Web Programming Project: Phase 1
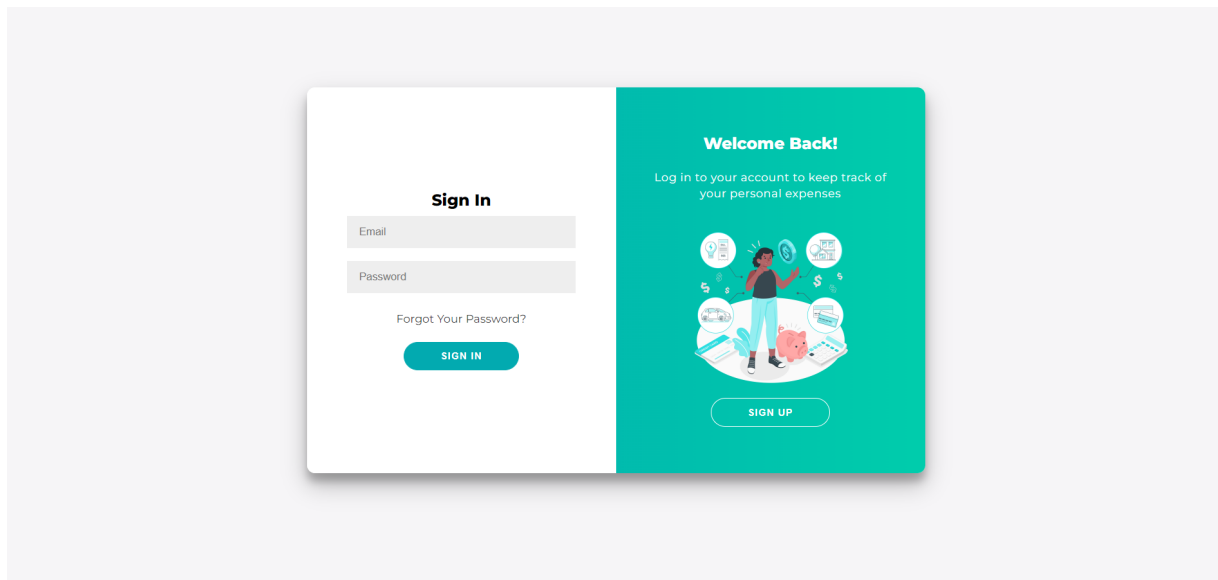
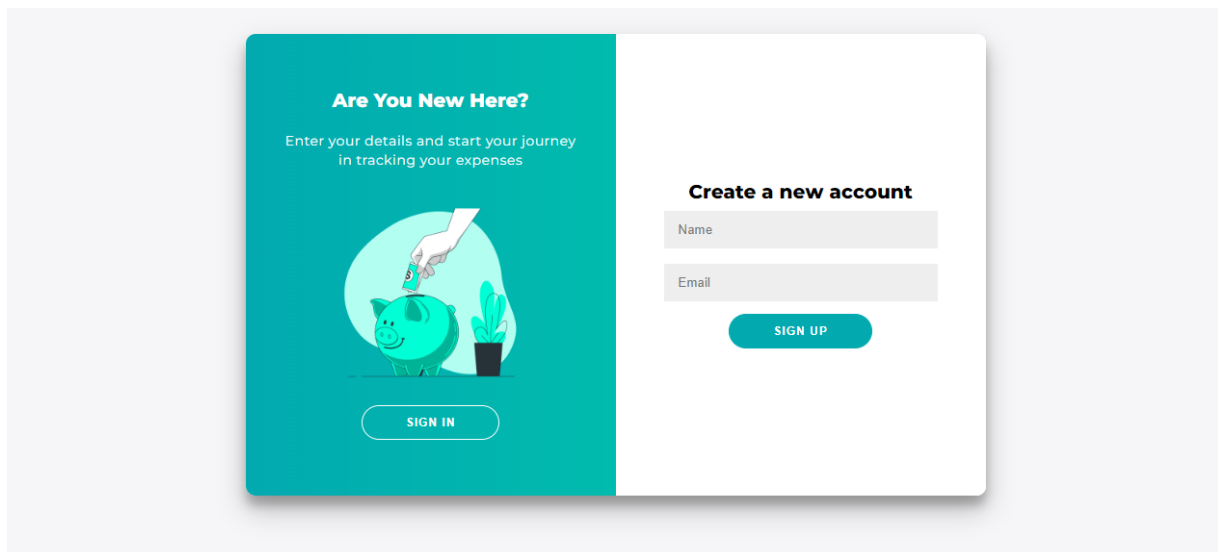**Repository Link:** https://github.com/Pavithra-Rajan/Finance-Management-System-UI

Hosted UI pages:

1. Login Page
2. Dashboard

UI designs
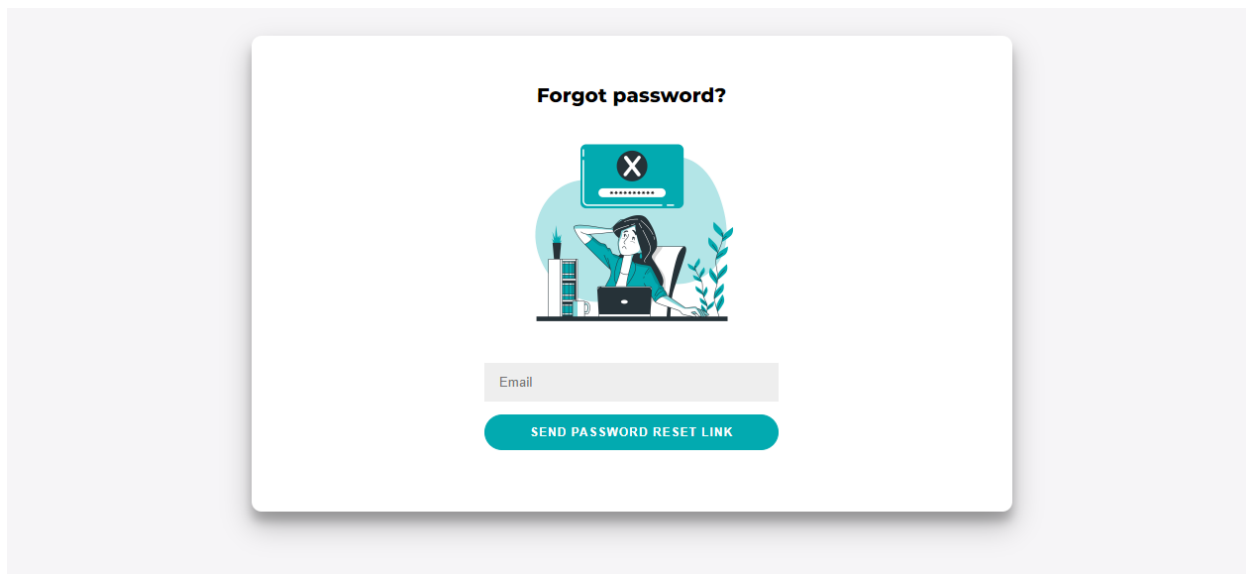
1. Sign-In

2. Sign Up
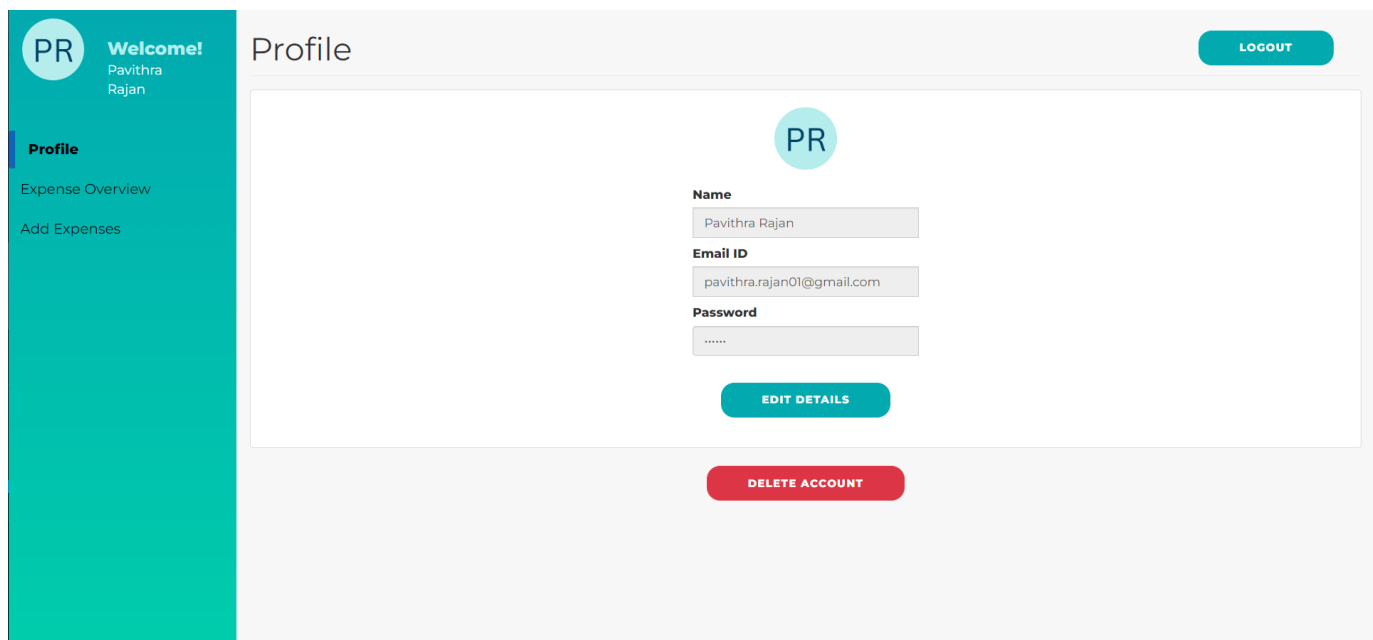
3. Forgot Password

4. Change/Edit User Details

Click on the edit details button, and the Name and Password fields will be displayed in editable input text boxes. The user can make the desired changes and click on save details to proceed.

Profile

PR

Name

Pavithra Rajan

Email ID

pavithra.rajan01@gmail.com

Password

......

SAVE DETAILS

DELETE ACCOUNT

5. Log out

    The user can log out from the system by clicking on the 'logout' button, which can be found on the top-right corner of the profile page.

6. Add expenses

PR **Welcome!**
    Pavithra
    Rajan

Profile

Expense Overview

Add Expenses

Add expenses

Date:

06/10/2022

Spent on:

Food

Amount in Rs.:

Amount

Remarks (if any):

Details on where the money was spent

SUBMIT

7. View expenses

    The user can view comprehensive charts to analyse the areas that they have spent. A table containing the date, tag, amount and remarks (if any) can be found below these charts.

8. Date range filter view

    The user can choose the 'from' and 'to' dates to filter the expenses. This can be found in the top-right corner of the expense overview page.
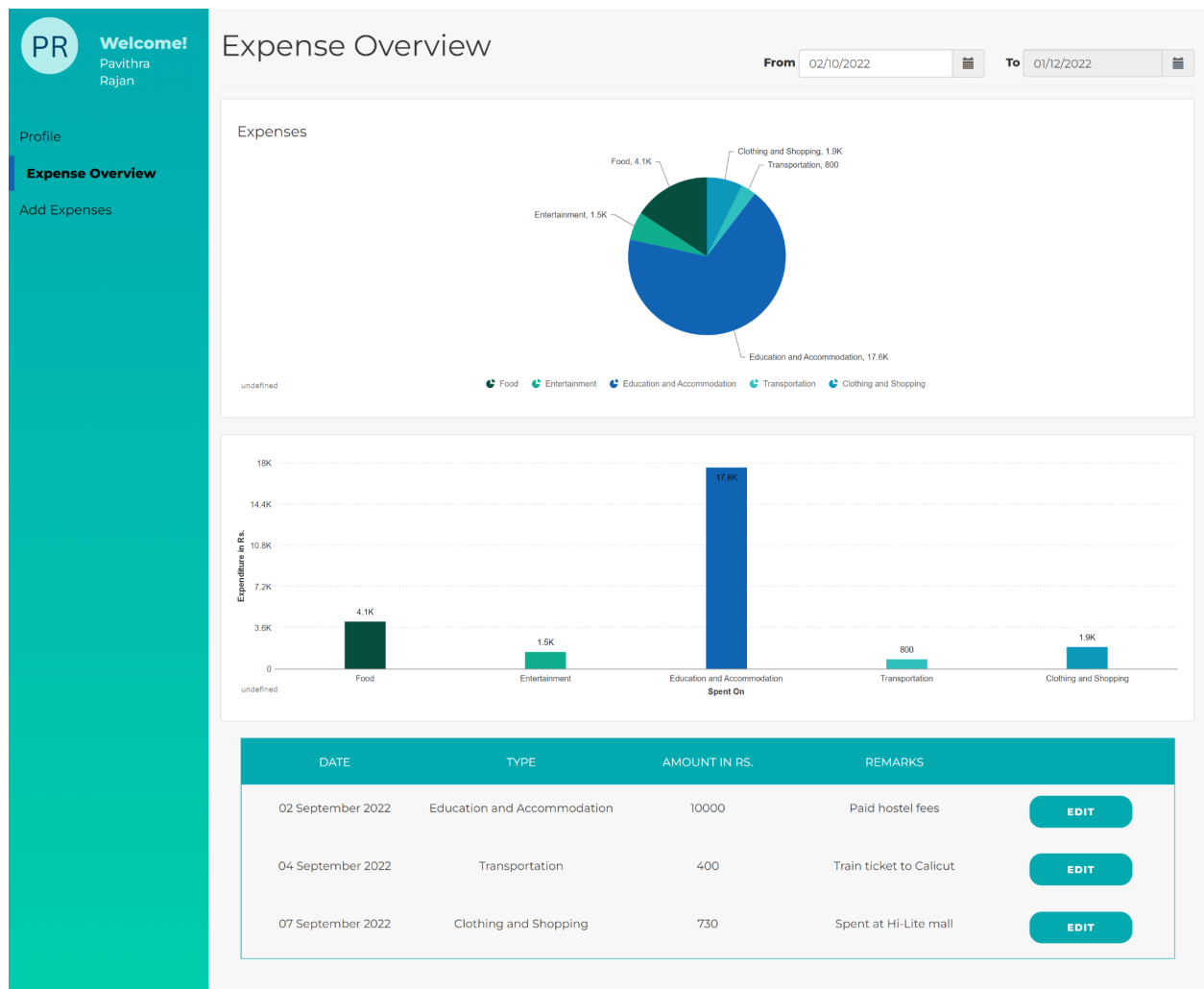
9. Edit or delete expenses

The user can click on the 'edit' button which is found in the table containing all the expenses. A modal will pop up and the user can edit the required details. The user can also delete the expense if needed.
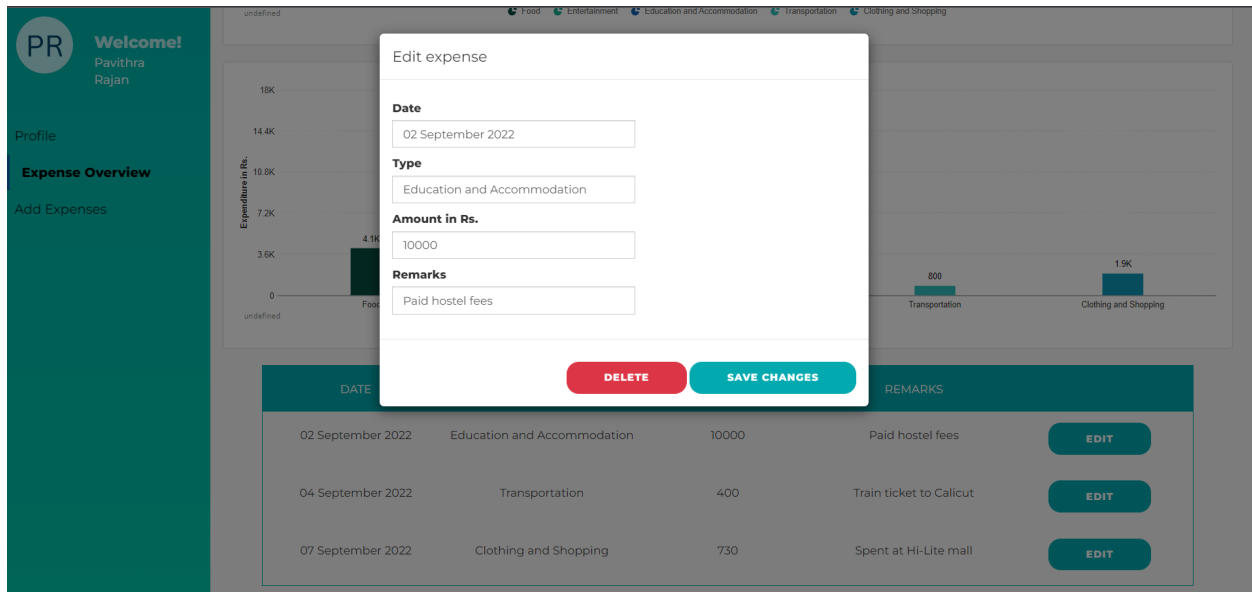
10. Sort expenses

The user can click on the table header (date, type or amount), and the table will be sorted in 'ascending' order based on the column clicked by the user. The table can be sorted in 'descending' order if the user clicks on the column header once again.

Use Cases

There is only one actor for the system and that is the User.

1. ## <u>Signup</u>

Purpose: Allows a user to register in the application
Priority: High
Preconditions: User must not have registered before
Postconditions: User will be successfully registered in the application and will be directed to login page
Actors: User
Extends:
1. Account with an entered email address already exists.
2. Incorrect email address.

Flow of Events:

Basic Flow:
1. Application asks for the user's email address and name.
2. The user enters the required details and submits the form.
3. User receives password on registered email address.
4. The user is directed to the login page.

Alternative Flow:
1. The user doesn't receive a password due to an incorrect email address.
2. The user has to sign up again.
3. If an account already exists with the entered email address, it alerts the user.

2. ## <u>Login</u>

Purpose: Allows users to get access to the application by logging in
Priority: High
Preconditions: The user must have registered in the application before.
Postconditions: The user will be directed to the dashboard and will be able to access the functionalities
Actors: User
Extends:
1. Entered password is incorrect
2. User has not registered in the application

Flow of Events:

Basic Flow:
1. Application requests for user details: email address and password
2. The user enters the required details and submits the form.
3. After successful password validation, the user is redirected to the dashboard.

Exceptions:
1. If the password entered is incorrect, then the user is asked to enter the password again
2. If the entered email address is not registered, then the user is asked to register (sign up) in the application.

3. Forgot Password

Purpose: Allows the user to get a new password
Priority: Medium
Preconditions: User must have registered in the application before.
Postconditions: User will be receiving a new password to their registered email.
Actors: User
Extends:
1. The user has not registered in the application

Flow of Events:
 Basic Flow:
1. Application asks for registered email address
2. The user enters the registered email address
3. Users receive a new password on their registered email address.
 Exceptions:
1. If the entered email address is not registered, then it shows a message showing the user to register.

4. Logout

Purpose: Allows the user to logout of the application
Priority: Medium
Preconditions: User must be logged into the application
Postconditions: User will be logged out
Actors: User
Extends: None
Flow of Events:
 Basic Flow:
1. User clicks on logout button
2. The user is redirected to the login page.

5. Delete Account

Purpose: Allows the user to delete account
Priority: Medium
Preconditions: User must be logged into the application
Postconditions: User account and details will be deleted from the database
Actors: User
Extends: Entered password is incorrect
Flow of Events:

Basic Flow:

1. User clicks on the delete button present in the profile page.
2. The application asks the user, 'Are you sure?'. The user clicks on 'Yes'.
3. The user is redirected to the login page.

6. Edit User Details

Purpose: Allows the user to edit their details.
Priority: Medium
Preconditions: The user must be logged into the application.
Postconditions: User details are updated.
Actors: User
Extends: View Profile
Flow of Events:
    Basic Flow:

1. The user clicks on the edit details button present on the profile page.
2. The user makes the desired changes.
3. User clicks the save button, and the details are updated.

    Exceptions:

1. The user edits the password and the new password is less than 6 characters.
2. The system alerts the user 'The password is too short'.

7. Adding expenses

Purpose: Allows the user to add expenses
Priority: High
Preconditions: User must be logged into the application
Postconditions: A new expense is added to the user account.
Actors: User
Extends: None
Flow of Events:
    Basic Flow:

1. User clicks on "Add Expense".
2. User fills in the details.
3. User clicks on "Add".

8. View expense

Purpose: Allows the user to view details of an expense
Priority: High
Preconditions:

1. Users must be logged into the application.
2. The expense to view should exist in the system.

Postconditions: Expense details are shown to the user.
Actors: User
Extends: None
Flow of Events:
    Basic Flow:

1. User clicks on the particular expense.
2. The details are shown to the user.

9. Edit expenses

Purpose: Allows the user to edit expenses
Priority: High
Preconditions:

1. The user must be logged into the application
2. The expense should exist in the system.

Postconditions: The expense details are updated.
Actors: User
Extends: View Expense
Flow of Events:
    Basic Flow:

1. User clicks on "Edit Expense".
2. User fills in the details to be updated.
3. User clicks on "Update".

10. Sort and filter expense

Purpose: Allows users to view expenses sorted and filtered by certain order and tags..
Priority: High
Preconditions: The user must be logged into the application
Postconditions: The expenses are shown after being sorted in the given order and filtered by given categories.
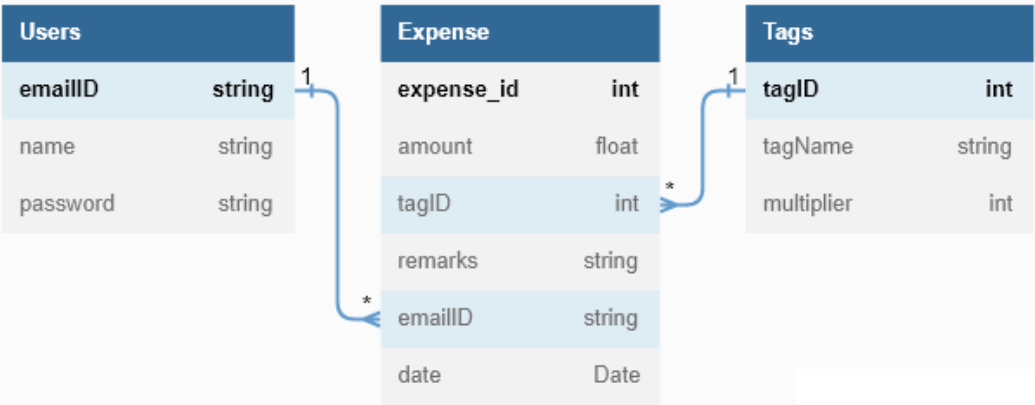**Actors:** User
Extends: None
Flow of Events:
    Basic Flow:

1. Click on the table header to sort the given column. In order to filter a particular expense, click on the filter icon.
2. Select the required filters.
3. Click on "OK".

Database Schema

Use Case Diagram