DALHOUSIE
UNIVERSITY

CSCI 5709: Advanced Topics in Web Services

Assignment - 2

by

Jahnavi Gajjala [B00969773]

Submitted to

Prof. Gabriella Mosquera
Department of Computer Science
Dalhousie University.

1. GoDine:

GoDine is a restaurant reservation application designed to enhance the dining experience for both customers and restaurant owners. With a focus on convenience, variety, and user engagement, GoDine positions itself as a comprehensive tool for modern diners. It transforms how customers discover and book dining spots, while simultaneously equipping restaurant owners with robust tools to optimize their reservation management and customer engagement.

2. Target User Insights:

The following are the users that we target with our application:

- General User or Customer
- Restaurant Owner
- Admin

Here are the **user personas** of our application:
The Urban Foodie (Customer): Maggie is tech-savvy, resides in a metropolitan area, and is always on the lookout for the next great dining experience. She appreciate the convenience and enjoy exploring new culinary trends.
The Busy Professional(Customer): With a packed schedule, Dara values efficiency. She seek a hassle-free booking experience for business lunches or dinners, often at the last minute.
The Social Planner(Customer): Responsible for organizing gatherings, Ben utilizes GoDine to plan events and find restaurants that cater to group dining needs with personalized menus.
Restaurant Owners: Alfredo is a restaurant owner, seeking to expand their customer base and streamline their booking processes using GoDine.

Here are the **user scenarios** of each of the user personas of our application. These scenarios cover the basic flow.
User Scenarios:
Spontaneous Dinner Plans: A couple decides to dine out and uses GoDine to find a restaurant with available tables and appealing deals.
Business Dining: A professional quickly books a table through GoDine for an impromptu meeting, selecting a restaurant based on location and client preference.
Event Planning: An organizer books a venue for a wedding reception using the event reservation feature, liaising with the restaurant to finalize details.

Here are the **prerequisites** to use our application.
User Requirements:
Technical Savvy: Minimal technical proficiency is expected for navigating the app interface.
Devices: Access to GoDine is required through a compatible device or computer with an internet connection.
Learning Curve: The application is designed for immediate use with no learning curve, providing an accessible platform for all users regardless of prior app experience.

3. User-Centered Design:

The following image shows the low fidelity prototype of the newsletter subscription feature. In this page, users get to view a sample of previous newsletters from different weeks, giving them an idea of the rich content that is accessible. This allows us to attract the user into subscribing to the newsletter. The newsletter will feature exclusive Offers, New Restaurant Alerts, Featured Restaurants, Food and Dining Trends, Recipe Section, Events and Experiences, Health and Nutrition Tips. Once the user reads it and finds it interesting, they can subscribe to the newsletter using the subscribe button at the bottom of the page. After subscribing, the user will receive the newsletter every week to their registered emails. To keep readers interested and coming back, the newsletter area is refreshed every week with fresh themes.

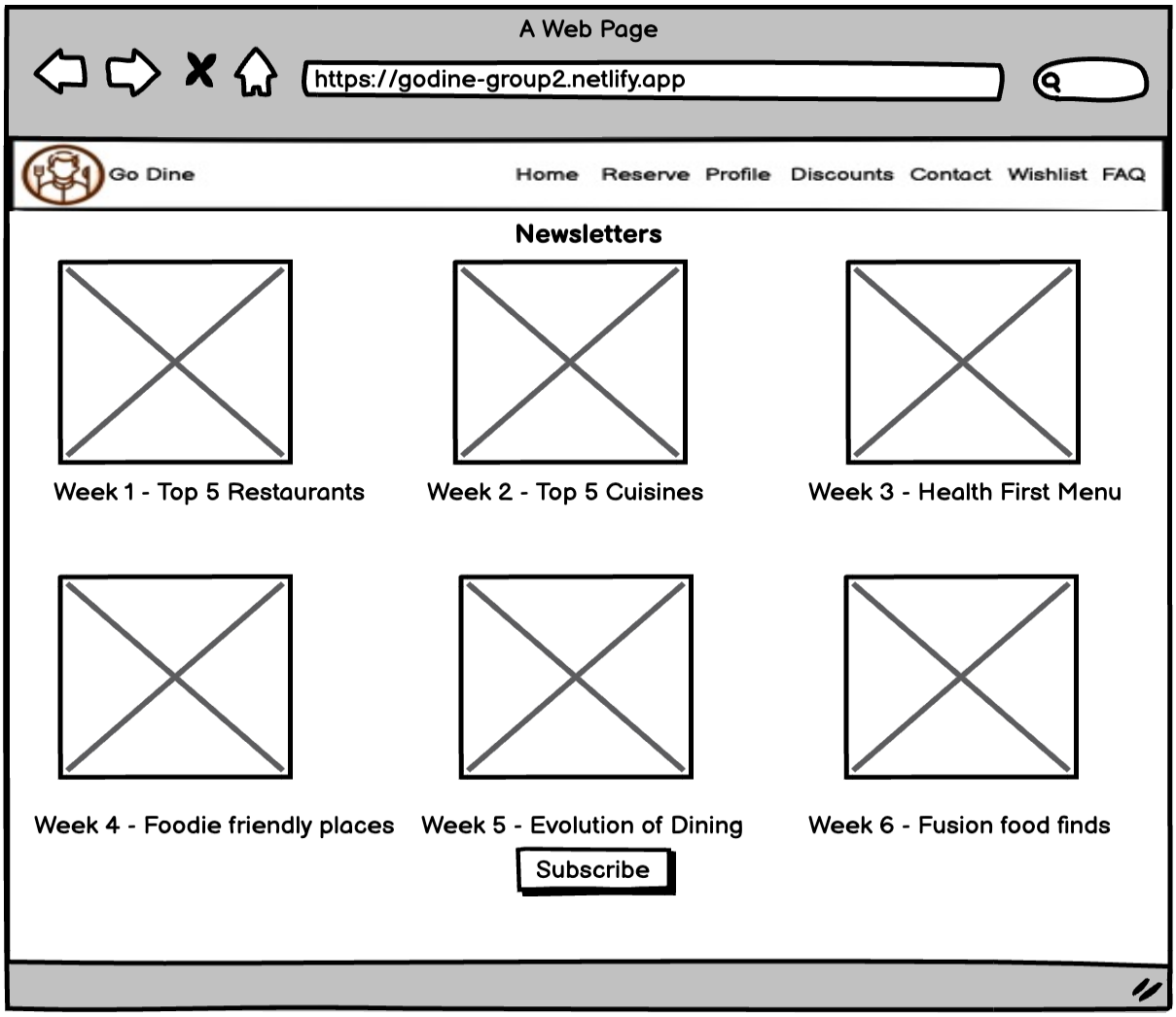

Figure 1: Newsletter Subscription Page Wireframe[4]

The design elements on this webpage are thoughtfully curated to captivate the audience and encourage them to subscribe to the newsletter. The following are the justifications for the design decisions.

Grid Layout for Newsletters: The newsletter samples are presented using a grid layout. It supports visual hierarchy by giving each newsletter identical weight, allowing users to easily review the material without feeling overwhelmed.

Sample Newsletters: An overview is given without overburdening the viewer with text by substituting visuals for the newsletter's actual content. It arouses interest and motivates

greater engagement.
Descriptive Titles: Every sample of a newsletter has a headline that summarizes its content, such as "Week 1 - Top 5 Restaurants". Users may make informed subscription decisions by knowing what to expect from the content because of this explicit labeling.
Subscription button: After visitors have looked through the newsletter samples, a "Subscribe" button is thoughtfully positioned at the bottom of the page. This placement takes use of the typical browsing behavior where users scroll down to view more content and are then prompted to take an action.
Typography and White Space: The design feels less crowded because there is enough white space surrounding all of the elements, and the text is readable. Users can concentrate on the text and read it more easily as a result.

Benefits for the customer:
Clarity and Focus: Users can concentrate on the material without being distracted by the uncluttered layout.
Informed Choice: Users may be more satisfied with the newsletter's content if samples are provided so they are aware of what they are getting into.
Ease of Use: Users are more likely to navigate a website with ease when it is made easy to use with familiar design conventions.

4. Application Architecture:

The diagram shows the architecture of a web application with a Node.js/Express backend and a React.js frontend. The React application communicates with the browser through services to make API calls and components for the user interface. Client-side navigation is handled via React Router. Requests are processed by middleware on the server side and then sent to Express routers, which forward them to the relevant controllers. They contain the logic to communicate with data models, which carry out CRUD operations with a MongoDB database.

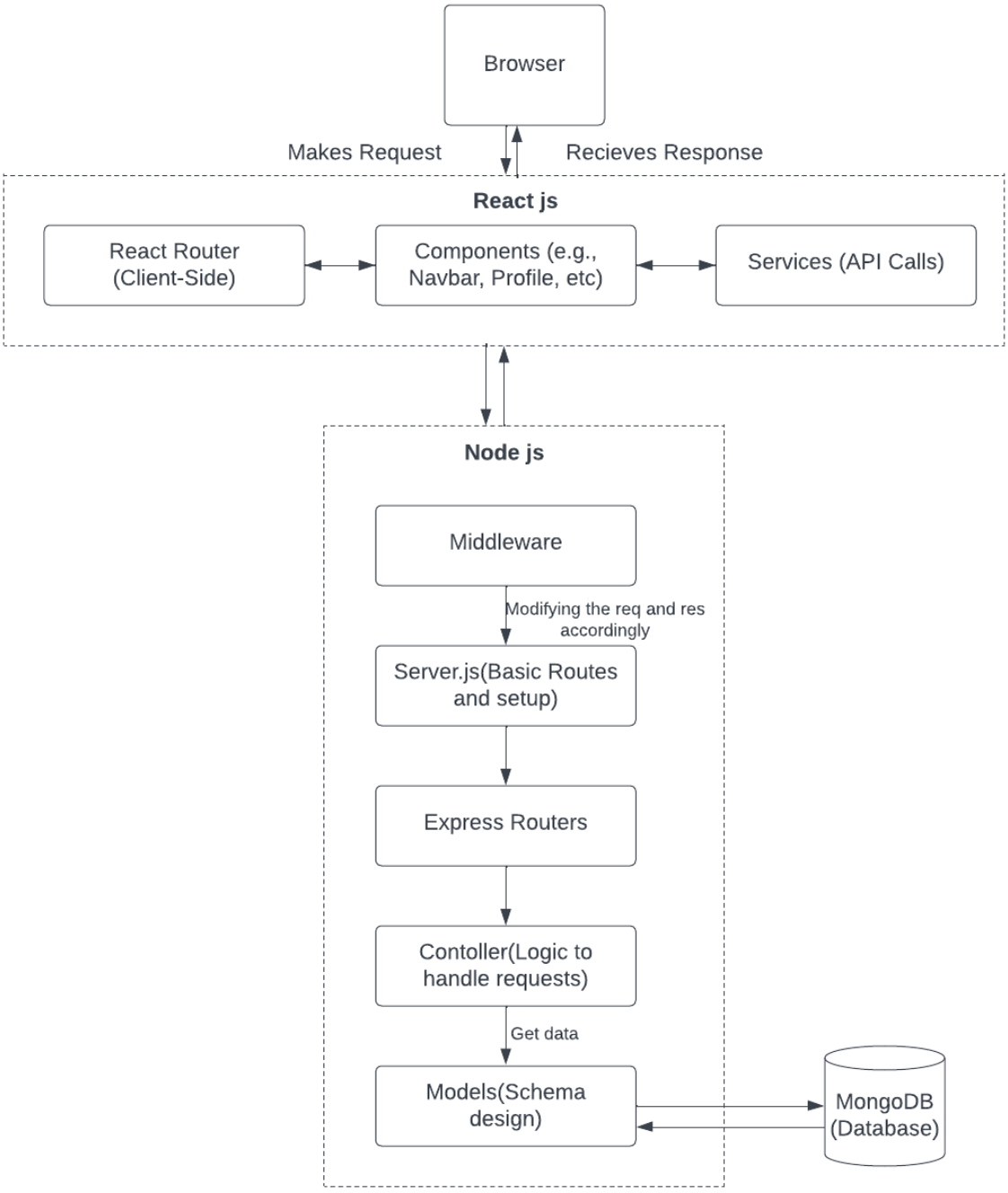

Figure 2: Application Architecture [2]

5. Interaction Design:

User Scenario:

On a busy Wednesday afternoon, Daniel, a financial analyst, and a food enthusiast who works in downtown Chicago, decides to use the GoDine app during his lunch break. His objective is to stay informed about the newest restaurants and food trends without actively seeking out this information. Daniel is especially looking for food-related events and healthy dining options that he can go to on the weekends or after work.
Intrigued by the weekly email feature that offers insights into the city's dining setting, He plans to sign up for the newsletter so that he can get updates directly to his inbox. Daniel thinks this will make it easier for him to schedule his meals and discover new restaurants that align with his preferences. [1]

He must follow the below **use case** process:

1. Daniel logs into the GoDine application.
2. The app displays the main dashboard with various options.
3. Daniel selects the 'Newsletter' option from the menu.
4. The system shows a page with a sample of the GoDine newsletter.
5. Daniel views the sample newsletter.
6. Satisfied with the content, Daniel clicks on ‘Subscribe’ to receive the newsletters.
7. The system confirms the subscription and indicates that newsletters will be sent to Daniel’s registered email.
8. Daniel accidentally clicks 'Unsubscribe' button in the user profile page.
    1. The system asks for confirmation to unsubscribe.
    2. Daniel clicks 'Cancel' on the unsubscribe prompt.
    3. The system retains Daniel's subscription and returns to the newsletter page.
9. The system updates his subscription status and sends a welcome email to Daniel’s registered email address.
10. Daniel checks his email inbox and finds the welcome email from GoDine.
11. If Daniel does not find the welcome email in his inbox.
    1. The system has a 'Resend Email' option in the newsletter subscription confirmation page.
    2. Daniel clicks 'Resend Email'.
    3. The system resends the welcome email to Daniel's registered email address.
12. Each week, Daniel checks his email inbox for the new GoDine newsletter.
13. The system automatically sends the weekly newsletter to Daniel's registered email address. [1]

The following images show the **task flow and click streams** of Newsletter Subscription feature of GoDine application. The process flow starts when user navigates to home page and clicks on Newsletter Subscription on the header. It redirects to the Newsletter page where the users get to view a sample of previous newsletters from different weeks. After viewing the newsletters, the user can subscribe to them if interested, using the subscribe button.

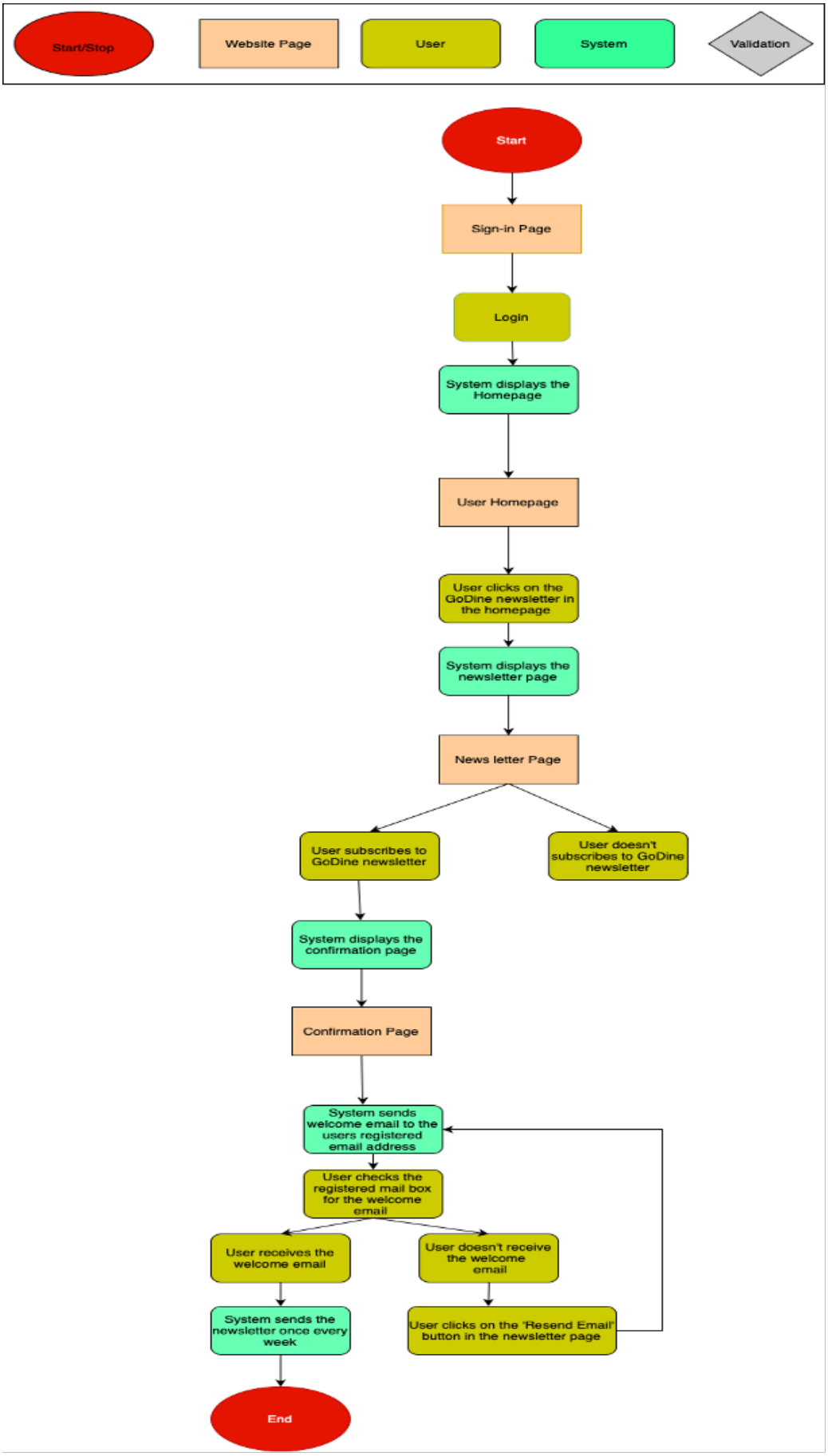

Figure 3: Task flow diagram applicable to Newsletter Subscription feature [1].

Figure 4: Clickstream applicable to Newsletter Subscription feature [2].

6. Process Workflow:

In the following workflow, a user visits a website and makes a request that prompts React components to display. The request is then sent to the back end where it's routed through a queue. Once the queue is verified as empty, the request is handled by a Node.js request handler which interacts with a MongoDB database. The result is sent back to the front end for the user to view. If the user clicks the subscribe button, an API request is sent, processed by middleware, which then communicates with MongoDB and sends an email confirmation to the user, after which the user views the confirmation message on the front end.

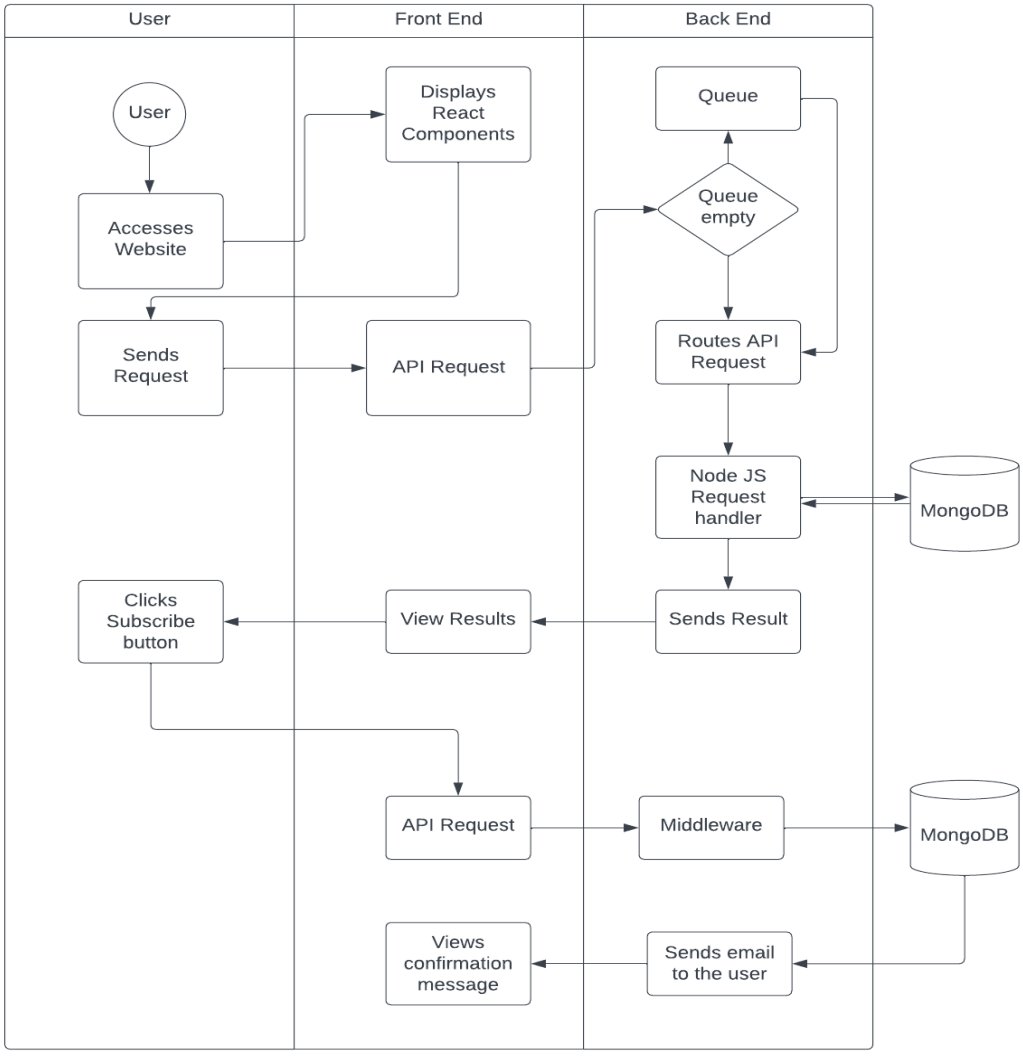

Figure 5: Process Workflow applicable to Newsletter Subscription [2].

7. Folder Structure:

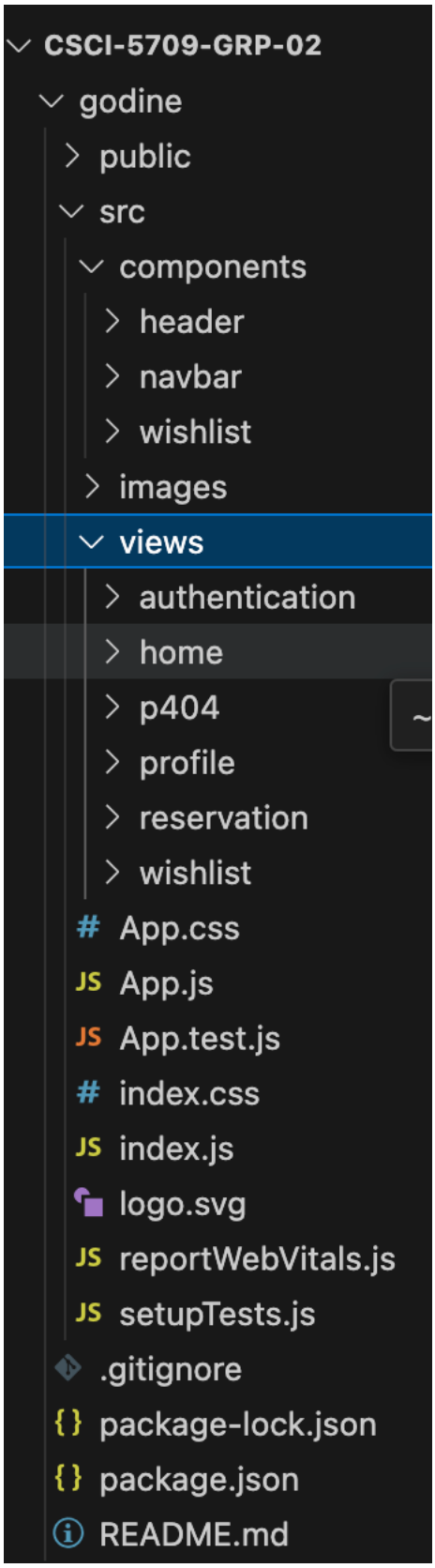

Figure 6: Client Side Folder Structure[3]

- **godine/**: Root directory.
- **node_modules/**: A folder that the project utilizes for all of its dependencies and libraries, made by the Node Package Manager (npm).
- **public/**: This directory stores the assets that should be accessible by the public when the app is deployed. For ex:
    - **favicon.ico**: A tiny symbol linked to the website that appears in the browser tab.
    - **index.html**: React components will be mounted within this file.
    - **logo192.png** and **logo512.png**: These are logo files in various resolutions that can be used as Progressive Web App (PWA) icons or for different device resolutions.
    - **manifest.json**: An application's name, icons, and start URL are defined in a JSON file that is used when the web application is installed on the user's home screen.
    - **robots.txt**: A text file created by webmasters that tells web robots—mostly search engine robots—how to navigate the pages on their website.
- **src/**: Source code is present
    - **components/**: Contains the components.
    - **images/**: Contains image files used in the application.
    - **views/**: Contains React component files that represent different views or pages in the application.
    - **App.css**: Main CSS file for styling the App component.
    - **App.js**: Principal React component that serves as the root for the rendered portion of the application.
    - **App.test.js**: Includes tests specific to the App component.
    - **index.css**: Contains global styles for the application in form of CSS.
    - **index.js**: The JavaScript entry point that renders the component tree of React to the Document Object Model.
    - **logo.svg**: The app's user interface uses an SVG logo picture.
    - **reportWebVitals.js**: A JavaScript file that allows to track the application's performance using web vitals.
    - **setupTests.js**: An application file used to configure tests for the React program.
- **.gitignore**: This file lists files that are purposefully left untracked and should be ignored by Git.
- **package-lock.json**: Any time npm alters the package.json file or the node_modules tree, this file is automatically generated.
- **package.json**: This file contains a variety of project-related metadata. This file contains data that npm needs in order to recognize and manage the project and its dependencies.
- **README.md**: A markdown file containing standard information about the program, installation instructions, usage guidelines, and occasionally codebase documentation.

Figure 7: Server Side Folder Structure[3]

- **godine_server/**: Root directory of the server-side application
    - **node_modules/**: Contains packages and modules installed via npm.
    - **src/**: Source directory where the main application code resides.
        - **config/**: Contains configuration files. For example: **nodemailer.js -** It is used for email functionality, possibly for sending emails through the application.
        - **controllers/**: Contains the controller files that hold the request handling logic. For instance, userController.js for operations pertaining to users, authController.js for authentication, and so forth.
        - **database/**: Probably includes scripts for database operations or database configuration, with db.js serving as the primary database connection file.
        - **models/**: Includes the data models that serve as a representation of the database collections' or tables' structure. The schema for each entity in the database is defined by models such as users.js, reservations.js, restaurant.js, and so on.
        - **populateData/**: Initial or test data (**populateData.js**).
        - **routes/**: The route definitions that link HTTP requests to the relevant controller functions are located in this directory. The client can communicate with different endpoints defined by files such as authRoutes.js and userRoutes.js.
- **.env**: Environment variables defined in a hidden file.
- **server.js**: The entry point of the application.
- **.gitignore**: A file that Git uses to decide which directories and files to skip over when committing changes.
- **package-lock.json**: Automatically generated when npm changes the node_modules tree or package.json.
- **package.json**: Defines the project's metadata, scripts, and the list of its dependencies that need to be installed.
- **.gitlab-ci.yml**: The application's build and testing specifications are outlined in the configuration file for GitLab's continuous integration service.

8. References:

[1] Y. Krishna Vaibhav, P. Tejas, K. Venkata Sreenivas Prasad, G. Jahnavi, S. Priyatam Reddy, B. Praveen Kumar Reddy "Project Proposal." Dalhousie University, [online document], 2024. [Accessed: 12-Mar-2024]

[2] "Intelligent Diagramming," *Lucid.com* [Online]. Available: https://lucid.app/ [Accessed: 12-Mar-2024]

[3] G. Jahnavi, "GoDine" Visual Studio Code, [Tool], [Accessed: 11-Mar-2024]

[4] "Balsamiq. Rapid, Effective and Fun Wireframing Software | Balsamiq," *Balsamiq.com* [Online]. Available: https://balsamiq.com/ [Accessed: 11-Mar-2024].