



# ASSIGNMENT 2

CSCI 5709

Dhrumil Amish Shah (B00857606)  
dh416386@dal.ca

## **A2.1 Application Features**

EnviClean is a web application designed for garbage disposal and collection. Below is the list of features provided by EnviClean:

- ⇒ List of **must to have** features:
  1. User Profile Management
  2. Social Clean Reporter Management – EnviClean Patron
  3. Notification Management System
  4. Garbage Collection Management System
  5. Garbage Depositor Management System
  6. Reward Points Management System
  7. Coupon Management System
  
- ⇒ List of **good to have** features:
  8. Blogs/Stories
  9. Depositor Leaderboard Management System
  10. Collector Leaderboard Management System

## **A2.2 Social Clean Reporter Management – EnviClean Patron**

This section covers the “Application Details” and “Application Workflow” of the feature **Social Clean Reporter Management – EnviClean Patron**.

### **A2.2.1 Application Details - EnviClean**

Today, the world is facing a global issue of waste management which has led to pressing consequences such as the emergence of infectious diseases, land and water pollution, obstruction of drains, and loss of biodiversity. The main cause of these problems is the communication barrier between the entities involved (the garbage depositor and picker) in the waste disposal and management process. To address these issues, we planned to build a platform - EnviClean that will bridge the communication gap between the people involved in the whole process.

EnviClean is a web application that primarily deals with waste management. We agreed upon providing some essential features such as garbage collection management, garbage depositor management, social clean reporter management, and notification management. Moreover, it efficiently handles issues such as the unavailability of periodic trash collections from residential houses, scarcity of service providers to pick up the trash based on the user's convenient time, and so on by some exceptional features available as a part of notification management in the application.

#### ⇒ **Target User Insight**

The web application targets serving two prime users: Garbage Depositors and Garbage Collectors. The potential users will access the web application to notify the garbage collectors of any unhandled or untreated waste and thus manage the waste disposal efficiently.

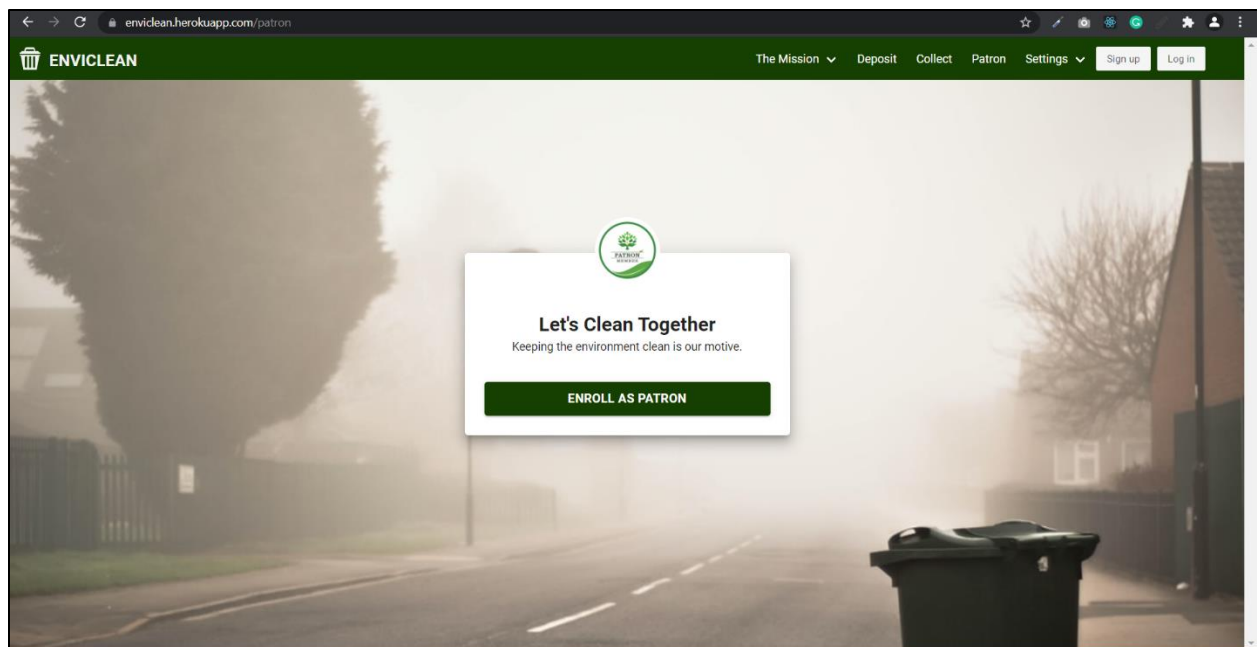
1. Garbage Depositor: The garbage depositors would use the application to post about trash or waste that needs to be treated. It provides a feature where the depositor can create a post of newly identified garbage in their house and locality with a picture and address details from where the garbage should be picked.

2. Garbage Collector: The garbage collector would be responsible to handle the garbage deposited by the depositors in the area they serve. The collectors can update the status of the posts created by the garbage depositors.

### ⇒ User-Centred Design Approach

I have implemented the **Social Clean Reporter Management – EnviClean Patron** feature for the EnviClean web application. This feature provides an interface for the EnviClean patron members to seamlessly identify all the trash they find when they are outside and upload it on the EnviClean website. Organizations and garbage collectors are intimated to make sure that the reported garbage is taken care of properly.

**Figure 1** displays the enroll as a patron web page that will be displayed to the registered users of the web application. This web page is created using a minimalistic design approach. A card is displayed to the user with a line "Let's Clean Together" and a full-width button that says "Enroll as Patron". The design of this web page is simple, keeping in mind the usability principles that make it a two-steps process. The user clicks on the "Enroll as Patron" button and is prompted with a dialog box displaying the terms and conditions for enrolling as a patron. **Figure 2** displays the terms and conditions dialog box.



*Figure 1 - Enroll as a Patron page [1]*

**Figure 2** displays the terms and conditions dialog box. On accepting the terms and conditions, the user will be successfully enrolled as a patron that will effectively close the dialog box and prompts the user to create their first post as an EnviClean Patron member (**figure 3**).

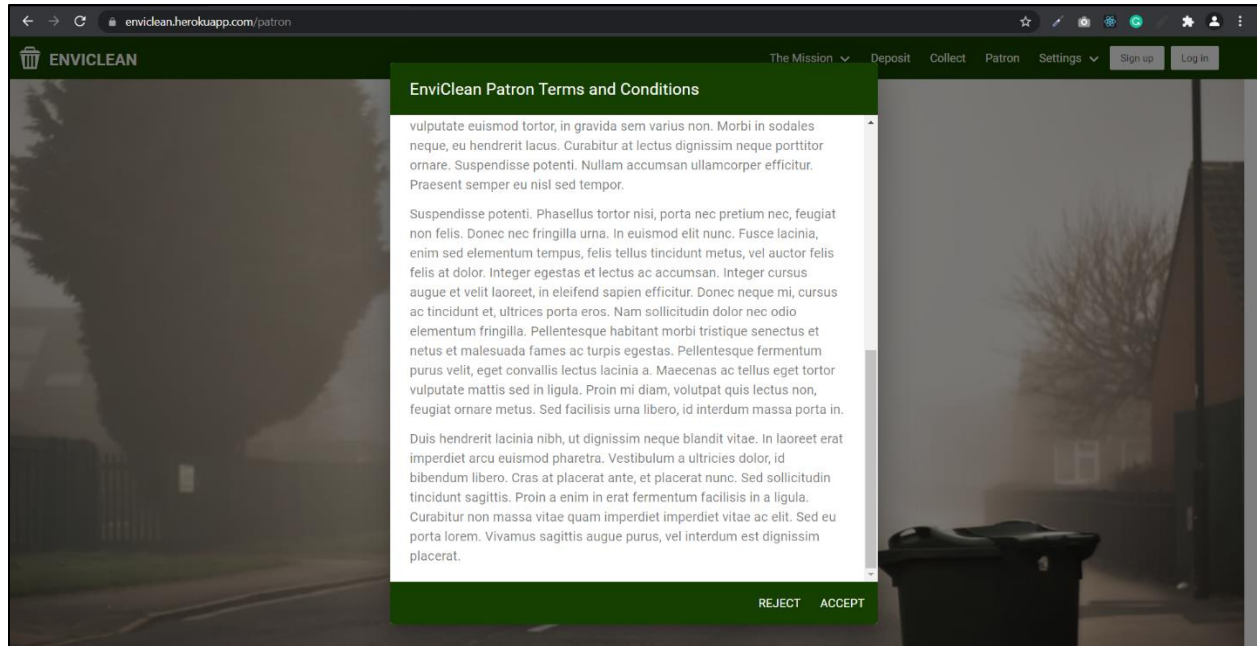


Figure 2 - Patron Terms and Conditions dialog [1]

**Figure 3** displays the web page where the users are prompted to create their first post as an EnviClean Patron member.

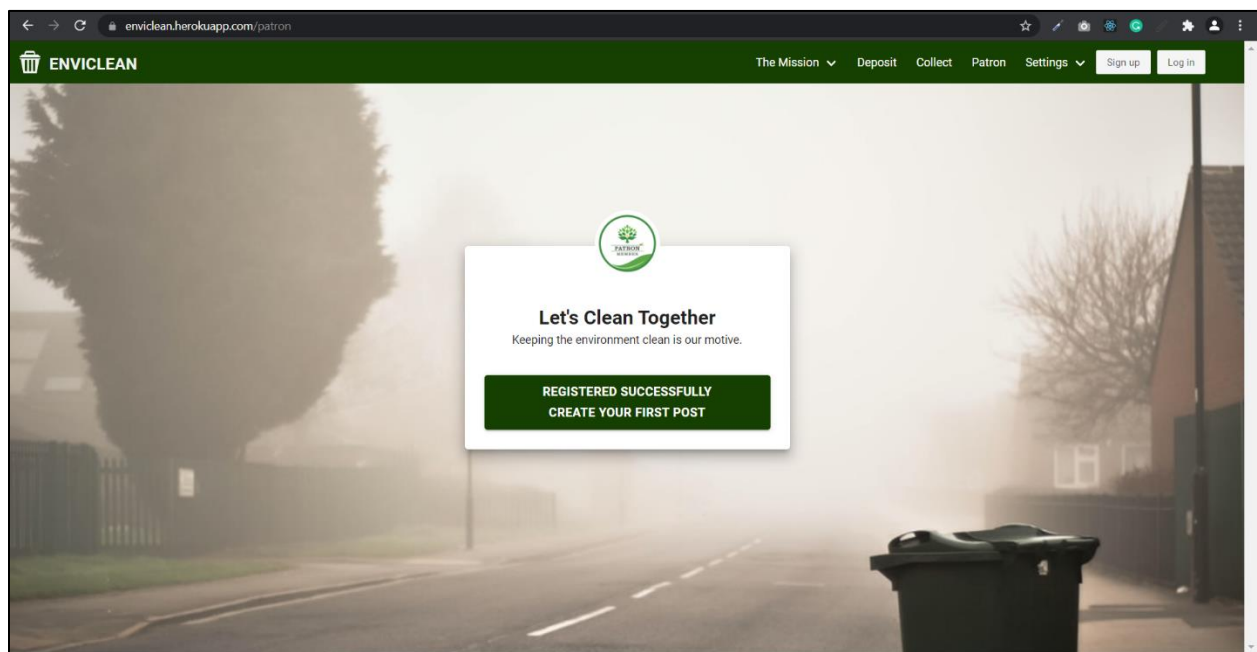
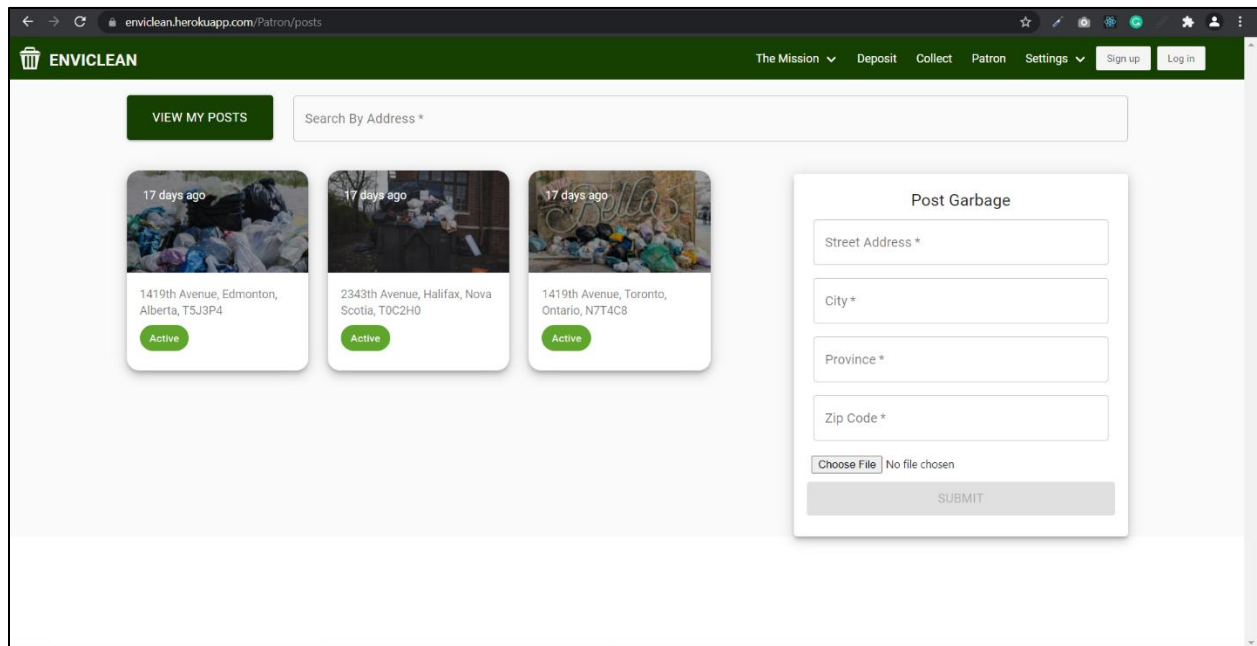


Figure 3 – Create the first post as an EnviClean Patron member page [1]

The task of enrolling registered users as EnviClean Patron members (Covered in **figure 1**, **figure 2** and **figure 3** above) is designed and developed keeping in mind that the user does not have to perform many tasks to become a Patron member in the application. Since the user is the registered user, the system does not prompt for entering the details provided during the registration in the

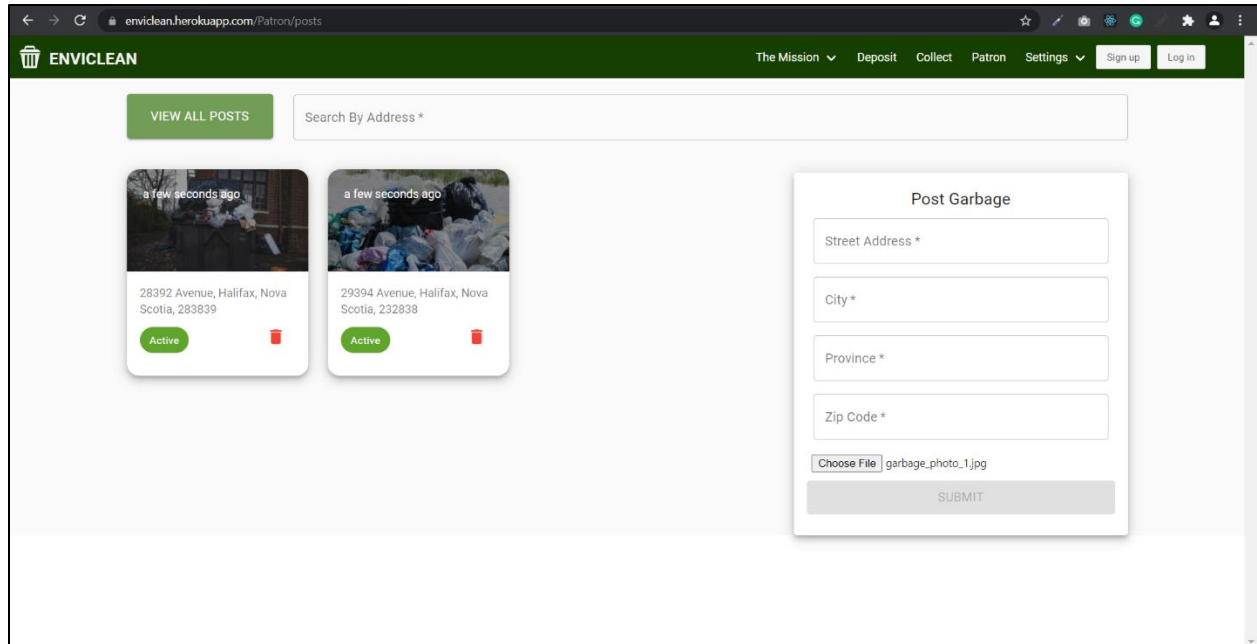
web application. This task is designed considering the usability principles. To ensure that the user's course of action is not disturbed, the terms and conditions are displayed in the dialog box on the same page rather than having them rendered on a separate web page.

**Figure 4** displays the web page that the user will be routed to when the user clicks on the “Create your first post!” button (as displayed in **figure 3**) if the user is freshly registered as a patron member or alternatively the user will be directly taken to the screen in **figure 4** from the Patron option available in navigation menu if the user is already a patron member. This page offers some of the prime features of the EnviClean application such as creating a garbage post wherein the user can create a post by entering the details where the trash is located along with a picture. Once the user enters all the required details correctly, the Submit button is enabled for the user to be able to create a garbage post. The patron member can view the **active** posts by the other users of the application on the cards displayed on the left side. Additionally, the users can also search for any particular post by entering any specifics of the address in the search bar.



*Figure 4 - View all posts page [1]*


Another task that can be performed is viewing the posts created by the patron member's account (**figure 5**). The 'VIEW MY POSTS' button on the top left side, adjacent to the search bar helps the users to view the posts created by them. The features offered here are similar to those displayed in **figure 4** with an add-on where the users can delete the active posts created by them. The delete post option is available only for the account using which the posts were created and only for active garbage collection requests.



*Figure 5 - View posts created by user page [1]*

The webpage displayed in **figures 4 and 5** covers all the major functionalities of the feature Social Clean Reporter Management - EnviClean Patron without making it look cluttered. It is designed keeping in mind the information architecture of the implemented webpage. Organizing the content in a grid-like structure makes the components appear evenly distributed and further enhances the readability of the webpage.

This feature is designed in a manner that displays the web pages on all devices evenly (i.e., 100% responsive). **Figure 6** displays the webpage on a mobile device (Moto G4). To make the webpage in **figures 4 and 5** user-friendly, I have aligned the search bar to appear on top and the "Post Garbage" creation container immediately below it.

 ENVCLEAN

VIEW MY POSTS

Search By Address \*

Post Garbage

Street Address \*

City \*

Province \*


Zip Code \*

Choose File

 No file chosen

SUBMIT

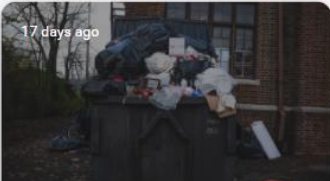
17 days ago



1419th Avenue, Edmonton, Alberta, T5J3P4

Active


17 days ago



2343th Avenue, Halifax, Nova Scotia, T0C2H0

Active

17 days ago



1419th Avenue, Toronto, Ontario, N7T4C8

Active

Figure 6 - Web page on a Moto G4 mobile device [1]



### **A2.2.2 Application Workflow**

The front-end and back-end of the EnviClean application are to be developed using the MVC (Model-View-Controller) architectural design pattern. User's interaction with the EnviClean web application is done using web browsers, accessed from various devices such as mobile phones, tablets, laptops and computers.

The front-end of the web application is the **React.js environment**. Various elements responsible for building the React.js environment are:

- **Components** - These are the various components that build the front-end of the application.
- **Routers** - They are responsible for navigating to various the webpages of the application.
- **Third-party libraries** - Libraries like Material-UI and Bulma are used for enhancing the EnviClean application to make it look elegant.
- **Services and global state management** - React's services and global state management offer flexibility to handle various states of data and trigger events based on these states throughout the application.
- **Back-end connectivity** - React supports a third-party library - Axios for integrating the front-end with the back-end.

The back-end of the web application is the **Node.js environment**. Various elements responsible for building the Node.js environment are:

- **Chrome's V8 JS engine** - It is a JavaScript engine that runs the Node.js application.
- **Express.js framework** - Express.js is built on top of Node.js and provides fundamental web application features, without hiding Node.js features.
- **Third-party libraries** - The external libraries used will enforce high-security protocols and for sending emails to the users of the application.
- **Routes (REST APIs)** - The routes for the web application are built using RESTful APIs and standards.
- **RDBMS driver** - The RDBMS driver creates a bridge between the back-end framework and the relational database.

The **RDBMS environment** hosts the relational database that will store the EnviClean data.

**Figure 7** displays the EnviClean application architecture.



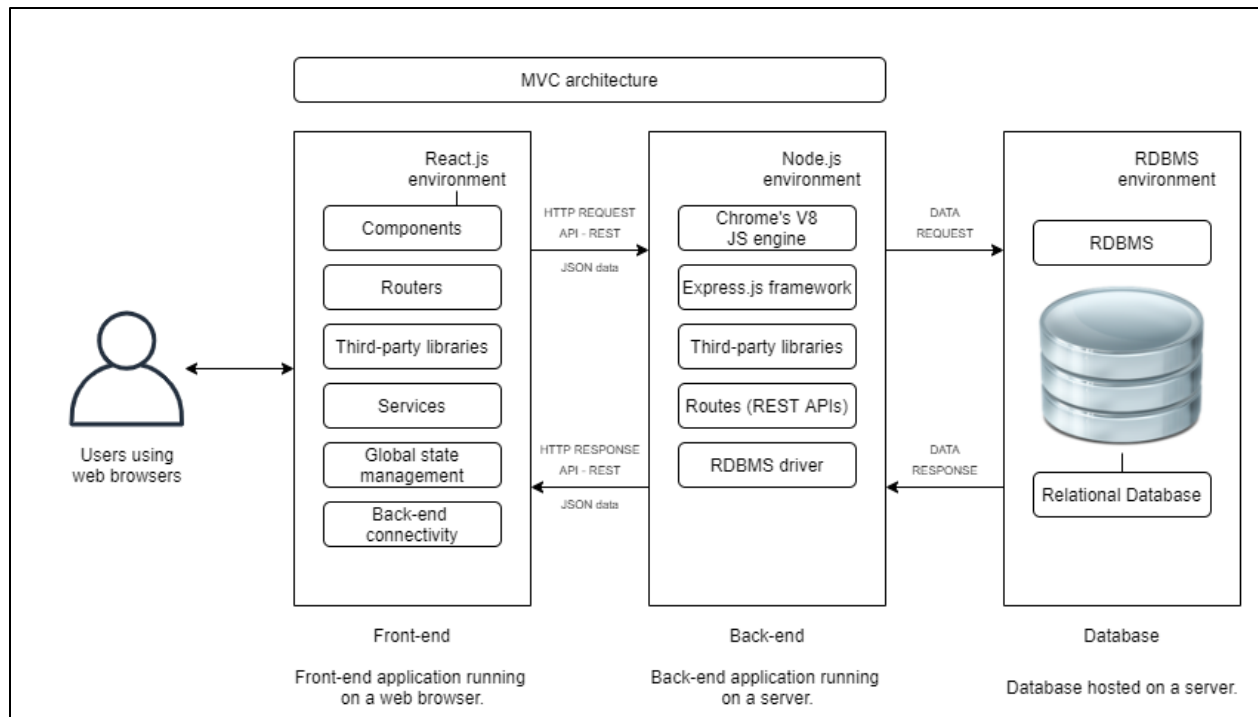


Figure 7 - EnviClean Application Architecture [2]

### ⇒ Interaction Design

Below is the list of tasks that are being developed by me as a part of **Social Clean Reporter Management – EnviClean Patron** for the EnviClean web application.

#### **1. Enroll as a social reporter and become an EnviClean Patron.**

Assuming the user is already a registered user of EnviClean.

##### Use Case

1. System displays EnviClean home page.
2. User clicks on the "EnviClean Patron" tab located on the navigation menu.
3. System displays the EnviClean patron page to the user.
4. User clicks on the "Enroll as Patron" button on the EnviClean patron page
5. System shows "Patron Terms and Conditions" dialogue to the user with two buttons, "Accept" and "Reject".
  - 5.1. User clicks on the "Reject" button.
  - 5.2. System displays "Terms and Conditions rejected" message and re-directs the user to the EnviClean patron page.
6. User clicks on the "Accept" button.
7. System redirects the user to the EnviClean patron page.
8. System displays the "Registered successfully. Create your first post" button.

##### Task Flow

**Figure 8** describes the task flow diagram of the task "Enroll as a social clean reporter and become an EnviClean Patron".

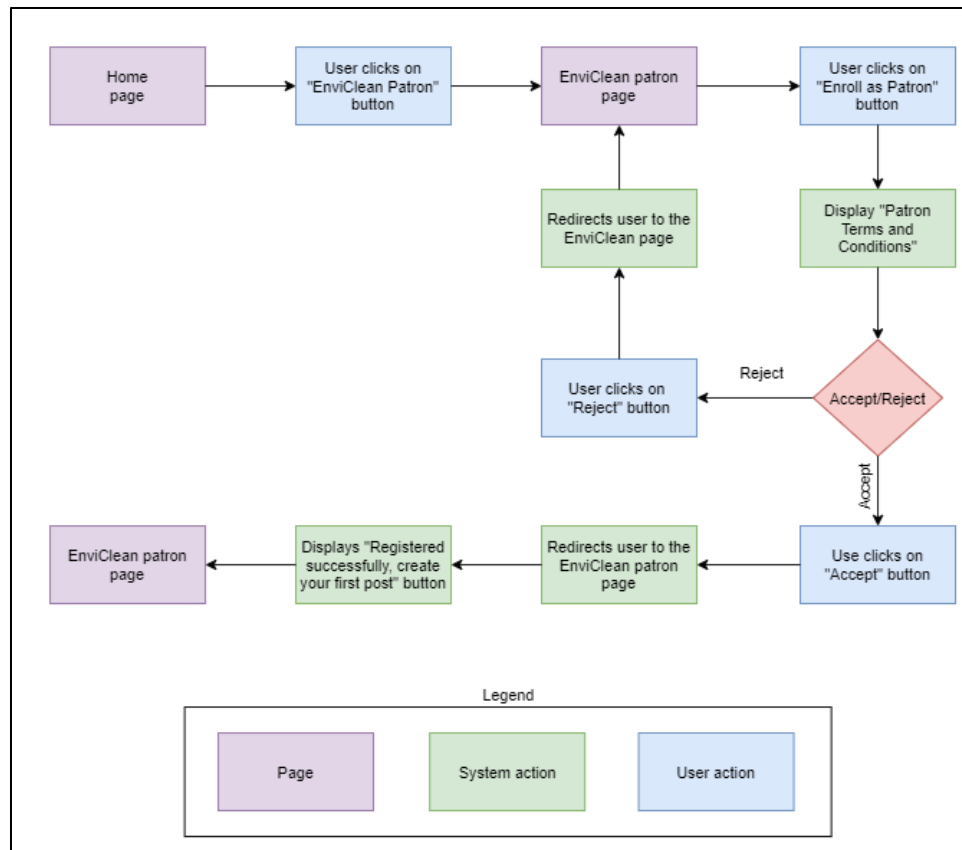


Figure 8 - Task Flow diagram of "Enroll as a social clean reporter and become an EnviClean Patron" [2]

### Click Stream

**Figure 9** describes the click-stream diagram of the task "Enroll as a social clean reporter and become an EnviClean Patron".

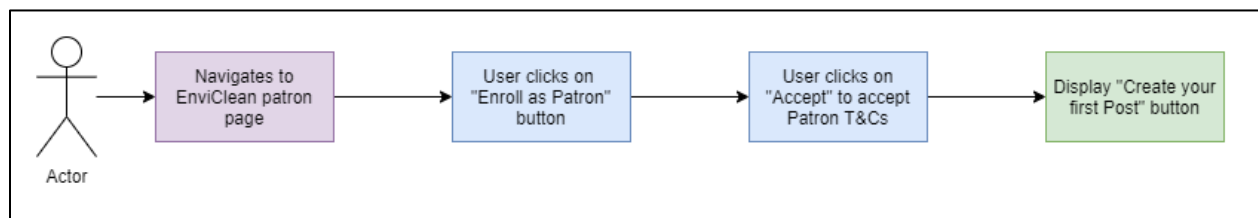


Figure 9 - Click Stream diagram of "Enroll as a social clean reporter and become an EnviClean Patron" [2]

## 2. Create a post regarding the garbage found.

Assuming the user is already a registered user of EnviClean and is already an EnviClean patron member.

### Use Case

1. System displays EnviClean home page.
2. User clicks on the "EnviClean Patron" tab located on the navigation menu.
3. System displays the EnviClean patron page to the user with all the posts created by other users.

4. System displays "Create New Post" card on the right side of the EnviClean patron page.
5. User enters the necessary information required to create a post.
  - 5.1. User enters the street address.
    - 5.1.1. System displays "Street address entered is incorrect."
    - 5.1.2. User enters the correct street address.
  - 5.2 User enters the city.
    - 5.2.1. System displays "City entered is incorrect."
    - 5.2.2. User enters the correct city.
  - 5.3 User enters province.
    - 5.3.1. System displays "Province entered is incorrect."
    - 5.3.2. User enters correct province.
  - 5.4 User enters zip code.
    - 5.4.1. System displays "Zipcode entered is incorrect."
    - 5.4.2. User enters correct zip code.
  - 5.5 User uploads image.
6. User clicks the "Create" button.
7. System displays "Post created successfully".
8. System displays all the posts with a newly created post on top.

#### Task Flow

**Figure 10** describes the task flow diagram of the task "Create a post regarding the garbage found".

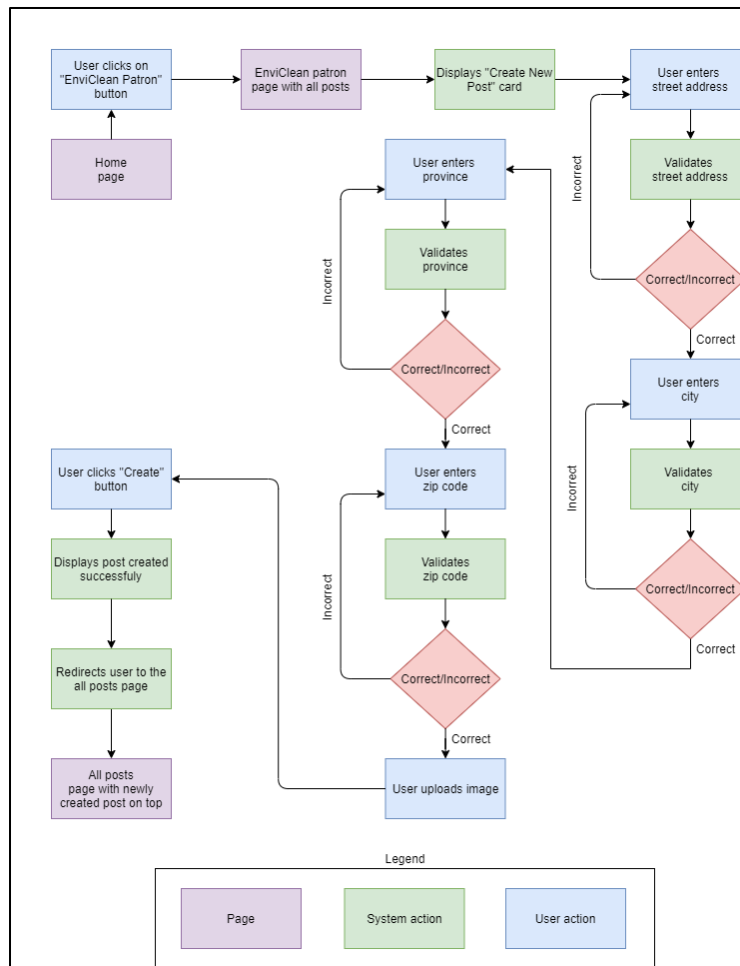


Figure 10 - Task Flow diagram of "Create a post regarding the garbage found" [2]

### Click Stream

**Figure 11** describes the click-stream diagram of the task "Create a post regarding the garbage found".

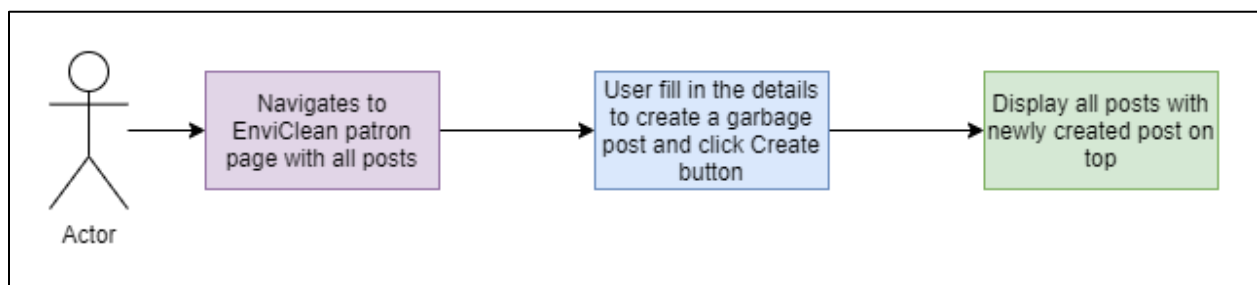


Figure 11 - Click Stream diagram of "Create a post regarding the garbage found" [2]

### 3. View posts created by other users in the vicinity.

Assuming the user is already a registered user of EnviClean and is already an EnviClean patron member.

Use Case

1. System displays EnviClean home page.
2. User clicks on the "EnviClean Patron" tab located on the navigation menu.
3. System displays the EnviClean patron page to the user with all the posts created by other users.
4. System displays a search bar where the user can enter a pin code.
5. User enters the zip code in the search bar.
  - 5.1. System displays "Zipcode entered is incorrect."
  - 5.2. User enters the correct zip code.
6. User clicks search icon.
7. System displays all the posts in the location entered by the user.

Task Flow

**Figure 12** describes the task flow diagram of the task “View posts created by other users in the vicinity”.

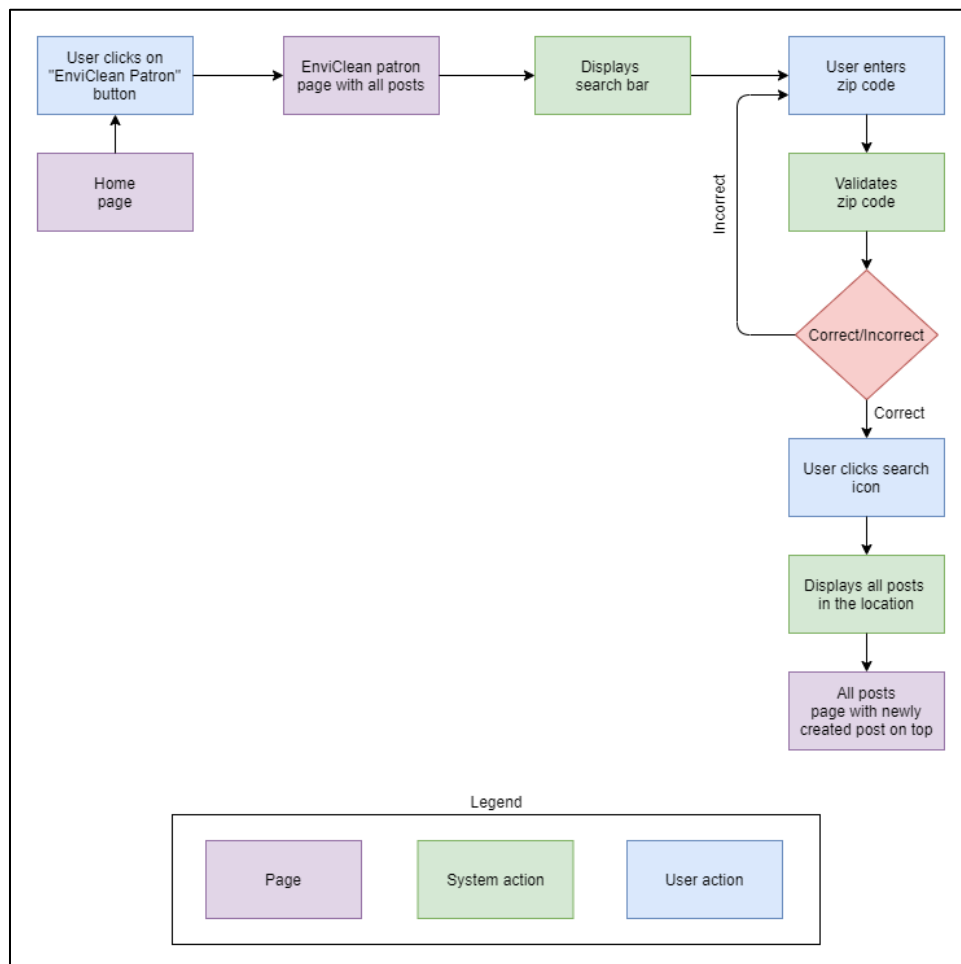
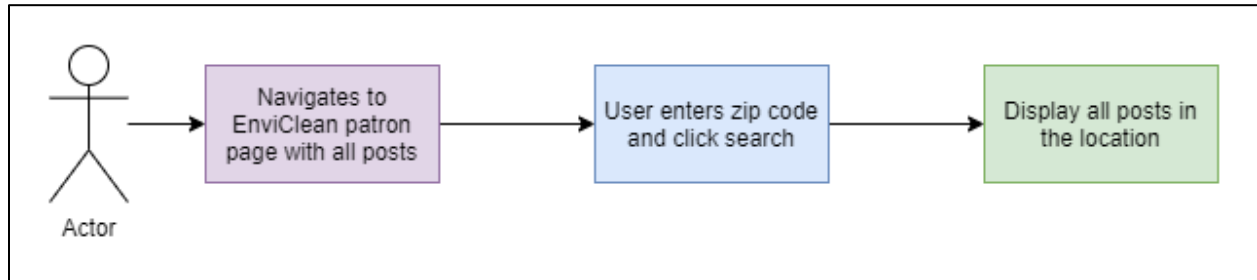


Figure 12 - Task Flow diagram of "View posts created by other users in the vicinity" [2]

Click Stream

**Figure 13** describes the click-stream diagram of the task “View posts created by other users in the vicinity”.



*Figure 13 - Click Stream diagram of "View posts created by other users in the vicinity" [2]*

#### 4. View posts created by users themselves.

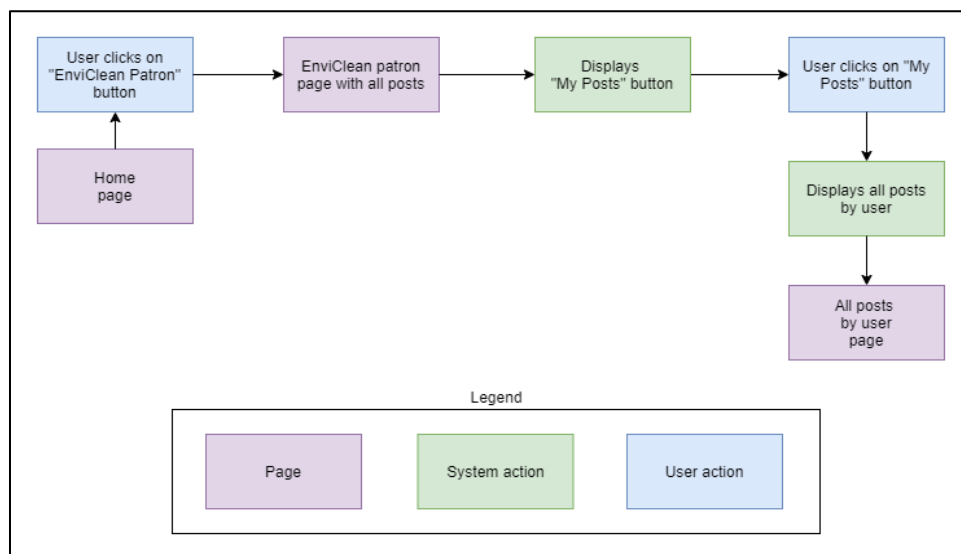
Assuming the user is already a registered user of EnviClean and is already an EnviClean patron member.

Use Case

1. Assuming Jane is already a registered user of EnviClean and EnviClean patron member.
2. System displays EnviClean home page.
3. User clicks on the "EnviClean Patron" tab located on the navigation menu.
4. System displays the EnviClean patron page to the user.
5. System displays the "My Posts" button.
6. User clicks on the "My Posts" button.
7. System displays all the posts created earlier by the user and their status.

Task Flow

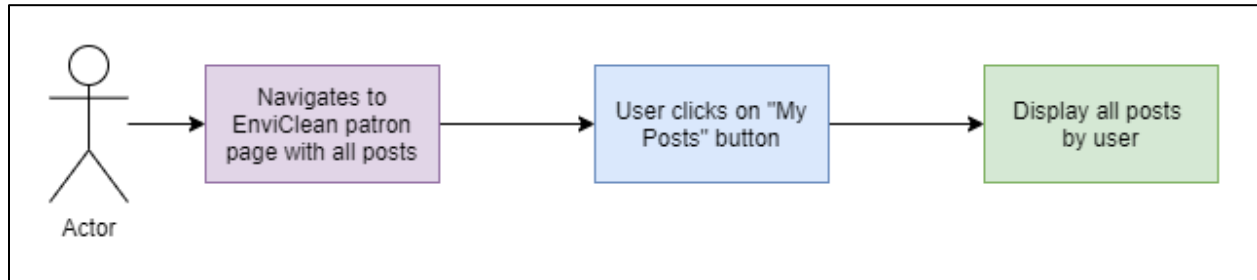
**Figure 14** describes the task flow diagram of the task “View posts created by users themselves”.



*Figure 14 - Task Flow diagram of "View posts created by users themselves" [2]*

Click Stream

**Figure 15** describes the click-stream diagram of the task “View posts created by users themselves”.



*Figure 15 - Click Stream diagram of "View posts created by users themselves" [2]*

⇒ **Process Workflow**

The Process Workflow diagram for the EnviClean web application is displayed in **Figure 16**. EnviClean application is developed using React.js library for the front-end, Node.js engine, and Express.js framework for the back-end and relational database for implementing the database component of the web application. The workflow for the feature Social Clean Reporter Management - EnviClean Patron starts with the user visiting the EnviClean web application using a web browser. To view all the created posts, the patron member will click on the "Patron" option available in the navigation menu. On click, the web application will route the patron member to the View all posts web page that will be rendered on the web browser using React components. This react-based component will render garbage posts created by all the users. When the patron member clicks on the “View all posts” button, the React-based component prepares a GET API request and sends it to the back-end router. The router then places the API request in the event queue of Node.js. The Node.js engine's single-threaded event handler checks for the requests to be handled in the event queue. The request is then picked up by the Thread pool and fulfilled by connecting to the relational database using the database driver. After the request is processed and the results are received, the controller then prepares the response (the post details in this case) and sends it across to the React-based component that displays the results to the patron members



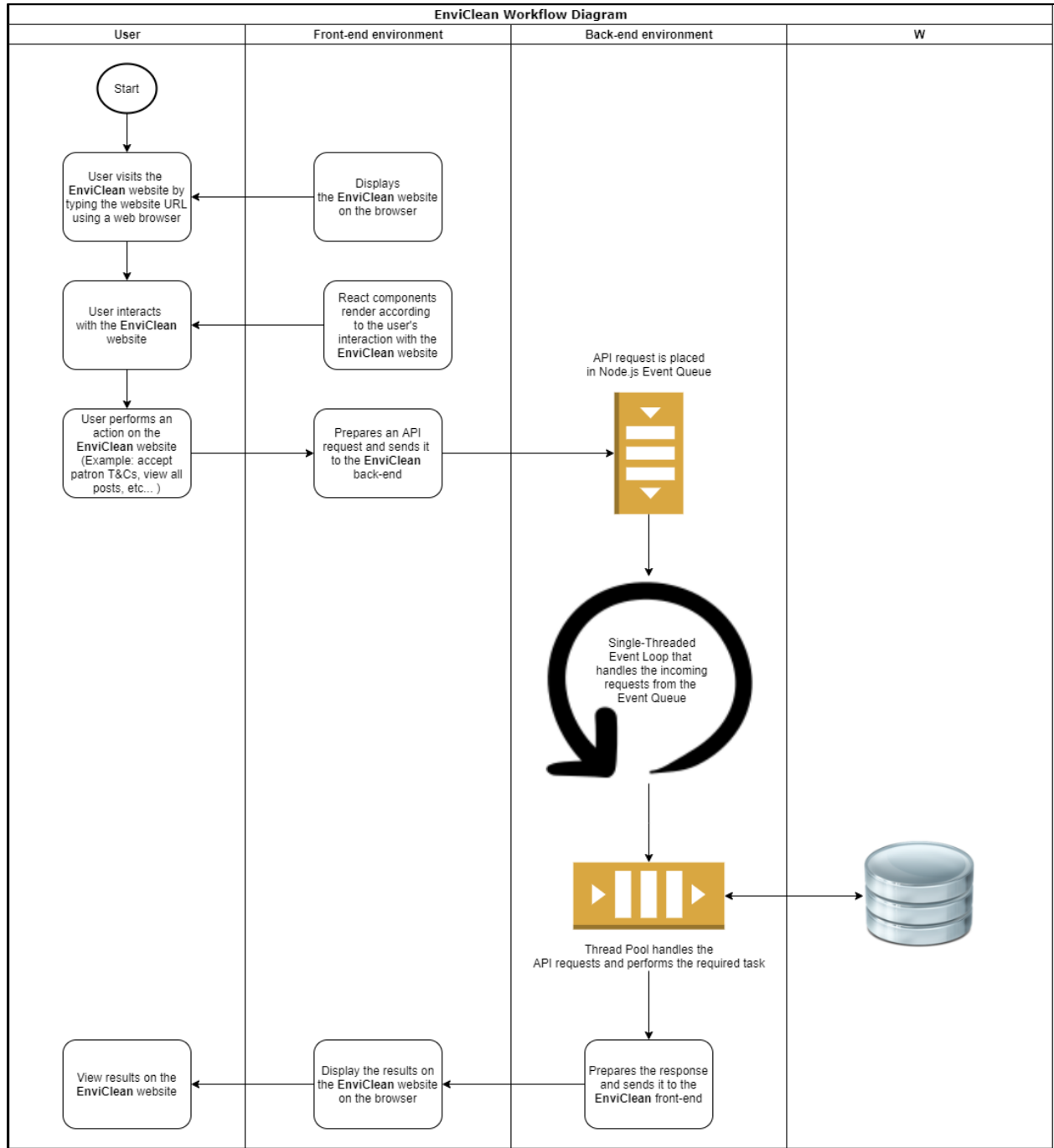


Figure 16 - EnviClean Process Workflow diagram [2]

### ⇒ Folder Structure

The folder structure of the EnviClean web application is divided based on the front-end (**client** folder) and the back-end(**server** folder).

**Figure 17** displays the EnviClean front-end folder structure. The application's front-end entry point is the "index.js" file. The "index.css" file holds the global application styling. The ".env" file store all the crucial information such as API keys. The "package.json" file contains information related to the project(e.g., name, version, repository name), external dependencies used, run scripts (e.g., start and test). The autogenerated "package-lock.json" file stores the information related to external dependencies used, and the "node\_modules" folder stores all the external code. The ".gitignore" file ensures that the API keys and external code are not uploaded to the remote repository. The "src" folder contains logic for the EnviClean web application. All components are stored in the "components" folder and back-end APIs are stored in the "api" folder. The logic for the global state management resides in "components", "actions" and "reducers" folders. Lastly, the "images" folder contains all the images used by EnviClean web application.

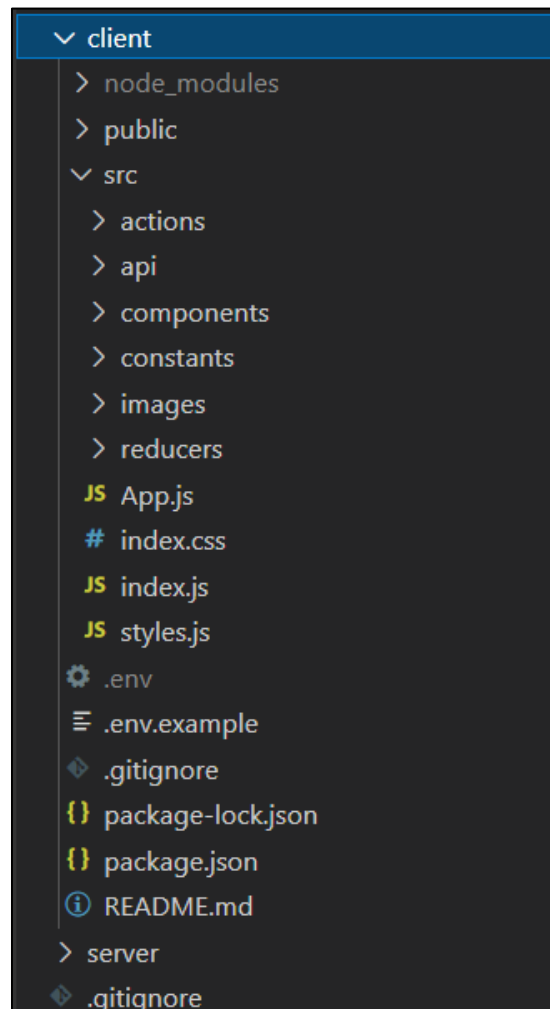


Figure 17 - Front-end folder structure (client folder)

**Figure 18** displays the EnviClean back-end folder structure. The application's back-end entry point is the "index.js" file. The ".env" file store all the crucial information such as database credentials and API keys. The "package.json" file contains information related to the application's back-end(e.g., name, version, repository name), external dependencies used, run scripts (e.g., start and test). The autogenerated "package-lock.json" file stores the information related to external dependencies used, and the "node\_modules" folder stores all the external code. The ".gitignore" file ensures that the ".env" file and "node\_modules" folder is not uploaded on the remote repository. The "routes" folder as the name suggests stores all the routes, and the logic for the routes is in the "controllers" folder. The "middlewares" folder contains the logic for the application's security and user's authentication. The "models" holds the application's data model.

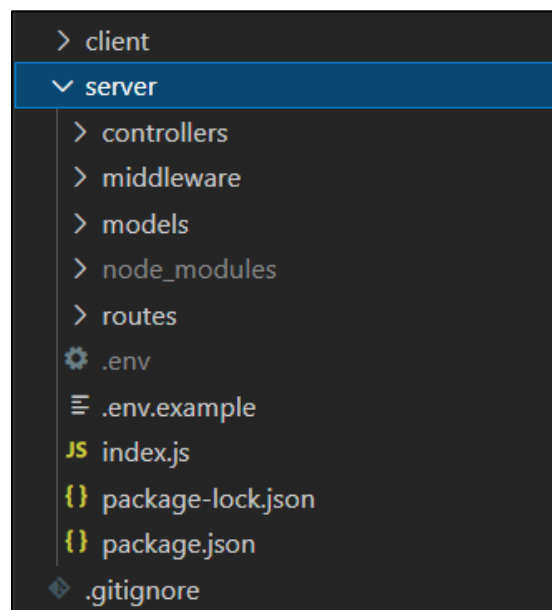


Figure 18 - Back-end folder structure (server folder)

## **References**

- [1] EnviClean, "EnviClean," D1\_Group6\_S21, [Online]. Available: <https://enviclean.herokuapp.com/>. [Accessed 01 July 2021].
- [2] Diagrams.net, "Diagram Software and Flowchart Maker," [Online]. Available: <https://app.diagrams.net/>. [Accessed 01 July 2021].