**Flash-Packer**

**Purpose:**

* Main purpose of this website is that customers can place order from their phone and from that only café kitchen workers will get the order information and will start cooking.
* Workers can update the stock of the required items to the database by removing the used stocks and adding the stocks if purchased new or used!

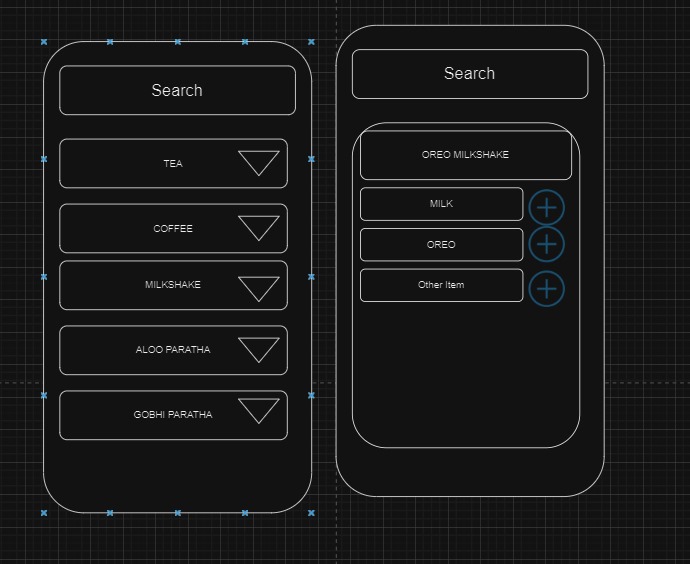
**Target audience:**

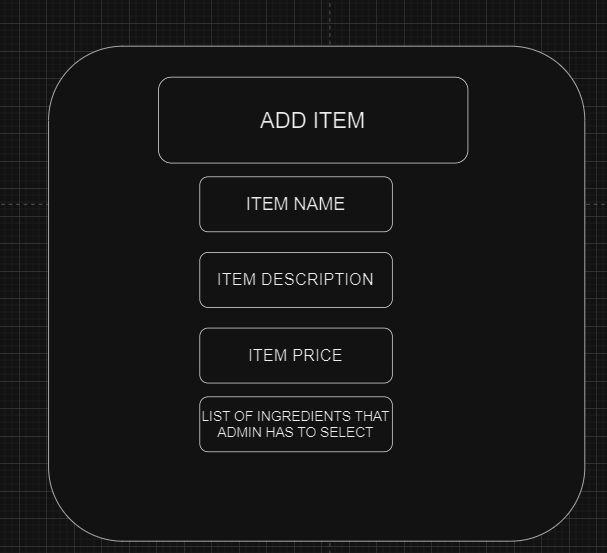
Cafe Customers

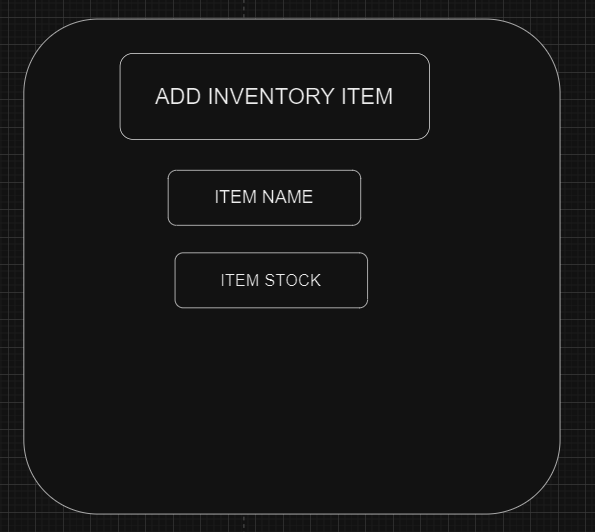
**Story:**

Customers comes and scans the QR code and then visits to the website, from which they can see the menu, there are many items and they can select from that and order it, that order will be redirected to the admin and the chief, chief accepts the order, and then cooks it, then ticks on ready, admin sends someone to pick the order from kitchen, order placed, customer gets the order, admin can see all the history and graphical and pictorial representation of data and can update the menu. Also, he can keep the record of the inventory!

**Design:**



****

****

ADMIN UI REQUIREMENTS:

* Admin can see (on home page)
  + Inspired from Mongodb interface
* Admin can do (on other page)
  + CRUD of New Menu Item
  + CRUD of Inventory Item
* Admin menu page
  + Can see menu list
    - Each item contains update and delete button
  + Update item ui
  + Add item ui
  + Add Button on menu page
* Admin Inventory page
  + Can see inventory list
    - Each item contains update and delete button
  + Update item ui
  + Add item ui
  + Add Button on menu page

**#The Add or update form requirement is as shown above**

**Needs:**

* Menu
* Inventory Details
* Recipe inventory required for particular order
* Designs

**Functionality:**

* Customer can see the menu
* Customer can order from that
  + Customer has to provide valid data of him/her
* Order history stored
* Bill invoice in the history | can send to user
* Worker
  + Can See Menu Item
  + Menu item contains ingredients
  + Can subtract the ingredients by selecting it
  + Can select the quantity for that type of product for example (floor, tea leaves, coffee powder etc) or countable for milk packets
  + Benefit: we can have the data of ‘for what item, which ingredients was used’
* Worker approach 2
  + Can see only ingredient lists
  + From that, it subtracts
* Admin can see
  + Café income
  + Order History
  + Inventory Item Stock
  + Menu
* Admin can do
  + CRUD of New Menu Item
  + CRUD of Inventory Item

**Database:**

**Going with the SQL**

SQL is a suitable choice for this project due to its ability to handle structured data and complex relationships, ensuring ACID compliance for financial transactions. The mature ecosystem, support for complex queries, and data integrity features make SQL databases reliable for maintaining consistent performance and facilitating flexible reporting and analysis.

Certainly! Here's the textual representation of the schema for your documentation:

**MenuTable:**

* MenuItemID (Primary Key)
* Name (VARCHAR(255))
* Description (VARCHAR(255))
* Price (DECIMAL(10, 2))

**InventoryTable:**

* InventoryItemID (Primary Key)
* Name (VARCHAR(255))
* Quantity (INT)
* Unit (VARCHAR(50))

**MenuItemIngredientTable:**

* MenuItemIngredientID (Primary Key)
* MenuItemID (Foreign Key referencing MenuTable.MenuItemID)
* InventoryItemID (Foreign Key referencing InventoryTable.InventoryItemID)
* Quantity (INT)

**OrderTable:**

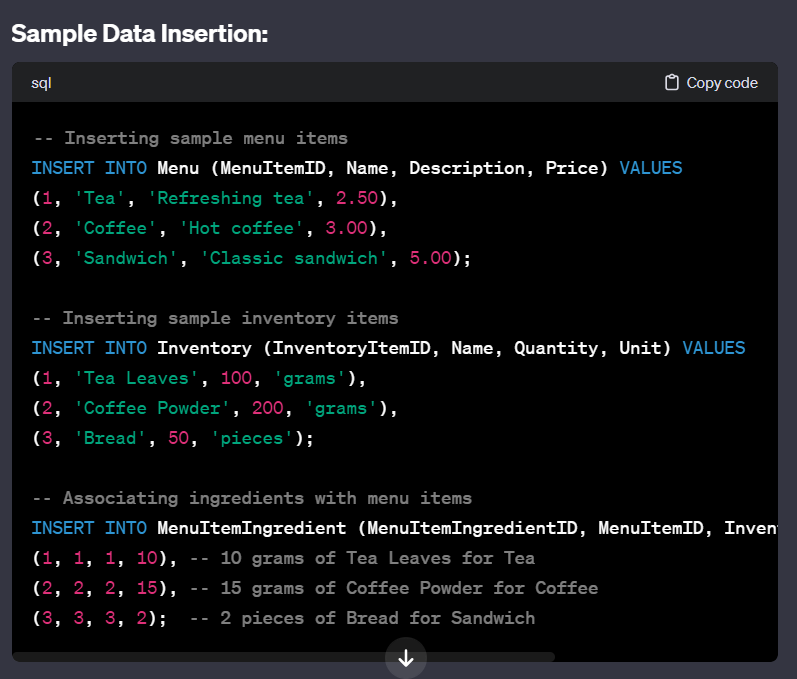
* OrderID (Primary Key)
* CustomerName (VARCHAR(255))
* CustomerEmail (VARCHAR(255))
* CustomerPhoneNumber (VARCHAR(20))
* OrderDateTime (DATETIME)
* TotalAmount (DECIMAL(10, 2))
* OrderStatus (VARCHAR(50))

**OrderItemTable:**

* OrderItemID (Primary Key)
* OrderID (Foreign Key referencing OrderTable.OrderID)
* MenuItemID (Foreign Key referencing MenuTable.MenuItemID)
* Quantity (INT)

**WorkerTable:**

* WorkerID (Primary Key)
* Username (VARCHAR(50) UNIQUE)
* Password (VARCHAR(255))
* FirstName (VARCHAR(50))
* LastName (VARCHAR(50))
* Role (VARCHAR(50))



CAN DO

Adding menu items

Adding inventory stock

Adding relation of dish and ingredient

Ordering

When order is placed, deducting inventory item stock according to order item and it’s quantity

i want to provide the order history to customer

Finding monthly earning

Finding yearly earning

Finding daily earning

when stock not available then customer can't order that item !

