GIS TRACKS FOODS PRODUCTS SYSTEM

Software Analysis and Design Document

MEMBERS:
IAN NDOLO MWAU - SCT 211-0034/2022
DAVID NZAMBULI - SCT 211-0068/2022
NEEMA OGAO – SCT211-0086/2022
MAUREEN NYAGA- SCT211-0052/2022
PHARIS KARIUKI-SCT211-0033/2022

INTRODUCTION

As the agricultural industry grows more complex and data-driven, the need for transparent, efficient, and intelligent supply chain solutions has become critical. This document presents the design and analysis of a GIS-based Food Tracking System aimed at enhancing traceability, accountability, and coordination across the agricultural supply chain.

The system harnesses the power of Geographical Information Systems (GIS) and modern digital tools to provide real-time visibility of food movement from farm to consumer. By integrating GPS tracking, the system accurately identifies farm locations, processing centers, distribution hubs, and transport routes, ensuring the timely delivery of fresh, high-quality produce.

This solution offers an interactive, user-friendly platform that visualizes the entire supply chain on a dynamic map interface. It allows administrators to monitor delivery progress, enables transporters to update location statuses, and provides farmers and other stakeholders with insights into how and where their produce moves. This transparency enhances inventory management, reduces waste, and helps align food supply with shifting market demands.

In this document, we outline the methodology used to analyze user requirements, define system functionalities, and design the key architectural components of the application. The goal is to lay a solid foundation for a scalable, adaptable, and sustainable solution. Ultimately, the GIS-based tracking system is designed to empower all participants in the agricultural value chain, creating a comprehensive, data-driven ecosystem that adds value at every stage.

FUNCTIONAL REQUIREMENTS

1. **User Management Module**

This module manages all user-related functions including registration, authentication, and access control.

- **User Registration & Authentication**
  - Users (Farmers, Transporters, Admins, Retailers) can register with personal and role-specific details.
  - The system supports secure login and logout functionalities.
  - Users can reset or recover passwords securely.
- **Role-Based Access Control (RBAC)**
  - **Admins:** Full system control including user, farm, and transport management.
  - **Farmers:** Manage their farm data and track produce movement.
  - **Transporters:** View assigned deliveries and update delivery statuses.
  - **Retailers (Optional):** View supply chain history and delivery statuses.

2. **Location Management Module**

Manages the creation and classification of physical locations involved in the supply chain.

- Users can add new locations by specifying:
  - Name, label/type (e.g., Farm, Processing Facility, Warehouse, Distribution Center, Supermarket, Restaurant)
  - Geographic coordinates and associated region(e.g., Central,Northern etc)
- Enhances visibility and traceability through integration with geospatial data systems.

3. **Farm Management Module**

Allows farmers to manage and update their farm-related information.

- Add and update farm profiles including:
  - Farm name, size, region, crop/livestock details, ownership
- Track ongoing farming activities and available produce
- Link farms to relevant locations in the supply chain for movement tracking

4. **Location Search & Filter**

Provides a shared interface for searching and filtering farms and locations across the supply chain.

Key Features:

- **Search by Name**
  - Keyword-based search for farms and locations.
- **Location Filters**

    - Filter by:
        - Region
        - Label (e.g., Farm, Processing Facility, Distribution Centre, Warehouse, Supermarket, Restaurant)
- **Filtered Location Results**
    - Displays number of matching locations.
    - Dropdown selection for easy access.
- **Farm Filters**
    - Filter by:
        - Farmer Name
        - Type of Produce
- **Filtered Farm Results**
    - Displays matching farms.
    - Enables selection for further actions.

5. **Geospatial Data & Mapping Module**

Integrates GIS technology to enable spatial tracking and real-time visibility of the food supply chain.

- **Farm Location Tracking**
    - GPS integration for locating and displaying farms on an interactive map
- **Facility and Route Mapping**
    - Visual representation of processing facilities and distribution centers
    - Mapping of transport routes and checkpoints
- **Real-Time Transport Tracking**
    - Monitor vehicle location, delivery status, and estimated arrival times
    - Status indicators for goods in transit

6. **Transport and GPS Tracking Module**

Manages delivery logistics and enables real-time tracking of produce transport for improved efficiency and visibility.

Key Features:

- **Delivery Scheduling & Assignment**
    - Assign delivery tasks with defined routes and vehicle details.
- **Delivery Status Updates**
    - Transporters can update progress (e.g., in transit, delivered, delayed).

- **Proof of Delivery**
    - Upload supporting evidence like images or digital signatures.
- **Live GPS Tracking**
    - Display vehicle locations and delivery progress on a live map.
    - Provide estimated arrival times using real-time GPS and traffic data.
- **Route Monitoring**
    - Detect route deviations and trigger alerts if needed.
- **Delivery History and Logs**
    - Store and access historical delivery data for auditing and performance analysis.
- **Performance Reports**
    - Generate analytics and reports on logistics efficiency.

7. Admin Management Module

Designed for administrators to manage users, monitor the system, and ensure smooth operation.

- **User & Role Management**
    - Add, remove, or edit users and assign system roles
- **System Monitoring**
    - Track system health, activity logs, and usage stats
- **Audit & Reports**
    - Access full reports on user actions, system events, and platform analytics
    - Ensure compliance and transparency through activity auditing

8. Mobile and Web Interface Module

Ensures accessibility and responsiveness across devices and user platforms.

- **Responsive Web Platform**
    - Fully functional on desktop, tablet, and web browsers
    - Enable farmers and transporters to access features on the go
    - Real-time notifications, delivery updates, and GPS tracking

NON-FUNCTIONAL REQUIREMENTS

1. **Performance**:
   - The system shall handle growing concurrent users without significant performance degradation.
   - Data retrieval and page loading times shall not exceed 3 seconds under typical usage conditions.
2. **Scalability**:
   - The system shall support scalability to accommodate an increasing number of users and data volume, ensuring performance remains consistent.
   - It shall be able to scale horizontally to handle additional loads.
3. **Availability**:
   - The system shall be available 99.9% of the time, excluding scheduled maintenance.
   - It shall provide automatic failover in case of system or server failures to ensure minimal downtime.
4. **Reliability**:
   - The system shall provide reliable data storage and backup mechanisms to prevent data loss.
5. **Security**:
   - All user data (personal, farm, and location details) shall be encrypted both at rest and in transit.
   - The system shall implement role-based access control (RBAC) to ensure secure access to sensitive data.
   - The system shall include protection against common security threats such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).
   - It shall implement secure user authentication and password recovery mechanisms.
6. **Usability**:
   - The system shall have an intuitive and user-friendly interface for farmers, transporters, and admins, requiring minimal training.
   - The system shall support mobile devices and be optimized for a responsive design, ensuring accessibility on smartphones and tablets.
7. **Maintainability**:
   - The system's architecture shall allow for easy maintenance, updates, and bug fixes without significant downtime.

- The system shall have clear documentation for developers and admins to facilitate troubleshooting and system updates.

8. **Interoperability**:
   - The system shall be able to support integration via REST/GraphQL APIs and Kafka pipelines.
9. **Compliance**:
   - The system shall comply with relevant legal and regulatory standards regarding data privacy and protection (e.g., GDPR, local data protection laws).
   - It shall follow industry best practices for secure handling and processing of sensitive user data.
10. **Audit and Logging**:
    - The system shall log all user actions related to critical activities (e.g., adding/editing farm data, location management, user registration).
    - Logs shall be stored securely and accessible for auditing purposes.

<u>SOFTWARE AND HARDWARE REQUIREMENTS</u>

Software Requirements

Frontend:

- React.js Framework
- Jest Testing Framework
- HTML5, CSS3, JavaScript
- Axios for API consumption

Backend:

- Django
- Django REST Framework
- PostgreSQL for relational data

Security:

- OAuth2 Authentication
- SSL/TLS Encryption

DevOps/Cloud Tools:

- Azure Cloud for hosting and optimization
- Docker (for containerization)
- GitHub (code repository & CI/CD)

Hardware Requirements

Client-Side (End Users)

- **Devices**: PC, tablet, or smartphone with internet access.
- **Screen Resolution**: Minimum 1280x720 (HD).
- **Processor**: 2.0 GHz dual-core or better for desktops; quad-core for mobile.
- **RAM**: At least 4 GB.
- **Storage**: Minimum 2 GB of free space.
- **OS**: Windows 10/11, macOS, Linux, Android 8.0+, iOS 12.0+.
- **Browser**: Chrome, Firefox, Edge, or Safari with JavaScript and WebGL support.
- **Network**: 2 Mbps internet speed for optimal performance.

Server-Side (For Hosting & Backend)

- **Processor**: Intel Xeon E5 or AMD EPYC, with 8-32 cores depending on load.
- **RAM**: Minimum 16 GB, recommended 32-64 GB for heavy traffic.
- **Storage**: SSDs with 500 GB minimum; additional for data retention.
- **Network**: 1 Gbps bandwidth with load balancing for availability.
- **OS**: Windows or Linux (Ubuntu Server or CentOS).
- **Database Servers**: PostgreSQL (4 cores, 16 GB RAM minimum)
- **Cloud Infrastructure**: Azure for scaling.

Development & Testing Environments

- **Dev Machine**: Intel i5, 8 GB RAM, 256 GB SSD.
- **CI/CD Tools**: GitHub Actions for automated testing and deployment.

SYSTEM ARCHITECTURE

System Architecture Description

The system architecture for the GIS-based food tracking system employs a layered approach to ensure separation of concerns, scalability, and maintainability.

Presentation Layer

This top-most layer provides user interfaces for different stakeholders:

- **Web Interface**: Browser-based access for all users
- **Mobile Interface:** Optimized interface for field operations, especially useful for drivers updating location data
- **Admin Dashboard**: Specialized interface for system administrators to manage users and system settings
- **GIS Visualization**: Interactive mapping interface to display spatial data related to farms and asset tracking

Application Layer

This layer contains the core business logic modules:

- **Authentication Module**: Manages user verification, role assignment, and access control
- **Farm Management Module**: Handles farm location data, operational information, and data updates
- **Asset Tracking Module**: Processes real-time location data, route optimization, and historical tracking

Service Layer

This middleware layer provides API interfaces and essential services:

- **API Gateway**: Central entry point for client applications, handling request routing and load balancing
- **GIS Services**: Specialized services for spatial data processing and geographic calculations
- **Data Services**: Manages data access, transformation, and validation
- **Security Services**: Implements security protocols including encryption and data protection

Data Layer

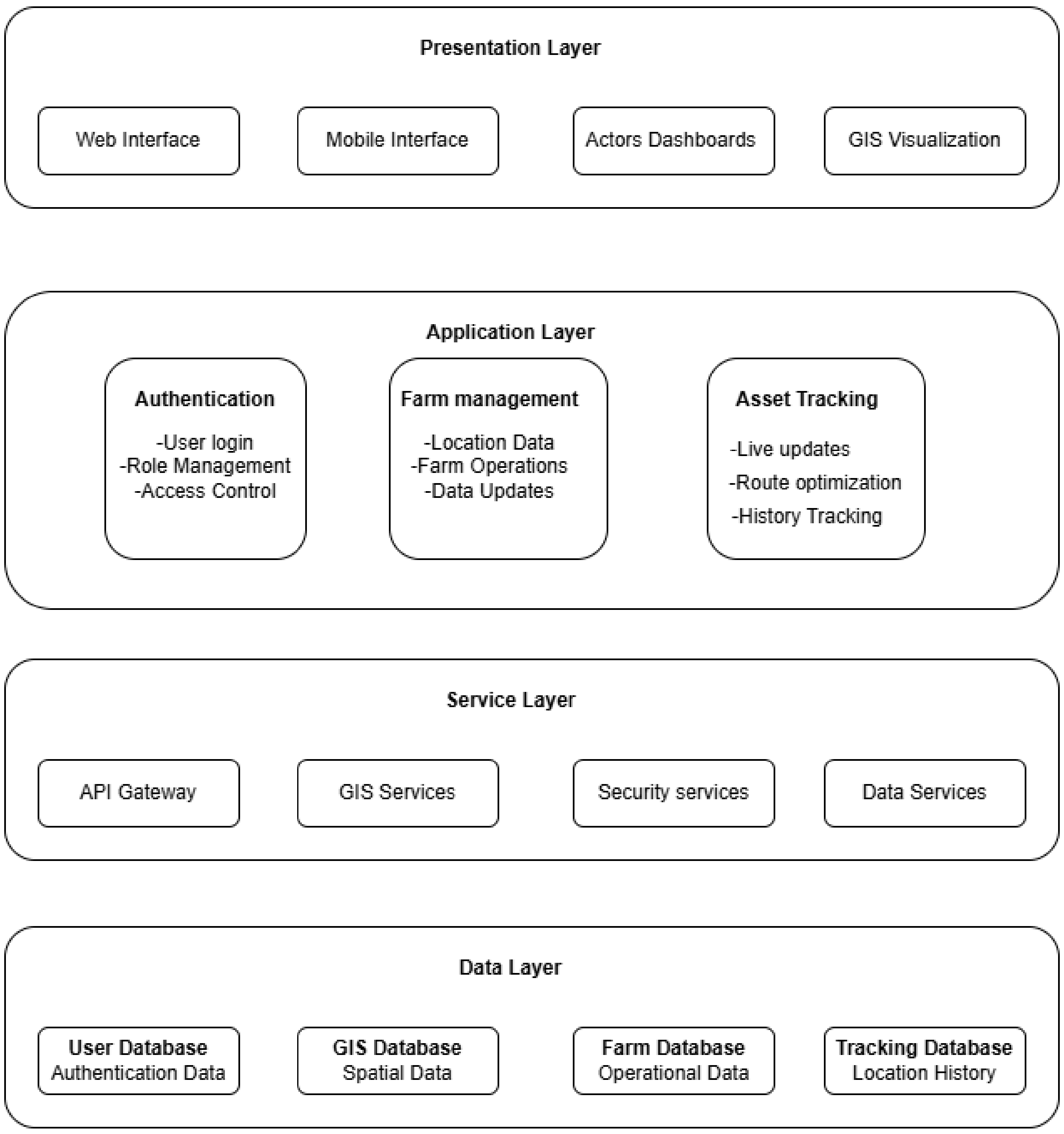
The foundation layer that manages persistent storage:

- **User Database**: Stores authentication data, user profiles, and role assignments

- **GIS Database**: Contains spatial data including farm boundaries, transport routes, and location histories
- **Farm Database**: Maintains farm operational data, production details, and related information
- **Tracking Database**: Archives real-time location data and movement histories for transport assets

DIAGRAM:

GIS- based Food Tracking System Architecture



Context Model

The Context Model diagram illustrates how the GIS-based Food Tracking System interacts with various stakeholders and external systems, showing its place within the broader ecosystem:

Primary Actors

1. **Administrator**: The central authority who manages the system, including user roles, system configuration, and overall data integrity. Administrators have the highest level of access and responsibility for maintaining system functionality.
2. **Farmer**: A key user who interacts with the system to update farm locations, production data, and operational information. Farmers rely on the system to map their properties and manage agricultural data.
3. **Driver**: The mobile stakeholder who provides real-time updates on transport vehicle locations and status. Drivers are crucial for maintaining the live tracking component of the supply chain.

External Systems

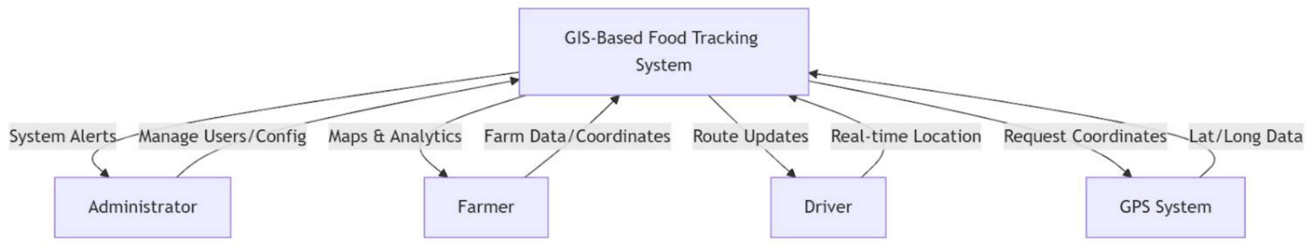
1. **GPS System**: An external technology service that provides precise location data for transport vehicles. This is essential for the asset tracking functionality.
2. **Google Maps API**: An external resource from google leveraged to provide map styles and layers and also navigation information.

Key Interactions

- **Administrators manage** the central system through configuration, user management, and system maintenance
- **Farmers update** the system with agricultural data and farm locations
- **Drivers track** assets by providing real-time location updates
- The system **integrates with GPS** for precise location tracking
- The system **provides data to customer portals** for supply chain transparency

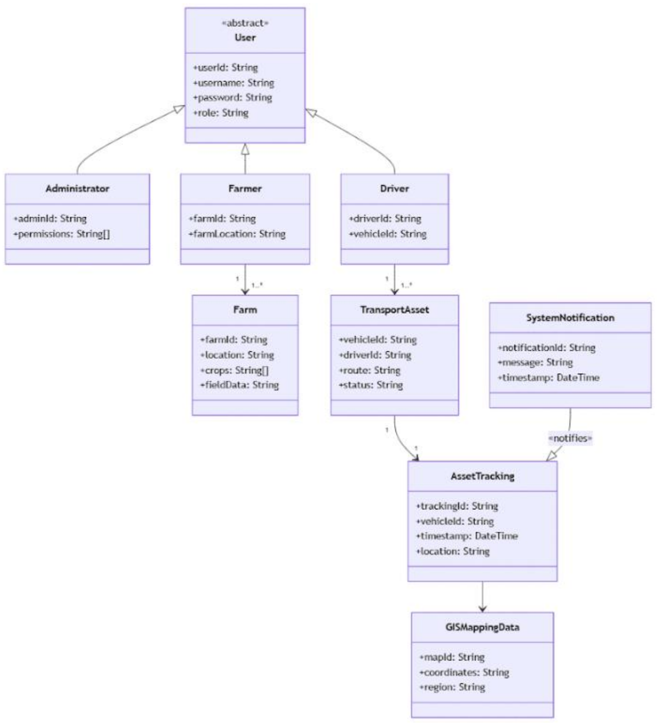
This context model demonstrates how the GIS-based Food Tracking System serves as the central hub connecting human stakeholders with technological systems to create a comprehensive food tracking ecosystem.

Diagram:



Class diagram

The Class Diagram outlines the core classes of the GIS-based food tracking system and their relationships. It showcases the primary entities, such as users (Administrator, Farmer, and Driver), and the classes representing farm data, transport assets, asset tracking, and system notifications. By defining the attributes and methods associated with each class, this diagram offers a structured view of the system's data model, making it easier to understand how the different components of the system interact and how data is stored and processed within the platform. It serves as a blueprint for the implementation of the platform's backend system.



<u>INTERACTIONAL MODELS</u>

Use Case Diagram

This use case diagram serves as a visual guide to understand the interactions between the key stakeholders (Administrator, Farmer, and Driver) and the GIS-based food tracking system.

Actor Descriptions:

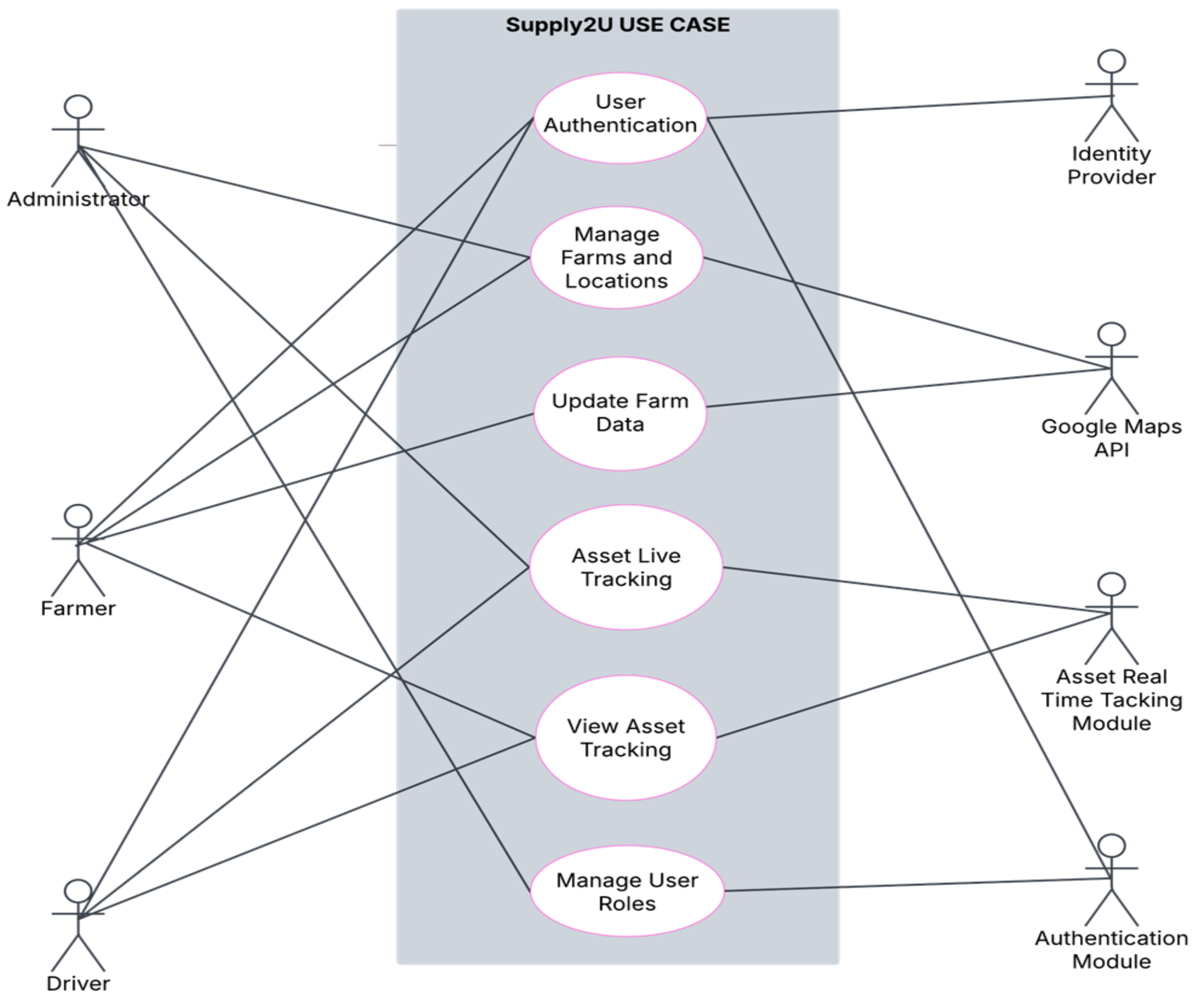
Include a section that provides detailed descriptions of each actor depicted in the diagram:

- **Administrator:** Responsible for system configuration, managing user roles, and maintaining overall data integrity. This role ensures that every component of the system functions securely and efficiently.
- **Farmer:** Uses the system to map and manage farms, update operational details, and ensure that all agricultural data is current. This actor benefits from functionalities that support optimal production practices.
- **Driver**: Engages with the system for live tracking of transport assets. Drivers update real-time location data, which is crucial for monitoring the secure and timely transportation of food products.

Use Case Explanations:

- **User Authentication:** Describes the process that ensures secure access for all users. Every interaction with the system is preceded by a robust authentication process to maintain security.
- **Manage Farms & Locations:** Details the functionality that allows farmers to add, update, or remove information related to farms, processing facilities, warehouses, and retail locations.
- **Update Farm Data:** Outlines how farmers can modify and maintain operational data related to their farming practices, ensuring the production process is well-documented and optimized.
- **Asset Live Tracking:** Explains how drivers update the system with real-time location data of transport vehicles, facilitating immediate tracking and monitoring of the food supply chain.
- **View Asset Tracking:** Covers how stakeholders can access real-time and historical tracking information, enhancing the visibility and reliability of the transportation process.
- **Manage User Roles:** Highlights the administrative functionality that allows for the creation, assignment, and modification of user roles, ensuring that only authorized personnel have access to specific system functions.

Diagram Integration:



Sequence diagrams

This section illustrates the dynamic interactions among the various system components and actors using sequence diagrams. These diagrams provide a step-by-step visual representation of how key operations within the GIS-based food supply chain system are executed. Below, you will find detailed descriptions and accompanying sample diagrams for five primary functionalities.

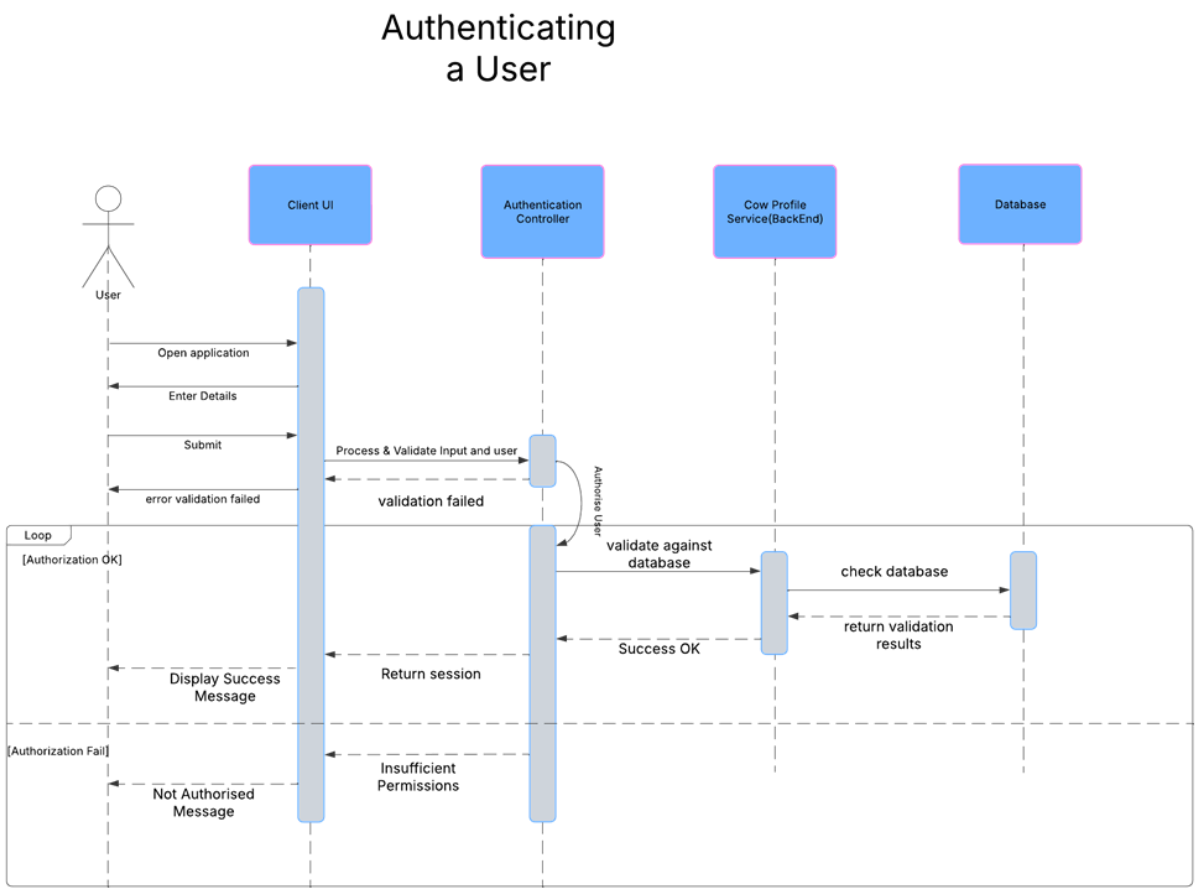
a. **User Authentication Sequence Diagram**

This sequence diagram demonstrates the authentication process that every user must undergo before accessing the system. It shows the flow of credentials from the user to the system, the validation against the user database, and the establishment of a user session.

Sequence Flow:

- **User:** Initiates login by providing credentials.
- **Client Interface:** Forwards the credentials to the Authentication Controller.
- **Authentication Controller:** Validates the credentials by consulting the User Database.
- **User Database:** Returns the validation results.
- **Session Manager:** Creates a session if the credentials are valid.
- **Client Interface:** Displays a dashboard or error message based on the outcome.

Diagram:



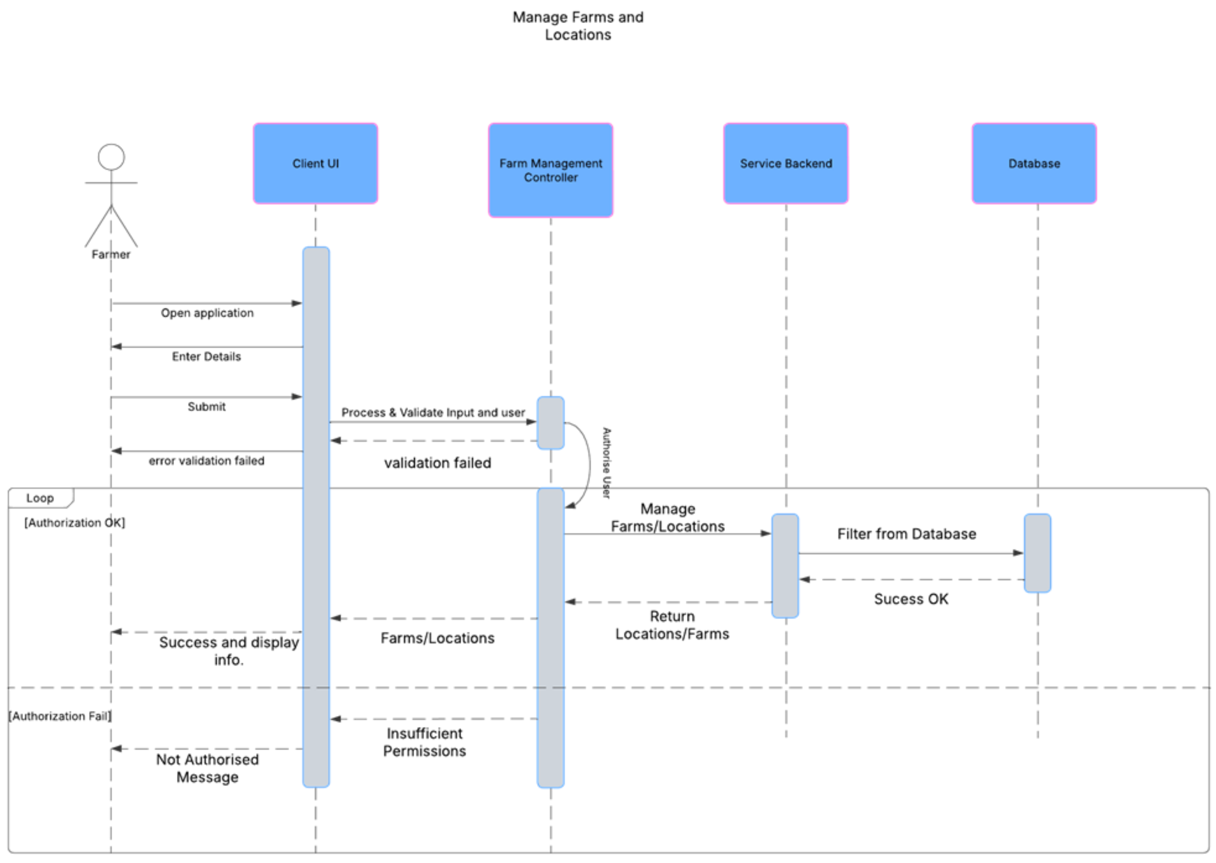
b. Manage Farms & Locations Sequence Diagram

This diagram captures the process through which farmers add or update information related to farms, processing facilities, warehouses, and retail locations. It outlines the data flow from the farmer's submission to storage in the system database.

Sequence Flow:

- **Farmer:** Submits new or updated farm/location data.
- **Client Interface:** Transmits the data to the Farm Management Controller.
- **Farm Management Controller:** Validates and processes the data.
- **System Database:** Stores or updates the information.
- **Client Interface:** Confirms the successful operation back to the farmer.

Diagram



c. Update Farm Data Sequence Diagram

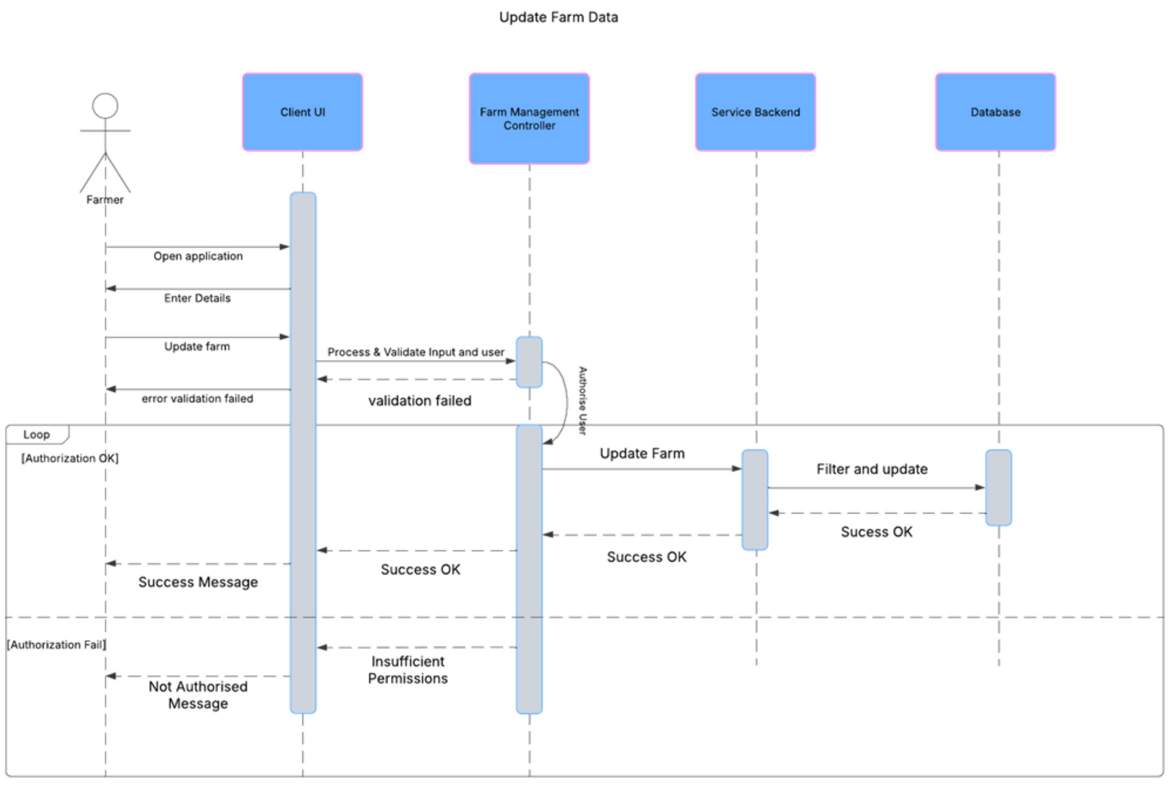
This sequence diagram illustrates how farmers update detailed operational data about their farms. It describes the flow of an update request from the client interface through the farm data controller to the system database.

Sequence Flow:

- **Farmer:** Initiates a request to update farm operational data.

- **Client Interface:** Sends the update request to the Farm Data Controller.
- **Farm Data Controller:** Processes and validates the update.
- **System Database:** Commits the updated information.
- **Client Interface:** Returns the update status to the farmer.

Sample Diagram:



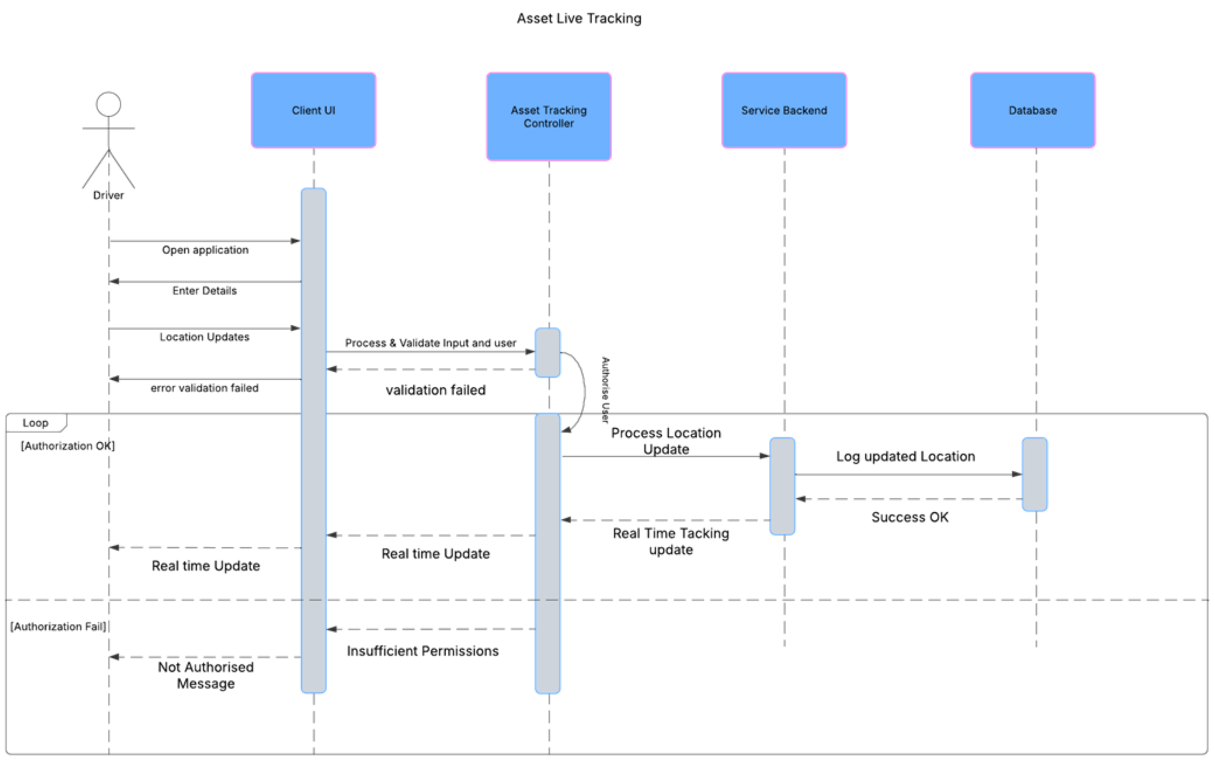
d. Asset Live Tracking Sequence Diagram

This diagram details the process involved in the live tracking of transport assets (trucks). It illustrates how location updates are transmitted from the driver through the system to ensure real-time monitoring.

Sequence Flow:

- **Driver:** Sends location updates from the truck.
- **Client Interface:** Transmits the location data to the Asset Tracking Controller.
- **Asset Tracking Controller:** Processes the location update.
- **Tracking Database/System:** Logs the updated location.
- **Client Interface:** Provides real-time tracking updates to the driver and other relevant stakeholders.

Sample Diagram



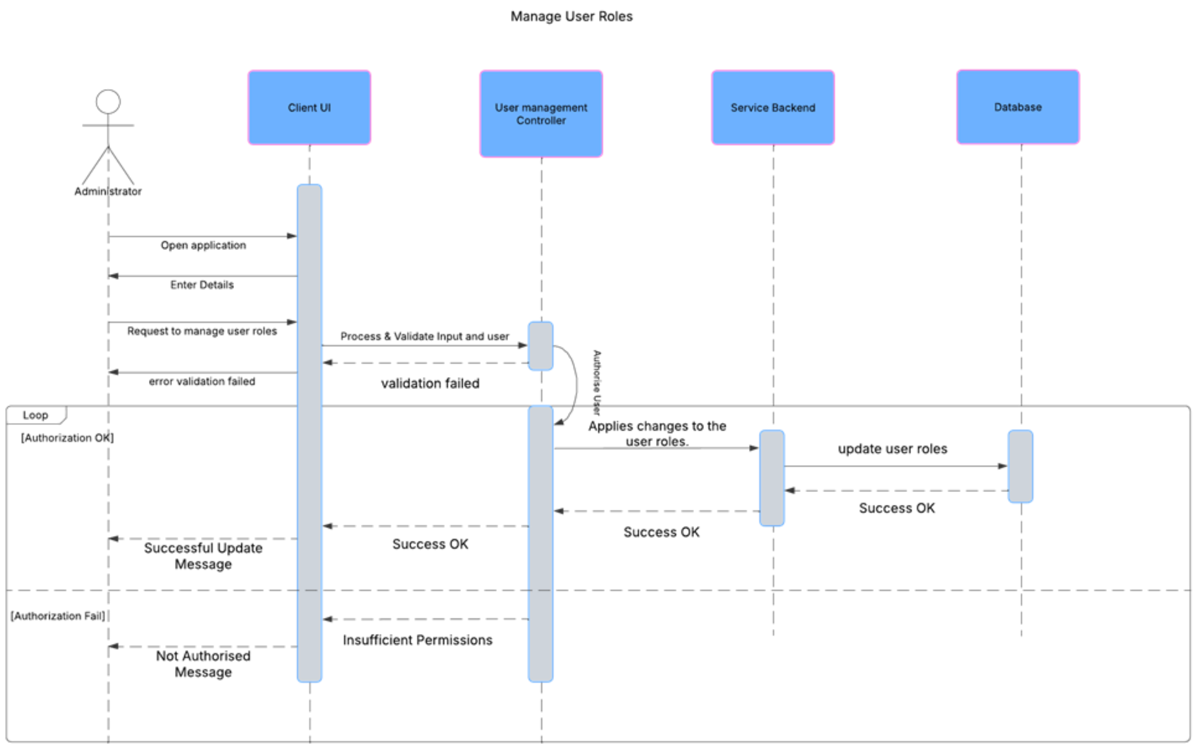
e. Manage User Roles Sequence Diagram

This sequence diagram represents the administrative process for managing user roles and permissions. It shows how the administrator's request flows from the client interface to the user management controller and, ultimately, to the system database for updating user roles.

Sequence Flow:

- **Administrator:** Initiates a request to manage user roles.
- **Client Interface:** Forwards the request to the User Management Controller.
- **User Management Controller:** Processes and validates the request.
- **System Database:** Applies changes to the user roles.
- **Client Interface:** Confirms the successful update back to the administrator.

Sample Diagram



Activity Diagrams

This section outlines the dynamic flow for four core functionalities of the system. Each activity diagram is accompanied by detailed documentation that describes the pre-conditions (what must be true before the process starts), the flow of events (step-by-step actions taken during the process), and the post-conditions (the state of the system after the process completes).

a. User Authentication

Pre-condition:

- The user must have an existing account with valid credentials.
- The system should be operational and accessible.

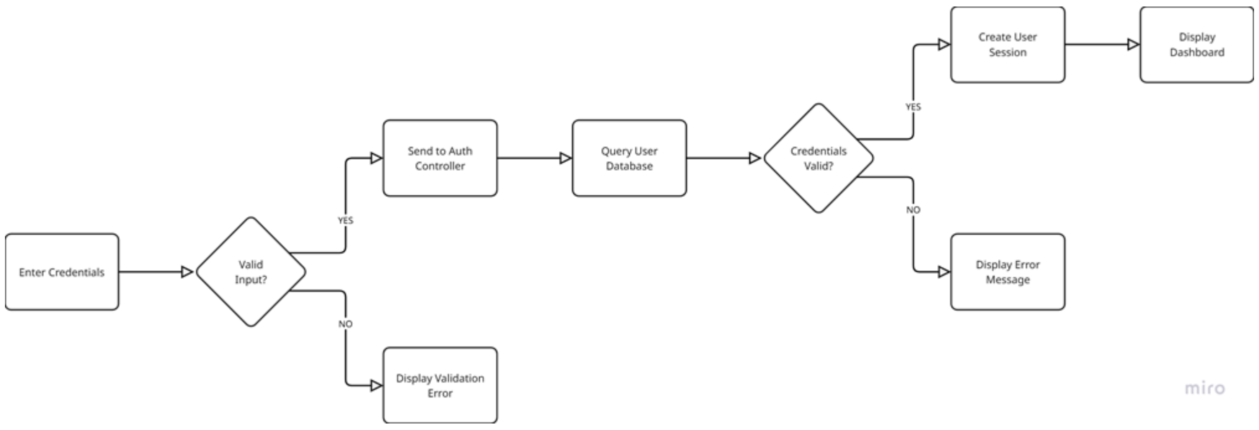
Flow of Events:

1. The user enters their credentials (username and password).
2. The input is validated for correctness (format and completeness).
3. The credentials are sent to the authentication controller for verification.
4. The system queries the user database to validate the credentials.
5. If the credentials are valid, a user session is created; if not, an error is returned.

Post-condition:

- If successful, the user is authenticated, and a session is established with access to the dashboard.
- If unsuccessful, an error message is displayed, and no session is created.

Activity Diagram



b. Manage Farms & Locations

Pre-condition:

- The farmer must be authenticated.
- The system must have access to the farm management module.

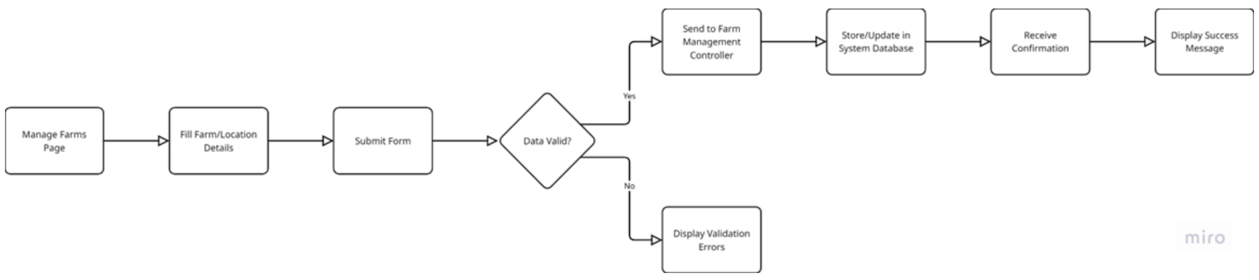
Flow of Events:

1. The authenticated farmer navigates to the Manage Farms page.
2. The farmer fills in the form with new or updated farm/location details.
3. The form is submitted and the input data is validated.
4. If the data passes validation, it is sent to the Farm Management Controller.
5. The system updates or stores the information in the system database.
6. A confirmation response is received and a success message is displayed.

Post-condition:

- The farm/location data is updated in the system, and the farmer receives confirmation of a successful update.
- If there are errors, appropriate validation messages prompt the user to correct the input.

Activity Diagram


Manage Farms Page
Fill Farm/Location Details
Submit Form
Data Valid?
Yes
Send to Farm Management Controller
Store/Update in System Database
Receive Confirmation
Display Success Message
No
Display Validation Errors
miro

c. **Asset Live Tracking**

Pre-condition:

- The driver must be authenticated.
- The tracking device on the truck is active and providing GPS data.

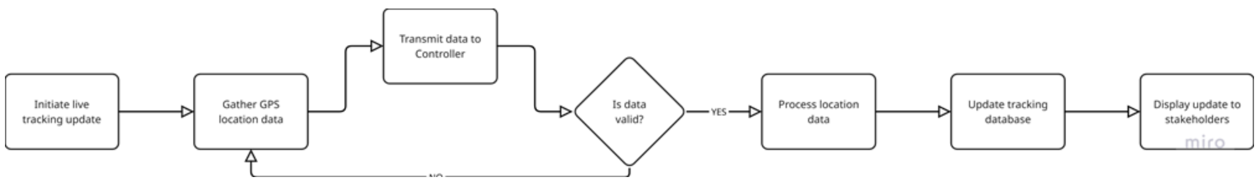
Flow of Events:

1. The driver initiates a live tracking update from the client interface.
2. The system gathers the current GPS location data from the truck.
3. The location data is transmitted to the Asset Tracking Controller.
4. The controller processes the incoming data and updates the tracking database.
5. The updated location is made available for real-time tracking and monitoring.

Post-condition:

- The system reflects the most recent location of the asset, ensuring that real-time tracking is accurate and accessible to relevant stakeholders (such as drivers, brokers, or farmers).
- If any issues occur during transmission, an error handling routine displays a notification for corrective action.

Activity Diagram



d. **Managing User Roles**

Pre-condition:

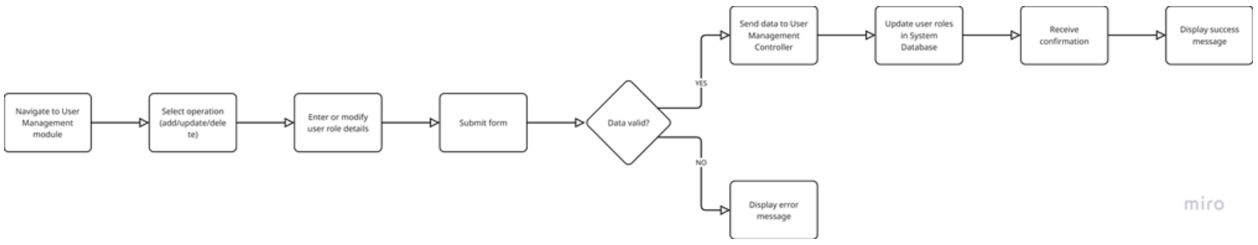
- The administrator must be authenticated and have proper privileges to manage user roles.
- The system's user management module must be accessible and operational.

Flow of Events:

1. The authenticated administrator navigates to the User Management module.
2. The administrator selects the desired operation (e.g., add, update, or delete user roles).
3. The administrator enters or modifies the user role details using the provided form.
4. The input data is validated for correctness and completeness.
5. If valid, the data is sent to the User Management Controller for processing.
6. The controller updates the role details in the system database.
7. A confirmation response is received, and a success message is displayed to the administrator.

Post-condition:

- The user roles are updated in the system's database, ensuring that the access permissions reflect the new configuration.
- If the update fails or validation errors occur, an appropriate error message is displayed, prompting the administrator to take corrective action.


Navigate to User Management module
Select operation (add/update/delete)
Enter or modify user role details
Submit form
Data valid?
YES
NO
Send data to User Management Controller
Update user roles in System Database
Receive confirmation
Display success message
Display error message
miro

DESIGN PATTERNS

Supply2U's architecture employs proven design patterns to ensure clean code organization, maintainability, and scalability across its agricultural supply chain platform. These patterns standardize development workflows while accommodating the diverse needs of farmers, transporters, and administrators.

MVC (Model-View-Controller)

The Model-View-Controller (MVC) pattern is used to maintain a clear separation between data, user interface, and business logic. This enhances modularity and makes the system highly scalable by allowing developers to update or modify one component without disrupting others.

- Model- The Model is the central part of the pattern, responsible for the business logic and data manipulation. For example:
  FarmModel: Manages geospatial farm data with PostGIS integration
  TransportModel: Processes real-time GPS coordinates using Apache Kafka streams
  UserModel: Handles Role-Based Access Control(RBAC) with Django's authentication backend.
- View- This is typically passive, meaning it does not contain business logic but only the logic to present the data to the user. It makes the user interface more flexible and easier to update or replace.
- Controller- Serves as the intermediary between the Model and View, processing user actions and retrieving relevant data. For example, RESTful APIs and websocket handlers.

WIREFRAMES AND UI/UX DESIGN

Overview

Intro

The Supply2U platform is built with a user-centric approach, ensuring intuitive navigation, efficiency, and accessibility. Every user interaction within the system is designed to be seamless, allowing stakeholders—whether farmers, distributors, buyers, or logistics personnel—to perform tasks effortlessly. While maintaining alignment with functional requirements, our UI/UX strategy prioritizes engagement, usability, and accessibility to create an interface that fosters inclusivity and efficiency.

The system tailors each user's experience based on their assigned role, offering a personalized workflow for different functionalities. Farmers manage farm areas and locations, logistics teams oversee deliveries, and administrators handle platform configurations. This structure ensures that Supply2U delivers an optimal interface that facilitates smooth supply chain operations while remaining responsive across multiple devices.

Tools

To develop high-fidelity wireframes and interfaces tools such as Figma – Used for prototyping, designing interactive mockups, and ensuring responsive layouts. Its features facilitate collaborative designing and also helps in incorporating real-time stakeholder feedback.

Wireframing process

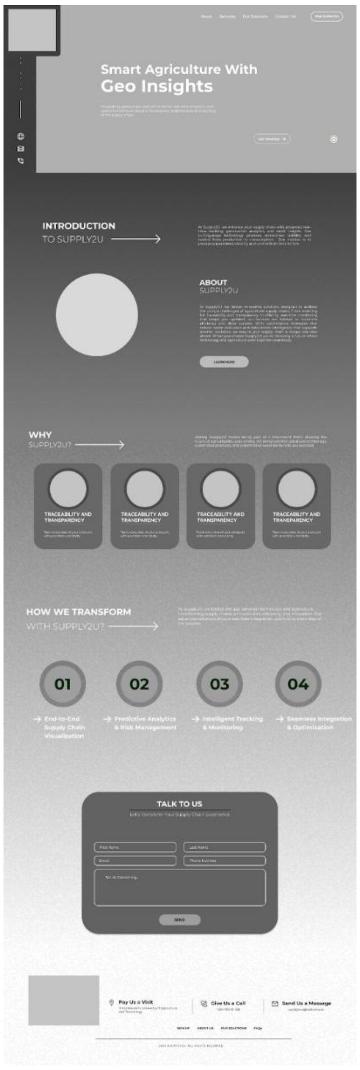
The wireframing phase establishes a skeletal framework for the system's layout, ensuring logical arrangement of elements before transitioning into high-fidelity designs. This process includes:

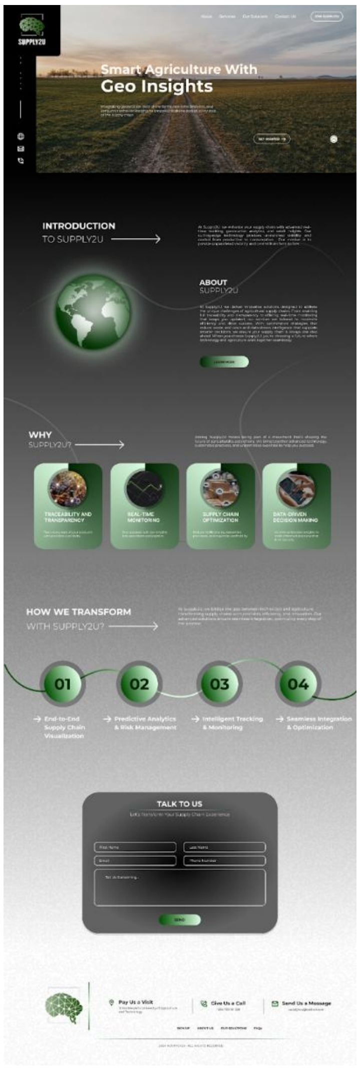
- Initial Concept Sketching – This stage involves making rough outlines of the interface components, navigation patterns, and user flow mapping.
- Low-Fidelity Wireframes – Basic layouts that define content placement, user actions, and system responses.
- High-Fidelity Prototypes - After refining the low-fidelity wireframes, Figma was used to create mockups with interactive components such as buttons, dropdowns, and forms with various states. Microinteractions too which include subtle animations, such as progress spinners, were integrated for enhancing user experience.

<u>Wireframes and mockups</u>

1. **Landing Page**

The Supply2U landing page is designed to provide a compelling introduction to the platform. It uses geolocation-driven visuals, ensuring that users see relevant imagery. The page features a call-to-action guiding users toward account registration while displaying more information about what Supply2U is to establish trust and credibility.




SUPPLY2U
Smart Agriculture With
Geo Insights
INTRODUCTION
TO SUPPLY2U
ABOUT
SUPPLY2U
WHY
SUPPLY2U?
TRACEABILITY AND TRANSPARENCY
REAL-TIME MONITORING
SUPPLY CHAIN OPTIMIZATION
DATA-DRIVEN DECISION MAKING
HOW WE TRANSFORM
WITH SUPPLY2U?
01
02
03
04
End-to-End Supply Chain Visualization
Predictive Analytics & Risk Management
Intelligent Tracking & Monitoring
Seamless Integration & Optimization
TALK TO US
Pay Us a Visit
Give Us a Call
Send Us a Message

2. Login/Signup

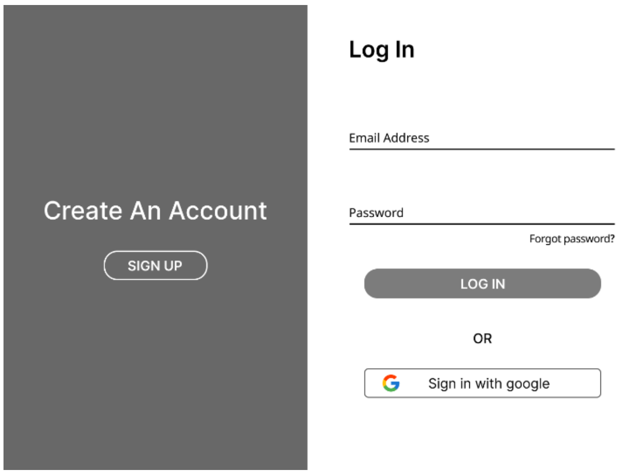
The Login/Signup interface for Supply2U is designed to be intuitive, secure, and accessible, ensuring seamless user onboarding for farmers, transporters, buyers, and administrators. The interface balances simplicity with advanced authentication options to accommodate users with varying digital literacy levels.

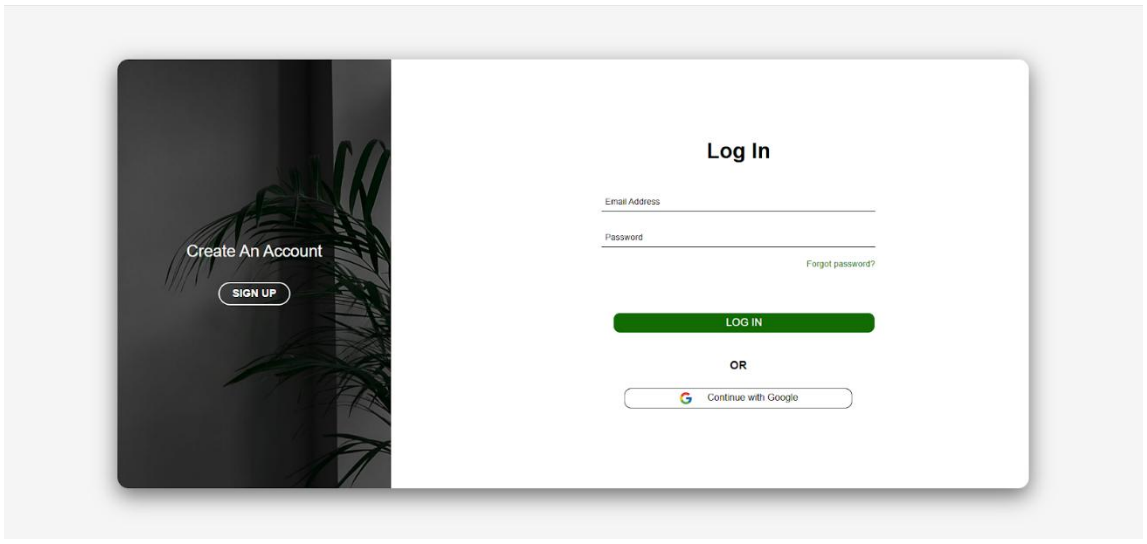
<u>Login Page</u>

Some of the key features include:

Multiple Authentication Methods- Users can log in with google or through traditional email/password credentials.

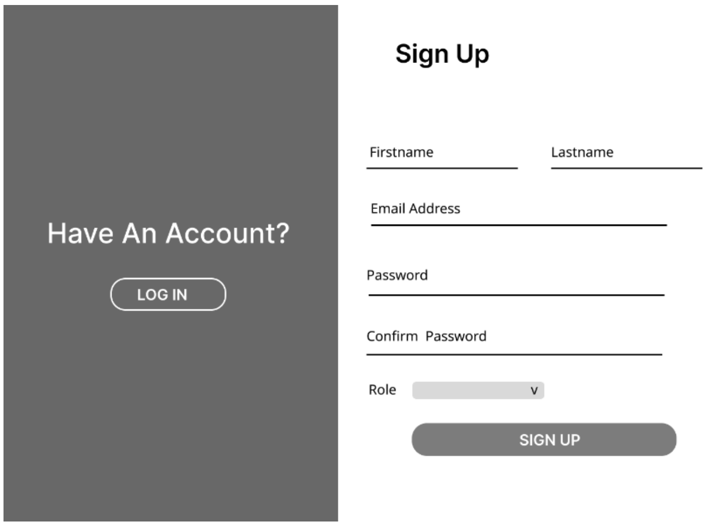
Password Recovery & Support- Users can reset passwords using their email.

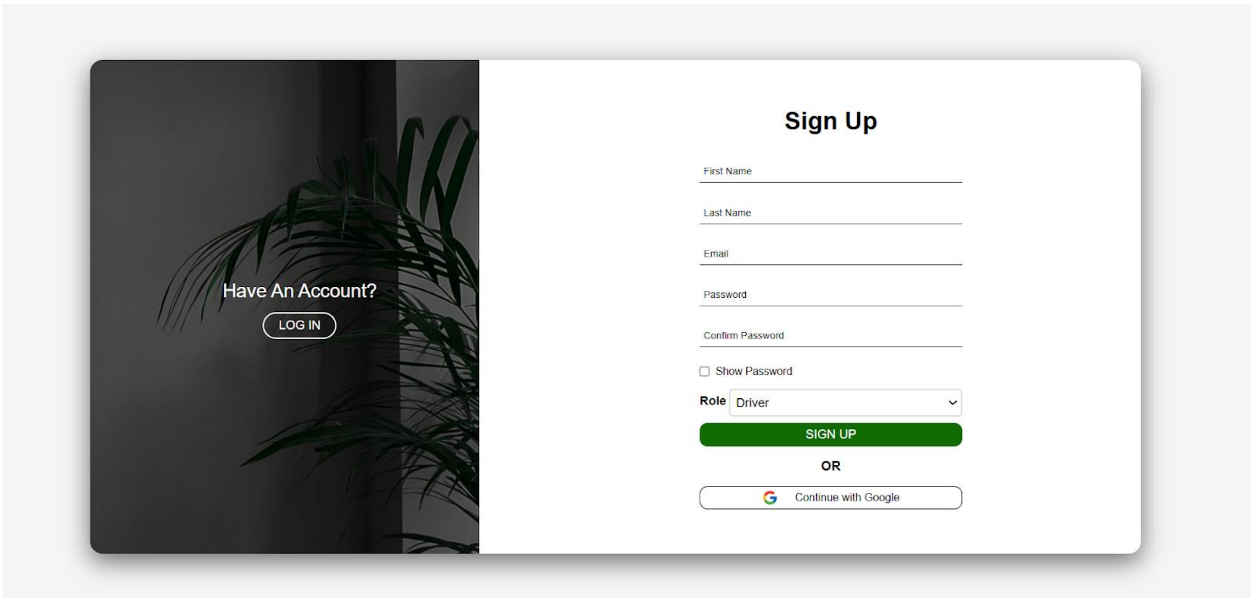




Signup Page

This page entails minimal input fields i.e. Only the most necessary details are collected such as the first and last name, role, email, and password.




Have An Account?

LOG IN

Sign Up

First Name

Last Name

Email

Password

Confirm Password

☐ Show Password

Role: Driver

SIGN UP

OR

Continue with Google

3. Role-Specific Dashboards

Each user role i.e. farmers, transporters, retailers, and admins, have a dedicated dashboard tailored to their needs:

Farmer Dashboard:

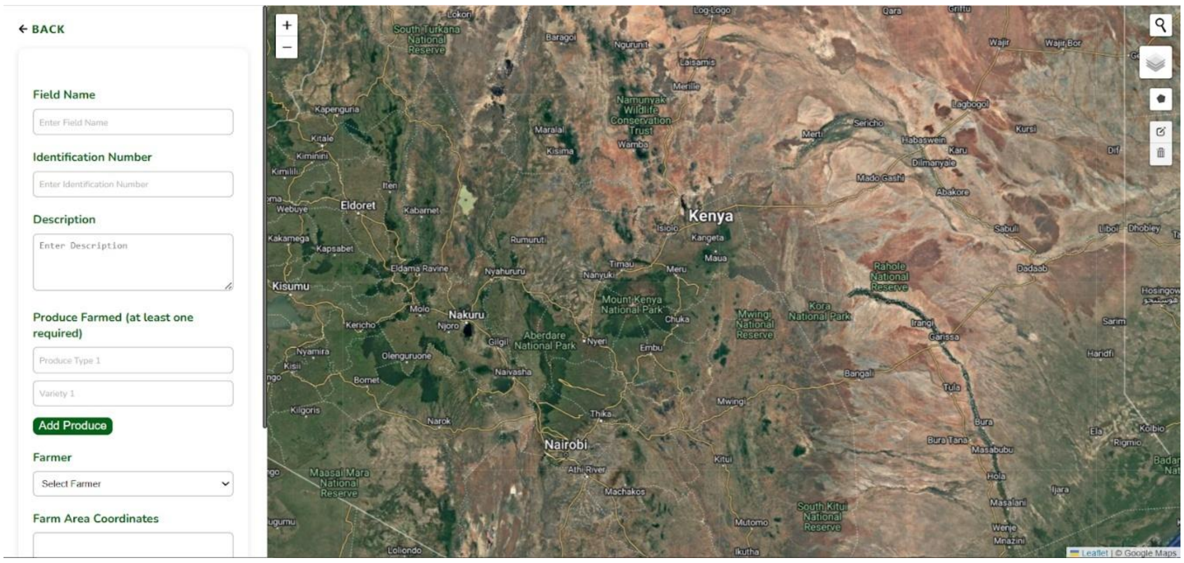
Transporter Dashboard: Real-time GPS tracking and traffic alerts for logistical support.

Administrator Dashboard: Customizable role management, system-wide analytics, and security controls for platform governance.

4. Mapping

Supply2U integrates geolocation tracking to enhance supply chain transparency. Farm locations are mapped dynamically, enabling retailers to source produce from their preferred regions while transporters receive optimized delivery routes based on traffic conditions.

Farmer / Retailer view



← BACK

Field Name

Enter Field Name

Identification Number

Enter Identification Number

Description

Enter Description

Produce Farmed (at least one required)

Produce Type 1

Variety 1

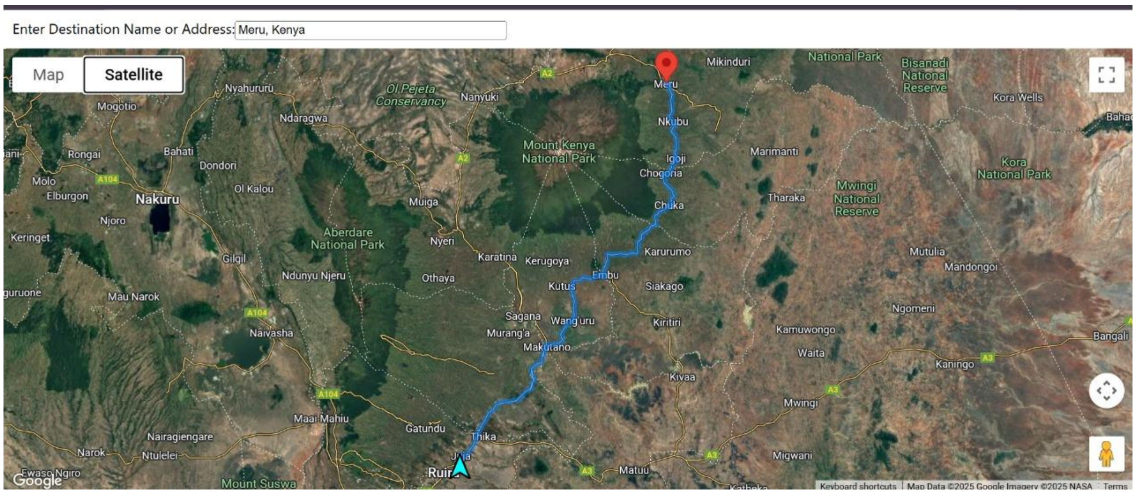
Add Produce

Farmer

Select Farmer

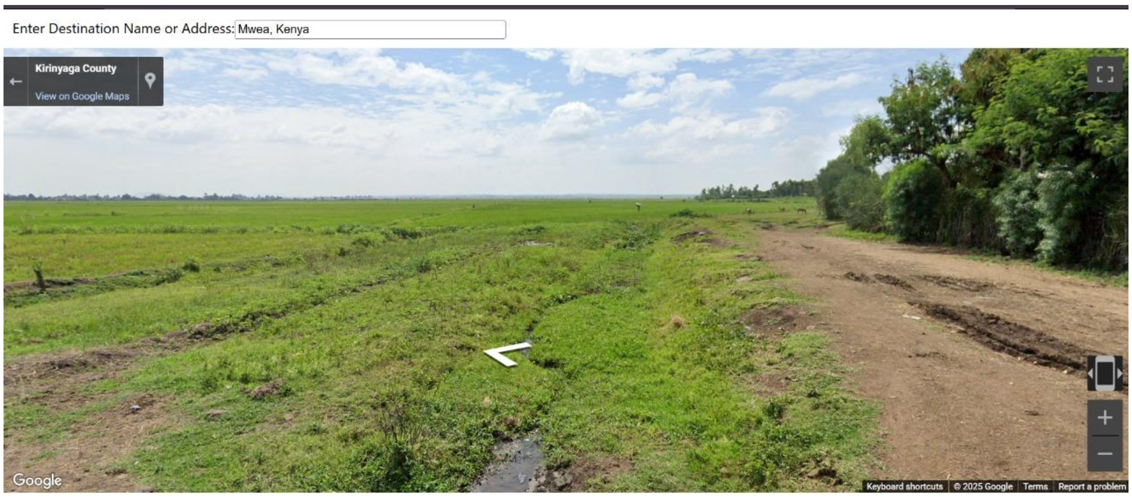
Farm Area Coordinates

Transporter View



Enter Destination Name or Address: Meru, Kenya

Map
Satellite
Meru
Nkubu
Chogoria
Chuka
Embu
Thika
Ruiru
Mount Kenya National Park
Aberdare National Park
Nakuru
Naivasha
Google
Keyboard shortcuts | Map Data ©2025 Google Imagery ©2025 NASA | Terms



Enter Destination Name or Address: Mwea, Kenya

Kirinyaga County
View on Google Maps
Google
Keyboard shortcuts | © 2025 Google | Terms | Report a problem

Screen Objects and User Interactions

To maintain usability and efficiency, key screen elements are designed with interaction patterns such as:

Large touch targets for farmers using gloves.

One-handed mobile navigation for transporters.

Drag-and-drop widgets for administrators managing configurations

Usability and Accessibility Best Practices

A focus on usability and accessibility ensures Supply2U meets the highest standards for inclusivity. The interface adheres to a minimum 4.5:1 contrast ratio, tested via WebAIM WAVE for optimal readability. Keyboard navigation is fully supported, enabling tab-based accessibility for users. Beyond accessibility, the system offers support in English.

DATABASE DESIGN

Conceptual Design

Entity Descriptions

USER

Represents any system user, such as a farmer, agro-dealer, transporter, etc.

Attributes:

- user_id (PK)
- name
- email
- phone
- role(e.g., Farmer, Retailer, Transporter)
- is_staff
- is_superuser

Relationships:

- A user of the role farmer owns one or more farms
- A user of the role retailer can place one or multiple orders
- Can offer or use transport

FARM

Describes individual farms owned by users.

Attributes:

- farm_id (PK)
- name
- location
- boundary
- region
- area
- created_at
- updated_at
- user_id (FK to User)

Relationships:

- Has one or more soils
- Can grow one or more crops
- Connected to a single user (owner)
- Has one or more weather instances

SOIL

Captures soil data for each farm.

Attributes:

- soil_id (PK)
- farm_id(FK to Farm)
- ph
- cation_exchange_capacity
- soil_type
- depth_measured
- date_measured
- organic_carbon

- humus
- electrical_conductivity
- recommended_applications

Relationships:

- Belongs to one farm

CROP

Describes crops grown or available.

Attributes:

- crop_id (PK)
- farm_id (FK to Farm)
- ideal_soil_type (ENUM TO SOIL_TYPE)
- variety
- water_requirement_mm_per_week
- temperature_range_celsius
- planting_season
- planted_date
- expected_harvest_date
- market_price_per_kg

Relationships:

- Grown on one farm
- Appears in one or more orders
- Enumerated from one soil type

ORDER

Represents a transaction for crop purchase or delivery.

Attributes:

- order_id (PK)
- user_id (FK to USER – buyer)
- crop_id (FK to CROP)
- quantity
- price
- Order_status
- payment_status

- date

Relationships:

- Linked to User and Crop
- May require transport

TRANSPORT

Handles logistics info for deliveries.

Attributes:

- transport_id (PK)
- user_id (FK to USER – driver)
- start_coordinates (FK to Farm - location)
- order_id (FK to Order)
- end_coordinates
- duration_minutes
- departure_time
- arrival_time
- route
- distance

Relationships:

- Linked to a driver (user)
- May be assigned to fulfill orders

WEATHER

Provides weather data for a specific farm.

Attributes:

- weather_id (PK)
- farm_id (FK to Farm)
- recorded_at
- temperature
- feels_like
- humidity
- rainfall
- wind_speed

- date

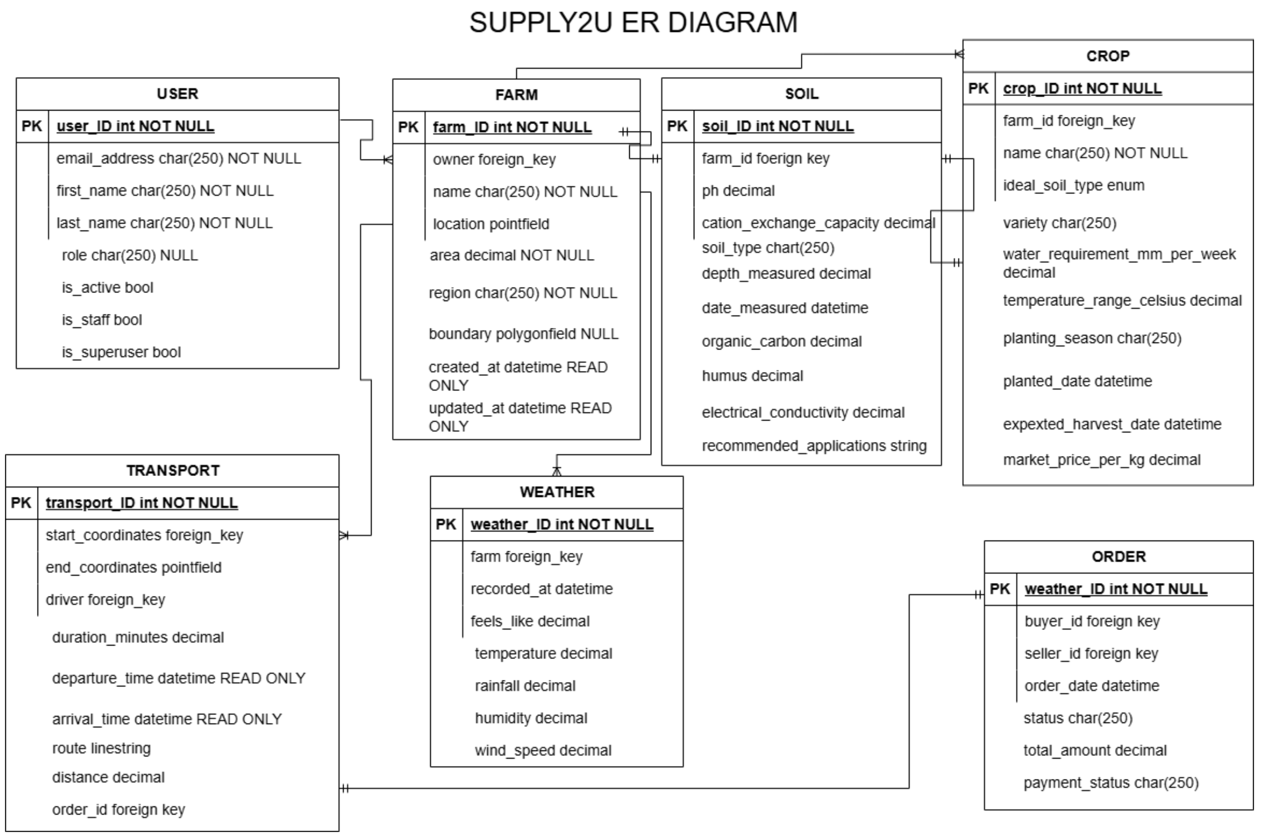
Relationships:

- Can be mapped to farms via location

Logical Design

Entity Relationships

Entity	**Related Entity**	**Relationship Type**	**Cardinality**
USER	FARM	One-to-Many	One user can have many farms
FARM	SOIL	One-to-Many	One farm has many soil samples
FARM	CROP	One-to-Many	One farm grows many crops
USER	ORDER	One-to-Many	One user can place many orders
ORDER	CROP	Many-to-One	Each order involves one crop
USER	TRANSPORT	One-to-Many	One user may manage multiple transport units
WEATHER	FARM	Many-to-One	Weather data mapped via location



SECURITY ARCHITECTURE

Security Goals

1. Confidentiality: Safeguard agricultural, retail, and logistical data.
2. Integrity: Ensure data is accurate and tamper-proof.
3. Availability: Guarantee uptime and reliability for all stakeholders.
4. Accountability: Track user activities and enforce data access control.

Architecture Layers & Security Controls

- User Access Layer (Web App)

Users:

- Smallholder farmers
- Agro-dealers
- Logistics personnel

Controls:

- HTTPS everywhere (TLS 1.3)
- Content Security Policy (CSP) to prevent XSS
- Session timeout and token renewal policies
- Responsive security design for mobile browsers
- Login protection (rate limiting, CAPTCHA on failure)

- Network & Communication Layer

Controls:

- Encrypted data in transit via HTTPS
- Firewall rules for backend server access
- API Gateway with throttling and rate limiting
- WAF (Web Application Firewall) for OWASP Top 10 protection

- Application Layer (Business Logic)

Modules:

- Geolocation Mapping
- Supply Chain Management
- Transportation Management

Controls:

- Role-based access control (RBAC)
- Input validation/sanitization
- Server-side validation
- Secure session management (JWT with refresh tokens)
- Least privilege for internal API calls
- Audit logs for critical operations

- Data Storage Layer

Data Types:

- User profiles
- Farm produce info
- Delivery routes
- Retail transactions
- Geo-coordinates

Controls:

- AES-256 encryption at rest
- Role-scoped database access
- Database firewalls and subnet isolation
- Automated backup + secure offsite storage
- Regular vulnerability scanning on the DB layer
- Logs stored with tamper-proof access

- Identity & Access Management (IAM)

Controls:

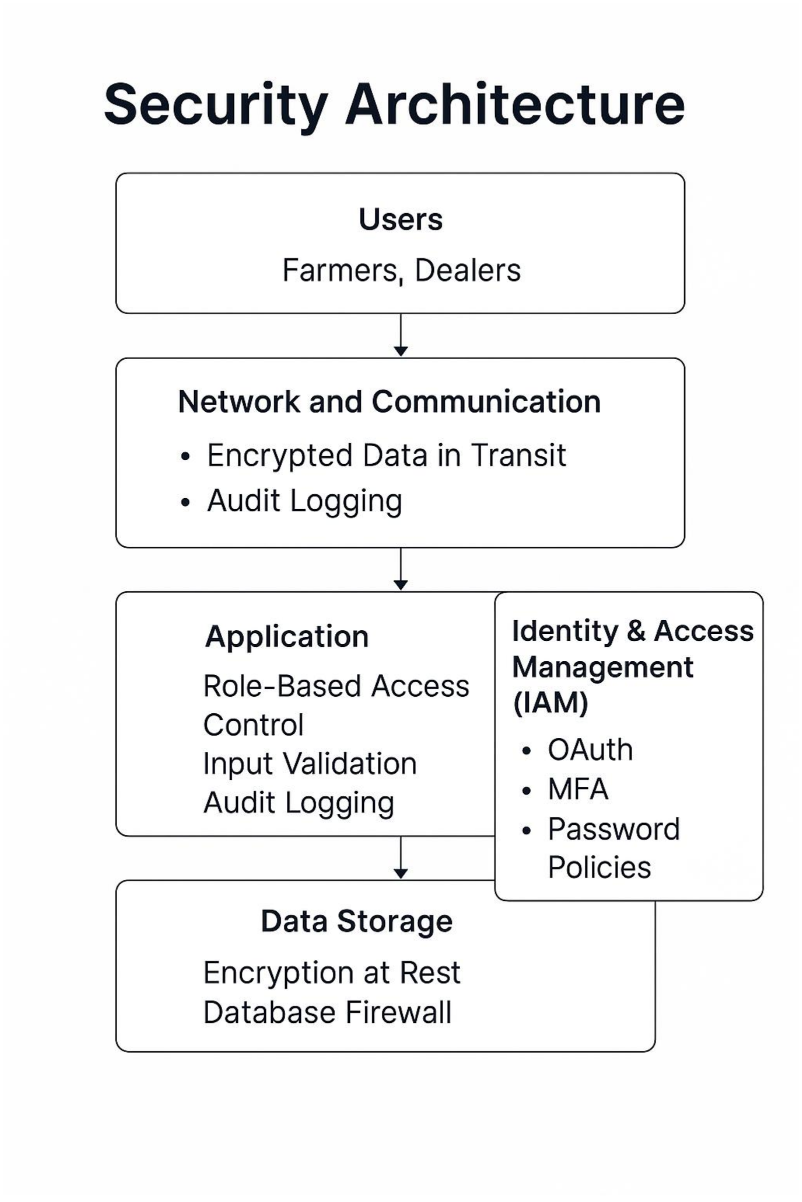
- Username/password with bcrypt hashing
- Multi-Factor Authentication (MFA) for admins
- Role hierarchy: Admin > Ops > Dealer > Farmer > Driver
- Password policies: complexity, rotation, expiry
- Account lockout on repeated failed login attempts

- Monitoring, Logging, and Response

Capabilities:

- Audit logging of login, data changes, and role updates
- Alerts for excessive access attempts or suspicious actions

Security Design



<u>SCALABILITY AND DEPLOYMENT</u>

Scalability

The system is designed to scale both vertically and horizontally as demand increases. The system supports several modules. To handle growth in each of these areas, scalability is addressed as follows:

- **Modular Architecture**: Each functional area (tracking, orders, registration, user management) is encapsulated in its own app/module within the Django project. This separation allows teams to scale features independently.

- **Horizontal Scaling**:

Frontend: Deployed on AWS Elastic Load Balancer (ELB) with auto-scaling groups to handle traffic spikes.

Backend: Microservices architecture using Kubernetes (EKS) for dynamic pod scaling based on CPU/memory usage.

Database: PostgreSQL (relational data) scaled via Amazon RDS Read Replicas. Cassandra (NoSQL for logs/tracking) uses a multi-node cluster for high write throughput.

- **Vertical Scaling**:

Server Upgrades: Adjust compute resources (vCPUs, RAM) for backend services (e.g., Django, Kafka) based on performance metrics.

Caching: Redis for session management and frequently accessed data (e.g., farm locations, delivery routes). CDN (CloudFront) for static assets (maps, UI components).

- **Real-Time Capacity**:

The system leverages PubNub and Django Channels to support hundreds of concurrent users and trucks in transit.

Load balancers (e.g., Nginx) distribute WebSocket and API requests efficiently to avoid bottlenecks.

- **Data Partitioning & Sharding**

Geospatial Data: Sharded by region (e.g., Northern, Central) to optimize query performance.

User Data: Partitioned by role (farmers, transporters) to reduce contention.

- **Asynchronous Processing**

Kafka queues for real-time GPS updates and order processing to decouple services.

Apache Spark for batch analytics (e.g., delivery performance reports).

Deployment

Deployment is planned in three environments: **development**, **staging**, and **production**.

- **Frontend Deployment (React)**:

Tool	**Vercel**	**Justification**

Hosting	- Global CDN with edge caching.	Faster load times for farmers in remote areas.
CI/CD	- Automatic builds/deploys on Git pushes.	Streamlines UI updates (e.g., map interface tweaks).
Optimizations	- Serverless Functions for API proxying.	Reduces backend load for static content (e.g., farm profiles).

- **Backend Deployment (Django + Channels)**:

Tool	**Railway**	**Justification**
Hosting	- Managed cloud instances with auto-scaling.	Handles spiky loads during harvest seasons.
Database	- PostgreSQL + PostGIS extension.	Native support for geospatial queries (e.g., "Find farms within 50km").
Real-Time	- WebSocket support via Daphne (ASGI).	Critical for live GPS tracking and route alerts.
CI/CD	- GitHub integration for zero-downtime deploys.	Ensures high availability (99.9% SLA).

- **Environment Strategy**

Environment	**Purpose**	**Frontend (Vercel)**	**Backend (Railway)**
Development	Feature testing	Preview URLs per Git branch.	Ephemeral instances with test DB.
Staging	UAT with farmers/transporters.	Password-protected deployment.	Mirrors production (scaled-down).
Production	Live system	Custom domain (app.supply2u.co).	Geo-replicated DB backups.

- **Containerization**:
    - All services are Dockerized for portability and ease of deployment.
    - Environment variables and secrets are managed securely via .env and vault services.
- **Access Control**:
    - The central dashboard allows full control.
    - External users access a scoped interface (e.g., their trucks, orders) using role-based permissions with JWT.
- **CI/CD**
    - GitHub Actions automates testing and deployment pipelines.

AGILE DEVELOPMENT PLAN

Sprint Cycles (2-Week Sprints)

Sprint	Focus Area	Deliverables
1	Core Modules Setup	User auth, farm management MVP, basic GIS integration.
2	Transport & GPS Tracking	Live tracking, route deviation alerts, delivery logs.
3	Admin & Reporting	Role management, audit logs, analytics dashboards.
4	Mobile Optimization	Responsive UI, offline mode for transporters.
5	Security & Compliance	RBAC enforcement, data encryption, GDPR compliance checks.
6	Performance Tuning	Load testing, caching strategies, API optimizations.

7	UAT & Bug Fixes	User acceptance testing, edge-case handling.
8	Deployment & Go-Live	AWS infrastructure provisioning, CI/CD pipeline finalization.

Sprint Activities:

- **Sprint Planning:** At the onset of each sprint, the team conducts a planning session to define the sprint goal and select user stories from the product backlog that align with this objective. This session ensures a mutual understanding of the tasks and their respective priorities.
- **Daily Standups:** Brief, 15-minute daily meetings are held for team members to share progress, discuss upcoming tasks, and highlight any impediments. These standups promote transparency and facilitate prompt issue resolution.
- **Sprint Review:** Upon sprint completion, the team showcases the developed features to stakeholders, gathering feedback to inform subsequent development efforts. This review ensures alignment with stakeholder expectations and project objectives.
- **Sprint Retrospective:** Following the review, the team reflects on the sprint process, identifying successes and areas for improvement. Actionable insights are documented to enhance future sprints.

Backlog Management and Prioritization

- **Tools:** The product backlog is maintained using platforms like ClickUp, providing a transparent and organized view of tasks and priorities.
- **Prioritization Technique:** User stories are evaluated and categorized based on the MoSCoW method:
  - **Must Have:** Critical functionalities essential for the system's core operation, such as real-time tracking, destination assignment, order creation, and user authentication.
  - **Should Have:** Important features that enhance user experience, including dashboard analytics and order history filters.
  - **Could Have:** Desirable additions like email/SMS notifications and QR code scanning capabilities for trucks, which can be incorporated if time permits.
  - **Won’t Have (for now):** Features like dedicated driver applications and customer chat integrations are deferred for future consideration.

RISK ANALYSIS

Risk	Impact	Likelihood	Mitigation Strategy
GPS Signal Loss	High	Medium	Enable offline mode with local caching; resume data transmission when connection is restored. Maintain the last known location on the dashboard.
Database Performance Bottlenecks	High	Low	Use of PostgreSQL optimization techniques: sharding large tables, implementing read replicas, and ensuring indexed geo queries for tracking.
Security Breaches (Unauthorized Access, Data Leakage)	Critical	Medium	Enforce Multi-Factor Authentication (MFA), role-based access, and secure token-based APIs. Conduct regular penetration testing and deploy a Web Application Firewall (WAF).
Third-Party API Failures (e.g., Google Maps, PubNub)	Medium	High	Use static map fallbacks and cached routes for continuity. Prioritize APIs with strong SLAs and monitor uptime via alerting systems.
User Adoption Resistance	Medium	High	Run targeted onboarding sessions with transporters and farmers. Use a phased rollout to gather feedback and adapt UI/UX to users' digital literacy levels.
Regulatory Non-Compliance	Critical	Low	Engage legal experts for GDPR and local data protection compliance. Implement automated data anonymization and data retention policies.
Server Overload During Peak Hours	High	Medium	Deploy load balancers (e.g., Nginx), auto-scale containers, and separate real-time services

			(WebSocket channels) from standard REST APIs.
Data Loss (Order History, Routes, Location Logs)	High	Low	Implement daily database backups, geo-redundant storage, and failover support for critical services.
Cloud Service Downtime (e.g., hosting provider outages)	High	Low	Host with providers offering 99.9%+ uptime SLAs. Configure backup servers and recovery scripts for quick switchover.
Deployment Failures or Bugs in Production	Medium	Low	Integrate CI/CD with staging environments and rollback options. Use automated testing and canary deployments.
Inaccurate Geolocation Data	Medium	Medium	Implement signal smoothing algorithms and fallback to user-entered destinations when needed.

TEAM COLLABORATION AND WORKFLOW

To ensure efficient development and seamless coordination among cross-functional teams, Supply2U follows an Agile (Scrum-based) workflow with clear roles, tools, and processes. Below is an in-depth breakdown of team collaboration strategies.

Team Structure & Roles

Role	**Responsibilities**

Product Owner (PO)	- Defines user stories and prioritizes backlog. - Acts as liaison between stakeholders and dev team.
Scrum Master	- Facilitates daily standups, sprint planning, and retrospectives. - Removes blockers.
Frontend Developers	- Implement UI/UX designs (React.js, Figma). - Optimize mobile responsiveness.
Backend Developers	- Build APIs (Django REST/GraphQL). - Integrate geospatial services (Google Maps API).
DevOps Engineers	- Manage CI/CD pipelines (GitHub Actions). - Monitor cloud infrastructure (AWS/Kubernetes).
QA Engineers	- Write automated tests (Jest, PyTest). - Conduct UAT with farmers/transporters.
UI/UX Designers	- Prototype wireframes (Figma). - Validate accessibility (WCAG 2.1 compliance).

Collaboration Tools

Tool	**Purpose**	**Usage Guidelines**
ClickUp	- Sprint planning, backlog grooming, and issue tracking. - Epics: User Auth, Live GPS Tracking. - Task dependencies and progress dashboards.	- Tasks linked to Goals (e.g., "Sprint 3: GPS Tracking"). - Custom statuses: To Do, In Review, Blocked, Done.
WhatsApp	- Real-time communication for urgent issues and quick coordination. - Group chats: #DevTeam, #Urgent-Alerts.	- @mentions for immediate attention. - Daily standup summaries posted here. - Avoid lengthy design discussions (use Figma comments).

GitHub	- Code repository + PR reviews. - Branching strategy: main (prod), dev (staging), feature/*.	- PRs require 2 approvals. - Link PRs to ClickUp tasks (e.g., "Closes #CU-123").
Figma	- Collaborative UI/UX design. - Prototype sharing for stakeholder feedback.	- Annotations for design specs. - Version history for audit trails.
Google Docs/Sheets	- Documentation hub (replaces Confluence). - Meeting notes, ADRs, and API specs.	- Shared folders with role-based access. - Live editing during sprint planning.
Zoom	- Sprint demos and stakeholder meetings.	- Recorded sessions stored in Google Drive.

CONCLUSION

This document has established a comprehensive and forward-thinking design for the Supply2U platform, aligning technical decisions with the real-world demands of institutional food logistics. By emphasizing scalability, usability, and integration of real-time technologies, the system is prepared to support efficient, transparent, and reliable supply chain operations. The strategies and structures outlined here will serve as a strong foundation for development, ensuring that Supply2U evolves into a practical and impactful solution for its target users.