UNIVERSITY NAME

## 

# **Food Farma**

## 

## **Design Document**

## 

## STUDENT NAME

## Table of contents

[**1. High-Level Design 3**](#_8xdxampi3b40)

[1.1 Overview 3](#_73xstnsjzaia)

[1.2 System Architecture 3](#_jdqikco9m6rk)

[1.3 Major Modules 5](#_c87llqyjqrj3)

[1.3.1 User Management Module 5](#_wlq584nhbg84)

[1.3.2 Product Management Module 6](#_5qlgfldx1juo)

[1.3.3 Order Management Module 7](#_suc3us2iyj60)

[1.3.4 Payment Processing Module 8](#_29u1n08ih0xm)

[1.3.5 Inventory & Analytics Module 8](#_pyacz0xkvasw)

[1.3.6 Marketing & Promotions Module 9](#_r5yzeip4x55e)

[**2. Detailed Design 11**](#_bmho8k2od3qj)

[2.1 Database Schema Overview 11](#_6wu4doitbosc)

[2.2 API Design Overview 12](#_pkojf5wkcbwz)

[**3. Conclusion 13**](#_bw8d5wyd621u)

## 

# 1. High-Level Design

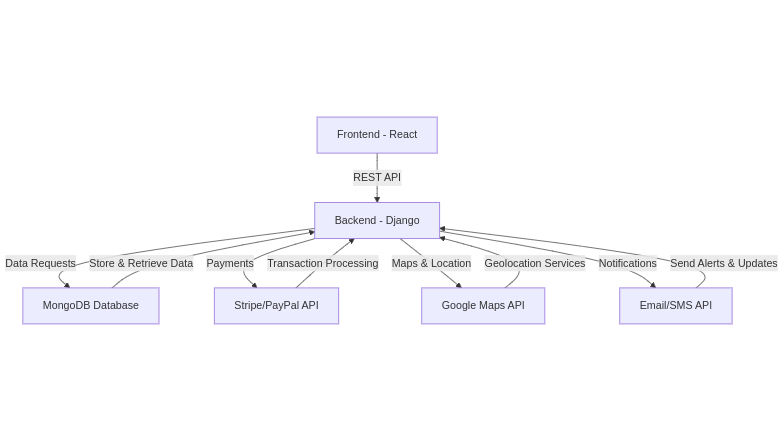
## 1.1 Overview

FoodFarma is an online platform designed to create a direct connection between local farmers and consumers. This platform enables farmers to showcase their products online, manage inventory, and fulfill orders efficiently while allowing consumers to browse, purchase, and track their orders seamlessly. The system follows a structured three-tier architecture comprising the frontend (React), backend (Django), and database (MongoDB), with integrations for payment processing, geolocation services, and notifications. Each of these components interacts through well-defined interfaces, ensuring smooth data flow and user experience.

## 1.2 System Architecture

The architecture of FoodFarma follows a modular approach, ensuring scalability and maintainability. The key components are:

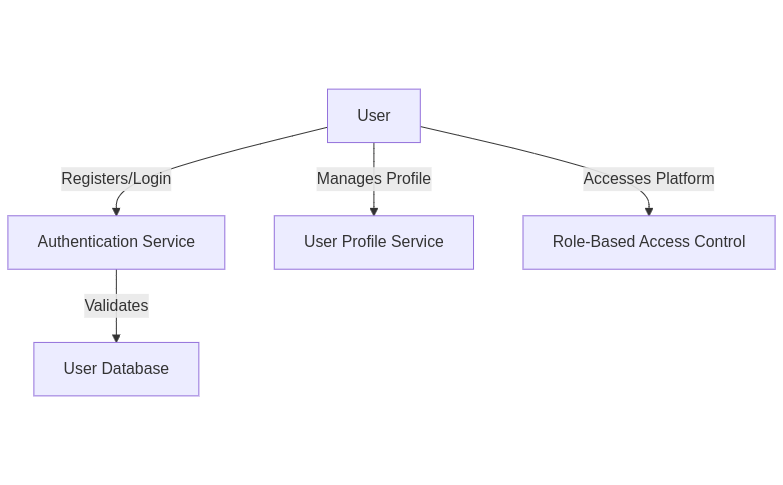
* **Frontend (React):** This is responsible for rendering the user interface, handling user interactions, and making API calls to the backend.
* **Backend (Django):** Acts as the core logic processor, handling business rules, API requests, authentication, and database interactions.
* **Database (MongoDB):** Stores information about users, products, orders, and transactions efficiently.
* **External APIs:** Includes integrations with Stripe/PayPal for payments, Google Maps API for geolocation, and Email/SMS APIs for notifications.



1. The Frontend (React) sends API requests to the Backend (Django) for user authentication, product retrieval, and order placement.
2. The Backend (Django) processes these requests, interacts with the MongoDB Database for data retrieval and storage, and returns responses to the frontend.
3. When a payment is initiated, the backend securely communicates with Stripe/PayPal APIs to handle transactions.
4. If an order requires location services (e.g., delivery tracking), the backend calls the Google Maps API to fetch geolocation data.
5. The backend also sends notifications (order updates, promotional messages) via Email/SMS APIs, ensuring timely communication with users.

## 1.3 Major Modules

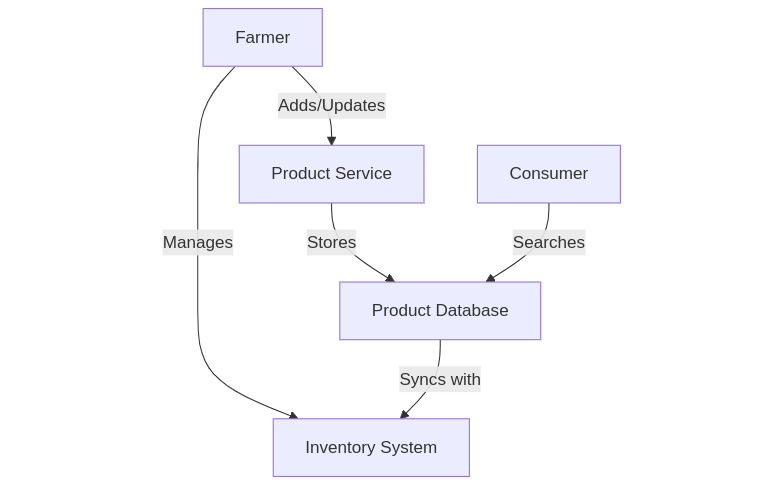
### 1.3.1 User Management Module

The User Management Module is responsible for authentication, authorization, and role-based access control. The system differentiates between farmers, consumers, and administrators, each with unique privileges. Farmers can list and manage products, consumers can browse and purchase items, and administrators oversee the entire system to ensure smooth operations.

1. Users interact with the authentication service to register and log in.
2. The authentication service validates credentials against the user database.
3. Role-based access control (RBAC) ensures that users can only access permitted functionalities based on their role (Farmer, Consumer, Admin).
4. The user profile service allows users to update personal details, addresses, and preferences.

### 1.3.2 Product Management Module

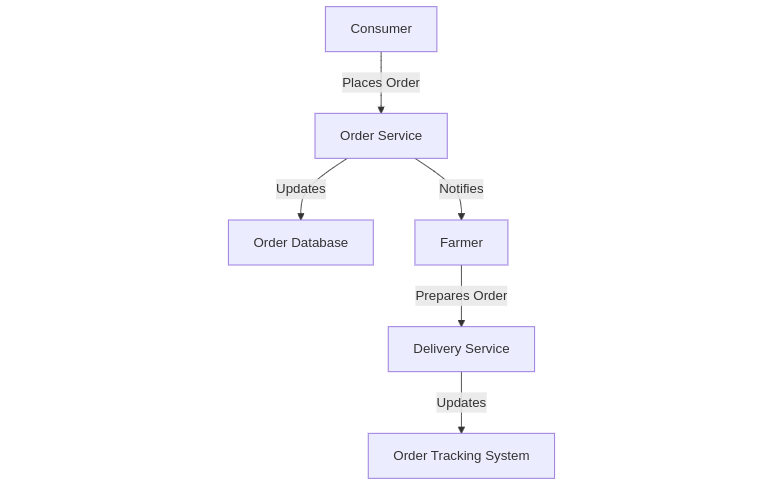
The Product Management Module allows farmers to create, update, and remove product listings. It includes options for adding product descriptions, images, categories, and stock levels. This module ensures data consistency by preventing stock depletion issues and providing real-time updates on product availability.



1. Farmers add, update, or remove products through the Product Service.
2. The Product Service stores all product details in the Product Database.
3. Farmers manage their inventory, and stock levels are updated in real-time.
4. Consumers can browse, search, and filter products.

### 1.3.3 Order Management Module

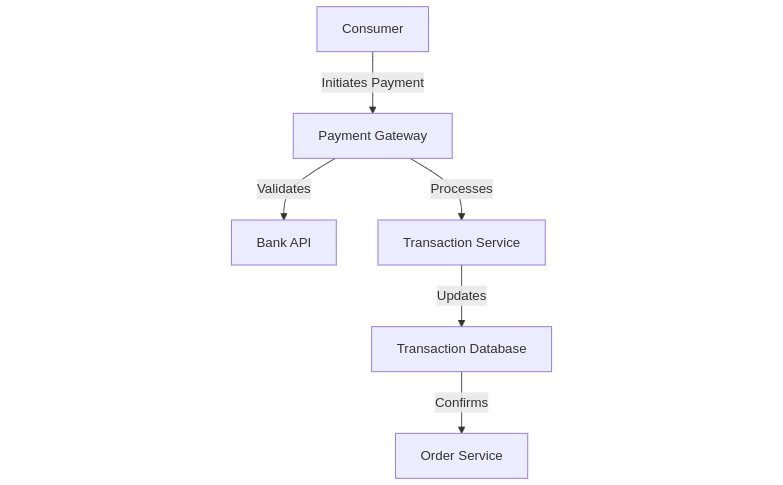
The Order Management Module handles order processing from cart selection to order fulfillment. When a consumer places an order, the system records the transaction, updates the inventory, and notifies the farmer. Consumers receive real-time order tracking updates, and the system allows order modifications before shipment.



1. Consumers place orders, which are processed by the Order Service.
2. The Order Database maintains order records.
3. Farmers receive notifications for new orders and initiate fulfillment.
4. The Delivery Service updates order status, ensuring real-time tracking for consumers.

### 1.3.4 Payment Processing Module

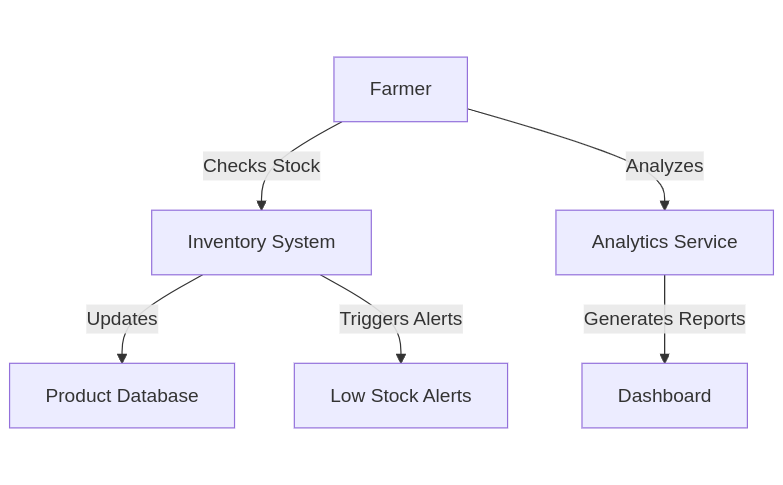
The Payment Processing Module integrates secure online payment gateways (Stripe/PayPal). It ensures encrypted transactions, maintains transaction histories, and supports refunds in case of order cancellations. The module adheres to financial compliance and security standards.



1. Consumers initiate payments via an integrated Payment Gateway (Stripe/PayPal).
2. The gateway validates transactions with the Bank API.
3. The Transaction Service records the payment details in the database.
4. Once confirmed, the Order Service proceeds with fulfillment.

### 1.3.5 Inventory & Analytics Module

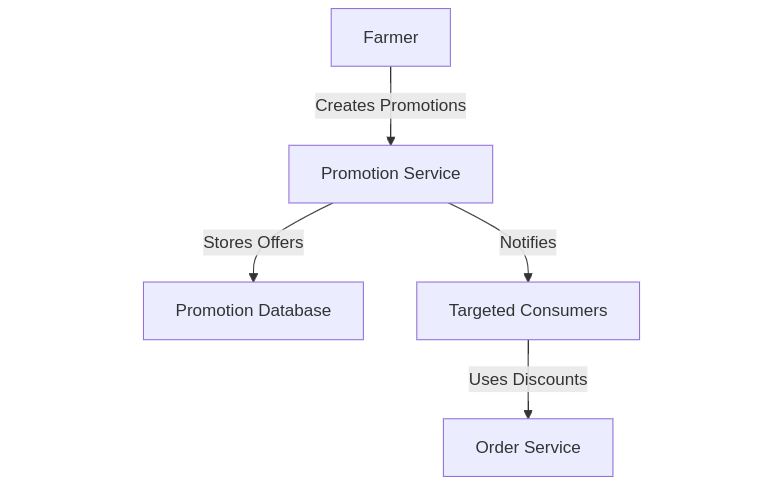
The Inventory & Analytics Module enables farmers to monitor stock levels, set alerts for low inventory, and access sales performance reports. The analytics feature provides insights into consumer trends, seasonal demand fluctuations, and revenue patterns, allowing farmers to make data-driven decisions.



1. Farmers monitor inventory levels and receive alerts for low stock.
2. The Inventory System syncs with the Product Database to keep stock levels updated.
3. The Analytics Service provides insights into sales trends and revenue.
4. A dashboard visualizes key business metrics, helping farmers make data-driven decisions.

### 1.3.6 Marketing & Promotions Module

The Marketing & Promotions Module empowers farmers to attract and retain customers through discounts, coupons, and loyalty programs. It also supports targeted promotions based on consumer purchase history and preferences.



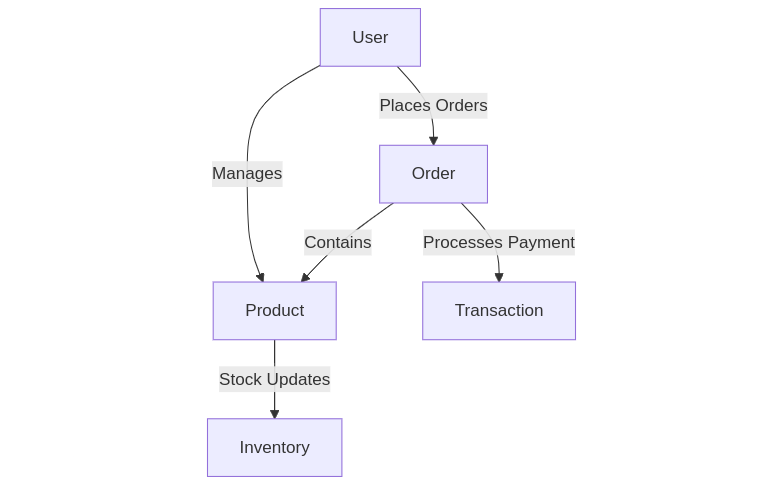
1. Farmers create and manage discounts, coupons, and loyalty programs.
2. The Promotion Database stores these promotional offers.
3. Consumers receive targeted offers based on their purchase history.
4. Discounts are applied at checkout through the Order Service.

# 

# 2. Detailed Design

## 2.1 Database Schema Overview

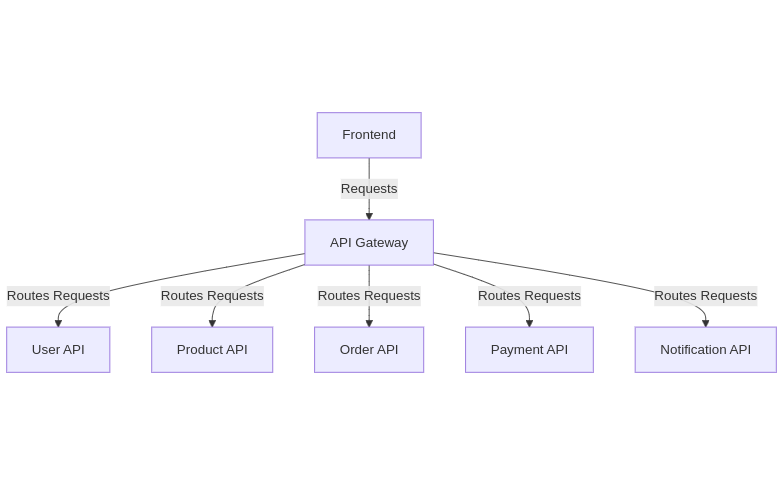
The FoodFarma database is designed to store user details, product information, orders, and transactions efficiently. The schema ensures fast retrieval of data and maintains consistency across different system components. The key entities include Users, Products, Orders, and Transactions.



1. Users are the core entities in the system, classified as farmers, consumers, or administrators.
2. A User places an Order, initiating the transaction process.
3. Orders contain Products, allowing multiple items in a single order.
4. When an order is confirmed, a Transaction is processed, handling payment securely.
5. Products are linked to Inventory, ensuring stock levels are updated in real time when purchases are made.

## 2.2 API Design Overview

The API design follows a RESTful architecture, allowing seamless communication between the frontend and backend. It includes endpoints for managing users, products, orders, payments, and notifications.



1. The Frontend interacts with the API Gateway, which acts as a single entry point for all requests.
2. Requests are routed to their respective APIs:
   * **User API:** Handles authentication, user profiles, and role-based access control.
   * **Product API:** Manages product listings, categories, and stock updates.
   * **Order API:** Processes orders, including cart management and order tracking.
   * **Payment API:** Handles transactions, refunds, and payment verification.
   * **Notification API:** Sends emails, SMS updates, and promotional alerts.

# 3. Conclusion

FoodFarma is designed to offer an efficient and scalable platform connecting local farmers with consumers. The high-level design focuses on major system components, while the detailed design ensures robust functionality through well-defined database structures, API endpoints, and modular architecture. Future enhancements will include AI-driven recommendations, predictive analytics, and enhanced security measures to further optimize user experience and operational efficiency.