

# **GIU Food-Truck Reservation System**

---

**Prepared BY:**

**Youssef Ibrahim 13004378 T21**

**Mahmoud Momen 13005416 T21**

**Ahmed Sameh 13005913 T21**

**Omar Tarek 13004881 T21**

**Adham Eletreby 13001491 T20**

**Abdullah Khaled 13004750 T19 (Team Leader)**

**Youssef Gomaa 13005157 T19**

**Ahmed Mostafa 13004273 T8**

## **GIU Food-Truck Reservation System – Updated SRS (Milestone 3 Integrated)**

### **1. Introduction**

This Software Requirements Specification (SRS) document defines the updated functional and non-functional requirements for the GIU Food-Truck Reservation System after incorporating all backend features and API requirements defined in Project Milestone 3.

### **2. Purpose**

The purpose of this system is to allow GIU students and staff to browse food-truck menus, pre-order meals, schedule pickup times, receive order notifications, and allow truck owners to manage menu items and orders. This update adds full support for backend operations, including menu management, cart functionality, order processing, and truck availability control.

### **3. Scope**

The system includes two main roles: Customers and Truck Owners. Customers can browse menus, add items to cart, place scheduled pickup orders, and view order history. Truck owners can manage menu items, track customer orders, and update truck order availability.

### **4. System Architecture**

The architecture includes a web-based frontend, a Node.js backend using Express and Knex, and a PostgreSQL database storing all system entities such as Users, Trucks, MenuItems, Orders, and Carts.

### **5. Functional Requirements (Updated from Milestone 3)**

#### **5.1 User Management**

- Register new users (customers or truck owners).

- Login and maintain sessions using token-based authentication.

## **5.2 Menu Item Management (Truck Owner)**

- Create new menu item (POST /api/v1/menuItem/new).
- View all my menu items (GET /api/v1/menuItem/view).
- View a specific menu item (GET /api/v1/menuItem/view/:itemId).
- Edit a menu item (PUT /api/v1/menuItem/edit/:itemId).
- Delete a menu item by marking it unavailable (DELETE /api/v1/menuItem/delete/:itemId).

## **5.3 Truck Management (Truck Owner)**

- View truck information (GET /api/v1/trucks/myTruck).
- Update truck order availability (PUT /api/v1/trucks/updateOrderStatus).

## **5.4 Truck Browsing (Customer)**

- View all available trucks (GET /api/v1/trucks/view).
- View menu items for a specific truck (GET /api/v1/menuItem/truck/:truckId).
- Search menu items by category (GET /api/v1/menuItem/truck/:truckId/category/:category).

## **5.5 Cart Management (Customer)**

- Add menu item to cart (POST /api/v1/cart/new) with enforcement of single-truck rule.
- View cart (GET /api/v1/cart/view).
- Edit cart item quantity (PUT /api/v1/cart/edit/:cartId).
- Delete item from cart (DELETE /api/v1/cart/delete/:cartId).

## **5.6 Order Management (Customer & Truck Owner)**

- Place order from cart (POST /api/v1/order/new).
- View my orders (GET /api/v1/order/myOrders).
- View order details (GET /api/v1/order/details/:orderId).
- Truck owner views truck orders (GET /api/v1/order/truckOrders).
- Truck owner updates order status (PUT /api/v1/order/updateStatus/:orderId).
- Truck owner views order details (GET /api/v1/order/truckOwner/:orderId).

## **6. Database Schema (From Milestone 3)**

The system uses a PostgreSQL database with required tables: Users, Trucks, MenuItems, Orders, OrderItems, Carts, and Sessions. Each table follows the constraints and relationships defined in the milestone, including primary keys, foreign keys, default timestamps, and status fields.

## **7. Non-Functional Requirements**

- Performance: API responses should be returned within 1 second.
- Security: Token-based authentication, hashed passwords, authorization checks.
- Reliability: All API operations must include error handling.
- Scalability: Database structure supports multiple users and food-trucks.
- Usability: Intuitive UI and clear API responses.

## **8. Constraints**

- Access restricted to GIU campus network users.
- No payment gateway integration in current version.

## **9. Future Scope**

Future updates could include analytics dashboards for tracking vendor performance, payment integration, and mobile app support.