**Integrated Culinary Services Platform**

**Table of Contents**

1. **Introduction**
   * Purpose
   * Scope
   * Overview
   * References
2. **Overall Description**
   * Product Perspective
   * Product Features
   * User Classes and Characteristics
   * Operating Environment
   * Constraints
3. **System Requirements**
   * Functional Requirements
   * Non-functional Requirements
4. **System Design**
   * Modules Overview
   * Database Design
5. **UML Diagrams**
   * Use Case Diagram
   * Data Flow Diagram (Level 0 and Level 1)

**1. Introduction**

**Purpose**

The **Integrated Culinary Services Platform** aims to digitize the operations of a food court, providing seamless user experiences from browsing menus to placing orders and managing them efficiently. The system offers a foundation for future enhancements like payment integration and feedback mechanisms.

**Scope**

The FCMS application facilitates:

* User authentication and profile management.
* Onboarding processes to introduce application features.
* Easy menu browsing and order management.
* Modular design to incorporate future features like payment gateways and feedback systems.

**Overview**

The system is structured around Firebase as the backend, with a Flutter-based frontend for cross-platform usability. The project integrates Firebase Authentication, Firestore Database, and intuitive UI/UX designs.

**2. Overall Description**

**Product Perspective**

The system replaces traditional token-based manual systems with an automated, efficient digital platform. It connects food court vendors and users in a centralized application, ensuring transparency and efficiency.

**Product Features**

* **Authentication Module**: User login and registration with Firebase.
* **Onboarding Module**: Interactive screens introducing app features.
* **Menu Module**: Comprehensive browsing of food items with prices, descriptions, and images.
* **Order Management**: Streamlined order placement and tracking.
* **Navigation Module**: Bottom navigation bar for easy access to core modules.

**User Classes and Characteristics**

1. **Customers**: Users browsing menus and placing orders.
2. **Food Court Administrators**: Oversee menu and order management.
3. **Developers**: Extend system functionality.

**Operating Environment**

* **Frontend**: Flutter framework (Dart language).
* **Backend**: Firebase services (Authentication, Firestore).
* **Target Platforms**: Android, iOS, and Web.

**Constraints**

* Internet connectivity is required for Firebase operations.
* Limited to devices supporting Flutter apps.

**3. System Requirements**

**Functional Requirements**

* **User Authentication**: Login and sign-up using Firebase.
* **Order Placement**: Users can browse menus and place orders.
* **Order Tracking**: Real-time order status updates.
* **Onboarding Screens**: Introduce features to new users.

**Non-functional Requirements**

* **Performance**: Must handle 100 concurrent users without degradation.
* **Scalability**: Accommodate additional features like payment gateways.
* **Usability**: Intuitive UI for users of varying technical proficiency.
* **Security**: Protect user data using Firebase security rules.

**4. System Design**

**Modules Overview**

1. **Authentication Module**:
   * Handles user registration and login.
   * Firebase Authentication ensures secure access.
2. **Onboarding Module**:
   * Guides new users through app features.
   * Multi-screen flow with descriptions and visuals.
3. **Menu Module**:
   * Displays food items with images, prices, and descriptions.
   * Filter and search functionalities.
4. **Order Management Module**:
   * Allows users to place and track orders.
   * Firebase Firestore stores order details.
5. **Navigation Module**:
   * Bottom navigation bar connects Home, Orders, and Profile tabs.

**Database Design**

* **Users**:
  + Fields: user\_id, name, email, password, profile\_image
* **Food Items**:
  + Fields: item\_id, name, price, image\_url, description
* **Orders**:
  + Fields: order\_id, user\_id, items, total\_price, status

**5. UML Diagrams**

**Use Case Diagram**

(Attach a visual diagram showing relationships between users and system functionalities.)

**Data Flow Diagram**

* **Level 0**: Overall process flow (User interaction with the system).
* **Level 1**: Detailed flow for modules like Authentication and Order Management.

(Include visual diagrams for both levels.)

**Outcomes of the Project**

* A secure and efficient food court management app.
* Simplified user interactions through intuitive designs.
* Foundation for future enhancements like online payments and feedback integration.
* A scalable architecture suitable for expanding food courts or similar applications.

**References**

1. Flutter Documentation: <https://flutter.dev/docs>
2. Firebase Documentation: <https://firebase.google.com/docs>
3. Material Design Guidelines: <https://material.io/design>