SOFTWARE DEVELOPMENT PRACTICES

ONLINE SHOPPING SYSTEM

CK Sruthi Shree

Dharshini S

Adithya Baba

Divya Dharshini

IT A

**OBJECTIVE:**

The online shopping system application is a software designed for the vendors and the customers for an easy way of shopping through online anytime, anywhere. It is where the customers look through the products to shop that have been posted by the vendors setting up online shops.

**USERS OF THE SYSTEM:**

* Admin
* Customer
* Vendor

**FUCTIONAL REUIREMENTS:**

* Build an application that can be easily accessed by the vendor to set up online shops and for the customers to purchase products through online
* USER REGISTRATION AND AUTHENTICATION:
* The page should open with a vendor/customer option, login/sign up page, profile, and cart.
* If a customer is going to buy a product, he/she must be signed up or registered.
* Unregistered users cannot go to the shopping cart.
* The newly registered customers should sign up with valid mail id, phone number, name and password, whereas existing customers can login using their mail id and password.
* PRODUCT CATALOG:
* The products are categorized accordingly and the customer can select according to their requirement.
* SHOPPING CART:
* The customer can add the products they are intrested in by adding them to the cart and they can check out later through it.
* The customer can place their order by entering their name, delivery address, and contact number.
* PAYMENT PROCESSING:
* The payment method for the purchased products will be through cash.
* We will extend this to credit card, debit card, etc, in the future.
* ORDER MANAGEMENT:
* The copy of the bill be sent to the customer to their registered mail id after placing the order.
* An option will be present in their order history in case of cancellation or return of the product.
* USER ACCOUNT MANAGEMENT:
* The customer can edit their profile through the profile dashboard.
* After the use of the application the customer can logout at the end of the session.
* SELLER ACCOUNT MANAGEMENT:
* The seller has a separate login and had the same procedure as the customer to register.
* **(to be done)**

**OUTPUT/POST CONDITION:**

* After the order placement a copy of the bill will be sent to the customer to their mail id that they have registered with.

**NON-FUNCTIONAL REQUIREMENTS:**

* SECURITY:
* Secured access to customer’s confidential data
* They should be categorized and stored in a secure manner (i.e., mail id, password, address, phone number etc.)
* PERFORMANCE:
* Peak Load performance
* eCommerce -< 3 sec
* Admin application < 2 sec
* Non-Peak Load performance
* eCommerce <2 sec
* Admin Applications < 2 sec
* AVAILABILITY:
* 99.9% available
* STANDARD FEATURES:
* Scalability
* Maintainability
* Resource Utilization
* Usability
* Security
* Reliability
* Availability
* Failover
* LOGGING AND AUDITING:
* The system should support logging (app/web/DB) and auditing at all levels.
* MONITORING:
* Should be able to monitor via as-in enterprise monitoring tools
* CLOUD:
* The solutions should be made cloud-ready and should have a minimum impact when moving away to cloud infrastructure.
* BROWSER COMPATIBLE:
* All latest browser.

**TECHNOLOGY STACK:**

**FRONT END:** CSS, HTML

**MATERIAL DESIGN:** Bootstrap / Bulma

**SERVER SIDE:**

**DATABASE:** MYSQL

**APPLICATION ASSUMPTION:**

* The login page should be the first page rendered when the application loads.
* Manual routing should be restricted by using AuthGuard by implementing the canActivate interface. For example, if the user enters as http://localhost:8000/signup

or

<http://localhost:8000/home>

the page should not navigate to the corresponding page instead it should redirect to the login page.

* Unless logged into the system, the user cannot navigate to any other pages.
* Logging out must again redirect to the login page.
* To navigate to the admin side, you can store a user type as admin in the database with a username and password as admin.
* Use admin/admin as the username and password to navigate to the admin dashboard.

**VALIDATIONS:**

* Basic email validation should be performed.
* Basic mobile validation should be performed.
* Password validations should be performed.

**PROJECT TASKS:**

**API ENDPOINTS:**

**USER:**

**ACTION URL METHOD RESPONSE:**

* Login /login POST true/false.
* Signup /signup POST true/false.
* **(to be done)**

**ADMIN:**

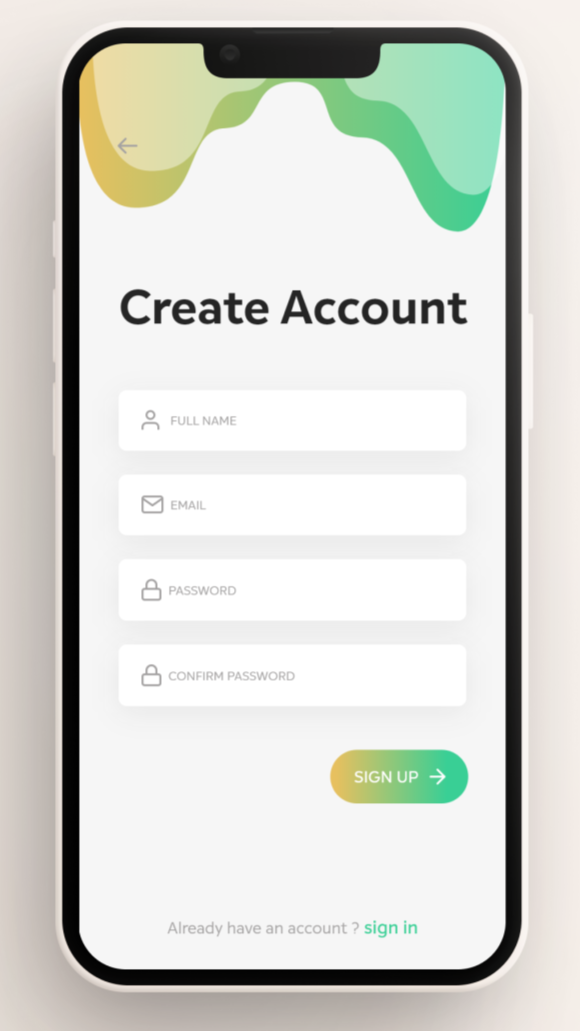
**ACTION URL METHOD RESPONSE:**

* **(to be done)**
* **(to be done)**

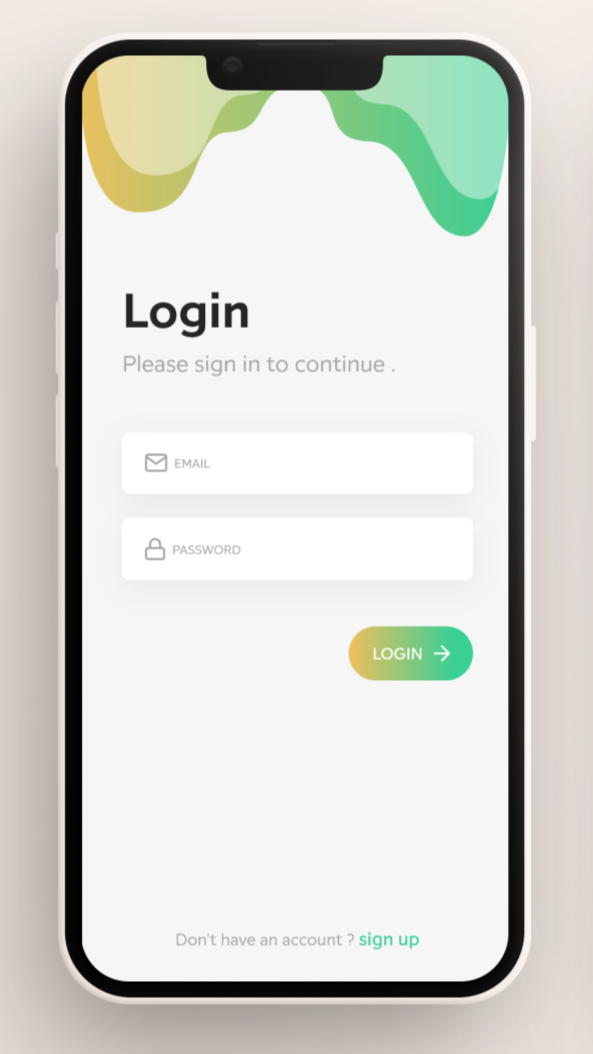
**FRONTEND:**

**CUSTOMER:**

* **Auth:** The customer can authenticate login and signup credential.
* **Register:** The new customer has options to sign up by providing their basic details.
* **Ids:**
* Name
* Mail id
* Password
* Re-type password
* Sign up button
* Login link
* **API endpoint URL:** <http://localhost:8000/signup>
* **Output screenshot:**

****

* **Login:** Already existing customer can login through their registered mail id and password.
* **Ids:**
* Mail id
* Password
* Login button
* Sign up link
* **API endpoint URL:** <http://localhost:8000/login>
* **Output screenshot:**

****

* **Dashboard:** A homepage where all the categorized products are available including the cart and profile.
* **Ids:**
* Category
* Profile
* Cart
* Logout button
* **API endpoint URL:** <http://localhost:8000/home>
* **Output screenshot:**

**ADMIN:**

* **Dashboard:** A dashboard where the list of product category is displayed on the admin side.

**BACKEND:**

* **Model Layer:**
* **User Model:** The user type (admin sell)

Backend:

Model Layer:

1. User Model: the user type (admin or customer) and all user

information are stored

a. Attributes:

i. email: String

ii. password: String

iii. username: String

iv. mobileNumber: String

v. active: Boolean

vi. role: String

2. Login Model: This class contains the email and password of the

user.

a. Attributes:

i. email: String

ii. password: String

3. Prevention Model: This contains the prevention methods of

COVID-19.

a. Attributes:

i. Wash Hands: Character

ii. Wear Mask: Character

iii. Clean Disinfect: Character

3. Statistics Model: This contains the statistics of COVID-19

confirmed cases, Total deaths, Total Recovered, New cases.

a. Attributes:

i. Confirmed Cases: String

ii. Total Death: String

iii. Total Recovered: String

iv. New Cases: String

Controller Layer:

1. Signup Controller: This class control the user signup

a. Methods:

i. saveUser(UserModel user): This method helps to store

users in the database and return true or false based on the database

transaction

2. Login Controller: This class controls the user login.

a. Methods:

i. check User (Login Model data): This method helps the user

to sign up for the application and must return true or false.