



**DALHOUSIE**  
**UNIVERSITY**

Master of Applied Computer Science

**CSCI 5709**

**Advanced Web Services**

**Assignment 2 (Individual)**

**B00924759**

## **1. Project**

**LastServe:** An initiative to eradicate world hunger by establishing a conduit for restaurants to give extra food to the hungry.

### **1.1 Project Background**

In Halifax, there are a disproportionately large number of underprivileged individuals, students, and people in need. Access to food is perceived by us as a fundamental human right, not a privilege. No one ought to struggle to acquire enough food each day or go to bed hungry. With the help of this project, eateries will be able to feed the needy in their local neighbourhood. Restaurants that have leftover food will update their public posting on our website each night with how much food is still available. Anyone in need of food can do so by setting up a time slot and coming to pick up their meal for that evening. By doing this, food waste in restaurants will be reduced and people won't go to bed hungry.

### **1.2 Target User Insight**

This website mostly caters to individuals on a restricted budget, restaurant owners, and volunteers. Those on a limited budget, such as students, the unemployed, and those with limited time are our potential users. Restaurant proprietors make up the second group of users. The restaurant would be listed on the website, and posts would be updated daily, with the amount of food still available. Those who have additional time on their hands and can assist restaurants in providing food to the hungry would make an additional group of users. The administrators are LastServe coordinators who oversee the site's user base and content. They will be provided with a unique set of tools that will allow them to do this by authorizing user registrations, examining, or deleting restaurant material as needed, and managing user engagement.

The website was designed to be very user-friendly and to take a short time to finish a task. The forms are created so that they are simple to read and just gather the necessary information without requesting excessive amounts of personal information. We kept the colour scheme subdued to accommodate all types of users to appeal to emotions and communicate the essence of our website.

### 1.3 User-Centred Design

The website has been created to enable users to find what they want in only a few clicks while taking the diverse demography of the target consumers and LastServe's goal into consideration. The website uses a consistent layout and colour scheme.

The feature chosen by me is User Profile Management. Following are the tasks and details about how a user-centric approach has been used to develop the website.

#### User Registration/Sign Up

The website's home page, as shown in **Figure 1**, has a straightforward design, and the Sign Up and Login buttons are clearly accessible in the navigation bar. **Figure 2** shows the registration form for creating an account on LastServe, which is self-explanatory and seeks just minimal but sufficient data from the user. As soon as the user finishes entering information in a field, appropriate validation errors are displayed. Once the details are verified and user details saved to the database, a success message is displayed to inform the user that an account has been created as seen in **Figure 3**.

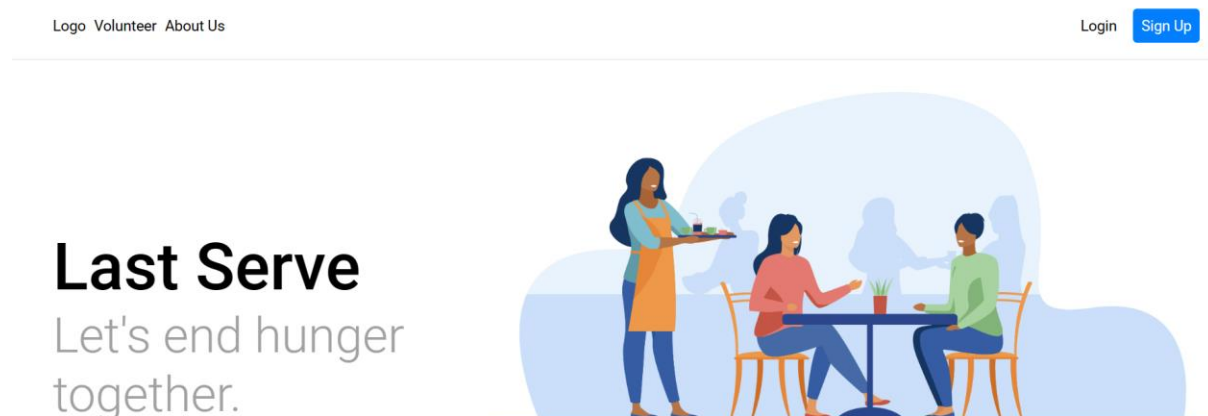


Figure 1: LastServe Home Page

## Register

First Name\*

First Name

Last Name\*

Last Name

Email\*

Email

Password\*

Password

Confirm Password\*

Confirm Password

\* Mandatory fields

Register

[Register Restaurant](#)




Figure 2: User registration form

## Register

First Name\*

First Name

Last Name\*

Last Name

Email\*

Email

Password\*

Password

Confirm Password\*

Confirm Password

\* Mandatory fields

Register

[Register Restaurant](#)

Success!

Account created successfully on LastServe

Close

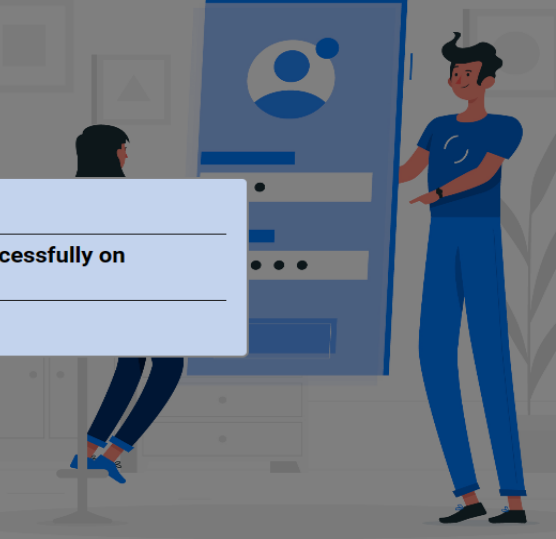


Figure 3: Success message on account creation

## Existing User Login

By logging into LastServe, users can view food availability posts posted by restaurants.

**Figure 4** shows the login page, which demands the user's Username and Password. The email ID associated with the user's account, as shown by the 'E-mail' placeholder in the field, would be the username. An invalid credentials error message is given if the username and password do not match.

In case the user does not remember the password used on LastServe while attempting to login, the login page additionally includes a link to reset password.

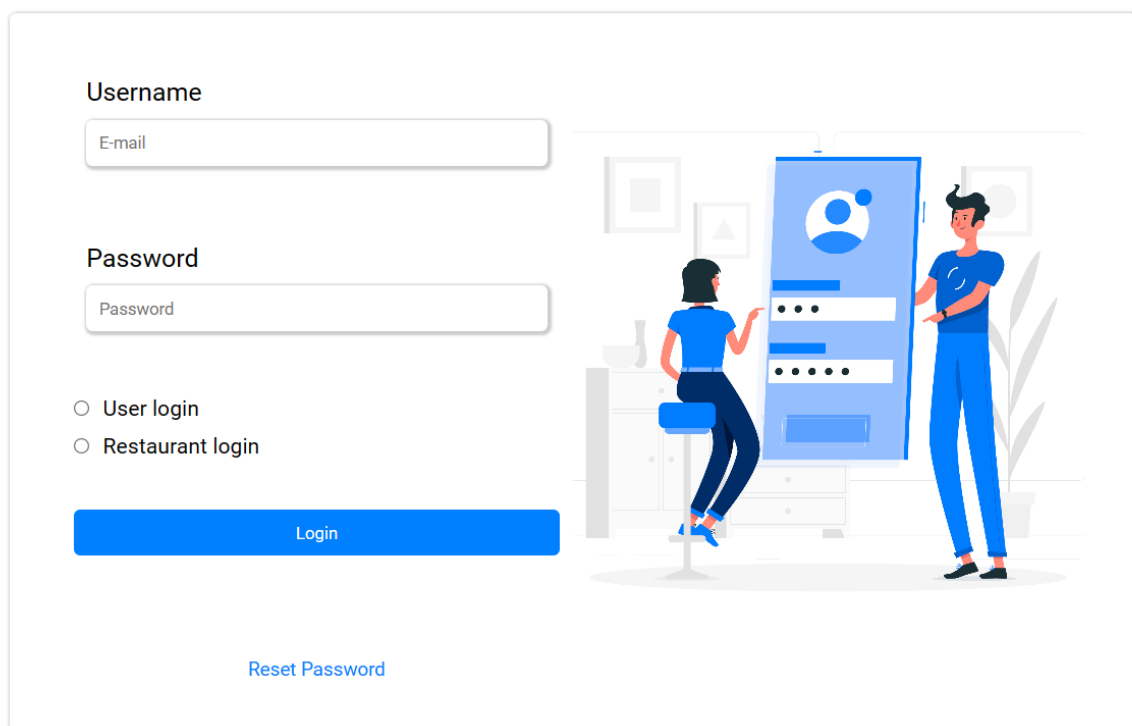

The image shows a user login page with a clean, modern design. On the left side, there are two input fields: 'Username' with a placeholder 'E-mail' and 'Password' with a placeholder 'Password'. Below these fields are two radio buttons: 'User login' (selected) and 'Restaurant login'. A large blue 'Login' button is positioned below the radio buttons. At the bottom of the form area, there is a blue link labeled 'Reset Password'. To the right of the form, there is a stylized illustration of a man and a woman in blue clothing interacting with a large, futuristic blue machine that has a circular logo on top and several buttons or screens on its front. The background of the illustration is light gray with some abstract shapes.

Figure 4: User Login page

## Password Reset

The process of resetting password has been designed to be hassle free as well as secure. As seen in **Figure 5**, users will have to enter the registered email address to which a reset key would be sent. A backend process will check if the entered key is same as the one sent on the email. If the keys match, users will be redirected to set the new password as seen in **Figure 7**. **Figure 8** shows a success message displayed to inform the user of password change.

## Reset Password

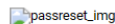


Registered E-mail\*

Send Reset Key

Figure 5: Reset password page.

## Reset Password



Registered E-mail\*

Reset Key\*

Set New Password

Resend Reset Key

Figure 6: Option to Resend the key or set a new password

## New Password

Password\*

Confirm Password\*

\* Mandatory fields

Reset Password




Figure 7: New password creation page

## New Password

Password\*

Confirm Password\*

\* Mandatory fields

Reset Password

**Success!**

New Password set successfully

Close


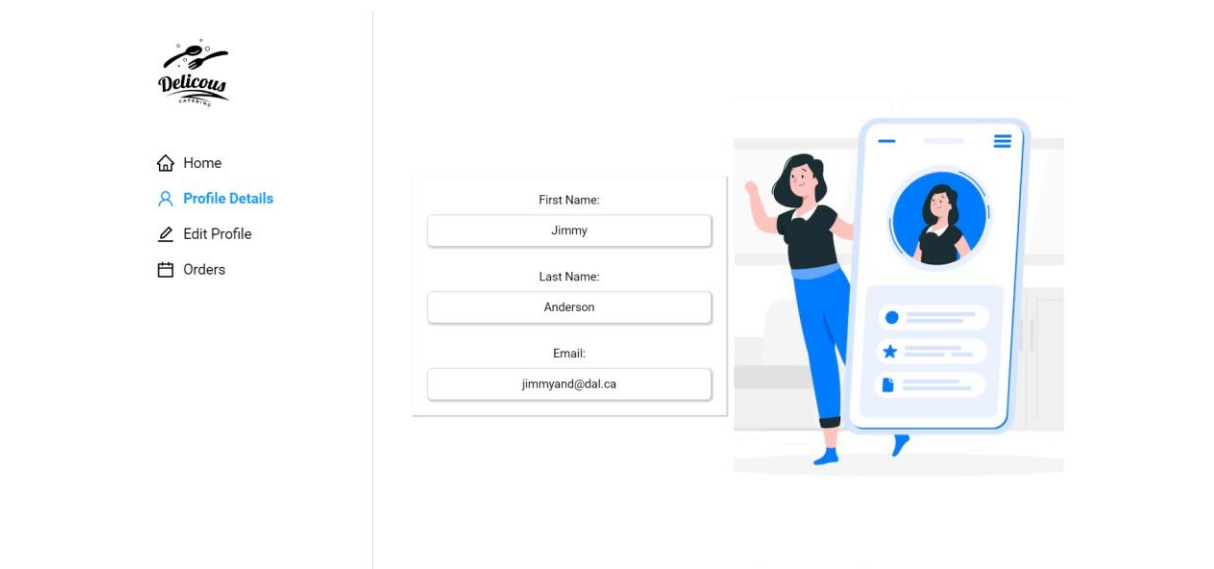


Figure 8: Password reset successful message

## View User Profile

A single point of access to all the user-available functionalities is provided by the user dashboard. Users can view personal information by clicking the 'Profile Details' link on the user profile page, as shown in **Figure 9**.



*Figure 9: User Details displayed on clicking Profile Details*

## Edit User Profile

As seen in **Figure 10** users can easily edit the first name and last used while creating the account. The email address however can not be edited and is highlighted by making the field dark. A success message is displayed to indicate that the changes have been saved as seen in **Figure 11**



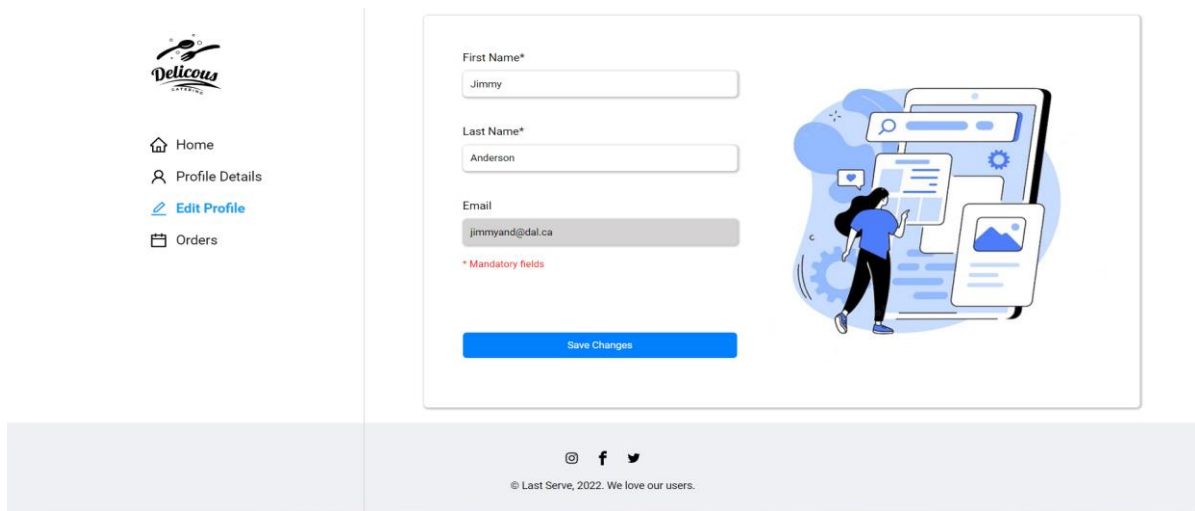


Figure 10: Edit user details page

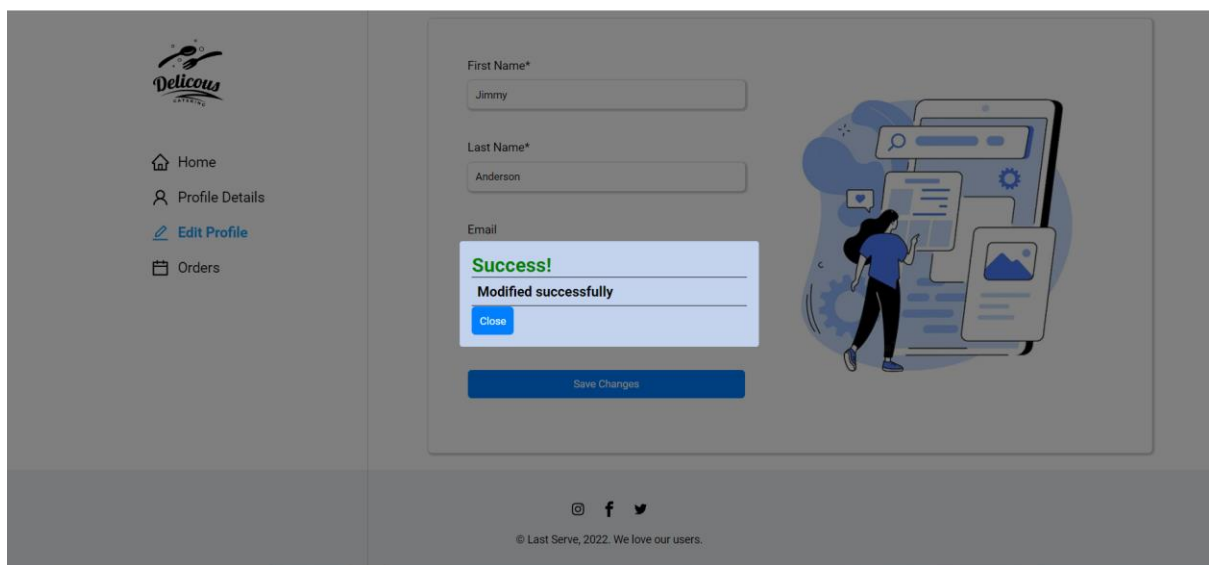


Figure 11: Success message on modifying details

## 2. Application Architecture

**Figure 12** shows the architecture of LastServe. The front-end of the application would be designed using React framework. We would be using Styled components as it allows CSS to be written in the JavaScript file (.jsx), to style the react components. We would use Axios, a promise-based HTTP client for node.js and the browser, to call the backend APIs.

The backend would be developed using node.js and express.js framework. We would use the RESTful (Representational State Transfer) guidelines for writing the APIs. Additionally, we would use MongoDB, a NoSQL database, hosted on-site to store the application data in JSON format to enable faster retrieval and performance.

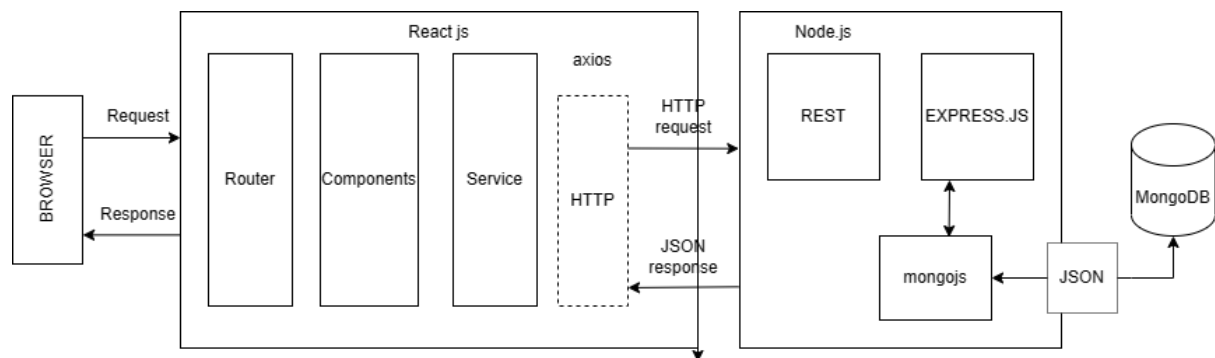


Figure 12: Application Architecture for LastServe

## 2.1 Interaction Design

The feature developed by me is 'User Profile Management'

Below are the tasks included under the feature.

### User Registration

User Scenario - At midnight, a student is starving but too worn out to prepare supper. Due to a limited budget, the student does not want to order food at a restaurant. The student wishes to register with a website that lists local restaurants that provide leftovers from the day at no charge. To finish the procedure, he will do the following.

**Figure 13** and **Figure 14** below shows the task flow and click stream diagram for the user registration process.

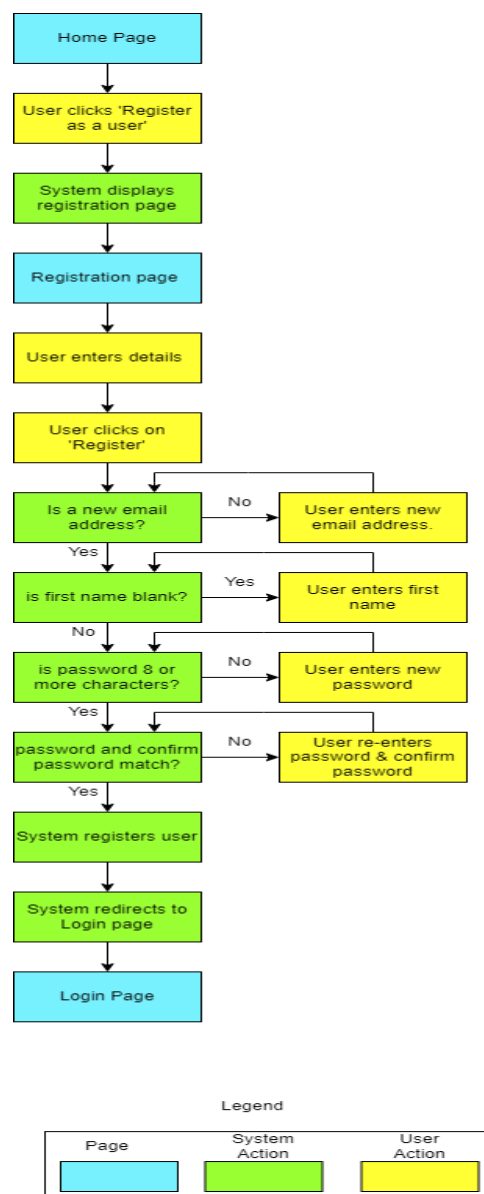


Figure 13: Task flow diagram for user registration

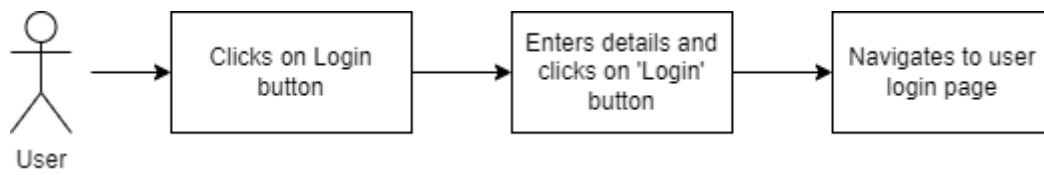


Figure 14: Click stream for user registration process

## Existing User Login

User Scenario - Login into account: A student is hungry and wants to check restaurants in the vicinity offering free leftover food on the LastServe website.

Figure 15 and Figure 16 show the task flow and click stream diagram for the login process.

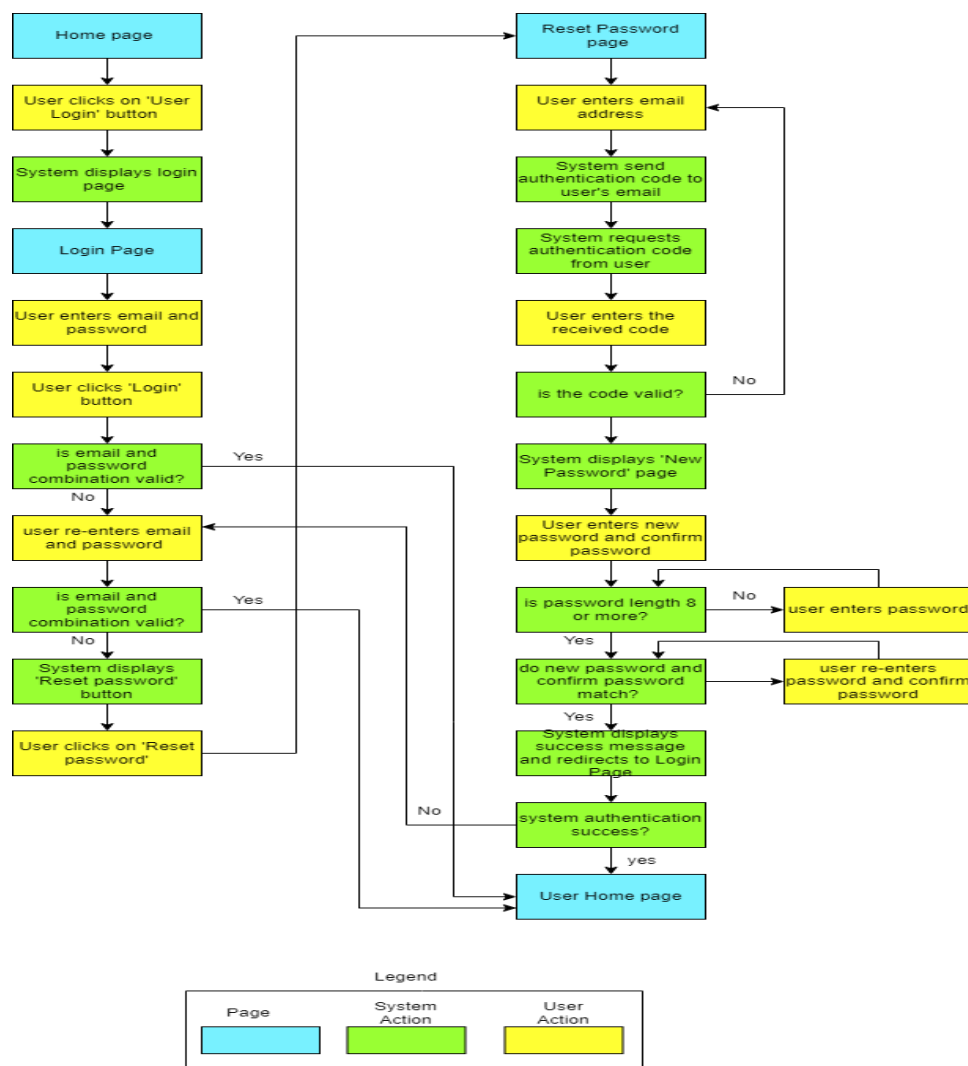


Figure 15: Task flow diagram for user login

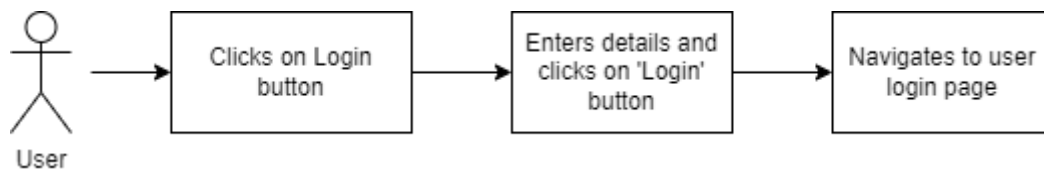


Figure 16: Click stream for user login process

## View User Profile

User Scenario - The student needs to check to the data entered while creating the account.

**Figure 17** and **Figure 18** depict the task flow and click stream diagram for view user profile task.

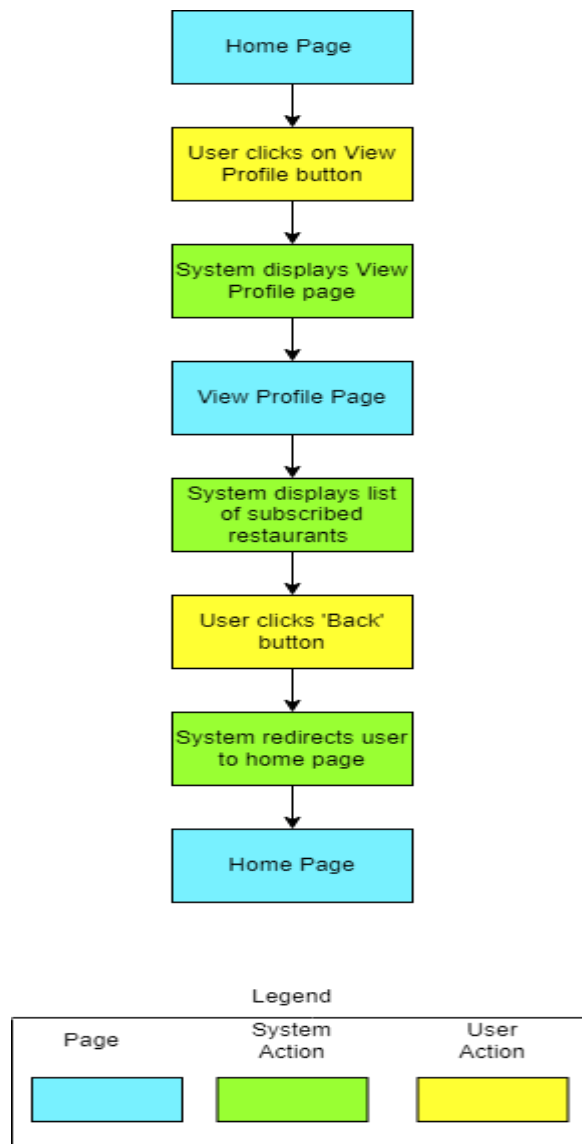


Figure 17: Task flow diagram for viewing profile details

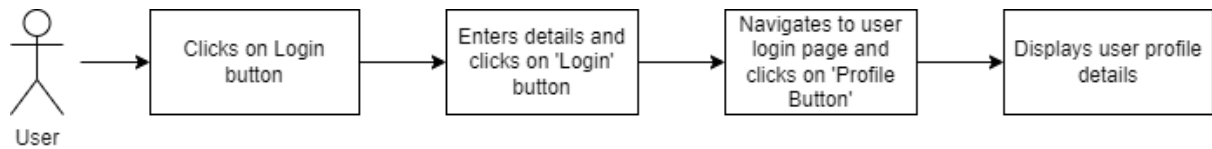


Figure 18: Click stream for view user profile details

## Edit User Profile

User Scenario - The student typed his name incorrectly when setting up an account. Also, even though it was an optional field, the student omitted the last name. The last time he visited a restaurant, he was denied access to pick up food since the name on the reservation was different from the name on his ID. To prevent future misunderstandings, the student wishes to modify his first name and add a last name.

Figure 19 and Figure 20 shows the task flow and click stream diagram for edit details task.

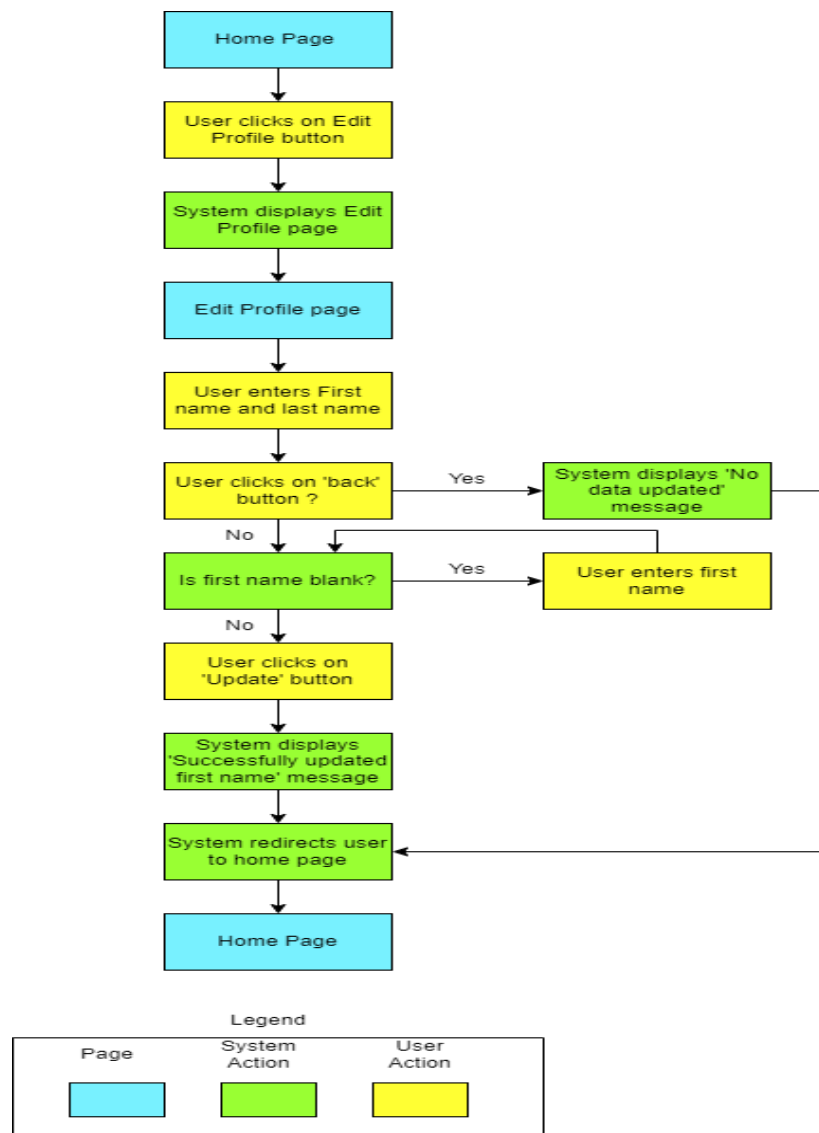


Figure 19: Task flow diagram to edit user details

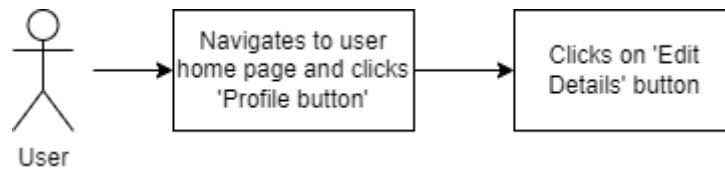


Figure 20: Click stream for edit user details task

## 2.2 Process Flow

**Figure 21** shows the workflow for LastServe application. When a user visits the website, the front-end environment displays the appropriate react components. When user makes a request, an axios HTTP request is sent with JSON content to the backend environment developed using node.js .The request is placed in the node.js event queue which is picked up for processing by the request handler. Nodejs environment interacts with the MongoDB database to process the request and sends back a JSON response. The result of the response is then rendered to the user.

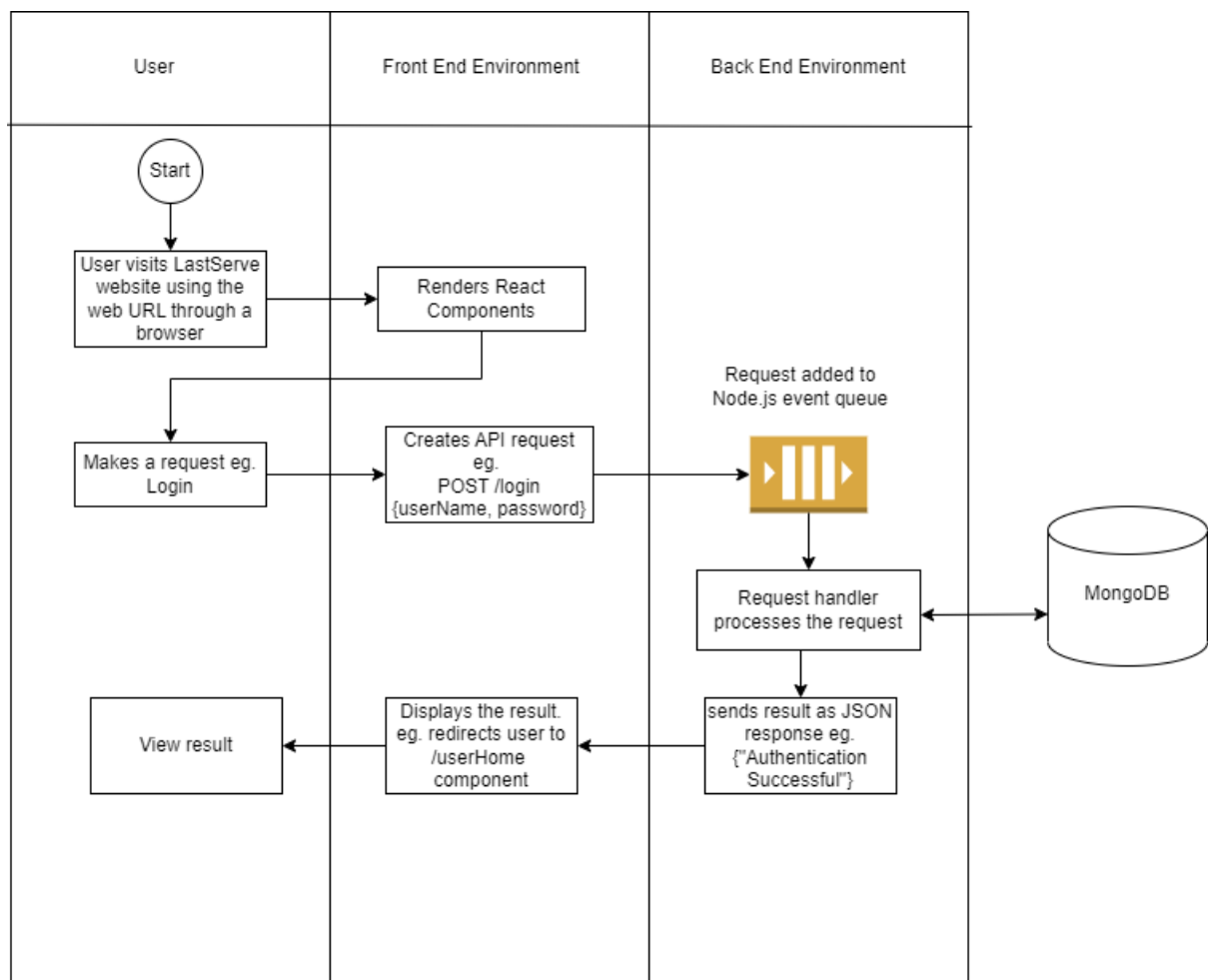


Figure 21: Process workflow of LastServe architecture

## 2.3 Folder Structure

The front end of the project would be developed using React JS framework. **Figure 22** displays the folder structure for the same.

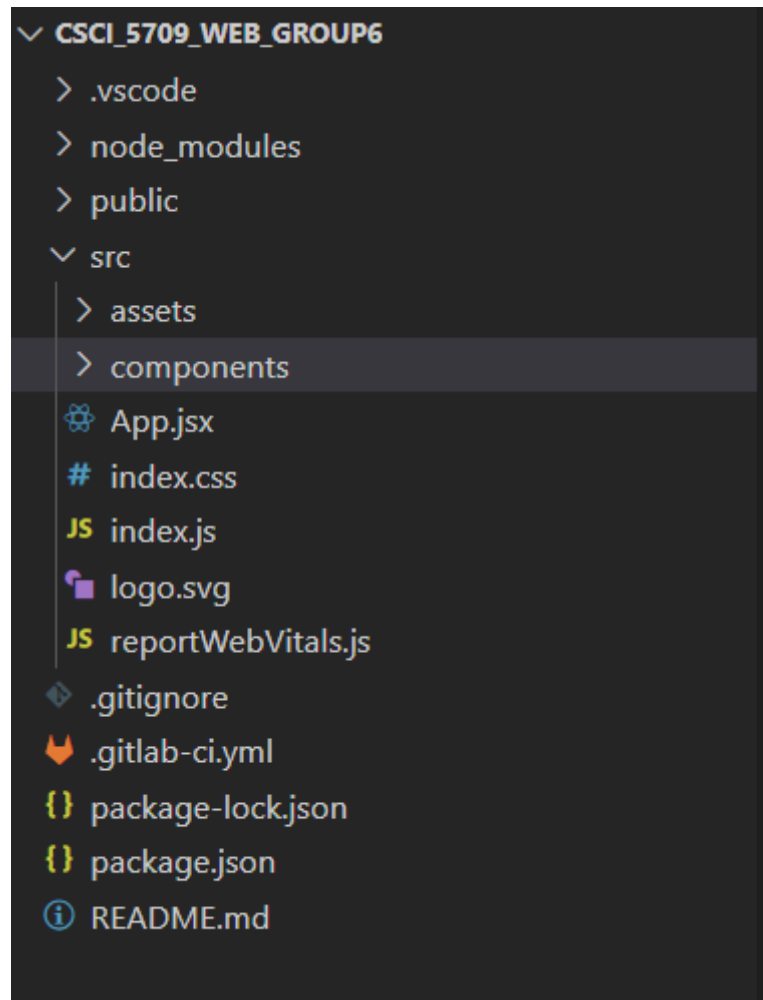


Figure 22: Reactjs folder structure

Table 1: Folders and description

Folder/File	Description
package.json	Contains metadata about a project like dependencies, version etc. and enables npm to start the project, run scripts, install dependencies, and publish to the NPM registry.
package-lock.json	Used to lock dependencies to a specific version number.
App.js	Contains the top component.
assets	Contains all the resources required in the project like images.
components	Contains all react components.



The backend of the project would be developed using NodeJs, ExpressJS and MongoDB database. **Figure 23** displays the folder structure for the same.

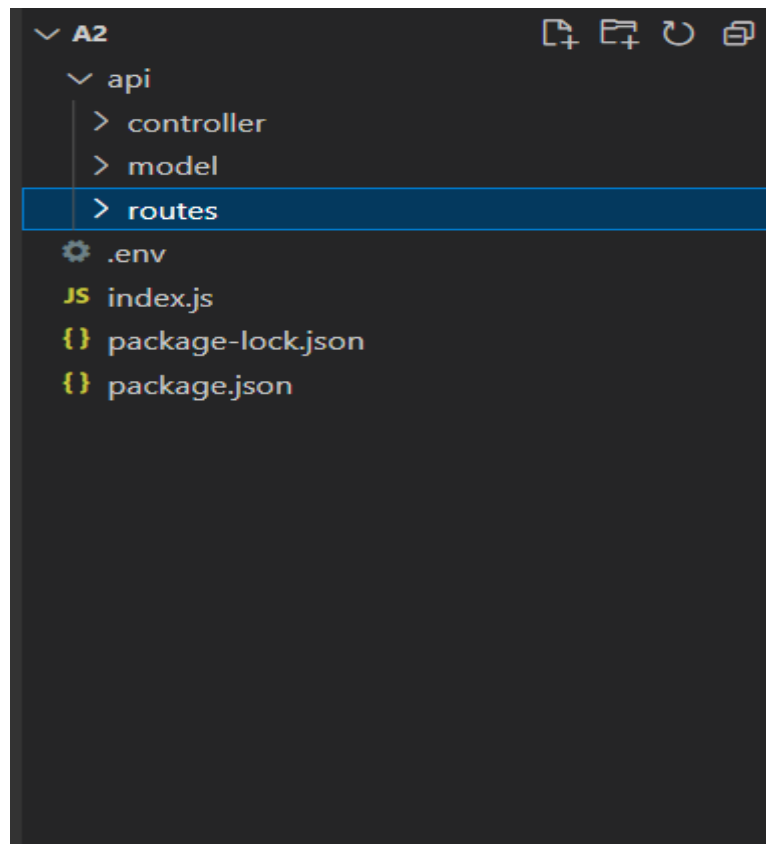


Figure 23: Folder structure for NodeJs

Table 2: Folders and description for the backend

Folder/File	Description
package.json	Contains metadata about a project like dependencies, version etc. and enables npm to start the project, run scripts, install dependencies, and publish to the NPM registry.
package-lock.json	Used to lock dependencies to a specific version number.
index.js	Handles startup, routing and other functions of the application and requires other modules to add functionality.
controllers	Contains files to handle incoming requests and return responses.
routes	Contains the route URLs of the APIs.
model	Contains files for the data model.

## References

- [1] E. Kring, "Role of mongodb in mern/mean stack - dzone," *dzone.com*, 17-Nov-2021. [Online]. Available: <https://dzone.com/articles/role-of-mongodb-in-mernmean-stack>. [Accessed: 11-Mar-2023].
- [2] "Draw.io - free flowchart maker and diagrams online," *Flowchart Maker & Online Diagram Software*. [Online]. Available: <https://app.diagrams.net/>. [Accessed: 10-Mar-2023].
- [3] "Sample\_A2(S20)." Dalhousie University, [online document], 2020. [Accessed 10-Mar-2023].
- [4] "LastServe," *Netlify.com*, 2023. [Online]. Available: <https://last-serve.netlify.app/>. [Accessed: 10-Mar-2023].