

## Project Documentation

**Project Name:** EPharmacy

### **Project Description:**

EPharmacy is a web-based application which provides a user-friendly platform for users to search and buy medicines online. It uses microservice architecture in which multiple microservices work together to give seamless experience to users from registration in the application to tracking their orders in real time. The backend of the application is built using Spring Framework whereas the frontend is built using Angular.

### **Major Modules:**

- User Registration and Login
- Medicine Catalogue Management
- Shopping Cart Management
- Payment Gateway Integration
- Order Tracking and Management

### **Architecture:**

This application uses Microservice Architecture to make it a robust, scalable and easy to maintain application. Each microservice is responsible for individual task and different microservices communicate with each other using REST APIs. The following microservices will be implemented:

- **Customer Microservice:** This microservice will be responsible for user authentication and managing user profile. It will provide APIs for user registration, authentication and updating user profile.
- **Medicine Microservice:** This microservice will manage the inventory of medicines. It will provide APIs for retrieving medicine information and updating stock levels.
- **Cart Microservice:** This microservice will manage the shopping cart of each user. It will provide APIs for adding, removing and updating items in the cart.
- **Payment Microservice:** This microservice will handle the payment process. It will integrate with a payment gateway to allow users to make payments for their orders.

- **Order Microservice:** This microservice will handle order placing, tracking and management. It will provide APIs for placing new orders, cancelling orders and retrieving order information.
- **Gateway Service:**
  - Acts as the entry point for all requests to the microservices
  - Uses Spring Cloud Gateway to handle routing, load balancing, and traffic management
  - Uses Spring Cloud Consul for centralized configuration, service discovery and registration

### **Technology Stack:**

- **Backend:** Spring Boot, Spring Data, Spring REST, Spring Cloud Consul
- **Frontend:** Angular, HTML5, CSS3, Bootstrap
- **Database:** MySQL

### **Non-Functional Requirements:**

- **Performance:** The application should be fast and responsive, with quick load times and minimal lag or delay.
- **Scalability:** The application should be able to handle large amounts of traffic and scale as needed to accommodate growth.
- **Security:** The application should be secure and protect user data from unauthorized access, with measures such as encryption and secure authentication.
- **Reliability:** The application should be reliable and available, with minimal downtime or outages.
- **Usability:** The application should be easy to use and intuitive, with a clear and consistent user interface.
- **Maintainability:** The application should be easy to maintain and update, with clear and well-organized code that is easy to understand and modify.
- **Compatibility:** The application should be compatible with a wide range of devices and browsers, with support for different screen sizes and resolutions.

### **Final Deliverables:**

- Application archive ( .jar ) with source code
- Database DDL Script
- Complete Source code

- Sample screenshots of important screens

## **User Stories:**

### **SET 01: Visitor**

#### **US 01: View Medicines**

As a visitor, I want to see the list of available medicines so that I can see the details like medicine name, manufacturers name, price, discount, image etc.

Acceptance criteria:

- The visitor should be able to see a list of all available medicines on the website/application.
- Display the list of available medicines in pagination format.
- The list should include the following information for each medicine: medicine name, manufacturer's name, price, discount, and an image.
- The medicines expiring in next 3 months should be displayed with 30% discount and the medicines expiring in next 6 months should be displayed with 20% discount.
- It should have Next page/Previous page option to navigate through the pages
- The visitor should be able to sort the medicines by any of the available criteria (e.g., by name, price, etc.).
- When the user clicks on any medicine, the medicine details should be displayed and if he tries to Add them to the cart without login, it should ask him to login to the application.

#### **US 02: Categorize Medicines**

As a visitor, I want to see the medicine categories, so that I can see the available medicines category-wise.

Acceptance criteria:

- Display the category of the medicines on home page. (categories like Covid Essentials, Diabetes, Ayush, Ayurvedic, Homeopathy etc.)

#### **US 03: Search Medicines**

As a visitor, I want to search for the medicines by providing medicine name as input, so that I can see the details of the required medicines.

Acceptance criteria:

- The search bar should be prominently displayed on the homepage.
- The user should be able to search for medicines by medicine name.
- The user should get suggestions as he types medicines name in search bar
- The search results should be displayed in a clear and organized manner, showing the name, price, and discount of each medicine.
- Display the list of medicines in pagination format
- Display proper error message in case medicine is not available

#### **US 04: Register**

As a visitor, I should be able to register to the application by providing the details like name, email, contact number, etc. so that I can get the benefits of a registered user.

Acceptance criteria:

- Validate the details entered by the user like Name, email, gender, date of birth ,mobile number, password
  - Name should only contain alphabets with single space between the words.
  - Email should be a valid email address with domains like ( in or com).
  - Only above 18 years old users should be allowed to register. Verify it using date of birth field.
  - Mobile number should start with 6,7,8 or 9 and should be of 10 digits.
  - Password should contain at least one uppercase, one lowercase, one digit, one special character, should be of minimum 7 character and maximum of 20 characters.
  - Password should be hashed to store in database.
- Address details include
  - Address name like Home or Work

- Address line1(house number/apartment number etc.), , Address line 2(street, landmark etc.)
- Area, City, State (state should be a dropdown), Pin code(6 digits).
- Only new users are allowed for the registration
- Display suitable success message after the registration process gets completed
- Display proper error message in case of any error or exception

## **US 05: Login**

As a registered user, I should be able to login into the application by providing email and password so that I can get the benefits of registered user

Acceptance criteria:

- Validate the credentials provided by the user.
- If 3 failed attempts are made within 30 seconds, the Customer Microservice circuit should go in open state for 1 minute.( *Implement it using Resilience4j*).
- Display a running timer of 1 minute while circuit is in open state also display proper message if user attempts to login when circuit is in open state.
- Customer should be able to reset password if he forgets his password linked to his email. Verify his mobile number for resetting password.
- Display proper error messages if the credentials are wrong.
- After successful login, the user is redirected to the home page.

## **US 06: Logout**

As a logged-in user, I should be able to log out of the application so that I can finish my session.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.
- The logout functionality should be available to the customer on every page or screen of the application.
- Clicking on the "Logout" button should immediately terminate the customer's session and clear their credentials.
- The customer should be redirected to the home page after logging out.

- Once logged out, the customer should not be able to access any pages or screens that require authentication until they log in again.

## **SET 02: Customer**

### **US 07: View and Update Profile**

As a customer, I should be able to view and update my profile details so that my profile is up to date.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.
- The customer should be able to view their profile details, including their name, email address, phone number, and date of birth.
- There are two types of customers: Regular and Prime. Initially every customer is registered as Regular. At any time, customers can upgrade themselves to Prime by choosing one of three plans (Monthly for Rs. 199, Quarterly for Rs. 399 and Yearly for Rs. 699).
- Benefit for Prime member: In addition to the normal discounts a prime member gets health coins equal to 10% of Total Order Value on every payment using debit card and gets coins equal to 15% of Total Order Value on every payment using credit card. Health coins(1 coin = INR 1) can be used for payment of any future order.
- The customer should be able to update their profile details as needed, including changing their name, email address, phone number.
- The system should validate any updates made to the profile details, ensuring that the customer provides valid and complete information.
- The updated profile details should be saved in the system and displayed to the customer immediately after the changes are made.
- The system should provide error messages if the customer enters invalid or incomplete information while updating their profile details.
- View Profile page should also provide a button to change password linked to customer's email id.

### **US 08: Change Password**

As a user, I want to change my password to ensure the security of my account.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.
- The customer should be able to access the view-profile page which allows them to change their password.
- The user should be required to enter their current password to confirm their identity before changing their password.
- The user should be able to enter a new password that meets any requirements specified by the application.
- The user should be required to confirm the new password by entering it twice to avoid errors.
- **The new password should not be from the last three passwords which customer already had previously.**
- The password change should be saved and stored securely by the system.
- The system should confirm to the user that their password has been changed successfully.
- The user should be logged out of their account after changing their password.
- The user should be able to login to their account using their new password.

## US 09: Add to Cart

As a registered customer, I should be able to add the medicines to the cart so that they can be purchased.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.
- The customer should be able to add medicines to their cart by clicking on an "Add to Cart" button from the medicines details page or search result page.

- Verify if the same item is already present in the cart. Display proper message.
- Quantity should also be mentioned while adding items to the cart.
- The customer should be able to view the contents of their cart at any time.
- The customer should be able to view the total cost of the medicines in their cart
- The system should save the customer's cart between sessions, allowing them to return to their cart and complete their purchase later if they choose.
- The customer should be able to proceed to checkout from their cart, or continue shopping if they choose.

### **US 10: Update Quantity**

As a registered customer, I should be able to change the quantity of the items in the cart. So that order for the medicines with correct quantity can be placed.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.
- The customer should be able to modify the quantity of each item in their cart or remove items from the cart entirely.
- The customer should be able to increase/decrease the medicine quantity through +/- button.
- A proper message should be displayed on updating the quantity.
- The subtotal should also be updated based on the quantity updated.
- Display proper error message in case of any error or exception

### **US 11: Delete Medicine**

As a registered customer, I should be able to delete the medicines in a cart so that medicines that are no longer required can be removed.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.



- The customer is allowed to choose and delete the medicine from his cart or delete all medicines at once from the cart.
- Customer should be prompted to confirm the delete operation before the delete operation is performed.
- On successful deletion, a successful message should be displayed to the customer.
- Display proper error message in case of any error or exception

### **US 12: View cart**

As a customer, I should be able to view my cart, so that I can get to know the medicines that I am purchasing.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.
- All the medicines added to the cart by the logged-in customer should be displayed.
- Details include the image, manufacturers name, medicine name, price, discount, manufacturing date and expiry date.
- For every medicine, a Remove button should be displayed, and Remove All medicines button should also be displayed at the top of the cart.
- There should be a button to proceed for buying the items in the cart.
- Display proper messages in case of the cart is empty

### **US 13: Place Order**

As a customer, I should be able to place order for the medicines that are added to my cart.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.
- For placing order, customer should be prompted to choose a shipping address from already added addresses and there should be an option to add a new address also.
- For making payment, list of already added Debit/Credit cards should be displayed to choose from and there should be an option to add a new card as well.

- On selecting the particular card, customer should be prompted to enter CVV of the chosen card. CVV should be validated against card to make payment.
- If customer enters invalid CVV 3 times within 30 seconds, then the Payment Microservice circuit should go in open state for 1 minute. *(Implement this using Resiliense4j).*
- *Order Microservice should make the rest call to Payment Microservice to make payment.*
- *There should be two instances of Payment Microservice running to dynamically load balance the requests coming from Order Microservice in round-robin fashion.*
- If the order is successfully placed then, a proper success message should be displayed with expected date of delivery.
- There should be a button to display the Order Summary giving the details of the order just placed.
- Customer should be given discount based on total order value as follows:
  - If the order value is between Rs. 1000 and 2000 then 10% discount.
  - If the order value is between Rs. 2000 and 3000 then 20% discount.
  - If the order value is above Rs. 3000 then 30% discount.
  - Bill calculation should be clearly depicted in Order Summary mentioning usage of health coins if any.
- Display a proper message in case of errors or exceptions

## **US 14: View Orders**

As a customer, I should be able to view all the orders placed by me, so that I can see the status of my orders.

Acceptance criteria:

- Only logged-in customers should be provided with this feature.
- The customer should be able to view a list of all their previous orders.

- Each order should display relevant information such as order id, date of purchase, medicines purchased, and order status, delivery status, delivery address and payment information
- The order history should be easily accessible from the customer's profile "My Orders" section.
- The Order Status and Delivery Status should change in real time as below:
  - Order Status should be 'PROCESSING' from the time of placing order till next **30 minutes**. After 30 minutes Order Status should change to 'CONFIRMED'.
  - When Order Status is 'CONFIRMED' then Delivery Status should change to 'IN\_TRANSIT' from 'AWAITING\_CONFIRMATION'.
  - After exactly **one day** being in 'IN\_TRANSIT' Delivery Status should change to 'OUT\_FOR\_DELIVERY' and after next **5 hours** Delivery Status should change to 'DELIVERED' and Order Status should change to 'COMPLETED'.
  - When the Order Status is 'PROCESSING' then only order can be cancelled and both Order Status and Delivery Status should change to 'CANCELLED'.

### **API Documentation:**

**API Name:** CustomerAPI

**Description:** This API is responsible for user registration, login and profile updating.

**Base URL:** /customer-api

#### **Endpoints:**

##### **1. Register**

**Endpoint:** POST /customer/register

**Description:** Create a new customer account.

Request Body:

```
{
  "customerName": "string",
  "customerEmailId": " string ",
  "contactNumber": " string ",
  "password": " string ",
  "gender": " string ",
  "dateOfBirth": LocalDate
  "address":
    {
      "addressName": " string ",
      "addressLine1": " string ",
      "addressLine2": " string ",
      "area": " string ",
      "city": " string ",
      "state": " string ",
      "pincode": " string "
    }
}
```

Responses:

- 200 Created: The customer account was successfully created.
- 400 Bad Request: A customer with the given email address already exists/invalid data

## 2. Login

**Endpoint : POST /customer/login**

**Description:** Authenticate the customer

Request Body:

```
{
  "customerEmailId": "string",
  "password": "string"
}
```

Responses:

- 200 OK: The customer authentication successful.
- 400 Bad Request: The email address or password is incorrect.

### 3. View Profile

**Endpoint:** GET /customer/{customerId}

**Description:** Fetch the customer profile

Path Variable : customer Id

Responses:

- 200 OK: Returns customer profile in the below format.

```
{
  "customerName": "string",
  "customerEmailId": " string ",
  "contactNumber": " string ",
  "password": " string ",
  "gender": " string ",
  "dateOfBirth": LocalDate
  "addressList": [
    {
      "addressName": " string ",
      "addressLine1": " string ",
      "addressLine2": " string ",
      "area": " string ",
      "city": " string ",
      "state": " string ",
      "pincode": " string "
    }
  ],
  "plan": {
    "planId":integer,
    "planName": "string",
    "planDescription": "string"
  },
  "planExpiryDate": LocalDate,
  "healthCoins": Integer
}
```

}

- 400 Bad Request: The customer with given id is not found.

#### 4. Update Profile

**Endpoint :** PUT /customer/update-profile

**Description:** Update the customer profile

Request Body:

```
{
    "customerId" : integer
    "customerName": "string",
    "customerEmailId": "string",
    "contactNumber": "string"
}
```

Responses:

- 200 OK: Customer profile updated successfully
- 400 Bad Request: The customer with given id is not found.

#### 5. Fetch Customer addresses

**Endpoint:** GET /customer/view-addresses/{customerId}

**Description:** Fetch all the addresses added by customer.

Path Variables: customerId

Responses:

- 200 OK: Fetch all the addresses successfully.
- 400 Bad Request: Invalid customerId/ No address found.

#### 6. Add Address

**Endpoint: POST /customer/add-address/{customerId}**

**Description:** Add the address for delivery.

Path Variables : customerId

Request Body

```
{
  "addressName" : "string",
  "addressLine1": "string",
  "addressLine2" : "string",
  "area" : "string",
  "state" : "string",
  "pincode": "string"
}
```

Responses:

- 201 CREATED: Address added successfully.
- 400 Bad Request: Invalid customerId.

## 7. Upgrade Customer

**Endpoint: PUT /customer/upgrade**

**Description:** Upgrade the customer to prime from regular

Request Body:

```
{
  "customerId" : 101,
  "plan":{"planId":1}
}
```

Responses:

- 200 OK: Customer upgraded to prime successfully.
- 400 Bad Request: The customer with given id is not found.

## 8. Change Password

**Endpoint:** PUT /customer/change-password

**Description:** Update the password for the customer account

Request Body

```
{
  "customerId" : integer,
  "oldPassword": "string",
  "newPassword": "string",
  "confirmPassword": "string"
}
```

Responses:

- 200 OK: Password changed successfully.
- 400 Bad Request: The customer with given id is not found/Incorrect old password/New passwords do not match.

**API Name:** MedicineAPI

**Description:** This API is responsible for fetching medicine details

**Base URL:** /medicine-api

**Endpoints:**

### 1. Get All Medicines

**Endpoint:** GET

/medicines/pageNumber/{pageNumber}/pageSize/{pageSize}

**Description:** Fetch all the medicines page wise

Path Variables : pageNumber, pageSize

Responses:

- 200 OK: Returns the list of all medicines

```
{
```



- ```
    "medicineId" : integer,  
    "medicineName": "string",  
    "manufacturer": "string",  
    "category": "string" ,  
    "manufacturing Date": LocalDate,  
    "expiryDate": LocalDate,  
    "price": integer,  
    "discountPercent" : integer  
}
```
- 400 Bad Request: No medicine found.

## 2. Get Medicine

**Endpoint:** GET /medicines/{medicineId}

**Description:** Fetch the details of medicine with given id

Path Variables : medicineId

Responses:

- 200 OK: Returns the details of medicine with given medicineId  
{  
 "medicineId" : integer,  
 "medicineName": "string",  
 "manufacturer": "string",  
 "category": "string" ,  
 "manufacturing Date": LocalDate,  
 "expiryDate": LocalDate,  
 "price": integer,  
 "discountPercent" : integer  
}
- 400 Bad Request: Medicine with given id is not found.

## 3. Get Medicine Category

**Endpoint:** GET /medicines/{category}

**Description:** Fetch all the medicines of given category

Path Variables : category

Responses:

- 200 OK: Returns the list of medicines with given category
- ```
{  
  "medicineId" : integer,  
  "medicineName": "string",  
  "manufacturer": "string",  
  "category": "string",  
  "manufacturing Date": LocalDate,  
  "expiryDate": LocalDate,  
  "price": integer,  
  "discountPercent" : integer  
}
```
- 400 Bad Request: No medicine found with given category.

#### 4. Update Medicine stock after order

**Endpoint:** PUT medicines/update-stock/medicine/{medicineId}

**Description:** Updates the quantity of medicine in the stock after successful order

Path Variables : **medicineId**

Request Body

```
{  
  "orderedQuantity":2  
}
```

Responses:

- 200 OK: Updates the stock successfully.
- 400 Bad Request : Invalid medicineId or stock not available.

**API Name:** CartAPI

**Description:** This API is responsible for fetching, adding and updating the medicines of the cart.

**Base URL:** /cart-api

## **Endpoints:**

### **1. Add to Cart**

**Endpoint:** POST /cart/add-medicine/{medicineId}/customer/{customerId}

**Description:** Add the medicine to customer's cart.

Path Variables : medicineId, customerId

Request Body

```
{  
    "quantity": integer  
}
```

Responses:

- 200 CREATED: Medicine added to cart successfully.
- 400 Bad Request: Invalid medicineId/customerId/quantity.

### **2. Get customers cart**

**Endpoint:** GET /cart/medicines/customer/{customerId}

**Description:** Fetch the medicines present in given customer's cart.

Path Variables : customerId

Responses:

- 200 OK: Fetch the list of medicines from customer's cart successfully.
- 400 Bad Request: No medicine found in the customer's cart.

### **3. Update quantity**

**Endpoint:** PUT /cart/update-quantity/medicine/{medicineId}/customer/{customerId}

**Description:** Update the quantity of medicine in customer's cart.

Path Variables : medicineId, customerId

Request Body

```
{  
    integer  
}
```

Responses:

- 200 OK: Quantity updated successfully.
- 400 Bad Request: Invalid medicineId/customerId/quantity.

#### 4. Delete Medicine

**Endpoint:** DELETE /cart/delete-medicine/{medicineId}/customer/{customerId}

**Description:** Delete the medicine from customer's cart.

Path Variables : medicineId, customerId

Responses:

- 200 OK: Medicine deleted from cart successfully.
- 400 Bad Request: Invalid medicineId/customerId.

#### 5. Delete All Medicines

**Endpoint:** DELETE /cart/delete-medicines/customer/{customerId}

**Description:** Delete all the medicine in customer's cart.

Path Variables : customerId

Responses:

- 200 OK: All the medicines successfully deleted from cart.
- 400 Bad Request: Invalid customerId or cart already empty.

**API Name:** OrderAPI

**Description:** This API is responsible for placing and cancelling orders and fetching the order details.

**Base URL:** /order-api

## **Endpoints:**

### **1. Place Order**

**Endpoint:** POST /order/place-order

**Description:** Place the order for medicines in the customer's cart.

Request Body

```
{
  "orderValueBeforeDiscount":Double
  "customer":{"customerId": Integer};
  "deliveryAddress":{"addressId": Integer };
  "card":{"
    "cardId":"string",
    "cvv":"string",
    "customerId": Integer
  }
}
```

Responses:

- 200 CREATED: Payment successful and Order placed successfully.
- 400 Bad Request: Invalid customerId/addressId/cardId.

### **2. Get Orders**

**Endpoint:** GET /order/view-orders/customer/{customerId}

**Description:** Fetch all the previous orders of customer with real time status.

Path Variables : customerId

Responses:

- 200 OK: Fetch the list of orders successfully.
- 400 Bad Request: No orders found or invalid customer id.

### 3. Cancel Order

**Endpoint:** PUT /order/cancel-order/{orderId}

**Description:** Cancel the order with given orderId.

Path Variables : orderId

Request Body

```
{  
    "cancelReason": "string"  
}
```

Responses:

- 200 OK: Order cancelled successfully.
- 400 Bad Request: No orders found with given orderId.

**API Name:** PaymentAPI

**Description:** This API is responsible for making payment and managing debit/credit cards.

**Base URL:** /payment-api

**Endpoints:**

#### 1. Make payment

**Endpoint: POST /payment/amount/{amountToPay}**

**Description:** Make the payment for ordering medicines in the customer's cart.

Path Variables : **amountToPay**

Request Body

```
{  
    "cardId": "string",  
    "nameOnCard" : "string",  
    "cardType":DEBIT/CREDIT,  
    "cvv" : "string",  
    "expiryDate" : LocalDate,  
    "customerId": Integer  
}
```

Responses:

- 201 CREATED: Payment made successfully.
- 400 Bad Request: Invalid customerId/cardId.

## 2. View cards

**Endpoint: GET /payment/view-cards/{customerId}**

**Description:** Fetch all the cards added by customer.

Path Variables : customerId

Responses:

- 200 OK: Fetch all the cards successfully.
- 400 Bad Request: Invalid customerId/ No card found.

## 3. Add Card

**Endpoint: POST /payment/add-card/{customerId}**

**Description:** Add the card for making payment in the customer's cart.

Path Variables : customerId

Request Body

```
{  
  "cardId": "string",  
  "nameOnCard" : "string",  
  "cardType":DEBIT/CREDIT,  
  "cvv" : "string",  
  "expiryDate" : LocalDate,  
  "customerId": Integer  
}
```

Responses:

- 200 CREATED: Payment made successfully.
- 400 Bad Request: Invalid customerId/cardId.

## Frontend Documentation

The frontend part of EPharmacy project is implemented using Angular Framework. It aims to create a user-friendly platform for customers to order medicines online. The frontend is responsible for displaying the user interface, handling user interactions, and communicating with the backend APIs to retrieve and update data.

### Project Setup

- Download the project from given link.
- Install Node.js and Angular CLI if not already installed.
- Run **npm install** to install project dependencies.
- Run **ng serve** to start the application.
- Navigate to <http://localhost:4200/> to view the app in the browser.

### Folder Structure

- **app**: app folder contains all the components, services, and models used in the project.



- **components:** components folder contains sub-folders for each component of the app.
- **models:** models folder contains data models used in the app.
- **services:** services folder contains all the services used for API communication.
- **app.module.ts** contains the main module for the app.
- **assets:** assets folder contains static assets such as images and stylesheets.
- **index.html** is the main HTML file for the app.

## Components

### App Component:

This is the parent component which displays the navigation menu, and it acts as the container for all other components. It includes links to login, registration, medicines, cart, customer profile, order components and logout button.

Please find below sample screenshots for your reference:

- Before signing in, the navigation bar should like:



- After signing in, the navigation bar should look like:



### Login Component:

The login component provides a form for users to login to the app. It checks the user credentials against the data stored in database. After successful login the user is redirected to home page. 'Create New Account' button is also present on login page for new users to create a new account.

- A sample Login page may look like:

The image shows a login form with a dark green header containing the text "Sign in or Sign up". Below the header, there is a message: "Sign up or Sign in to access your orders, special offers, health tips and more!". The form contains two input fields: "Email Address" and "Password". Below these fields is a blue button labeled "Sign in". Underneath the button is the text "OR", followed by a green button labeled "Create New Account".

### Register Component:

The register component provides a form for users to register themselves to app. It should prompt the user to enter his details such as full name, email, gender, date of birth and mobile number. Users should also set the password for his account. The details entered by users should be validated as per validations given below:

- Name should only contain alphabets with single space between the words.
- Email should be a valid email address with domains like ( in or com).
- Gender should be Male, Female or Other.
- Only above 18 years old users should be allowed to register. Verify it using date of birth field.
- Mobile number should start with 6,7,8 or 9 and should be of 10 digits.
- Password should contain at least one uppercase, one lowercase, one digit, one special character, should be of minimum 7 character and maximum of 20 characters.
- On page 2 user should enter his address details as below.
  - Address name should be a string such as Home or Work.
  - Address line1(house number/apartment number etc.).
  - Address line 2(street, landmark etc.)
  - Area, City, State (state should be a dropdown),
  - Pin code should be of 6 digits.

- Sample Registration page may look like:(page1 contains user details and page2 contains address details):

The image displays two sequential steps of a registration process, separated by a green arrow pointing from left to right. Both screens have a dark green header with the text "Provide your details".

**Left Screen (Step 1: Sign Up):** The progress bar shows "1 Sign Up" as the active step, "2 Address" as the next step, and "3 Done" as the final step. The form fields include:

- Full Name: A text input field with a person icon and placeholder text "Enter full name".
- Email: A text input field with an envelope icon and placeholder text "Enter email address".
- Gender: A dropdown menu with a person icon and placeholder text "--Select your gender--".
- Date of Birth: A date picker with a calendar icon, placeholder text "mm/dd/yyyy", and a clear button.
- Mobile Number: A text input field with a phone icon and placeholder text "Enter mobile number".
- Password: A text input field with a lock icon and placeholder text "Enter password".

A green "Next" button is at the bottom.

**Right Screen (Step 2: Address):** The progress bar shows "1 Sign Up" as a completed step, "2 Address" as the active step, and "3 Done" as the final step. The form fields include:

- Address Name: A text input field with a person icon and placeholder text "Enter Address Name(Home / Work)".
- Address Line 1: A text input field with a location pin icon and placeholder text "Enter address(House No, Building No, road name)".
- Address Line 2: A text input field with a location pin icon and placeholder text "Enter address( Street, Colony)".
- Area: A text input field with a location pin icon and placeholder text "Enter area".
- City: A text input field with a location pin icon and placeholder text "Enter city".
- State: A dropdown menu with a checkmark icon and placeholder text "--Select state--".
- Pin Code: A text input field with a "PIN" icon and placeholder text "Enter PIN Code".


A green "Next" button is at the bottom.

- For invalid data, the validation messages should be displayed as:

**Provide your details**


1 Sign Up — 2 Address — 3 Done

Full Name

 Enter full name


Name should contain only alphabets and single space between words!

Email

 i


Email id should be in valid format (example@abc.com or example@abc.in)!

Gender

 --Select your gender--


Select your gender!

Date of Birth

 mm/dd/yyyy


Date of Birth is required!

Mobile Number

 Enter mobile number

Mobile number should contain 10 digits starting with 6,7,8 or 9!

Password

 Enter password


- Must contain at least 1 lowercase!
- Must contain at least 1 digit!
- Must contain at least 1 uppercase!
- Must contain at least 1 special character!
- Must be at least 7 characters!
- Must be at most 20 characters!

Please provide a valid password!

Next

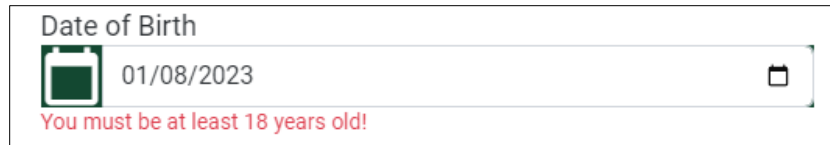
- If date of birth is a future date, the error message should be displayed as:

Date of Birth

 06/11/2024

Date of Birth cannot be in the future!

- If age is below 18, the error message should be displayed as:

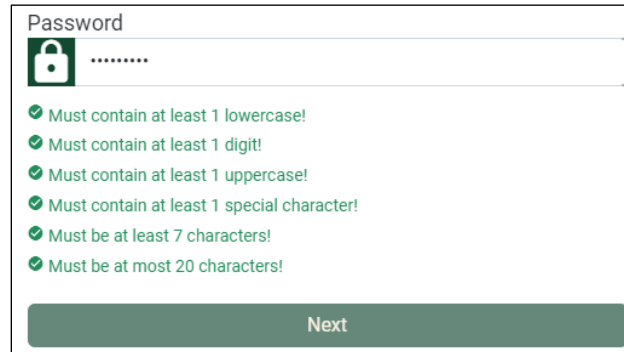


Date of Birth

01/08/2023

You must be at least 18 years old!

- If password meets the requirements, then it should be displayed as:



Password

.....

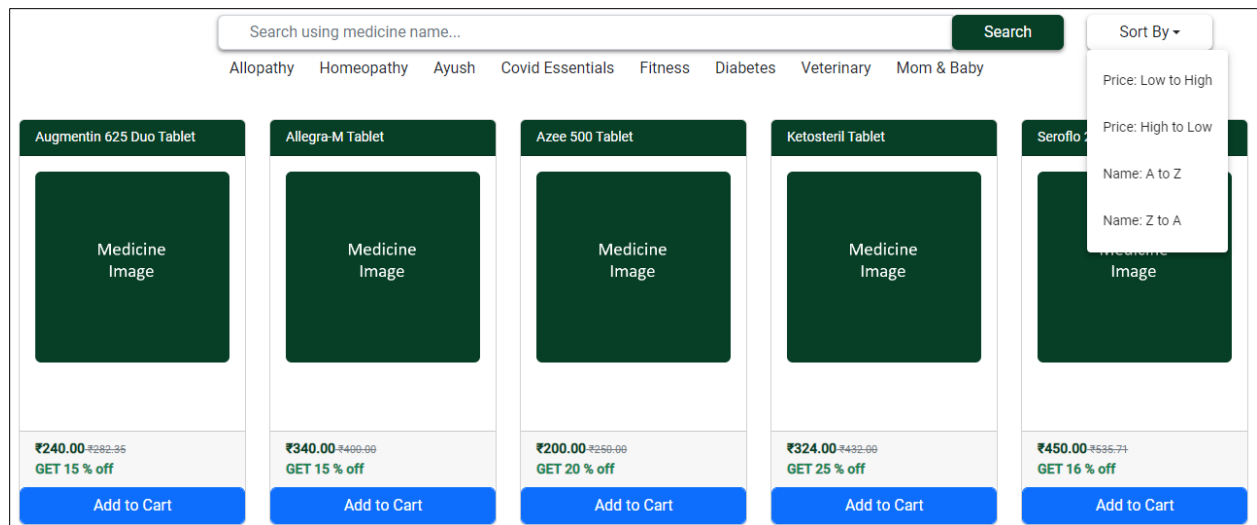
- ✔ Must contain at least 1 lowercase!
- ✔ Must contain at least 1 digit!
- ✔ Must contain at least 1 uppercase!
- ✔ Must contain at least 1 special character!
- ✔ Must be at least 7 characters!
- ✔ Must be at most 20 characters!

Next

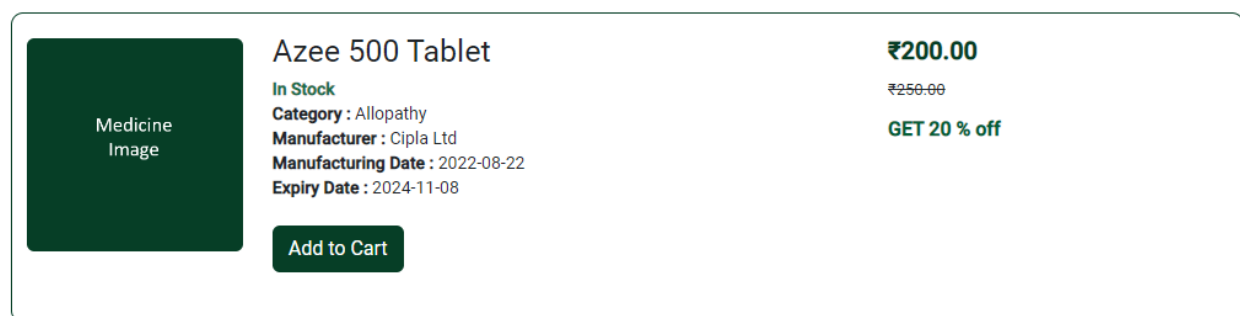
### Medicine Component:

Medicine component displays the landing page of the application. It contains a search bar to search the medicines by name, a list of categories Allopathy, Homeopathy etc. to see medicines category wise and a button to sort medicines by name or price. It also displays the list of all available medicines in pagination format.

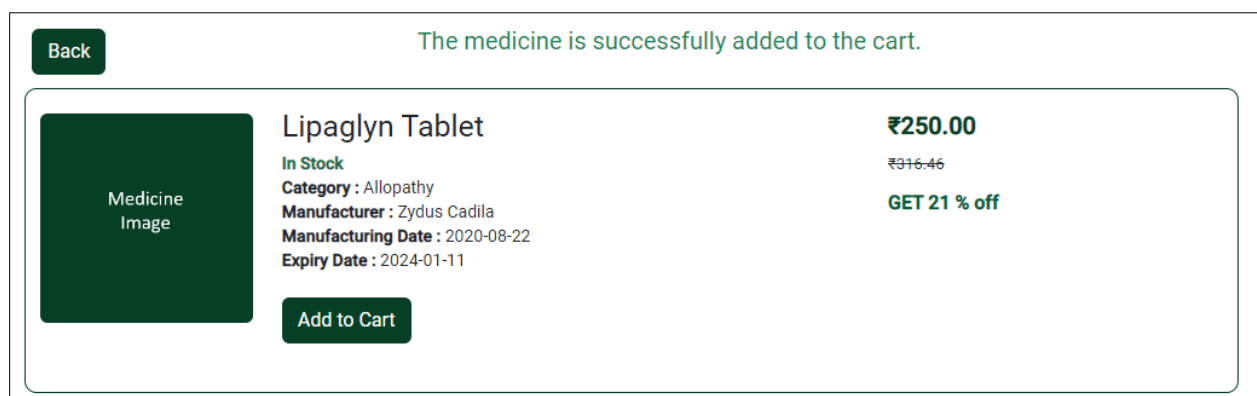
- When the user start typing in the search bar then medicine names containing the typed characters should appear as suggestions in drop down list.
- When user clicks on a category, then all the medicines of that category should be displayed.
- Sort By should display the options to sort medicines based on name A to Z and Z to A and based on price High to Low and Low to High.
- The home page should be displayed as:



On clicking on any medicine the details of medicine such as Name, category, manufacturing date, expiry date, stock availability, price and discount along with a button to add to cart should be displayed as:



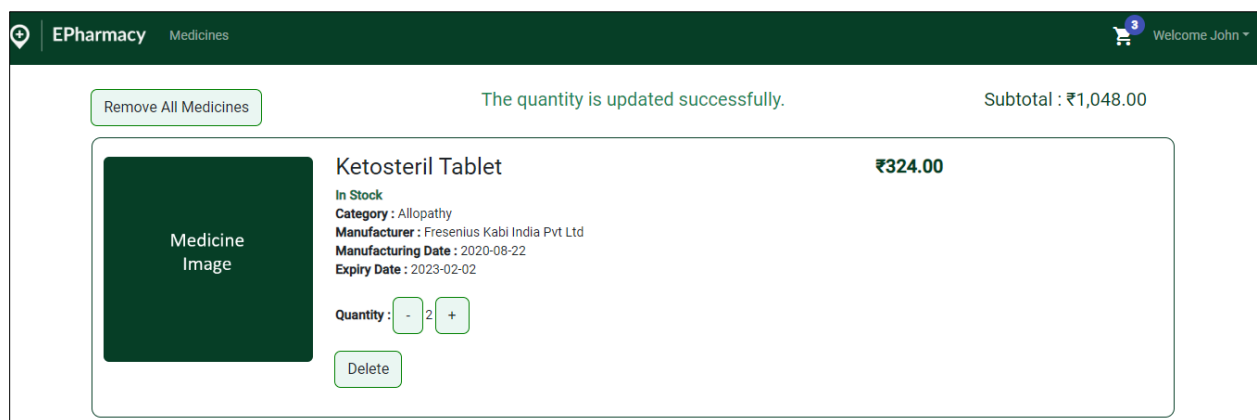
- On adding medicines to cart, a proper message should be displayed as:



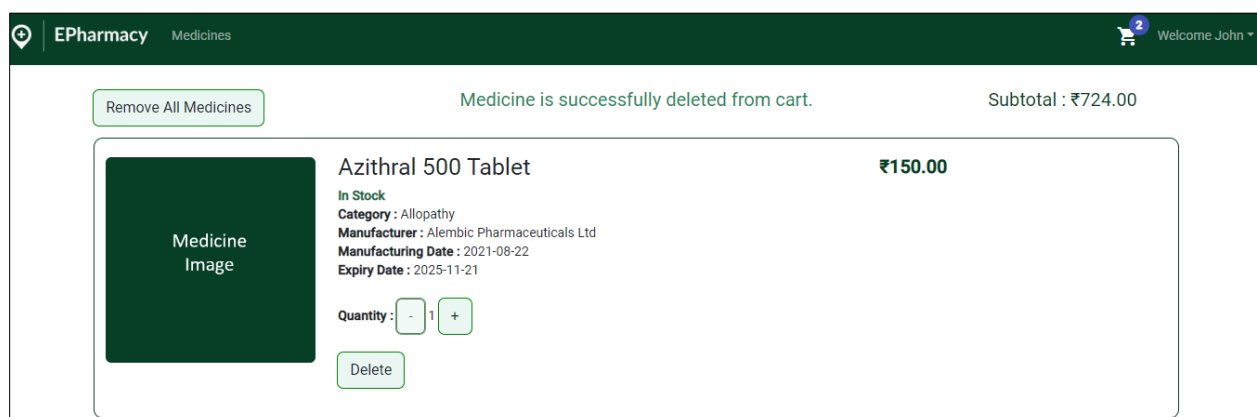
**Cart Component:**

The cart component displays the list of medicines added to the cart. Customers can update the quantity of medicines and remove medicines from the cart. It also displays the total price of the items in the cart.

- '+' and '-' buttons are used to update the quantity of medicines in the cart. Quantity should not go below 1. '-' button should be disabled when quantity is 1.
- On updating quantity subtotal should be updated immediately.
- On updating the quantity of medicines in cart, a proper message should be displayed as:

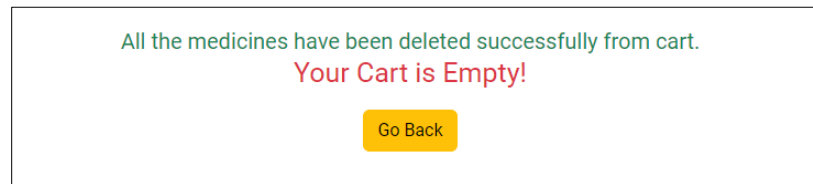


- Delete button is used to delete the medicine from cart.
- On deleting the medicine from cart, proper message should be displayed as:



- Remove All Medicine button is used to delete all the medicines from cart.

- On deleting all the medicines from cart, proper message should be displayed as:

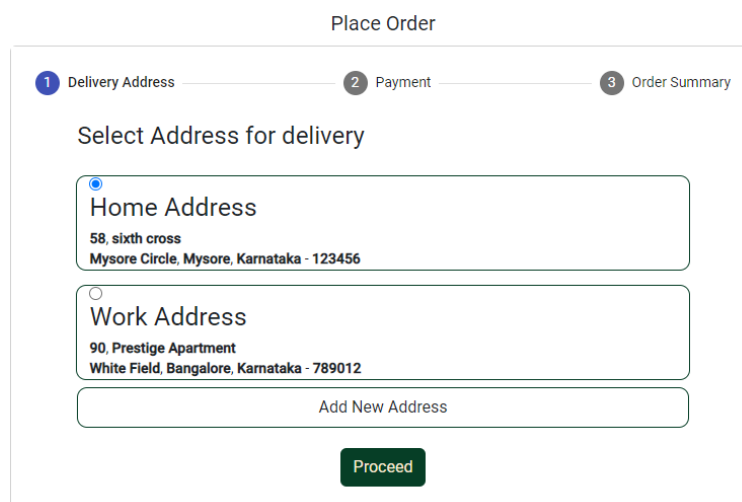


## Place-Order Component:

Place-Order component displays the necessary pages to place order. It allows customer to select or add a new delivery address, select or add a new card for payment and view order summary after successful payment.

### Delivery address:

- A customer should be able to choose an address for delivery so that the medicines are delivered at his address.
- Already added addresses should be displayed with a radio button to choose any address.
- When an address is selected then only Proceed button should be enabled.
- Already added addresses should be displayed as:

A screenshot of a web form titled "Place Order". At the top, there is a progress bar with three steps: "1 Delivery Address" (active, highlighted with a blue circle), "2 Payment", and "3 Order Summary". Below the progress bar, the text "Select Address for delivery" is displayed. There are two address options, each with a radio button. The first option is "Home Address" (selected), with details "58, sixth cross" and "Mysore Circle, Mysore, Karnataka - 123456". The second option is "Work Address" (unselected), with details "90, Prestige Apartment" and "White Field, Bangalore, Karnataka - 789012". Below these options is a button labeled "Add New Address". At the bottom center of the form is a green button labeled "Proceed".

### Making Payment:

- A customer should be able to make payment for the order. On clicking the Proceed button, Payment page is displayed



- On Payment page, Amount to be paid with proper calculation of amount including discount and usage of health coins should be displayed.
- Already added cards should be listed with radio button to choose card.
- When a card is chosen, customer should be asked to enter CVV to validate the card for making the payment.
- A proper message after payment and a button to view order summary should be displayed as:

Place Order

✓ Delivery Address
**2** Payment
3 Order Summary

Item(s) Subtotal :	₹1,624.00
Discount :	10%
Health Coins to use :	0
Total :	₹1,461.60
<b>Amount to be paid :</b>	<b>₹1,461.60</b>

Select Card for paying ₹1,461.60

☒

XXXX XXXX XXXX 1234

Type : DEBIT\_CARD

Valid Through : 2023-04-07

Name on Card : Steve

Enter CVV :

[Add New Card](#)

Payment is successful and order is placed. Thank You for ordering. Your order will be delivered before : 2023-03-12

[Order Summary](#)

### Order summary:

- A customer should be able to see the order summary so that he can keep his order details for future reference.
- Order Summary should be generated, just after placing order as shown below:

## ORDER SUMMARY

ORDER ID	ORDER PLACED ON	ORDER VALUE
2	Mar 7, 2023, 11:11:35 AM	₹1,624.00
ITEMS ORDERED		
Azee 500 Tablet Mfg by : Cipla Ltd		
		Unit Price : ₹200.00 Quantity Ordered : 2
Ketosteril Tablet Mfg by : Fresenius Kabi India Pvt Ltd		
		Unit Price : ₹324.00 Quantity Ordered : 1
Seroflo 250 Inhaler Mfg by : Cipla Ltd		
		Unit Price : ₹450.00 Quantity Ordered : 1
Heptral 400mg Tablet Mfg by : Abbott		
		Unit Price : ₹450.00 Quantity Ordered : 1
DELIVERY ADDRESS		
STEVE: 58, sixth cross, Mysore Circle Mysore, Karnataka - 123456		
PAYMENT INFORMATION		
PAYMENT METHOD		
DEBIT_CARD		Item(s) Subtotal : ₹1,624.00 Discount : 10% -----
BILLING ADDRESS		Total : ₹1,461.60 -----
STEVE: 58, sixth cross, Mysore Circle Mysore, Karnataka - 123456		Grand Total : ₹1,461.60

## View-Order Component:

View-Order component displays all the orders placed by the customer till current time. It displays status of all the orders and other details such as order id, date of order, delivery status, shipping address, expected date of delivery. It also gives the information of items ordered and payment information of each order.

### View Orders:

- A customer should be able to view his all orders so that he can see the current status of all orders.
- Each order should be displayed with details such as order id, date of order, delivery status, shipping address, expected date of delivery a Cancel Order button.
- Cancel Order button should be enabled only if the order status is PROCESSING, otherwise disabled.
- Each order looks like:

## My Orders

ORDER ID	ORDER STATUS	ORDER PLACED ON	DELIVERY STATUS	ARRIVING ON	SHIP To -
1	CONFIRMED	Mar 7, 2023, 10:30:26 AM	IN_TRANSIT	Mar 7, 2023	

ITEMS ORDERED

Medicine Image

**Letroz Tablet**

Category : Allopathy  
Manufacturer : Sun Pharmaceutical Industries Ltd  
Manufacturing Date : 2020-08-22  
Expiry Date : 2023-11-21

Unit Price : **₹150.00**  
Quantity Ordered : **1**

PAYMENT METHOD  
DEBIT\_CARD

**BILLING ADDRESS**  
STEVE 58, sixth cross, Mysore Circle  
Mysore, Karnataka - 123456

PAYMENT INFORMATION

Item(s) Subtotal :	₹150.00
Discount :	0%
Total:	₹150.00
<b>Grand Total:</b>	<b>₹150.00</b>

Cancel Order

### Cancel Order:

- A customer should be able to cancel order when the order status is 'PROCESSING'.
- On the click of Cancel Order button, one pop up should appear to confirm the cancellation of order by asking order id and reason for cancelling the order.
- There should also be a 'Confirm' button which is disabled until the orderId and cancel reason is provided.
- The pop up may look like:

Cancel Order

Confirm Order Id

Reason for Cancelling Order

Confirm

- On providing valid details and confirming the order cancellation the by clicking on 'Confirm' button, a proper message should be displayed as:

Cancel Order

Confirm Order Id

6

Reason for Cancelling Order

Medicines delivery delayed

Confirm

Your order has been cancelled successfully. Your money will be credited within 48 hours

- After cancelling order the, in order history the Order Status should be updated to 'CANCELLED'. Delivery Status and Delivery Date should also be updated accordingly.
- A cancelled order may look like:

ORDER ID 6	ORDER STATUS CANCELLED	ORDER PLACED ON Mar 13, 2023, 12:41:01 PM	DELIVERY STATUS DELIVERY_FAILED	ARRIVING ON N.A.	SHIP To
---------------	---------------------------	--	------------------------------------	---------------------	---------

ITEMS ORDERED

Medicine Image

Seroflo 250 Inhaler  
Category : Allopathy  
Manufacturer : Cipla Ltd  
Manufacturing Date : 2020-08-20  
Expiry Date : 2024-06-28

Unit Price : ₹450.00  
Quantity Ordered : 2

PAYMENT INFORMATION

PAYMENT METHOD

DEBIT\_CARD

BILLING ADDRESS

STEVE 90, Prestige Apartment, White Field  
Bangalore, Karnataka - 789012

Item(s) Subtotal :

₹900.00

Discount :

0%

Total:

₹900.00

Grand Total:

₹900.00

## Add-Address Component:

This component provides a form for customers to add their address details in database so that the same address can be used as delivery address while placing order. The customer must enter his address details such as address name, house or building number, area, city, state and pin code. Only valid details should be accepted.

- Address Name should can be any string such as Home, Work etc.
- Address line 1 should contain house number/building number etc.
- Address line 2 should contain street name/number, landmark etc.

- Area and City should contain a valid string.
- State should be displayed as drop-down list of Indian states.
- Pincode should be a valid 6 digits number.
- The address form should be displayed as:

The form is titled "Provide Address Details" and contains the following fields:

- Enter Address Name(Home / Work)**: A text input field with a person icon.
- Enter address line 1 (House No. or Building No.)**: A text input field with a building icon.
- Enter address line 2 (street name or landmark)**: A text input field with a location pin icon.
- Enter area**: A text input field with a location pin icon.
- Enter city**: A text input field with a city skyline icon.
- Select state--**: A dropdown menu with a checkmark icon.
- Enter PIN Code**: A text input field with a PIN icon.

At the bottom of the form is a green button labeled "Add Address".

### Add-Card Component:

This component provides a form for customers to save his debit/credit card details in database so that the card can be used for making payment. The customer must provide the details such as card number, card type, expiry date, cvv and name on card. Only valid details should be accepted.

- Card number should be of 16 digits.
- Card should be Debit or Credit.
- Expiry date should be a future date.
- Name on card should contain only English letters with single space between words.
- CVV should be of 3 digits.
- The card form should be displayed as:

Provide Card Details

Card Number

Name on Card

CVV

Expiry Date

Card Type


Add Card

## Customer Component:

This component is used for displaying customer profile, updating customer details such as email Id, contact number, password. It also allows customer to upgrade themselves to be Prime members by choosing different plans.

- Customer Profile should be accessible from navigation menu 'My Profile' tab.
- On displaying Customer Profile, customer should get access to buttons Edit Profile, Change Password and Upgrade to Prime
- Customer profile should be displayed as:

Customer Profile



Customer Name:

Steve

Email:

steve@infosys.com

Contact Number:

9880253413

Gender:

Male

Date of Birth:

1999-09-09

Address:

Home Address : 58, sixth cross, Mysore Circle, Karnataka-123456

Work Address : 90, Prestige Apartment, White Field, Karnataka-789012

Edit Profile

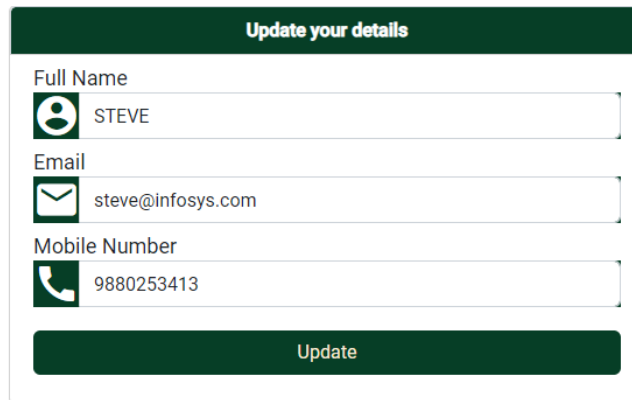
Change Password

Upgrade to Prime

## Edit Profile:

- A customer should be able to edit his details such as name, email and contact number. On clicking on Edit Profile button, update profile form should be displayed.

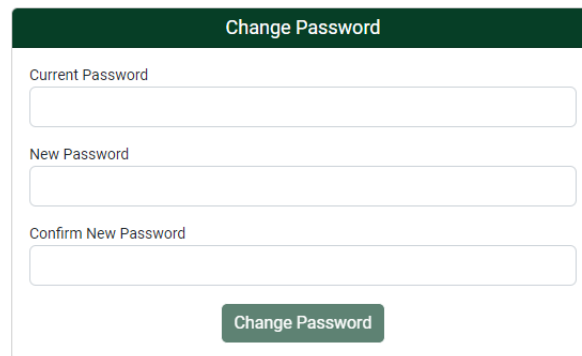
- Update form should open with prefilled current details of customer.
- Full Name should contain only letters and single space between words.
- Email should be a valid email address with domain com or in.
- Mobile number should of 10 digits starting with 6,7,8 or 9.
- Customer should be able to update any of these details.



The form is titled "Update your details" in a dark green header. It contains three input fields, each with a green icon on the left: a person icon for "Full Name" (prefilled with "STEVE"), an envelope icon for "Email" (prefilled with "steve@infosys.com"), and a telephone icon for "Mobile Number" (prefilled with "9880253413"). A dark green "Update" button is at the bottom.

### Change Password:

- A customer should be able to change password for his account so that he can ensure security of his account. On clicking 'Change Password' button a Change Password form should appear.
- Change Password form should have 3 mandatory fields to fill i.e. Current Password, New Password and Confirm New Password. A sample form may look like:



The form is titled "Change Password" in a dark green header. It contains three text input fields labeled "Current Password", "New Password", and "Confirm New Password". A dark green "Change Password" button is at the bottom.

- New Password should meet the following password requirement:
  - Password must contain at least 1 lowercase, 1 uppercase, least 1 digit and at least 1 special character.
  - The minimum and maximum lengths of password must be 7 and 20 characters respectively.

- Confirm password field should display error message as 'Passwords do not match' when New Password and Confirm Password fields do not match as shown below:

The screenshot shows a 'Change Password' form with a dark green header. It contains three input fields: 'Current Password', 'New Password', and 'Confirm New Password'. The 'New Password' field is followed by six green checkmark icons and their corresponding requirements: 'Must contain at least 1 lowercase!', 'Must contain at least 1 digit!', 'Must contain at least 1 uppercase!', 'Must contain at least 1 special character!', 'Must be at least 7 characters!', and 'Must be at most 20 characters!'. Below the 'Confirm New Password' field, a red error message 'Passwords do not match.' is displayed. The 'Change Password' button is present but appears disabled.

- Change Password button in above form should be disabled until Passwords do not match. And it should be enabled only on password match as shown below:

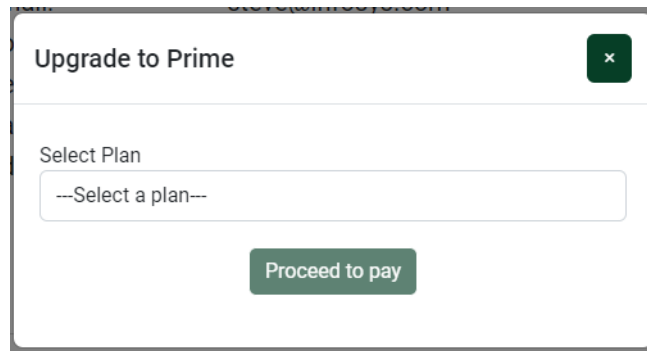
This screenshot shows the same 'Change Password' form, but the 'Change Password' button is now a solid dark green, indicating it is enabled. The error message is no longer present, suggesting the passwords now match.

### Upgrade to Prime:

- A regular customer should be able upgrade himself to 'Prime' by choosing one of the three available plans (Monthly, Quarterly or Yearly) so that he can get the benefit of prime member.
- For Monthly plan customer should pay Rs. 199, for Quarterly Rs. 399 and for Yearly Rs. 699.

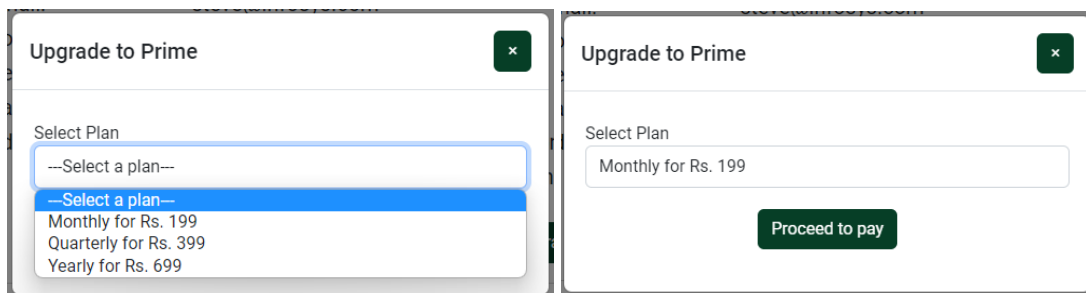


- On clicking the 'Upgrade to Prime' button, a pop-up should appear as:



The screenshot shows a modal dialog titled "Upgrade to Prime" with a close button (X) in the top right corner. Inside the dialog, there is a label "Select Plan" above a dropdown menu. The dropdown menu currently displays "--Select a plan--". Below the dropdown menu is a green button labeled "Proceed to pay".

- After selecting a plan, 'Proceed to pay' button is enabled, and the customer can proceed to make payment as per the plan chosen.



The image contains two side-by-side screenshots of the "Upgrade to Prime" dialog. The left screenshot shows the dropdown menu open, displaying three options: "--Select a plan--", "Monthly for Rs. 199", "Quarterly for Rs. 399", and "Yearly for Rs. 699". The right screenshot shows the dropdown menu closed, with the selected option "Monthly for Rs. 199" visible in the input field. In both screenshots, the "Proceed to pay" button is visible and enabled.

- On clicking 'Proceed to pay' button, customer can access all his card and choose one for making payment. On successful payment, a proper message should be displayed as:

Upgrade to Prime

×

Select Plan

Monthly for Rs. 199

Select card to pay ₹199.00

☒

XXXXXXXX XX  
XXXXXXXX XX  
XXXXXXXX XX  
XXXXXXXX 1234  
Type : DEBIT\_CARD  
Valid Through : 2023-04-07  
Name on Card : Steve

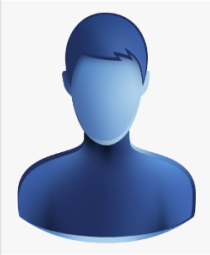
Enter CVV : ...

Make Payment

Congratulations! You are a prime member now. You  
 will get health coins on every purchase till 13 April  
 2023.

- The updated profile of prime member should be displayed with distinguishable prime membership, the expiry date of plan subscribed and the number of health coins currently available in the account. A sample profile may look like:

Customer Profile



Prime Member\*

Plan expires on : 13 April 2023

Customer Name:

Steve

Email:

steve@infosys.com

Contact Number:

9880253413

Gender:

Male

Date of Birth:

1999-09-09

Health Coins\*

0

Address:

Home Address : 58, sixth cross, Mysore Circle, Karnataka-123456

Work Address : 90, Prestige Apartment, White Field, Karnataka-789012

Edit Profile

Change Password

Upgrade to Prime