Pharmacy Store

SIDHANT KR AGRAWAL(2021495) || PARAS DHIMAN(2021482)

PROJECT ENTITIES AND RELATIONSHIPS
Some basic entities are Patients, Doctors, Prescription, Medicine,Pharmacist,Suppliers,Policy Providers,Orders,Appointments,Branches,Stock,Payments,Medical History
Relationships:
- Patients - Prescription (one to many)
- Prescription - Doctor (many to one)
- Prescription - Pharmacist (many to one)
- Prescription - Medicines (many to many)
- Medicine - Suppliers (many to many)
- Patient - Policy (many to many)
- Branches - Stock (one to many)
- Stock - Suppliers (many to many)
- Patient - Appointment (one to one)
- Appointment - Doctor(many to one)
- Patient - Medical History (one to one)
- Doctor - Branches (many to many)
- Pharmacist - Branches (one to one)

PROJECT SCOPE
Users can access and modify their account information upon registering and logging into the app. They could look through various goods, see information about each one, including its name, description, price, and photographs, and even put items in a virtual shopping basket to see how much their purchase would cost. A shopping cart and order history would be available to customers. They might look for goods either by brand or

kind. All the data is stored and maintained in MySQL's database(i.e going to be the primary service provider).
Things, as well as orders for those products, might be managed by an administrator.
Customers could pay for their orders.
The program would be developed to function correctly on a wide range of screen sizes.

TECH STACK
Front-end:

- *React* : React would develop the application's user interface. React enables for reusable UI components and quick updates to the user interface in response to user actions or data changes.
- *HTML & CSS* : HTML would structure and CSS would style the application's web pages. HTML tags produce headers, paragraphs, and forms, whereas CSS styles govern font size, colour, and layout.
- *JavaScript* : JavaScript would produce dynamic front-end components. JavaScript code may establish event listeners that react to user activities like button clicks and request data from the back-end.

Back-end:

- *Django & Python* : These technologies would develop the application back-end. Django is a Python web framework for constructing online applications. Python would handle front-end requests, database operations, and business logic on the application's back-end.
- *MySQL* : MySQL would store and retrieve application data. SQL queries would extract and update customer, product, and order

history from MySQL database tables in response to front-end requests.

This stack would provide a dynamic and interactive front-end driven by React and a solid and efficient back-end constructed using Django and Python, with MySQL handling data storage and administration.

Functional Requirements
The application will provide a dynamic and interactive front-end driven by React.
It would enable users to create and log in to an account, browse products, add items to their cart, and view order history.
There will be a solid and efficient backend constructed using Django and Python, with MySQL handling data storage and administration. The backend will interact with the database, which should :

- Securely store customer information and authenticate customer login attempts.
- able to retrieve and display product information
- create and manage cart
- process and validate payments
- Handle customer account updates