**GIU Food Truck Order System**

**Software Requirements Specification (SRS)**

**1. Introduction**

**1.1 Purpose**

The GIU Food-truck System is a comprehensive web-based application designed to enable the members of the GIU community to reserve menu items from specific food-trucks on campus for pick-up within a given time window, thus alleviating the problem with long-waiting times for food and beverage items during peak periods in between classes. This system enables seamless menu browsing, order placement, and order management for food-truck customers as well as providing a similarly intuitive, efficient, and scalable solution for food service providers.

**1.2 Intended Audience**

* GIU students, faculty, and staff (patrons)
* Food-truck service providers (shop owners)
* System administrators
* Project stakeholders and TAs

**1.3 Scope**

This SRS covers the requirements for the GIU Food-truck System web application, including customer and shop interfaces, order management, and basic administration. It does not include payment processing, external authentication, or integration with third-party systems at this stage.

**1.4 Overview**

The document outlines the product vision, high-level architecture, functional and non-functional requirements, interface requirements, constraints, and future scope.

**2. Product Vision and Scope**

**2.1 Vision**

To streamline food ordering and pickup on campus, reducing wait times and improving the experience for both customers and food-truck operators.

**2.2 Scope**

**The system will:**

* Allow users to register, log in, and select their role (customer or shop)
* Enable customers to browse menus, add items to a cart, and schedule pickup times
* Allow shops to manage menus, view incoming orders, and update order statuses
* Provide real-time order status updates and notifications

**The system will NOT:**

* Process online payments (future scope)
* Integrate with external authentication or credit card services (future scope)
* Support delivery (pickup only)

**3. High-Level Architecture**

* **Frontend:** React.js (SPA), HTML5, CSS3
* **Backend:** Node.js with Express.js (RESTful API)
* **Database:** MongoDB (Mongoose ODM)
* **Authentication:** JWT-based (local, no external providers)
* **Hosting:** To be determined (local for development)
* **APIs:** RESTful endpoints for all major operations

**Architecture Diagram:** **[Browser/React] <-> [Node.js/Express API] <-> [MongoDB]**

**4. Functional Requirements**

**4.1 User Stories**

**Food-truck Patron (Customer)**

* As a customer, I want to register and log in so I can place orders.
* As a customer, I want to browse food-truck menus and see item details.
* As a customer, I want to add items (with extras) to my cart.
* As a customer, I want to select a pickup time within allowed hours.
* As a customer, I want to view my order history and status.
* As a customer, I want to receive notifications when my order is being prepared and when it is ready.

**Food-truck Service Provider (Shop)**

* As a shop, I want to register and log in to manage my food-truck.
* As a shop, I want to add, edit, or remove menu items and set preparation times.
* As a shop, I want to view incoming orders, see customer pickup times, and update order statuses (pending, preparing, ready, completed, cancelled).
* As a shop, I want to be notified when it’s time to start preparing an order.

**Administrator**

* As an admin, I want to view all users and orders for monitoring and support.

**4.2 Use Case Diagram**

A diagram of a product order system

AI-generated content may be incorrect.

**<<include>>**



**5. Non-Functional Requirements**

* **Performance:** Page load times under three seconds.
* **Scalability:** Support at least 500 simultaneous users.
* **Security:** JWT authentication, input validation, rate limiting.
* **Reliability:** Automated daily database backups, disaster recovery plan.
* **Usability:** Responsive design for mobile and desktop.
* **Operating Hours:** System available 24/7; food-truck order windows configurable by shop.

**6. Interface Requirements**

* **Internal:** RESTful API between frontend and backend.
* **External:** No integration with payment gateways, external authentication, or third-party systems at this stage.
* **Possible Extensions:** Integration with payment gateways, SMS/email notification services, and university authentication in the future.

**7. Constraints**

* Only GIU community members may register as customers.
* Only authorized food-truck operators may register as shops.
* Pickup times must be within shop-defined operating hours.
* No online payment or external authentication in this version.

**8. Future Scope of the Project**

* Online payment integration (credit/debit cards, mobile wallets)
* University SSO authentication
* Real-time chat between customer and shop
* Advanced analytics for shop owners
* Integration with campus map for food-truck locations
* Push notifications (mobile/web)
* Support for delivery orders