Amazon RDS provides easy database setup as the group can develop and test the application with shared databases with provided database, host, and password instead of testing it in the local mysql environment.

The Amazon RDS that we used is a free tier where we can use up to 750 hours of 1 database instance each month with 20GB of SSD storage each month for free for 1 year which is more than sufficient for us to develop and test our application.

### React (Front-end library)

Our front-end is built using React (Meta Open Source, 2023), a front-end library that enables creating dynamic front-end pages using a series of reusable and customizable components. Components allow for complicated front-end pieces to be reduced to smaller, more manageable parts that can be shared amongst the website. This was especially beneficial in a group project whereby the work was able to be split up easily and shared amongst ourselves. React also provides functionality that allows for dynamic updating of front-end elements through the use of 'hooks'. These hooks can be passed between components and when used automatically refreshes front-end components intelligently to display the dynamic state without re-rendering an entire page. Finally, React is an open source library created by Meta so it adheres to industry standards and there are no licensing issues to impact our project.

### React-router (Routing Library)

React-router (reactrouter.com, n.d.) is a library that enables React front-end applications to implement client-side routing. We used this as the final piece of our solution for creating a one-page web app so that the user experience doesn't require waiting for a



page to be loaded in the browser after being fetched from the server. React-router allows routes to be set up that can be navigated to anywhere in the website and the page will be re-rendered using the associated React component. We chose to use this to implement our routing since it is designed to be used with React apps and the functionality we required didn't warrant using a much larger and complicated routing framework such as Next.js. Again, react-router is open source.

### Material UI (Front-end Library)

Material UI ([mui.com](https://mui.com), n.d.) is a front-end library that includes an extensive suite of React components and styles. Our front-end was created completely in React and Material UI allowed us to import most of the basic components and set up the structure of our website fast. The components are highly customisable and flexible, allowing us to combine them with our own React functionality easily, and modify them to the way we wanted it. Additionally, the library is open source so there are no licensing issues impacting our usage. Also, Material UI is designed by Google and adheres to their Material design philosophy, so using the library helps to keep us aligned with industry design standards.

### Google Distance Matrix (API)

The Distance Matrix API (Google Developers, n.d.) is a resource we used to filter eateries based on location. The endpoint is able to return distance data from two provided locations. On our front page we allow users to enter their location and a maximum distance in kilometers to set up a radius of an area of where they want to eat. With these two pieces of information, we can pass each restaurant location to be displayed to the user along with the location of the user to the Distance Matrix API and filter out any results that have a distance greater than the set maximum distance. The API is free to use when used with a Google Developers account, which we had set up.

### Google Places (API)

The Google Places API (Google Developers, n.d.), provides detailed information about locations based on Google's own mapping of the real world. It is able to convert simple descriptions of places and return many related results with useful metadata. We were able to use it with our website to implement the autocomplete location input feature on the home page. This feature takes a text input of a location and displays a list of relevant classified locations to be selected. The chosen location's information is parsed and processed to be ready to use by the Distance Matrix API as described above. Again, this API is free to use with our previously set up Google Developers account.

## Implementation challenges:

### User Context

The view of our website's front-end is determined by the user interacting with it. A user not logged in will only see a limited amount of functionality compared to someone that is an eatery owner will have a completely different view as they are a separate stakeholder that has different requirements such as managing their restaurant posts on their profile compared to someone searching for a place to eat. Because of this, our app utilizes the 'Context' React hook that is passed to each of the main page components. This hook contains the current state of the user and we set it up to handle a nullable boolean state. This allows us to define that when the context is set to null, the user is not logged in. Set to true and the user is logged in but not an eatery manager, and finally false is logged in and an eatery manager. This is a simple piece of functionality that easily allows us to modify the displayed components based on the user just by having a boolean check in front of styles. For example, on the restaurant profile the update and edit buttons for the content are only displayed when the 'userContext' is set to false. This way the update components don't appear to a non-logged in user to

prevent them from sending failing requests to the backend and when they do log in as an eatery manager the components are re-rendered instantly and available to use.

### Reusable Components

Since our website is designed to be consumed by 2 main stakeholders, users and eatery managers it is required to change its view to match the functionality each stakeholder expects. However, there is a significant overlap in what each stakeholder expects to use and so, rather than copying similar functionality between the 2 views, many components were designed to be reused for both sides. This was done by allowing for dynamic updates based on the identity of the user, displaying eatery management functionality for eatery managers and user functionality for users. The reusable components were customisable with boolean flags that would render certain styles and layers when identity conditions were met.

Similarly, the restaurant posts displayed on the home page for users are rendered from the same component that is displayed on the user's profile page.

## Installation Manual:

This project uses the virtual machine option. Specifically, it uses the windows version of virtualbox 6.1.46 based on the Lubuntu 20.4.1 LTS virtual machine image. For instructions on how to import this image follow this tutorial:

<https://www.linuxvmimages.com/how-to-use/how-to-import-vm-images-in-virtualbox/>

Virtual Machine Credentials:

Username: lubuntu

Password: lubuntu

(to become root, use sudo su)

To download docker on the respective virtual machine and use it as expressed below, it is recommended that the following guide is followed:

1. Open a terminal and git clone the following repository  
<https://github.com/uns-cse-comp3900-9900-23T2/capstone-project-3900f18ameanencodingmachine.git>
2. In the backend directory, create a .env file and copy the details in .env\_template into the newly created file. Once created, insert all the required fields.
3. Download and start docker on your computer. Ensure that docker daemon is running and listening for connections.
4. Use the docker command docker-compose build to build the app and docker-compose up to run the built app.
5. Visit <http://localhost:80/> to start using the application.

Database setup (if you want to start from empty database, otherwise skip this):

1. Make sure to update the Ubuntu VM with “sudo apt update” and enter whatever password from when you login into Ubuntu VM
2. Then download the MySQL using the command “sudo apt install mysql-server”
3. Make sure that the mysql server is started by using “sudo service mysql start”
4. Go to sql directory
5. Now, to access the Amazon RDS MySQL database use the command “mysql -h DB\_HOST -u DB\_USER -p” where DB\_HOST and DB\_USER is found on .env file. When prompted for password enter DB\_PASSWORD (also from .env)
6. Then input command “source schema.sql”, you may ignore the warning. This will create two databases (one for application and one for test) with the same table schema.
7. Use “\q” to exit the mysql

Note: to test using the jest file, change database field in config variable at database/db-config/db\_connection.js to DB\_DATABASE\_TEST from .env file

## Reference list:

1. Meta Open Source (2023). React. [online] react.dev. Available at: <https://react.dev/>.
2. reactrouter.com. (n.d.). Home v6.4.1. [online] Available at: <https://reactrouter.com/en/main>.
3. mui.com. (n.d.). React Components - Material UI. [online] Available at: <https://mui.com/material-ui/>.
4. Amazon Web Services, Inc. (n.d.). Amazon RDS Free Tier | Cloud Relational Database | Amazon Web Services. [online] Available at: [https://aws.amazon.com/rds/free/?loc=ft#Free\\_Tier](https://aws.amazon.com/rds/free/?loc=ft#Free_Tier).
5. Google Developers. (n.d.). Overview | Distance Matrix API. [online] Available at: <https://developers.google.com/maps/documentation/distance-matrix/overview>.
6. Google Developers. (n.d.). Overview | Places API. [online] Available at: <https://developers.google.com/maps/documentation/places/web-service/overview>