**Software Requirements Specification (SRS)**

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to describe the requirements for the Food Ordering Web Application. This application allows customers to order food online from various restaurants, and enables restaurants to manage orders, deliveries, and menus.

**1.2 Scope**

This system is a web-based platform with two main user types: **Customers (Buyers)** and **Restaurants**. Customers can browse menus, add items to their cart, and place orders. Restaurants can manage their menu, view and dispatch orders, and track deliveries. The system also includes notification features for new orders.

**1.3 Definitions**

* **Buyer/Customer:** A user who orders food.
* **Restaurant:** A business that receives and fulfills food orders.
* **Order:** A request by a buyer for food items.
* **Delivery:** The process of getting the order to the customer.
* **Menu:** The list of food items offered by a restaurant.

**2. Overall Description**

**2.1 Product Perspective**

The system is a standalone web application built with React (frontend) and Node.js/Express (backend), using MySQL as the database.

**2.2 Product Functions**

* Customer registration and login (with OTP verification).
* Restaurant registration and login.
* Menu management for restaurants.
* Browsing and searching for food items.
* Cart management for customers.
* Order placement and order history for customers.
* Order management for restaurants (view, dispatch, delivered).
* Delivery tracking.
* Notification badge for new orders in the restaurant dashboard.

**2.3 User Classes and Characteristics**

* **Customer:** Can browse, search, add to cart, place orders, and view order history.
* **Restaurant:** Can manage menu, view incoming orders, mark orders as dispatched, and view delivered orders.

**2.4 Operating Environment**

* Web browser (Chrome, Firefox, Edge, etc.)
* Node.js server
* MySQL database

**2.5 Design and Implementation Constraints**

* Responsive design for desktop and mobile.
* Secure handling of user data and authentication.
* Real-time or near real-time updates for order notifications.

**3. System Features**

**3.1 User Authentication**

**3.1.1 Customer Registration and Login**

* Customers can sign up using email, name, and password.
* OTP verification is required for registration.
* Customers can log in using email and password.

**3.1.2 Restaurant Registration and Login**

* Restaurants can register with name, email, password, description, and image.
* Restaurants can log in using email and password.

**3.2 Menu Management (Restaurant)**

* Restaurants can add, edit, and delete menu items.
* Menu items include name, price, and image.
* Menu is displayed to customers for browsing and ordering.

**3.3 Food Browsing and Search (Customer)**

* Customers can view all available restaurants and their menus.
* Customers can search for food items using keywords.

**3.4 Cart Management (Customer)**

* Customers can add menu items to their cart.
* Customers can view, update, or remove items from the cart.
* Cart persists until the order is placed or cleared.

**3.5 Order Placement (Customer)**

* Customers can place an order by providing delivery details (name, address, phone).
* Orders are saved in the database with status "Placed".
* Customers receive confirmation of order placement.

**3.6 Order Management (Restaurant)**

* Restaurants can view all incoming orders for their restaurant.
* Orders display customer details, order items, total, and order date.
* Restaurants can mark orders as "Dispatched".
* Dispatched orders are removed from the active orders list and shown in the delivered orders list.

**3.7 Delivery Management**

* Dispatched orders are shown in the "Delivered" section for the restaurant.
* Delivery details include customer info, delivery person, delivery time, and status.

**3.8 Notifications**

* Restaurants see a notification badge on the "Orders" link in the navbar showing the count of new (un-dispatched) orders.
* The badge updates automatically when new orders are placed or dispatched.

**3.9 Order History (Customer)**

* Customers can view their past orders and order status.

**4. External Interface Requirements**

**4.1 User Interfaces**

* **Responsive Web UI:** Built with React and Bootstrap for both customers and restaurants.
* **Navigation Bar:** Shows relevant links based on user type and login status.
* **Notification Badge:** Shows new order count for restaurants.

**4.2 Hardware Interfaces**

* Standard web browser on PC or mobile device.

**4.3 Software Interfaces**

* **Frontend:** React.js
* **Backend:** Node.js with Express
* **Database:** MySQL

**4.4 Communications Interfaces**

* HTTP/HTTPS for all client-server communication.
* RESTful API endpoints for all operations.

**5. System Architecture**

* **Frontend:** React components for Home, Cart, Orders, Restaurant Dashboard, Delivered, Navbar, etc.
* **Backend:** Express routes for authentication, menu, orders, delivery, notifications.
* **Database:** Tables for users, restaurants, orders, menu items, deliveries.

**6. Functional Requirements**

**6.1 Registration and Login**

* The system shall allow customers and restaurants to register and log in.
* The system shall verify customer registration with OTP.

**6.2 Menu Management**

* The system shall allow restaurants to add, edit, and delete menu items.

**6.3 Order Placement**

* The system shall allow customers to place orders from their cart.
* The system shall save orders in the database with status "Placed".

**6.4 Order Management**

* The system shall allow restaurants to view all orders for their restaurant.
* The system shall allow restaurants to mark orders as "Dispatched".
* The system shall move dispatched orders to the delivered section.

**6.5 Notifications**

* The system shall show a notification badge with the count of new orders for restaurants.
* The system shall update the badge in real time or at regular intervals.

**6.6 Delivery Management**

* The system shall allow restaurants to view delivered orders with all details.

**6.7 Security**

* The system shall securely store passwords (hashed).
* The system shall validate all user inputs.

**7. Non-Functional Requirements**

* **Performance:** The system should respond to user actions within 2 seconds.
* **Scalability:** The system should support multiple restaurants and customers simultaneously.
* **Reliability:** The system should be available 99% of the time.
* **Usability:** The UI should be intuitive and easy to use for all users.
* **Security:** All sensitive data should be encrypted and protected.

**8. Database Requirements**

* **Tables:** buyers, restaurants, orders, menu\_items, delivery.
* **Indexes:** On primary keys and foreign keys for fast lookup.
* **Relationships:** Orders linked to buyers and restaurants; deliveries linked to orders.

**9. Future Enhancements**

* Real-time notifications using WebSockets.
* Payment gateway integration.
* Restaurant analytics dashboard.
* Customer reviews and ratings.

**10. Appendix**

* **Technologies Used:** React, Node.js, Express, MySQL, Bootstrap.
* **Deployment:** Can be deployed on any standard web server with Node.js and MySQL support.

**End of SRS Document**