# **Bar Management**

### Requirements

- **1.** Users are able to login to the system using credentials.
- 2. Users Can Add new Items to the database.
- 3. Users Can Add a new Table in the respective category.
- 4. Once logged in users can Select an existing table to start taking orders.
- 5.User can select an existing table to update or edit the order Items.
- 6.User can print the bill which makes the current table marked for collecting payments.
- 7. Once User collects the payment the paid option will free the table and get removed from the love dashboard.
- 8. The user can Download an excel file of the daily sales.
- 9. Users can add the purchase information from the Bill from the Distributor along with discounts and other options.

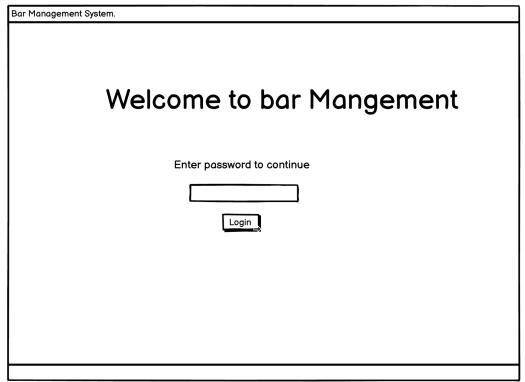
Here one of the improvements will be that we are going to provide the government names compared to the local names. So we are maintaining the updated list every time.

### UI Wire-Frames.

Balsamiq : Presentation <u>link</u>

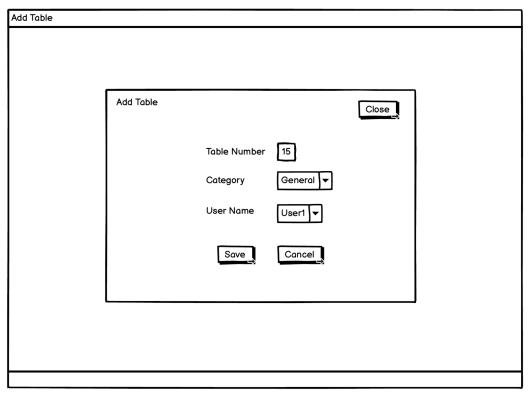
Click on presentation to view the application

Login Page 1/5

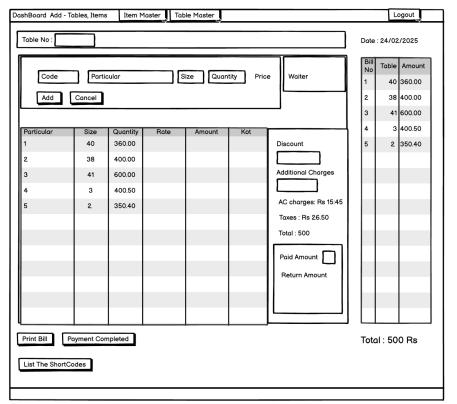


Here We are not including the User Information Since we maintain a single login

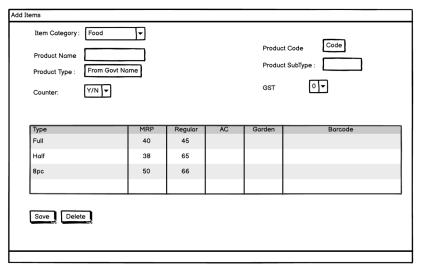
Table Master 5 / 5



Main Dash Board 2/5



Item Master (Food) 4 / 5

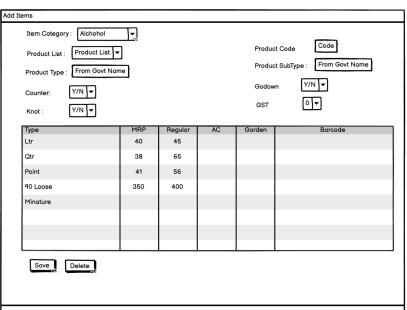


Product Type : From Govt Name

Here Based on the Item Category and Product Item Selected

The Other inputs will change accordingly. Example Product Subtype, Go down etc..

#### Item Master (Alchohol)



3 / 5
Product Type: From Govt Name

Here Based on the Item Category and Product Item Selected

The Other inputs will change accordingly. Example Product Subtype, Go down etc..

## Class Diagrams

Note: Still working on it and will update with final in some time

- **User** (user\_id, username, password\_hash, role, created\_at)
- **Item** (item\_id, name, category, size, price, mrp, barcode, stock\_quantity, created\_at)
- **Table** (table\_id, table\_number, category, status, created\_at)
- **Order** (order\_id, table\_id, user\_id, status, timestamp)
- **OrderItem** (order\_item\_id, order\_id, item\_id, quantity, price)
- **Bill** (bill\_id, order\_id, total\_amount, discount, taxes, final\_amount, payment\_status, generated at)
- **Payment** (payment\_id, bill\_id, amount\_paid, payment\_method, paid\_at)
- **SalesReport** (report\_id, date, total\_sales, generated\_by)
- Purchase (purchase\_id, distributor\_name, bill\_number, total\_amount, discount, final\_amount, purchase\_date)

#### Each class has associations:

- User places Orders.
- Orders belong to a Table.
- Orders contain multiple OrderItems.
- Bills are generated for Orders.
- Payments are linked to Bills.
- Sales Reports are generated by Users.
- Purchases are recorded separately.

## **Database Design**

Note: Still working on it and will update with final one in some time

Let's design our database so for that first we need to have a class diagram.

```
CREATE TABLE users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    role ENUM('admin', 'waiter', 'manager') NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE items (
```

```
item id INT PRIMARY KEY AUTO INCREMENT,
   name VARCHAR(100) NOT NULL,
   category ENUM('beer', 'whiskey', 'vodka', 'wine', 'soft drinks') NOT
NULL,
    size ENUM('Ltr', 'Qrt', 'Point', 'Nip', 'Miniature', '90 Loose',
'Large', 'Small') NOT NULL,
   price DECIMAL(10,2) NOT NULL,
   mrp DECIMAL(10,2) NOT NULL,
   barcode VARCHAR(50) UNIQUE NOT NULL,
   stock_quantity INT NOT NULL,
   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
CREATE TABLE tables (
   table id INT PRIMARY KEY AUTO INCREMENT,
   table_number VARCHAR(10) UNIQUE NOT NULL,
   category ENUM('indoor', 'outdoor', 'VIP', 'bar') NOT NULL,
   status ENUM('available', 'occupied', 'awaiting payment') DEFAULT
   created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
CREATE TABLE orders (
   order_id INT PRIMARY KEY AUTO_INCREMENT,
   table id INT NOT NULL,
   user id INT NOT NULL,
   status ENUM('pending', 'completed') DEFAULT 'pending',
   timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
   FOREIGN KEY (table id) REFERENCES tables(table id) ON DELETE
CASCADE,
   FOREIGN KEY (user_id) REFERENCES users(user_id) ON DELETE CASCADE
);
CREATE TABLE order_items (
   order_item_id INT PRIMARY KEY AUTO_INCREMENT,
   order_id INT NOT NULL,
   item_id INT NOT NULL,
   quantity INT NOT NULL,
   price DECIMAL(10,2) NOT NULL,
   FOREIGN KEY (order_id) REFERENCES orders(order_id) ON DELETE
CASCADE.
   FOREIGN KEY (item_id) REFERENCES items(item_id) ON DELETE CASCADE
);
CREATE TABLE bills (
   bill_id INT PRIMARY KEY AUTO_INCREMENT,
```

```
order id INT NOT NULL,
   total_amount DECIMAL(10,2) NOT NULL,
   discount DECIMAL(10,2) DEFAULT 0.00,
   taxes DECIMAL(10,2) DEFAULT 0.00,
   final amount DECIMAL(10,2) NOT NULL,
   payment_status ENUM('pending', 'paid') DEFAULT 'pending',
   generated at TIMESTAMP DEFAULT CURRENT TIMESTAMP,
   FOREIGN KEY (order_id) REFERENCES orders(order_id) ON DELETE CASCADE
);
CREATE TABLE payments (
    payment id INT PRIMARY KEY AUTO INCREMENT,
   bill id INT NOT NULL,
   amount_paid DECIMAL(10,2) NOT NULL,
   payment_method ENUM('cash', 'credit_card', 'UPI') NOT NULL,
   paid at TIMESTAMP DEFAULT CURRENT TIMESTAMP,
   FOREIGN KEY (bill_id) REFERENCES bills(bill_id) ON DELETE CASCADE
);
CREATE TABLE sales_report (
   report id INT PRIMARY KEY AUTO INCREMENT,
   date DATE NOT NULL UNIQUE,
   total_sales DECIMAL(10,2) NOT NULL,
   generated_by INT NOT NULL,
   FOREIGN KEY (generated_by) REFERENCES users(user_id) ON DELETE
CASCADE
);
CREATE TABLE purchases (
   purchase id INT PRIMARY KEY AUTO INCREMENT,
   distributor_name VARCHAR(100) NOT NULL,
   bill_number VARCHAR(50) UNIQUE NOT NULL,
   total amount DECIMAL(10,2) NOT NULL,
   discount DECIMAL(10,2) DEFAULT 0.00,
   final_amount DECIMAL(10,2) NOT NULL,
   purchase_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```