
**University Institute of Information Technology,**

**PMAS-Arid Agriculture University,**

**Rawalpindi Pakistan**

# FarmConnect

***By***

| | |
|---|---|
| **Muhammad Shahid Faqeer** | **20-ARID-906** |
| **Muhammad Ahmad** | **20-ARID-882** |
| **Ismail Khan** | **20-ARID-863** |

***Supervisor***

**Prof. Dr. Yaser Hafeez**

***Bachelor of Science in Software Engineering***

***(2020-2024)***

**The candidate confirms that the work submitted is their own and appropriate credit has been given where reference has been made to the work of others**.

# DECLARATION

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software documentation and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other. We will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

| | | |
|---|---|---|
| Muhammad Shahid Faqeer | Muhammad Ahmad | Ismail Khan |
| --- | --- | --- |

# CERTIFICATE OF APPROVAL

It is to certify that the final year project of BS (SE) "FarmConnect" was developed by "Muhammad Shahid Faqeer**, 20-Arid-906"**, "Muhammad Ahmad**, 20-Arid-882"** and "**Ismail Khan, 20-Arid-863"** under the supervision of "Prof. Dr. Yaser Hafeez" and that in their opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Software Engineering.

---

**Supervisor**

---

**(Mr. Imran Khurram OR Mr. Zeeshan Javed)**

**External Examiner**

---

**Administrator UIIT**

# Executive Summary

FarmConnect is a groundbreaking mobile application designed to revolutionize how farmers connect, collaborate, and exchange knowledge. Unlike existing agricultural apps, FarmConnect takes a holistic approach, offering a comprehensive social media platform tailored specifically for farmers. With advanced features like personalized profiles, robust search capabilities, a newsfeed, group communities, and a marketplace, the user-centric design ensures an intuitive experience.

FarmConnect leverages cutting-edge technology to create an all-encompassing social media platform for farmers, enhancing connectivity and collaboration. By providing personalized profiles, a dynamic newsfeed, and group communities, it ensures a tailored experience. The marketplace and robust search capabilities further support resource exchange and knowledge sharing. Developed with a foundation in software engineering principles, the app prioritizes security, scalability, and user-friendly design. Emphasizing ethical practices, FarmConnect continuously evolves based on user feedback, striving to address the unique challenges of agriculture and empower the farming community.

Drawing on a diverse skill set from a Software Engineering degree program, FarmConnect integrates programming fundamentals, data structures, algorithms, and more. The project spans various disciplines, incorporating web technologies, mobile app development, computer networks, information security, and formal methods in software engineering. Emphasizing ethical and professional practices, FarmConnect aims to provide a secure, scalable, and user-friendly platform for farmers to enhance productivity and foster a connected farming community.

As a forward-looking initiative, FarmConnect prioritizes continuous user feedback to refine and reimagine existing functionalities, ensuring an even more effective tool for farmer networking, education, and resource exchange. With a commitment to addressing unique challenges in agriculture, FarmConnect stands as a transformative force, modernizing and empowering the farming sector.

In conclusion, FarmConnect is more than just an app; it’s a transformative tool designed to modernize and empower the farming community. By integrating advanced technology and user feedback, it addresses the unique challenges of agriculture, fostering a connected, productive, and informed farming sector.

# Acknowledgement

All praise is to Almighty Allah who bestowed upon us a minute portion of His boundless knowledge by virtue of which we were able to accomplish this challenging task.

We are greatly indebted to our project supervisor “Professor Dr. Yaser Hafeez” for personal supervision, advice, valuable guidance and completion of this project. We are deeply indebted to him for encouragement and continual help during this work.

And we are also thankful to our parents and family who have been a constant source of encouragement for us and brought us the values of honesty & hard work.

| | | |
|---|---|---|
| Muhammad Shahid Faqeer | Muhammad Ahmad | Ismail Khan |
| --- | --- | --- |

# Abbreviations

| | |
|---|---|
| **SRS** | Software Requirement Specification |
| **PC** | Personal Computer |
| UI | User Interface |

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1: Introduction

FarmConnect is a pioneering mobile application designed to revolutionize agricultural practices by providing a specialized social media platform for farmers. Addressing the limitations of existing agricultural apps, FarmConnect integrates advanced features such as personalized user profiles, group communities, a marketplace, and secure communication. The project draws on a diverse skill set from the Software Engineering degree program, emphasizing real-world applications in programming, data structures, algorithms, web and mobile development, and more. With a user-centric design, data security protocols, and continuous refinement based on user feedback, FarmConnect aims to create a cohesive and empowering network for farmers, fostering collaboration, knowledge-sharing, and enhanced productivity within the farming community.

## 1.1. Brief

FarmConnect is a groundbreaking mobile application aimed at revolutionizing the way farmers connect, collaborate, and share knowledge within the agricultural community. Our app will serve as a social media platform specifically designed to meet the unique needs of farmers, empowering them to enhance productivity, access valuable insights, and foster a strong farming network.

## 1.2. Relevance to Course Modules

The FarmConnect project is highly relevant to various courses studied during your degree program. Here's how it relates to the specific courses you've mentioned:

a. **Programming Fundamentals and OOP:** These courses provide the foundational programming knowledge needed for software development, including the skills to build and maintain the FarmConnect mobile app and web dashboard.

b. **Data Structures and Algorithms:** Knowledge of data structures and algorithms is essential for optimizing the efficiency and performance of the FarmConnect platform.

c. **Operating System:** Familiarity with operating systems is crucial for server setup, maintenance, and deployment of FarmConnect.

d. **Modern Programming Languages:** Java programming skills are essential for the development of the Android mobile app component of FarmConnect.

e. **Software Engineering:** FarmConnect represents a real-world software engineering project, incorporating various software development methodologies, best practices, and tools.

f. **Software Requirement Engineering and Software Design and Architecture:** These courses contribute to the design and planning phases of FarmConnect, ensuring that user requirements are addressed in the architecture of the platform.

g. **Web Technologies, Web Engineering, and Mobile App Development:** These courses equip us with the knowledge and skills to develop the web dashboard and mobile app components of FarmConnect.

h. **Computer Networks:** Understanding computer networks is critical for ensuring data transfer and communication within the FarmConnect platform is secure and efficient.

i. **Information Security:** This course is directly relevant to safeguarding user data and ensuring the security of the FarmConnect platform.

j. **Introduction to Computer Technology and Object-Oriented Analysis and Design:** These courses provide the foundational knowledge of computer technology and object-oriented design principles that are applied in FarmConnect.

k. **Software Construction and Software Quality Engineering:** These courses contribute to the coding and quality assurance phases of FarmConnect development, ensuring robust and reliable software.

l. **Software Project Management:** Project management concepts are essential for the successful planning, execution, and delivery of the FarmConnect project.

m. **Database Management Systems (DBMS):** Knowledge of database management systems is essential for designing and managing the database used to store user data in FarmConnect.

n. **Human Computer Interaction:** These courses provide insights into designing a user-friendly interface for the FarmConnect mobile app and web dashboard.

o. **Formal Methods in Software Engineering:** The use of formal methods contributes to the overall quality and reliability of FarmConnect software.

p. **Professional Practices:** This course equips you with the skills and knowledge required for professional software development, including ethical and legal considerations.

The FarmConnect project draws upon a broad spectrum of skills and knowledge from Software Engineering degree program, demonstrating the interdisciplinary nature of software development projects and their real-world applications. It is a culmination of your education in software engineering, programming, and related fields.

## 1.3. Project Background

The FarmConnect software addresses the challenge of fostering collaboration and knowledge-sharing within the farming community. Existing agriculture apps lack a holistic approach to connecting farmers with networking, information exchange, and collaboration tools. Our motivation to develop this system arises from the absence of a comprehensive platform tailored to farmers' unique needs, encompassing advanced features like group communities, marketplace, and secure communication. While similar systems exist, they often lack crucial functionalities. Re-implementing FarmConnect offers a chance to refine these features, enhancing user experience and adaptability. We aim to learn skills in user-centered application design, integrating diverse functionalities, and building a secure and scalable platform for real-world agriculture practices.

FarmConnect offers an integrated solution by providing personalized user profiles, advanced search, newsfeed, group communities, and marketplace. Through a user-centric design, it ensures ease of use and intuitive navigation. The system's algorithm utilizes user preferences and tags for effective content discovery. This project emphasizes data security protocols to facilitate secure communication. By amalgamating various agricultural features into a single application, FarmConnect eliminates fragmentation and fosters a robust farming community. The re-implementation focuses on refining functionalities based on user feedback, ensuring a more effective tool for farmer networking, education, and resource exchange.

## 1.4. Literature Review

There are just few applications ideas that are related to our idea social media farming app (FarmConnect). Few of them are discussed below:

a) **Agri Setu – Agriculture app**

The AgriSetu app serves as a comprehensive platform connecting farmers with their diverse agricultural needs, from seeds and fertilizers to equipment and advisory services. It facilitates market linkage and features real-time weather updates, market prices, farming-related magazines, a community for farmer interactions, expert consultations, and soil testing information. This app streamlines the agriculture supply chain, allowing farmers to optimize their productivity and sell their product directly to institutional buyers.

**Short-Comings:**

a. Events and Forums
b. Photo Analysis for Expert Feedback

**b) Bakhabar Kissan**

The Bakhabar Kissan app empowers Pakistani farmers through an agricultural digital hub, connecting them with experts and the agricultural value chain. It provides user-friendly interface that offers dynamic insights on weather, crop and livestock advisory, modern techniques, and disaster management. Key features include crop guidance from soil prep to post-harvest, livestock management practices, photo analysis for expert feedback, modern

farming techniques, instructional videos, localized weather forecasts, and product information for agricultural machinery. This app serves as a comprehensive solution for farmers' needs, fostering community and technological empowerment.

**Short-Coming:**

a. Marketplace for Products and Services
b. Events and Forums
c. Secure Transactions

**c) Kissan Madadgar**

Kissan Madadgar, an initiative by Concave Agri, offers real-time farming advisory to Pakistani farmers, enhancing productivity. Features include advice on over 50 crops, daily crop updates, real-time weather information, Kissan Madadgar Videos, informative articles, news, and a helpline with a WhatsApp account.

**Short-Comings**

a. Groups and Communities
b. Marketplace for Products and Services
c. Events and Forums
d. Photo Analysis for Expert Feedback

## 1.5. Analysis from Literature Review (in the context of your project)

The table 1.5.1 presented below provides an overview of related applications, comparing them to FarmConnect.

| Features | Agri Setu – Agriculture app | Bakhabar Kissan | Kissan Madadgar | FarmConnect |
|---|---|---|---|---|
| **User Profiles and Networking** | Yes | Yes | Yes | Yes |
| **Newsfeed and Updates** | Yes | Yes | Yes | Yes |
| **Groups and Communities** | Yes | Yes | No | Yes |
| **Messaging and Collaboration** | Yes | Yes | Yes | Yes |
| **Marketplace for Products and Services** | Yes | No | No | Yes |
| **Knowledge Sharing and Education** | Yes | Yes | Yes | Yes |
| **Events and Forums** | No | No | No | Yes |
| **Notifications and Alerts** | Yes | Yes | Yes | Yes |
| **Weather Updates and Crop Monitoring** | Yes | Yes | Yes | Yes |

| | | | | |
|---|---|---|---|---|
| **User-generated Content Sharing** | Yes | Yes | Yes | Yes |
| **Expert Consultation** | Yes | Yes | Yes | Yes |
| **Photo Analysis for Expert Feedback** | No | Yes | No | No |
| **Secure Transactions** | Yes | No | No | No |
| **Dynamic Insights on Disaster Management** | No | Yes | No | No |

*Table 1.5 Literature Review*

## 1.6. Methodology and Software Lifecycle for This Project

For the development of FarmConnect, the chosen software methodology is Agile. This decision is based on the project's dynamic nature, the need for continuous collaboration with stakeholders, and the requirement to iterate quickly based on user feedback. Agile methodology's iterative and incremental approach aligns well with the goal of delivering a functional product while continuously improving and adapting to changing needs. This methodology allows for flexible adjustments to the evolving requirements of the agricultural community, ensuring that the final product effectively addresses their needs and challenges.

### 1.6.1. Rationale behind Selected Methodology

The rationale for selecting the Agile methodology for this project is rooted in its compatibility with the project's nature and goals.

#### *1.6.1.1. Rationale behind Selected Methodology*

Agile was chosen as the project's methodology because of its iterative and incremental approach, which suits the dynamic nature of FarmConnect. It enables continuous collaboration with stakeholders and the ability to quickly adapt based on user feedback. This flexibility is crucial in delivering a product that effectively addresses the evolving requirements of the agricultural community.

# Chapter 2: Problem Definition

This chapter discusses the precise problem to be solved.

## 2.1. Problem Statement

The challenge is to facilitate collaboration and knowledge-sharing within the farming community, as existing agricultural apps lack comprehensive tools for networking, information exchange, and collaboration.

## 2.2. Deliverables and Development Requirements

**Deliverables:**
These project deliverables collectively represent the objectives and outcomes of the FarmConnect project, providing a comprehensive social media platform tailored to the unique needs of the agricultural community.

a. **FarmConnect Mobile Application:** The primary deliverable is the fully functional FarmConnect mobile application for Android devices. The application will be designed, developed, and thoroughly tested to ensure a seamless user experience.

b. **FarmConnect Web Dashboard:** A secondary deliverable is the FarmConnect web dashboard. This web-based interface will be developed using Laravel and will provide users with additional functionalities, including data analysis, research tools, and administrative controls.

c. **User Profiles and Networking Features:** Users will have the ability to create and manage personalized profiles, showcasing their expertise, farming practices, and location. Advanced search and discovery features will enable users to connect with like-minded farmers, facilitating the creation of valuable networks and partnerships.

d. **Newsfeed and Updates Functionality:** A centralized newsfeed within the mobile app and web dashboard will allow users to share updates, photos, and videos related to their farming practices. Users will be able to engage with posts through actions such as liking, commenting, sharing, and promoting a supportive community.

e. **Groups and Communities Management:** FarmConnect will enable farmers to create or join groups based on specific interests, crops, regions, or farming techniques. Group discussions, announcements, and event sharing will facilitate knowledge exchange and collaboration among farmers.

f. **Messaging and Collaboration Tools:** The project will deliver private messaging and group chat functionalities, ensuring seamless communication between farmers.

Farmers will be able to share documents, images, and multimedia files, enhancing collaboration and information sharing.

g. **Marketplace for Agricultural Products and Services:** Within the FarmConnect app, users will have access to a dedicated marketplace for buying, selling, or exchanging agricultural products, equipment, and services.

h. **Knowledge Sharing and Education Features:** FarmConnect will encourage farmers to share their expertise through articles, tips, and tutorials. A categorized knowledge base and tagging system will make it easy for users to discover and access relevant agricultural information.

i. **Notifications and Alerts System:** Real-time notifications will keep users informed about new messages, comments, likes, group activities, and upcoming events. Customizable notification preferences will ensure that users stay connected without feeling overwhelmed.

j. **Weather Updates:** Integration with weather APIs will provide farmers with accurate and timely weather forecasts.

**Development Requirements:**

The development requirements for FarmConnect, the agricultural social media platform, encompass a range of technical, design, and operational elements necessary to bring the project to fruition. Here are the key development requirements for FarmConnect: Deliverables and development requirements.

1. **Software Development:**
   a. **Mobile App Development:** The core component of FarmConnect is the mobile application. This requires Android app development, utilizing Java or Kotlin, for compatibility with Android devices.
   b. **Web Development:** A web dashboard will be developed using web technologies, with Laravel as the backend framework. Front-end development using HTML, CSS, and JavaScript is essential to create an intuitive and user-friendly web interface.

2. **Database Management:**
   a. **Database Creation:** A relational database management system (RDBMS) such as MySQL is required to store user profiles, posts, marketplace listings, and other critical data.
   b. **Database Administration:** Ongoing database administration will be essential to ensure data integrity, security, and scalability.

3. **API Integration:**
   a. **Weather APIs:** Integration with weather APIs will provide farmers with accurate and timely weather forecasts, requiring API integration and data synchronization.

b. **Mobile App APIs:** Development and integration of APIs for the mobile app to enable features like user profiles, messaging, and notifications.
c. **ChatGPT API Integration:** Facilitation with chatbot for farmers.

4. **User Interface (UI) and User Experience (UX) Design:**
   a. **UI/UX Design:** Designers will need to create user-friendly and visually appealing interfaces for both the mobile app and web dashboard.
   b. **Responsive Design:** The UI/UX should be responsive to ensure usability on a variety of devices and screen sizes.

5. **Networking and Data Security:**
   a. **Network Security:** Implementation of security measures to protect user data and ensure secure communication within the app and web platform.
   b. **Data Encryption:** Data encryption methods must be applied to protect user data and communications.

6. **Cloud Infrastructure:**

   **Cloud Hosting:** The project may require cloud hosting services to host and scale the application and database.

7. **Testing and Quality Assurance:**
   a. **Testing Environment:** Setting up a testing environment to conduct rigorous testing, including functional testing, usability testing, and security testing.
   b. **Bug Tracking:** Implementing a system for tracking and resolving bugs and issues identified during testing.

8. **Documentation:**
   a. **Development Documentation:** Creation of technical documentation for developers, including code documentation and architecture descriptions.
   b. **User Documentation:** Preparing user manuals and guides for app and web platform users.

9. **Project Management:**

   **Project Management Tools:** Utilizing project management tools and methodologies (e.g., Agile, Scrum) to manage development tasks, timelines, and team collaboration.

10. **Hardware Requirements:**

    **Development Devices:** Developers may require a range of Android devices for testing the mobile app and a laptop with at least 8 GB Ram, core i 5($6^{th}$ Generation), and 250GB SSD for development.

11. **Server and Hosting:**

    Selection and provisioning of servers or cloud hosting services to ensure the platform's availability, scalability, and performance.

## 2.3. Current System

The AgriSetu™ app is a comprehensive agricultural platform that seamlessly connects Indian farmers and stakeholders, offering a wide range of services and real-time information. Farmers can access market prices, weather updates, create personal profiles and pages, stay informed through farming-related magazines, engage with a community of experts and fellow farmers, and consult agriculture specialists for personalized advice. The app also provides a directory of nearby soil testing laboratories. AgriSetu™ is a one-stop solution, enhancing the efficiency and profitability of farming in India. Figure 2.1 gives some ideas of AgriSetu UI.

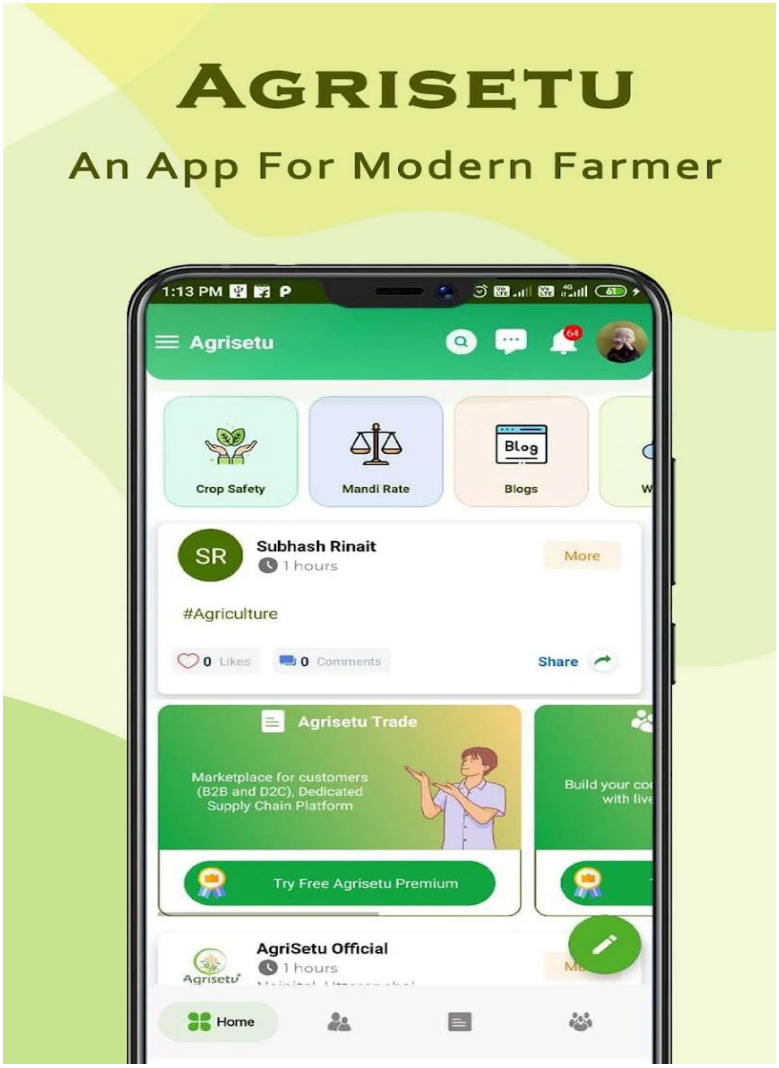


*Figure 2.3 AgriSetu*

AgriSetu™ revolutionizes the farming landscape in India by providing farmers with essential tools and resources at their fingertips. With real-time information, expert consultations, and a robust community, the app empowers farmers to make informed decisions and improve their productivity. AgriSetu™ is dedicated to enhancing the efficiency and profitability of Indian agriculture.

# Chapter 3: Requirement Analysis

Software Requirements Specification (SRS) report should be included in this chapter.

## 3.1. Functional Requirements

The functional requirements are outlined in the table below.

| Functional Requirements | Requirements Description |
|---|---|
| **User Authentication** | 1. Users must be able to register with a unique User ID, mobile number, and password.<br>2. Mobile numbers must be validated through OTP during registration.<br>3. Users should be able to log in securely using their credentials.<br>4. The system should provide a secure sign-out option for users. |
| **Profile Management** | 1. Users should have the ability to create and manage personalized profiles, highlighting their expertise, farming practices, and location.<br>2. Advanced search and discovery features should enable users to connect with like-minded farmers. |
| **Post Interaction** | 1. Users can create posts with text, images, and videos.<br>2. The system supports likes, comments, and shares on posts.<br>3. A centralized newsfeed displays updates from other farmers.<br>4. Users can create and manage groups based on interests or farming practices. |
| **Messaging** | Users can engage in private and group chat, sharing documents, images, and files. |
| **Marketplace** | A dedicated marketplace allows users to buy, sell, or exchange agricultural products and services. |
| **Advertising and Buying** | 1. Users can advertise products in the marketplace.<br>2. The system supports buying products with negotiation features and secure transactions. |
| **View Stories (Web Dashboard Only)** | Users can view and respond to stories through the web dashboard. |
| **Privacy Settings** | Users have control over privacy settings, managing who can view their profile and posts. |
| **Reporting** | Users can report inappropriate content, and reported posts are reviewed by FarmConnect admins. |

*Table 3.1 FR*

## 3.2. Non-Functional Requirements

The non-functional requirements are outlined in the table below.

| Non-Functional Requirements | Requirements Description |
|---|---|
| **Scalability** | The system should easily grow to handle more users and data. |
| **Reliability** | The platform should be available 99.9% of the time. |
| **Security** | User data and communications should be encrypted.<br>Secure authentication and authorization should prevent unauthorized access. |
| **Usability** | The user interface should be easy to use for farmers with varying levels of tech experience. |
| **Compatibility** | The web dashboard should be compatible with popular browsers. |
| **Maintainability** | The system should be easy to update and maintain.<br>Code should be well-documented. |
| **Data Backup and Recovery** | No user data will be lost as it is stored on cloud. |
| **Compliance** | The system should follow data protection and privacy laws. |
| **User Support** | There should be a support system to help users with issues. |

*Table 3.2 NFR*

## 3.3. Use Cases

Utilizing use cases is a popular and effective method for capturing requirements. Prior to drafting functional requirements, employing use cases aids in comprehending user expectations.

FarmConnect offers a range of use cases designed to enhance the connectivity and productivity of farmers. Through personalized profiles, farmers can create detailed profiles that showcase their farming practices, achievements, and areas of expertise, fostering a sense of community and professional networking. The newsfeed feature keeps users informed about the latest agricultural trends, market updates, and innovations, enabling them to stay ahead in their field. Group communities within the app allow farmers to join and participate in discussions, share experiences, and collaborate on common interests or challenges.

Furthermore, FarmConnect's robust marketplace empowers farmers to buy and sell products, tools, and services directly within the app, streamlining transactions and expanding their market reach. The advanced search capabilities enable users to quickly find relevant information, resources, and connections, enhancing their decision-making process. By integrating these features, FarmConnect not only simplifies daily farming activities but also promotes a collaborative environment where farmers can share knowledge, seek advice, and support each other, ultimately leading to increased productivity and a stronger agricultural community. Continuous user feedback ensures that FarmConnect evolves to meet the changing needs of farmers, making it an indispensable tool for modern agriculture.

### 3.3.1. Use Case Model

The following use case model provides an overview of functional requirements before delving into detailed use cases.

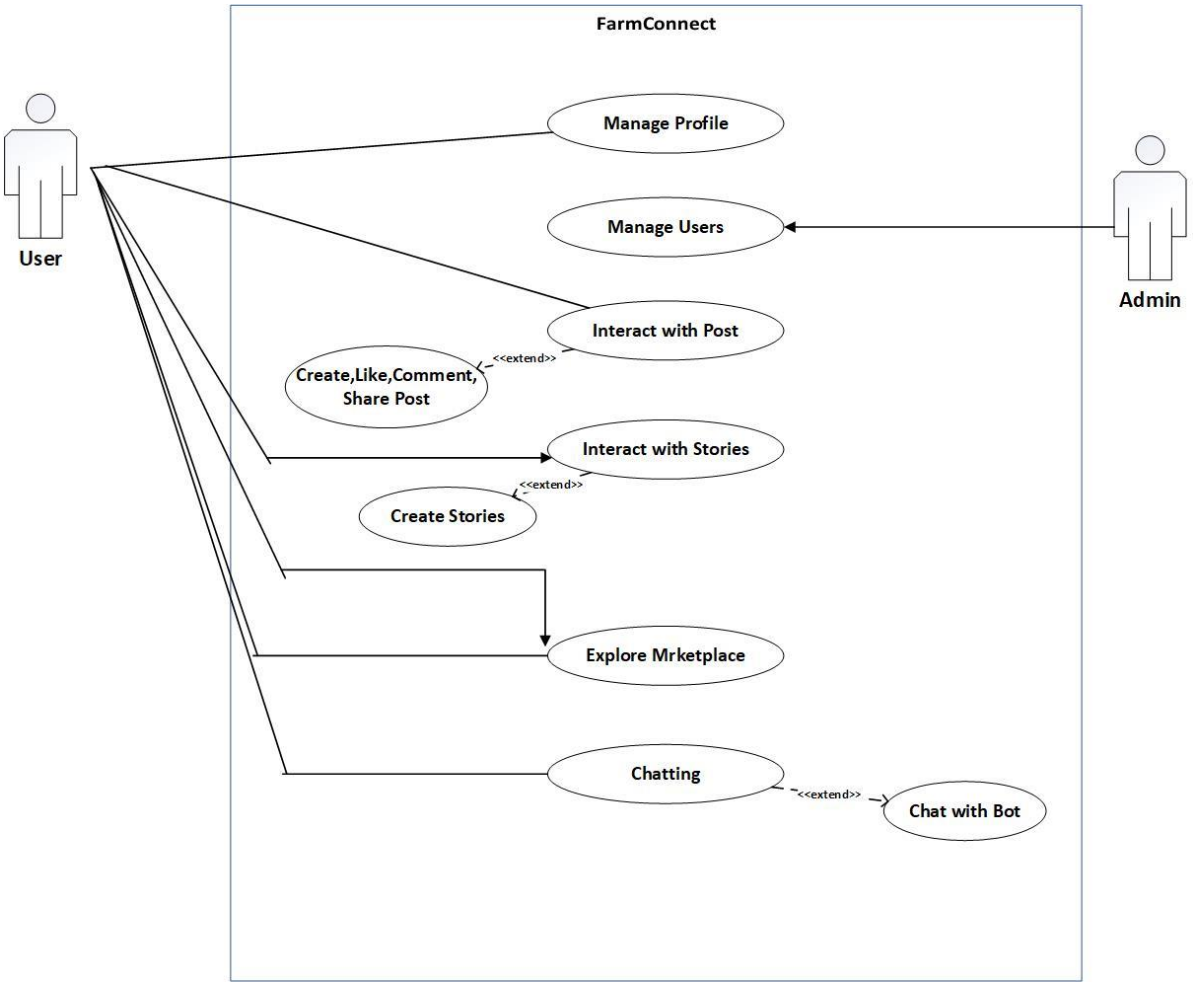


*Figure 3.3.2 Use Case*

### 3.3.2. Actors Description

1. **User:**
   - **Description:** Represents individuals who use the FarmConnect platform.
   - **Roles:**
     **Registered User:** Engages in various activities on the platform, including posting, interacting with posts, using marketplace features, and participating in chats.

2. **Seller:**
   - **Description:** Represents individuals or entities selling agricultural products or services on the FarmConnect platform.
   - **Roles:**
     a. **Product Listing:** Creates and manages listings for agricultural products or services in the marketplace.

b. **Interaction with Buyers:** Responds to inquiries, negotiates deals, and manages transactions with potential buyers.

3. **Buyer:**
   - **Description:** Represents individuals or entities interested in purchasing agricultural products or services on the FarmConnect platform.
   - **Roles:**
     a. **Browsing and Searching:** Explores the marketplace, searches for specific products, and views product details.
     b. **Interaction with Sellers:** Contacts sellers, negotiates deals, and initiates transactions for purchasing.

4. **Admin:**
   - **Description:** Represents administrators who manage and oversee the FarmConnect platform.
   - **Roles:**
     a. **Content Moderation:** Monitors and moderates user-generated content, including posts and comments.
     b. **User Management:** Manages user accounts, addresses issues, and ensures platform integrity.

### 3.3.3. Use Case Description

The following tables presents you use cases as well as guides you about FarmConnect.

| | |
|---|---|
| **Use Case ID:** | UC-1.1.1 |
| **Use Case Name:** | Register User |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a user registering on the FarmConnect platform, including the validation of the mobile number via OTP. |
| **Trigger:** | The user initiates the registration process. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection. |
| **Postconditions:** | 1. User receives a confirmation of successful registration.<br>2. User can now log in to the platform. |
| **Normal Flow:** | 1. User opens the FarmConnect mobile app.<br>2. User taps the "Register" button.<br>3. The system displays a registration form.<br>4. User enters their UserID, which may be an email address or a unique username.<br>5. User enters their mobile number.<br>6. User creates a password.<br>7. User agrees to the terms and conditions.<br>8. User taps the "Register" button. |

| | |
|---|---|
| | 9. The system generates an OTP and sends it to the provided mobile number.<br>10. User receives the OTP via SMS.<br>11. User enters the OTP received on their mobile.<br>12. The system validates the OTP.<br>13. If the OTP is valid, the system registers the user.<br>14. User receives a confirmation message. |
| **Alternative Flows:** | **[Alternative Flow 1 - Invalid OTP]:**<br>12a. In step 12, if the OTP entered by the user is invalid.<br>1. The system displays an error message.<br>2. User re-enters the OTP.<br>3. The use case resumes from step 12. |
| **Exceptions:** | **[Mobile Number Already Registered (Step 9)]**<br>In step 9, if the mobile number is already registered.<br>1. The system displays an error message indicating that the mobile number is already associated with an account.<br>2. User is prompted to log in or recover their account if necessary. |
| **Includes:** | None |
| **Special Requirements:** | **Security and Data Protection:**<br>• User passwords must be securely stored and hashed to protect user accounts.<br>• User data, including mobile numbers, must be encrypted in transit and at rest. |
| **Assumptions:** | 1. The user has a valid mobile phone number that can receive SMS messages.<br>2. The user is responsible for keeping their OTP confidential.<br>3. The user understands and accepts the terms and conditions of using FarmConnect.<br>4. The user has already downloaded and installed the FarmConnect mobile app.<br>5. The user's mobile phone has an active internet connection.<br>6. The FarmConnect platform is accessible to the user. |
| **Notes and Issues:** | 1. Continuous monitoring and logging of registration activities are essential for security and auditing purposes.<br>2. Ensure that the OTP delivery process is robust and reliable, as it plays a critical role in user verification.<br>3. Consider providing a mechanism for users to recover their accounts if they encounter issues during registration, such as incorrect OTP entry.<br>4. User should be registered with at least 8 digits password. |

*Table 3.3.3.1: UC-1.1.1 Registration*

| | |
|---|---|
| **Use Case ID:** | UC-1.1.2 |
| **Use Case Name:** | User Login |
| **Actors:** | Primary Actor: User |

| | |
|---|---|
| **Description:** | This use case describes the process of a registered user logging into the FarmConnect platform to access their account. |
| **Trigger:** | The user initiates the login process. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and has a valid account on FarmConnect. |
| **Postconditions:** | 1. User successfully logs in and gains access to their FarmConnect account.<br>2. User's personalized data and features are available within the app. |
| **Normal Flow:** | 1. User opens the FarmConnect mobile app.<br>2. The system displays the login screen.<br>3. User enters their UserID (email or username) and password.<br>4. User taps the "Login" button.<br>5. The system validates the entered credentials.<br>6. If the credentials are correct, the system logs the user into their account.<br>7. User gains access to their FarmConnect account. |
| **Alternative Flows:** | **[Alternative Flow 1 - Forgotten Password with Mobile Number and OTP]:**<br>1. In step 3 of the normal flow, the user realizes they have forgotten their password.<br>2. The user taps the "Forgot Password?" link on the login screen.<br>3. The system prompts the user to enter their registered mobile number.<br>4. User enters their mobile number.<br>5. The system generates an OTP and sends it to the provided mobile number.<br>6. User receives the OTP via SMS.<br>7. User enters the OTP received on their mobile.<br>8. The system validates the OTP.<br>9. If the OTP is valid, the system allows the user to create a new password.<br>10. User creates a new password and confirms it.<br>11. The system updates the user's password.<br>12. User is redirected to the login screen and can now log in with the new password. |
| **Exceptions:** | **Invalid Credentials (Step 5):**<br>1. In step 5, if the entered credentials are incorrect.<br>2. The system displays an error message indicating that the credentials are invalid.<br>3. User is prompted to re-enter the correct UserID and password. |
| | **Invalid OTP (Step 8):**<br>1. In step 8 of the alternative flow if the entered OTP is incorrect.<br>**2.** The system displays an error message indicating that the OTP is invalid.<br>**3.** User is prompted to re-enter the correct OTP. |

| | |
|---|---|
| **Includes:** | Register User |
| **Special Requirements:** | **Security and Data Protection:**<br>• User credentials must be securely stored and hashed to protect user accounts from unauthorized access or data breaches.<br>• User data, including passwords and personal information, must be encrypted in transit and at rest to safeguard user privacy. |
| | **Session Management:**<br>Implement proper session management to ensure that user sessions are securely maintained, and users are automatically logged out after a certain period of inactivity. |
| **Assumptions:** | 1. **User Account Existence:** It is assumed that the user attempting to log in has a valid and existing FarmConnect account with a registered UserID (email or username) and password.<br>2. **Internet Connection:** It is assumed that the user's mobile device has an active and stable internet connection to access the FarmConnect platform.<br>3. **Correct UserID and Password:** Users are expected to provide the correct UserID and password associated with their FarmConnect account for a successful login.<br>4. **User Familiarity:** Users are assumed to be familiar with the process of logging into mobile applications and have knowledge of their own login credentials.<br>5. **Mobile Device Compatibility:** It is assumed that the FarmConnect mobile app is compatible with the user's mobile device and operating system.<br>6. **Privacy and Data Protection:** Users are responsible for maintaining the privacy and security of their login credentials and personal information. They should not share their login information with others.<br>7. **No Unauthorized Access:** The system assumes that unauthorized users will not have access to a registered user's mobile device or login information.<br>8. **OTP Delivery:** In the alternative flow where a user forgets their password and requests an OTP, it is assumed that the mobile number associated with the user's account is valid and accessible for OTP delivery.<br>9. **OTP Validity:** It is assumed that the OTP generated for password reset is valid for a reasonable period and can be used within that time frame.<br>10. **User Cooperation:** Users are expected to follow the instructions provided during the password reset process, including entering the OTP correctly and creating a new password. |
| **Notes and Issues:** | 1. Password Recovery: Providing users with the ability to recover their forgotten passwords through OTP validation is a valuable feature to prevent users from being locked out of their accounts. |

| | |
|---|---|
| | 2. Account Lockout Policy: Consider implementing a policy that temporarily locks user accounts after a specified number of unsuccessful login attempts to deter brute force attacks.<br>3. Password Policies: The system should enforce strong password policies, including complexity requirements, to enhance account security. |

*Table 3.3.3.2: UC-1.1.2 User Login*

| | |
|---|---|
| **Use Case ID:** | UC-1.1.3 |
| **Use Case Name:** | Sign Out |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user signing out of their FarmConnect account within the mobile app. |
| **Trigger:** | The user initiates the process of signing out of their FarmConnect account. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and logged into their FarmConnect account. |
| **Postconditions:** | The user's session is terminated, and they are successfully signed out of their FarmConnect account. |
| **Normal Flow:** | 1. User opens the FarmConnect mobile app.<br>2. User navigates to the account or profile section.<br>3. The system provides an option to "Sign Out" or "Log Out."<br>4. User selects the "Sign Out" or "Log Out" option.<br>5. The system prompts the user to confirm the sign-out action.<br>6. User confirms the sign-out action.<br>7. The system terminates the user's session and logs them out of the app.<br>8. The user is returned to the app's login or landing page. |
| **Alternative Flows:** | None. |
| **Exceptions:** | **Cancel Sign Out (Step 6):**<br>1. In step 6 of the normal flow, if the user decides to cancel the sign-out action.<br>2. The system cancels the sign-out request, and the user remains logged in. |
| **Includes:** | User Login |
| **Special Requirements:** | 1. Users should be required to confirm the sign-out action to prevent accidental logouts.<br>2. The system should clear any locally stored user data upon sign-out for security and privacy. |
| **Assumptions:** | 1. Users understand the implications of signing out and are familiar with the process.<br>2. Sign-out is a secure process, and user data is protected. |
| **Notes and Issues:** | 1. The user experience for signing out should be straightforward and user-friendly. |

| | |
|---|---|
| | 2. Support for password or biometric re-authentication may be considered as an additional security layer when signing back in after sign-out. |

*Table 3.3.3.3: UC-1.1.3 Sign Out*

| | |
|---|---|
| **Use Case ID:** | UC-2.1.1 |
| **Use Case Name:** | Manage Profile |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user managing and updating their profile information on the FarmConnect platform. |
| **Trigger:** | The user initiates the process of managing their profile, which may include editing personal information, updating profile picture, and other profile-related actions. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and has a valid account on FarmConnect.<br>**4.** User is logged into their FarmConnect account. |
| **Postconditions:** | The user's profile information is updated according to their actions, and changes are reflected in their FarmConnect profile. |
| **Normal Flow:** | 1. User opens the FarmConnect mobile app.<br>2. User navigates to the "Profile" section.<br>3. The system displays the user's profile with options to manage it.<br>4. User selects "Edit Profile" or similar action.<br>5. The system presents a form or interface for the user to edit and update their profile information, which may include:<br>• Name<br>• Profile picture<br>• Contact information<br>• Bio or description<br>• Farm Story<br>• Other relevant information<br>6. User makes the desired changes.<br>7. User saves the changes.<br>8. The system updates the user's profile with the edited information. |
| **Alternative Flows:** | **Alternative Flow 1 - Cancel Profile Editing (Step 7):**<br>1. In step 7 of the normal flow, the user decides to cancel the profile editing.<br>2. The system discards the changes and returns to the user's profile display without saving any updates. |
| **Exceptions:** | **Profile Update Error (Step 7):**<br>1. In step 7 of the normal flow, if there is an error in updating the user's profile information.<br>2. The system displays an error message indicating the update failure.<br>3. User is prompted to reattempt the update. |
| **Includes:** | None. |

| | |
|---|---|
| **Special Requirements:** | The user should be able to upload and set a profile picture or avatar.<br>User information changes should be reflected consistently across the FarmConnect platform. |
| **Assumptions:** | 1. Users are familiar with updating their profile information on social media platforms.<br>2. Users have the necessary permissions to edit and update their profile. |
| **Notes and Issues:** | 1. User experience design for profile management should be intuitive and user-friendly.<br>2. The system should validate user-provided information and handle data consistency and integrity.<br>3. The handling of user-generated content, including profile pictures, should adhere to content guidelines and standards. |

*Table 3.3.3.4: UC-2.1.1 Manage Profile*

| | |
|---|---|
| **Use Case ID:** | UC-2.2.1 |
| **Use Case Name:** | Create Post |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user creating a post on the FarmConnect platform, including the option to upload an image or video as part of the post. |
| **Trigger:** | The user initiates the process of creating a new post. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and has a valid account on FarmConnect.<br>4. User is logged into their FarmConnect account. |
| **Postconditions:** | 1. A new post is created and added to the FarmConnect platform.<br>2. The post is visible to other users based on the user's privacy settings.<br>3. If an image or video is uploaded, it is associated with the post and visible to other users. |
| **Normal Flow:** | 1. User opens the FarmConnect mobile app.<br>2. The system displays the user's home feed or the post creation screen.<br>3. User taps the "Create Post" button.<br>4. The system presents a post creation form, including fields for text content.<br>5. User enters the text content for the post.<br>6. User taps the "Upload Image/Video" option.<br>7. The system allows the user to select an image or video from their device's gallery.<br>8. User selects an image or video.<br>9. The system uploads the selected image or video.<br>10. User taps the "Post" button to create the post. |

| | |
|---|---|
| | 11. The system saves the post to the user's profile and displays it in the feed. |
| **Alternative Flows:** | **[Alternative Flow 1 - Without Image/Video]:**<br>1. In step 6 of the normal flow, the user chooses not to upload an image or video.<br>2. The system proceeds with post creation without an image or video attached. |
| **Exceptions:** | **Upload Error (Step 9):**<br>1. In step 9 of the normal flow, if there is an error in uploading the selected image or video.<br>2. The system displays an error message indicating the upload failure.<br>3. User is prompted to reattempt the upload or continue without an image or video. |
| **Includes:** | 1. **Upload Media:** This use case includes the functionality for users to upload images or videos as part of their posts. The "Image/Video Upload" use case allows users to select and upload media files from their device's gallery.<br>2. **Manage Privacy:** As part of creating a post, the user may set privacy settings to control who can view the post. The "Privacy Settings" use case allows users to define the audience for their posts. |
| **Special Requirements:** | 1. Images and videos must adhere to specified file format and size limits for successful upload.<br>2. Privacy settings should allow users to control who can view their posts. |
| **Assumptions:** | 1. Users are familiar with the process of creating and sharing posts on social media platforms.<br>2. Users have the necessary permissions to access their device's image and video gallery for upload.<br>3. Uploaded content should comply with FarmConnect's content guidelines and standards. |
| **Notes and Issues:** | 1. User experience design for selecting and uploading images or videos should be intuitive and user-friendly.<br>2. Content moderation and reporting mechanisms should be in place to handle inappropriate or harmful content. |

*Table 3.3.3.5: UC-2.2.1 Create Post*

| | |
|---|---|
| **Use Case ID:** | UC-2.2.2 |
| **Use Case Name:** | Upload Media |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user uploading an image or video as part of creating a post on the FarmConnect platform. |
| **Trigger:** | The user initiates the process of uploading an image or video while creating a post. |

| | |
|---|---|
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and has a valid account on FarmConnect.<br>4. User is logged into their FarmConnect account.<br>5. The user is in the process of creating a post. |
| **Postconditions:** | The selected image or video is successfully uploaded and associated with the user's post. |
| **Normal Flow:** | 1. User is in the process of creating a new post (part of the "Create a Post with Image/Video Upload" use case).<br>2. User selects the "Upload Image/Video" option.<br>3. The system allows the user to select an image or video from their device's gallery.<br>4. User chooses an image or video from their gallery.<br>5. The system initiates the upload of the selected image or video.<br>6. The system displays a progress indicator during the upload.<br>7. Upon successful upload, the system associates the image or video with the user's post. |
| **Alternative Flows:** | **[Alternative Flow 1 - Cancel Upload (Step 5)]:**<br>In step 5 of the normal flow, the user decides to cancel the upload.<br>The system cancels the upload process and returns to the post creation form. |
| **Exceptions:** | **Upload Error (Step 5):**<br>1. In step 5 of the normal flow, if there is an error in uploading the selected image or video.<br>2. The system displays an error message indicating the upload failure.<br>3. User is prompted to reattempt the upload or continue without an image or video. |
| **Includes:** | None |
| **Special Requirements:** | 1. Images and videos must adhere to specified file format and size limits for successful upload.<br>2. A clear and user-friendly progress indicator should be provided to inform users about the upload status. |
| **Assumptions:** | 1. Users are familiar with the process of creating and sharing posts on social media platforms.<br>2. Users have the necessary permissions to access their device's image and video gallery for upload. |
| **Notes and Issues:** | 1. User experience design for selecting and uploading images or videos should be intuitive and user-friendly.<br>2. Uploading large media files may require additional considerations, such as optimization and compression.<br>3. Error handling during the upload process should provide informative messages to guide the user in case of failures. |

*Table 3.3.3.6: UC-2.2.2 Upload Media*

| | |
|---|---|
| **Use Case ID:** | UC-2.2.3 |
| **Use Case Name:** | View Posts |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user viewing posts created by other users within the FarmConnect mobile app. Posts may include text, images, videos, and other content related to agriculture. |
| **Trigger:** | The user initiates the process of viewing posts by navigating to the Posts section within the FarmConnect mobile app. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and logged into their FarmConnect account.<br>4. The user is within the Posts section of the FarmConnect app. |
| **Postconditions:** | The user has successfully viewed posts created by other users within the FarmConnect mobile app. |
| **Normal Flow:** | 1. User opens the FarmConnect mobile app.<br>2. User navigates to the Posts section within the app.<br>3. The system displays a list of available posts, which may include text, images, videos, and other content.<br>4. User can select a specific post to view.<br>5. The system displays the selected post in a user-friendly format.<br>6. User can view the post content, including text, images, and videos.<br>7. User can interact with the post, such as liking, commenting, or sharing, depending on the available options. |
| **Alternative Flows:** | **Alternative Flow 1 - No Available Posts (Step 3):**<br>1. In step 3 of the normal flow, if there are no available posts to view.<br>2. The system displays a message indicating that there are no posts currently available. |
| **Exceptions:** | **Loading Error (Step 3):**<br>1. In step 3 of the normal flow, if there is an error in loading available posts.<br>2. The system displays an error message indicating the loading failure.<br>3. User may attempt to reload or troubleshoot the issue. |
| **Includes:** | None. |
| **Special Requirements:** | 1. Posts should be presented in an engaging and user-friendly format.<br>2. Users should have the ability to interact with posts, such as liking, commenting, or sharing.<br>3. The system should provide a clear and intuitive navigation to the Posts section within the mobile app |
| **Assumptions:** | 1. Users are familiar with the concept of viewing and interacting with posts in social media platforms.<br>2. Posts are updated and available for viewing regularly within the mobile app. |
| **Notes and Issues:** | 1. User experience for viewing and interacting with posts should be designed to enhance engagement and user satisfaction. |

| | |
|---|---|
| | 2. Consider providing a clear message to users that posts can only be viewed within the mobile app. |

*Table 3.3.3.7: UC-2.2.3 View Posts*

| | |
|---|---|
| **Use Case ID:** | UC-2.2.4 |
| **Use Case Name:** | Like Post |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user liking a post created by another user within the FarmConnect mobile app. |
| **Trigger:** | The user initiates the process of liking a post within the app. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and logged into their FarmConnect account.<br>4. The user is viewing a post within the app. |
| **Postconditions:** | The user has successfully liked the post, and their like action is recorded. |
| **Normal Flow:** | 1. User is viewing a post within the FarmConnect mobile app.<br>2. The user selects the "Like" button associated with the post.<br>3. The system registers the user's like action for the post.<br>4. The "Like" button changes to indicate that the user has liked the post. |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Includes:** | None |
| **Special Requirements:** | The "Like" button should provide visual feedback to indicate that the user has successfully liked the post. |
| **Assumptions:** | 1. Users are familiar with the concept of liking posts in social media platforms.<br>2. Posts are regularly updated and available for liking within the mobile app. |
| **Notes and Issues:** | None |

*Table 3.3.3.8: UC-2.2.4 Like Posts*

| | |
|---|---|
| **Use Case ID:** | UC-2.2.5 |
| **Use Case Name:** | Comment on Post |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user leaving a comment on a post created by another user within the FarmConnect mobile app. |
| **Trigger:** | The user initiates the process of leaving a comment on a post within the app. |

| | |
|---|---|
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and logged into their FarmConnect account.<br>4. The user is viewing a post within the app. |
| **Postconditions:** | The user has successfully left a comment on the post, and their comment is recorded. |
| **Normal Flow:** | 1. User is viewing a post within the FarmConnect mobile app.<br>2. The user selects the "Comment" or "Add a Comment" option associated with the post.<br>3. The system provides a text input field for the user to enter their comment.<br>4. User enters their comment and selects the "Post" or "Send" option.<br>5. The system registers the user's comment and displays it below the post. |
| **Alternative Flows:** | None |
| **Exceptions:** | None. |
| **Includes:** | None |
| **Special Requirements:** | 1. The system should provide a user-friendly text input field for entering comments.<br>2. Comments should be displayed in an organized and user-friendly manner below the post. |
| **Assumptions:** | 1. Users are familiar with the concept of commenting on posts in social media platforms.<br>2. Posts are regularly updated and available for commenting within the mobile app. |
| **Notes and Issues:** | None |

*Table 3.3.3.9: UC-2.2.5 Comment on Posts*

| | |
|---|---|
| **Use Case ID:** | UC-2.2.6 |
| **Use Case Name:** | Share Post |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user sharing a post created by another user within the FarmConnect mobile app. |
| **Trigger:** | The user initiates the process of sharing a post within the app. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and logged into their FarmConnect account.<br>4. The user is viewing a post within the app. |
| **Postconditions:** | The user has successfully shared the post, and the shared post is visible on their profile or timeline. |
| **Normal Flow:** | 1. User is viewing a post within the FarmConnect mobile app.<br>2. The user selects the "Share" or "Repost" option associated with the post. |

| | |
|---|---|
| | 3. The system provides options for the user to add a comment or message with the shared post.<br>4. User enters their comment or message (optional) and selects the "Share" or "Post" option.<br>5. The system shares the post to the user's profile or timeline, making it visible to their connections or followers. |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Includes:** | None |
| **Special Requirements:** | 1. The system should provide options for users to add comments or messages when sharing a post.<br>2. Shared posts should be visible on the user's profile or timeline. |
| **Assumptions:** | 1. Users are familiar with the concept of sharing posts in social media platforms.<br>2. Shared posts are regularly updated and visible on the user's profile. |
| **Notes and Issues:** | None. |

*Table 3.3.3.10: UC-2.2.6 Share Posts*

| | |
|---|---|
| **Use Case ID:** | UC-2.2.7 |
| **Use Case Name:** | Report Post |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user reporting a post within the FarmConnect mobile app. Reporting a post allows users to bring attention to content that violates community guidelines or is otherwise inappropriate. |
| **Trigger:** | The user initiates the process of reporting a post within the app. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and logged into their FarmConnect account.<br>4. The user is viewing a post within the app that they wish to report. |
| **Postconditions:** | 1. The user has successfully reported the post with a specific reason for the report.<br>2. The reported post is flagged for review by the FarmConnect admin. |
| **Normal Flow:** | 1. User is viewing a post within the FarmConnect mobile app.<br>2. The user selects the “Report” or “Flag” option associated with the post.<br>3. The system presents a dialog box or form for the user to provide a reason for the report.<br>4. User enters a text-based reason for reporting the post.<br>5. The user submits the report.<br>6. The system records the report, including the reported post’s details and the user’s reason for reporting it. |

| | |
|---|---|
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Includes:** | None. |
| **Special Requirements:** | 1. The system should provide a user-friendly text input field for the user to enter their reason for reporting the post.<br>2. Reported posts should be flagged for review by FarmConnect admin and should not be visible to other users until the review is completed. |
| **Assumptions:** | 1. Users are familiar with the concept of reporting posts in social media platforms.<br>2. Reported posts are promptly reviewed by the FarmConnect admin for appropriate action. |
| **Notes and Issues:** | None |

*Table 3.3.3.11: UC-2.2.7 Report Posts*

| | |
|---|---|
| **Use Case ID:** | UC-2.2.8 |
| **Use Case Name:** | Manage Privacy |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user setting privacy settings for a post they are creating on the FarmConnect platform. |
| **Trigger:** | The user initiates the process of creating a new post and configuring its privacy settings. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and has a valid account on FarmConnect.<br>4. User is logged into their FarmConnect account.<br>5. The user is in the process of creating a post (part of the "Create a Post with Image/Video Upload" use case). |
| **Postconditions:** | The user has configured the privacy settings for their post, determining who can view it. |
| **Normal Flow:** | 1. User is in the process of creating a new post (part of the "Create a Post with Image/Video Upload" use case).<br>2. User selects the "Privacy Settings" option.<br>3. The system displays a list of privacy settings options, which may include:<br>• Public: Visible to all FarmConnect users.<br>• Friends: Visible to the user's connections or friends on FarmConnect.<br>• Private: Visible to the user only.<br>4. User selects one of the privacy settings options.<br>5. The system saves the selected privacy setting for the post. |
| **Alternative Flows:** | **[Alternative Flow 1 - Cancel Privacy Settings (Step 4)]:** |

| | |
|---|---|
| | In step 4 of the normal flow, the user decides to cancel configuring privacy settings.<br>The system returns to the post creation form without saving any changes to privacy settings. |
| **Exceptions:** | **Privacy Setting Selection Error (Step 4):**<br>1. In step 4 of the normal flow, if there is an error in selecting a privacy setting.<br>2. The system displays an error message indicating the selection failure.<br>3. User is prompted to reattempt the selection. |
| **Includes:** | None. |
| **Special Requirements:** | 1. Clear and concise descriptions should be provided for each privacy setting option to help users understand their choices.<br>2. Privacy settings should be consistently applied to posts to ensure privacy preferences are respected. |
| **Assumptions:** | 1. Users are familiar with the concept of privacy settings in online social platforms.<br>2. Users have the necessary permissions to configure privacy settings for their own posts. |
| **Notes and Issues:** | User experience design for selecting and configuring privacy settings should be user-friendly and straightforward.<br>An option to change privacy settings for a post after it is created should be considered in the system's design. |

*Table 3.3.3.12: UC-2.2.8 Manage Privacy*

| | |
|---|---|
| **Use Case ID:** | UC-3.1.1 |
| **Use Case Name:** | Chatting |
| **Actors:** | Primary Actor: User<br>Secondary Actor: Chatbot |
| **Description:** | This use case describes the process of a registered user engaging in chat conversations with other users and interacting with a chatbot on the FarmConnect platform. |
| **Trigger:** | The user initiates the process of starting a chat conversation with another user or interacting with the chatbot. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and has a valid account on FarmConnect.<br>4. User is logged into their FarmConnect account.<br>5. The user is within the chat section of the FarmConnect app. |
| **Postconditions:** | 1. The user has engaged in a chat conversation with another user or the chatbot.<br>2. The chat conversation is saved in the user's chat history for reference. |
| **Normal Flow:** | 1. User opens the FarmConnect mobile app.<br>2. User navigates to the chat section. |

| | |
|---|---|
| | 3. The system displays the list of available chat conversations and contacts.<br>4. User selects an existing chat conversation or starts a new chat with another user.<br>5. The system opens the selected chat conversation.<br>6. User can type and send messages to the other user.<br>7. The system delivers the messages to the recipient.<br>8. User can also initiate a chat with the chatbot, if available, by selecting the chatbot contact.<br>9. The chatbot responds to the user's messages based on its programmed functionality. |
| **Alternative Flows:** | **Alternative Flow 1 – End Chat (Step 4):**<br>1. In step 4 of the normal flow, the user decides to end a chat conversation.<br>2. The system closes the chat conversation and returns to the chat list. |
| **Exceptions:** | **Message Delivery Error (Step 7):**<br>1. In step 7 of the normal flow, if there is an error in delivering messages to the recipient.<br>2. The system displays an error message indicating the delivery failure.<br>3. User may reattempt message delivery or troubleshoot the issue. |
| **Includes:** | Send Message. |
| **Special Requirements:** | 1. The chatbot's functionality should be clearly defined and programmed to provide meaningful interactions with users.<br>2. Users should have the ability to initiate and manage multiple chat conversations. |
| **Assumptions:** | 1. Users are familiar with chat-based communication in social media platforms.<br>2. Users have the necessary permissions to initiate chat conversations and interact with the chatbot. |
| **Notes and Issues:** | 1. User experience design for chat conversations should provide an intuitive and user-friendly interface.<br>2. Privacy and security considerations for chat communications should be addressed.<br>3. The chatbot's behavior, capabilities, and limitations should be communicated to users. |

*Table 3.3.3.13: UC-3.1.1 Chatting*

| | |
|---|---|
| **Use Case ID:** | UC-4.2 |
| **Use Case Name:** | Explore Marketplace |
| **Actors:** | Primary Actor: User<br>Secondary Actor: Seller |
| **Description:** | This use case describes the process of a registered user browsing, uploading product details, and interacting with sellers to discuss |

| | |
|---|---|
| | product details and delivery within the FarmConnect Marketplace. It excludes actual transactions. |
| **Trigger:** | The user initiates the process of accessing the FarmConnect Marketplace to browse products, list items for sale, and interact with sellers regarding product details and delivery. |
| **Preconditions:** | 1. User has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered and has a valid account on FarmConnect.<br>4. User is logged into their FarmConnect account.<br>5. The user is within the Marketplace section of the FarmConnect app. |
| **Postconditions:** | The user has viewed product details, uploaded their product details for sale, and engaged in discussions with sellers regarding products and delivery. |
| **Normal Flow:** | 1. User opens the FarmConnect mobile app.<br>2. User navigates to the Marketplace section.<br>3. The system displays a list of available categories or product listings.<br>4. User selects a category or performs a search for specific products.<br>5. The system displays relevant product listings.<br>6. User selects a product listing to view more details.<br>7. User can view product details, including images, description, and price.<br>8. User can start a chat conversation with the seller.<br>9. Seller responds to the user's inquiry about product details and delivery.<br>10. User can also select an option to list their own items for sale.<br>11. If the user selects "Sell," the system guides the user through the process of creating a product listing, which may include adding product details, images, pricing, and description.<br>12. The system records the user's product listing for other users to view. |
| **Alternative Flows:** | **Alternative Flow 1 - Cancel Listing (Step 11):**<br>1. In step 11 of the normal flow, the user decides to cancel the item listing.<br>2. The system discards the listing and returns to the product browsing interface. |
| **Exceptions:** | **Chat Error (Step 8):**<br>1. In step 8 of the normal flow, if there is an error in starting a chat conversation with the seller.<br>2. The system displays an error message indicating the chat initiation failure.<br>3. User may reattempt to start a chat. |
| **Includes:** | **UC-1.2.7 (View Product Details):** This sub-use case allows the user to view detailed information about a product listing, including images, |

| | |
|---|---|
| | description, and price. This should be included when the user selects a product listing to view more details.<br>**UC-1.2.8 (Upload Product Details):** This sub-use case enables the user to list their own items for sale, providing details such as product name, description, images, pricing, and more. This should be included when the user selects the "Sell" option.<br>**UC-1.2.9 (Chat):** This sub-use case covers the user's ability to initiate a chat conversation with a seller to discuss product details, negotiate, and inquire about delivery. This should be included when the user starts a chat with a seller. |
| **Special Requirements:** | 1. Sellers should have the ability to manage and edit their product listings.<br>2. A messaging feature should be implemented for users to interact with sellers.<br>3. Product categories and filters should be provided to enhance the browsing experience. |
| **Assumptions:** | 1. Users are familiar with online marketplace interactions.<br>2. Users have the necessary permissions to list items for sale, view product details, and initiate chat conversations with sellers. |
| **Notes and Issues:** | 1. User experience design for product browsing, chat interaction, and listing creation should be user-friendly and intuitive.<br>2. The focus is on interactions and discussions, not actual financial transactions. |

*Table 3.3.3.14: UC-4.1.1 Marketplace*

| | |
|---|---|
| **Use Case ID:** | UC-4.2.1 |
| **Use Case Name:** | Advertise Product |
| **Actors:** | Primary Actor: User (Seller) |
| **Description:** | This use case describes the process of a registered user (seller) advertising a product for sale within the FarmConnect Marketplace. |
| **Trigger:** | The user (seller) initiates the process of listing and advertising their product for sale in the Marketplace. |
| **Preconditions:** | 1. User (seller) has downloaded and installed the FarmConnect mobile app.<br>2. The user has an active internet connection.<br>3. User is registered as a seller on FarmConnect.<br>4. User is logged into their FarmConnect account.<br>5. The user is within the Marketplace section of the FarmConnect app. |
| **Postconditions:** | The user's product is listed for sale in the Marketplace, and it is available for other users to browse and inquire about. |
| **Normal Flow:** | 1. User (seller) opens the FarmConnect mobile app.<br>2. User navigates to the Marketplace section.<br>3. The system displays an option to "Advertise Product" or similar.<br>4. User selects the "Advertise Product" option. |

| | |
|---|---|
| | 5. The system guides the user through the process of creating a product listing, which may include adding product details, images, pricing, and description.<br>6. User completes the product listing and selects the "List for Sale" option.<br>7. The system records the user's product listing, making it available for other users to browse. |
| **Alternative Flows:** | **Alternative Flow 1 - Cancel Listing (Step 6):**<br>1. In step 6 of the normal flow, the user decides to cancel the item listing.<br>2. The system discards the listing and returns to the Marketplace section. |
| **Exceptions:** | **Listing Error (Step 6):**<br>1. In step 6 of the normal flow, if there is an error in creating the product listing.<br>2. The system displays an error message indicating the listing creation failure.<br>3. User may reattempt the listing or troubleshoot the issue |
| **Includes:** | Explore Marketplace |
| **Special Requirements:** | 1. The system should provide user-friendly options for creating and editing product listings.<br>2. Product images, descriptions, and pricing should be supported to create attractive listings.<br>3. Users (sellers) should have the ability to manage and edit their product listings after advertising them |
| **Assumptions:** | 1. Users (sellers) are familiar with listing products for sale on online marketplaces.<br>2. Users (sellers) have the necessary permissions to advertise and manage their product listings. |
| **Notes and Issues:** | 1. User experience design for creating product listings should be intuitive and user-friendly.<br>2. Support for editing and updating existing product listings may be required.<br>3. The system should handle the storage and display of product images and information accurately and efficiently. |

*Table 3.3.3.15: UC-4.2.1 Advertising*

| | |
|---|---|
| **Use Case ID:** | UC-4.2.2 |
| **Use Case Name:** | Buy Product |
| **Actors:** | Primary Actor: User<br>Secondary Actor: Seller |
| **Description:** | This use case describes the process of a user purchasing a product from a seller through the FarmConnect app. The app facilitates collaboration between the user and the seller, including discussions about shipment and payment within the chat feature. |

| | |
|---|---|
| **Trigger:** | The user initiates the process of purchasing a product within the app. |
| **Preconditions:** | 1. The user is registered and logged into their FarmConnect account.<br>2. The user is viewing a product listing in the app.<br>3. The seller has listed the product for sale within the app. |
| **Postconditions:** | The user and seller have agreed on the terms of the purchase, including shipment and payment details, within the app's chat feature. |
| **Normal Flow:** | 1. The user views a product listing within the FarmConnect app.<br>2. The user initiates a chat conversation with the seller through the app.<br>3. The user and seller collaborate in the chat to discuss and agree on the product's price, payment method, and shipment details.<br>4. The user and seller finalize the purchase agreement within the chat. |
| **Alternative Flows:** | None in this scenario. |
| **Exceptions:** | None in this scenario. |
| **Includes:** | Explore Marketplace |
| **Special Requirements:** | 1. The FarmConnect app should provide a user-friendly chat feature for users to easily communicate with sellers.<br>2. Users and sellers should have the flexibility to discuss and agree on payment and shipment details within the chat. |
| **Assumptions:** | 1. Users and sellers are responsible for ensuring the accuracy and security of the payment and shipment process.<br>2. Users are familiar with the standard process of purchasing products and handling payments and shipments. |
| **Notes and Issues:** | None |

*Table 3.3.3.16: UC-4.2.2 Buy Product*

| | |
|---|---|
| **Use Case ID:** | UC-5.1.1 |
| **Use Case Name:** | View Stories (Web Dashboard Only) |
| **Actors:** | Primary Actor: User |
| **Description:** | This use case describes the process of a registered user viewing stories created by social users within the FarmConnect web dashboard. Stories related to agriculture are shared by other users within the FarmConnect platform, and they are only accessible through the web dashboard. |
| **Trigger:** | The user initiates the process of viewing stories by navigating to the Stories section within the FarmConnect web dashboard. |
| **Preconditions:** | 1. User has access to the FarmConnect web dashboard.<br>2. The user has an active internet connection.<br>3. User is registered and logged into their FarmConnect account.<br>**4.** The user is within the Stories section of the FarmConnect web dashboard. |
| **Postconditions:** | The user has successfully viewed stories created by other social users within the FarmConnect web dashboard. |

| | |
|---|---|
| **Normal Flow:** | 1. User accesses the FarmConnect web dashboard.<br>2. User navigates to the Stories section within the web dashboard.<br>3. The system displays a list of available stories, which may include images, videos, and text.<br>4. User can select a specific story to view.<br>5. The system displays the selected story in a user-friendly format.<br>6. User can view the story content, including text, images, and videos. |
| **Alternative Flows:** | **Alternative Flow 1 - No Available Stories (Step 3):**<br>1. In step 3 of the normal flow, if there are no available stories to view.<br>2. The system displays a message indicating that there are no stories currently available. |
| **Exceptions:** | **Loading Error (Step 3):**<br>1. In step 3 of the normal flow, if there is an error in loading available stories.<br>2. The system displays an error message indicating the loading failure.<br>3. User may attempt to reload or troubleshoot the issue. |
| **Includes:** | None. |
| **Special Requirements:** | 1. Stories should be presented in an engaging and user-friendly format within the web dashboard.<br>2. Users should be able to navigate to the Stories section easily within the web dashboard. |
| **Assumptions:** | 1. Users are familiar with the concept of viewing and interacting with stories in a web dashboard environment.<br>2. Stories are updated and available for viewing regularly on the web dashboard. |
| **Notes and Issues:** | 1. User experience for viewing stories within the web dashboard should be designed to enhance engagement and user satisfaction.<br>2. Consider providing a clear message to mobile app users that stories can only be viewed on the web dashboard for transparency. |

*Table 3.3.3.17: UC-5.1.1 View Stories*

| | |
|---|---|
| **Use Case ID:** | UC-5.1.2 |
| **Use Case Name:** | Respond to Story (Web Dashboard) |
| **Actors:** | Primary Actor: Visitor (unregistered user) |
| **Description:** | This use case describes the process of a visitor, who is not registered, responding to a story displayed on the FarmConnect web dashboard. Visitors can view and engage with stories shared by registered users. |
| **Trigger:** | The visitor initiates the process of responding to a story on the FarmConnect web dashboard. |
| **Preconditions:** | 1. The visitor has access to the FarmConnect web dashboard.<br>2. The visitor is not a registered user and does not have an active FarmConnect account.<br>3. Stories are available and visible on the web dashboard. |

| | |
|---|---|
| **Postconditions:** | The visitor has successfully responded to the story, and their response is recorded as a visitor comment. |
| **Normal Flow:** | 1. The visitor accesses the FarmConnect web dashboard.<br>2. The visitor navigates to a story they wish to respond to.<br>3. The visitor selects the “Comment” or “Respond” option associated with the story.<br>4. The system provides a text input field for the visitor to enter their comment.<br>5. The visitor enters their comment and selects the “Post” or “Send” option.<br>6. The system records the visitor’s comment as a response to the story. |
| **Alternative Flows:** | None. |
| **Exceptions:** | None. |
| **Includes:** | None. |
| **Special Requirements:** | 1. The system should provide a user-friendly text input field for entering visitor comments.<br>2. Visitor comments should be displayed below the story and attributed to the visitor's unique identifier (e.g., IP address or session token). |
| **Assumptions:** | 1. Visitors are familiar with the concept of leaving comments on web-based content.<br>2. Stories are regularly updated and visible to visitors on the FarmConnect web dashboard. |
| **Notes and Issues:** | None |

*Table 3.3.3.18: UC-5.1.2 Respond Stories*

| | |
|---|---|
| **Use Case ID:** | UC-5.1.3 |
| **Use Case Name:** | Manage Platform |
| **Actors:** | Primary Actor: Admin |
| **Description:** | This use case outlines the actions an admin can perform to manage the FarmConnect platform and its user base efficiently. |
| **Trigger:** | The admin logs into the FarmConnect admin dashboard and initiates management tasks. |
| **Preconditions:** | 1. The admin has access to the FarmConnect admin dashboard.<br>2. The admin is logged into the admin account. |
| **Postconditions:** | 1. The FarmConnect platform is well-managed and adheres to community guidelines.<br>2. User accounts and content are monitored and moderated effectively. |
| **Normal Flow:** | 1. The admin logs into the FarmConnect admin dashboard.<br>2. The admin is presented with a dashboard showing relevant metrics, reports, and moderation tools. |

| | |
|---|---|
| | 3. The admin can view and analyze user statistics, including new registrations, active users, and flagged content.<br>4. The admin can access and review reported content, including posts, comments, and user profiles.<br>5. The admin can take actions such as warning, suspending, or banning users based on the severity of violations.<br>6. The admin can communicate with users, providing warnings or guidance when needed.<br>7. The admin can update or modify the FarmConnect platform settings and configurations.<br>8. The admin can perform regular backups and maintenance tasks to ensure the platform's stability. |
| **Alternative Flows:** | None. |
| **Exceptions:** | If the admin encounters technical issues with the admin dashboard, they report the issue to the technical support team. |
| **Includes:** | None. |
| **Special Requirements:** | 1. The admin dashboard should provide a user-friendly interface for effective monitoring and management.<br>2. Moderation tools should include options for warning, suspending, or banning users, as well as hiding or removing content.<br>3. The admin should have access to analytics and reports to make informed decisions about platform management. |
| **Assumptions:** | 1. The admin is familiar with the community guidelines and standards for content moderation.<br>2. The admin has the necessary permissions to access and modify platform settings. |
| **Notes and Issues:** | The admin's actions should be logged for auditing and accountability purposes. |

*Table 3.3.3.19: UC-5.1.3 Manage Platform*

FarmConnect offers a variety of use cases tailored to enhance the connectivity and productivity of farmers. Personalized profiles allow farmers to showcase their expertise and achievements, fostering professional networking and community building. The app's newsfeed provides real-time updates on agricultural trends and market information, helping farmers make informed decisions. Group communities enable farmers to engage in discussions, share experiences, and collaborate on common challenges. The integrated marketplace facilitates the buying and selling of products, tools, and services, streamlining transactions and expanding market reach. Advanced search capabilities allow users to quickly find relevant information and resources. Additionally, FarmConnect connects farmers with agricultural experts for personalized advice.

## 3.4. System Sequence Diagram:

A system sequence diagram for FarmConnect illustrates the interactions between the user and the system during key processes. It begins with the user logging into the app, where the system validates credentials and displays the personalized profile. Next, the user can access the newsfeed, prompting the system to retrieve and display the latest updates. When a user joins a group community, the system processes the request and displays relevant discussions. The marketplace interaction involves the user searching for products, with the system retrieving and displaying matching items. For expert consultations, the user requests advice, and the system connects them with an agricultural specialist. These interactions are visually mapped out to ensure seamless and efficient communication between the user and FarmConnect's system components.
Here is System Sequence Diagram for FarmConnect:

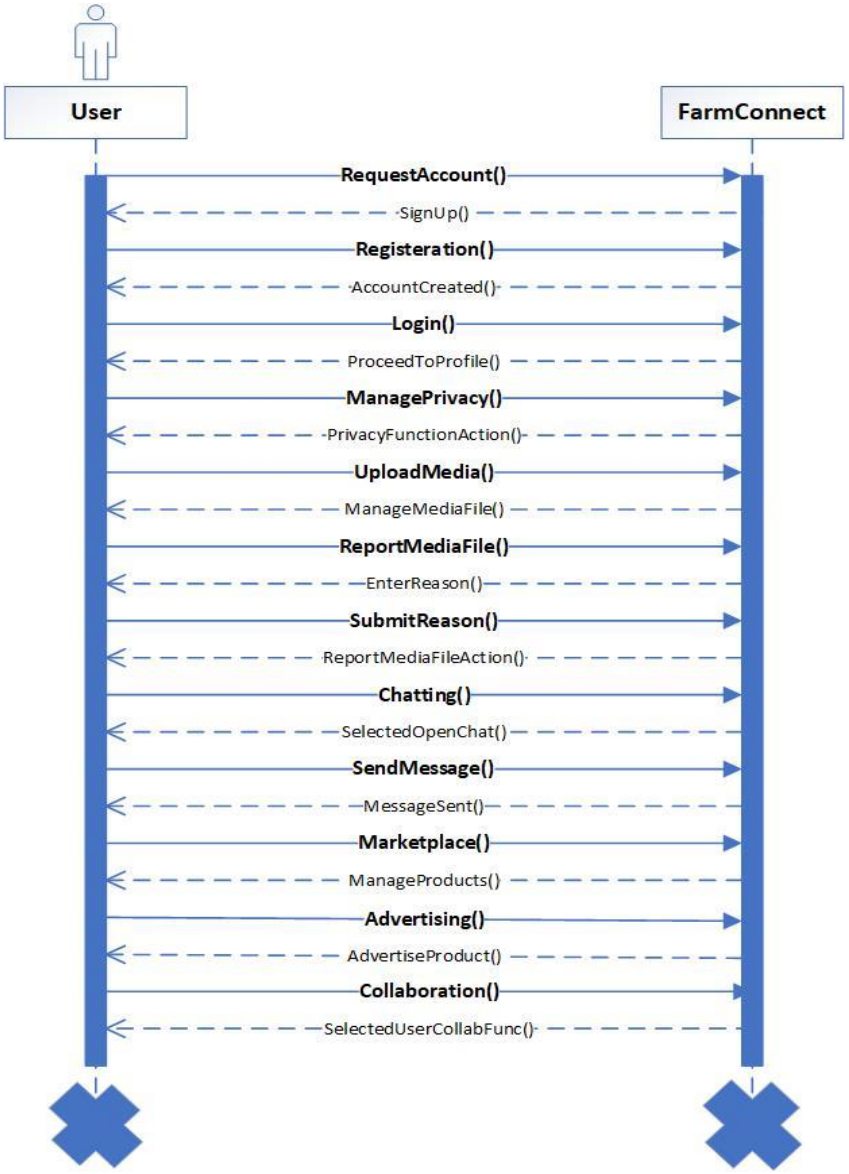


*Figure 3.4 System Sequence Diagram*

## 3.5. Data Flow Diagram:

A Data Flow Diagram (DFD) for FarmConnect outlines the flow of data between various system components and users. The primary external entity is the Farmer, who interacts with the app to perform different functions. The User Interface component collects data inputs, such as login credentials, profile information, and search queries. This data is processed by the Application Server, which handles authentication, data retrieval, and business logic.

A Data Flow Diagram (DFD) for FarmConnect shows how user inputs, like login details and profile updates, are processed by the Application Server. The server interacts with modules such as Newsfeed, Marketplace, and Community, retrieving and displaying data. All information is stored securely in the Database, ensuring seamless data flow and system functionality.

Data flows from the server to several modules, including the Newsfeed Module, which aggregates and presents updates, and the Marketplace Module, which manages product listings and transactions. The Community Module facilitates interactions within group discussions, while the Expert Consultation Module connects users with specialists. Additionally, the Database stores user profiles, market data, and transaction history, ensuring that information is accessible and secure.

A Data Flow Diagram (DFD) for FarmConnect depicts how data moves through the system. It begins with user inputs, such as login details and profile information, which are processed by the Application Server. The server interacts with various modules, including the Newsfeed, Marketplace, Community, and Expert Consultation modules, to retrieve and display relevant data. All user and transaction data are stored in the Database, ensuring secure and efficient data management. The DFD visually represents these interactions, illustrating how data flows from users through the system components to provide a cohesive and functional experience.

The diagram visually represents these data flows, highlighting how user inputs and system responses interact to provide a seamless and integrated experience.

Below are all levels of DFD's for FarmConnect:

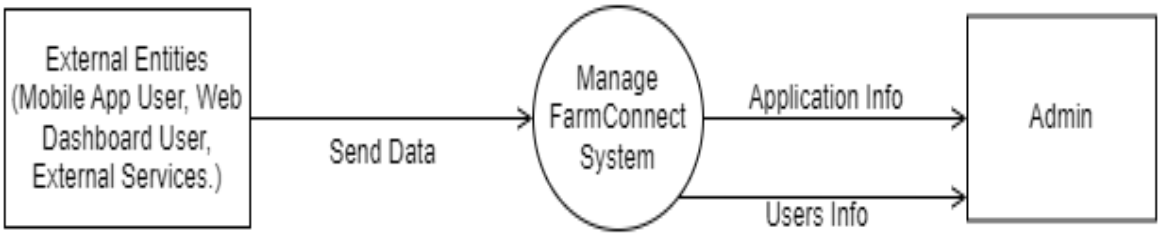


*Figure 3.5.1 DFD Level 0*

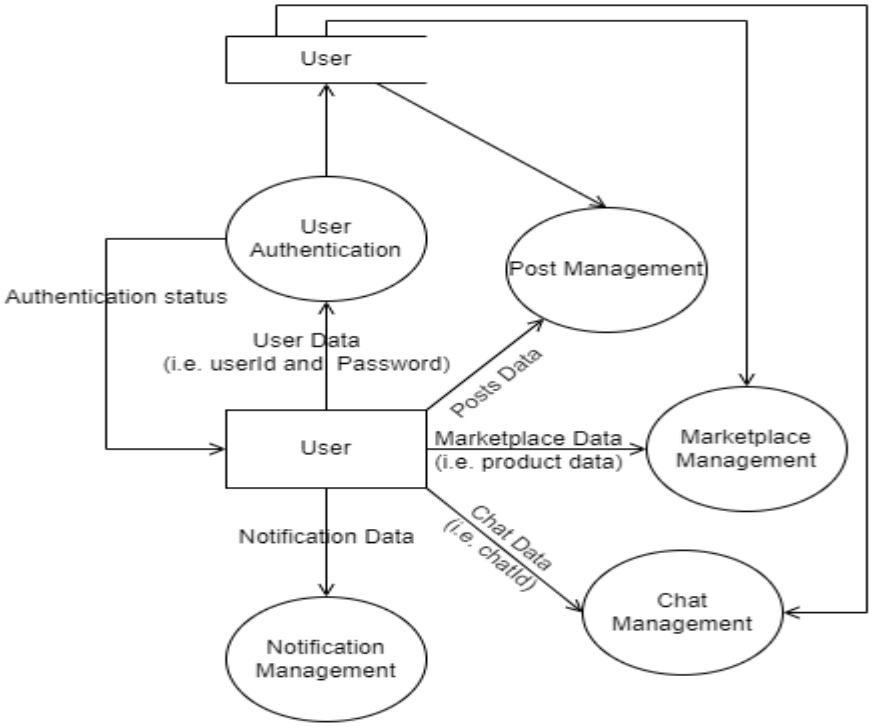


*Figure 3.5.2 DFD Level 1*

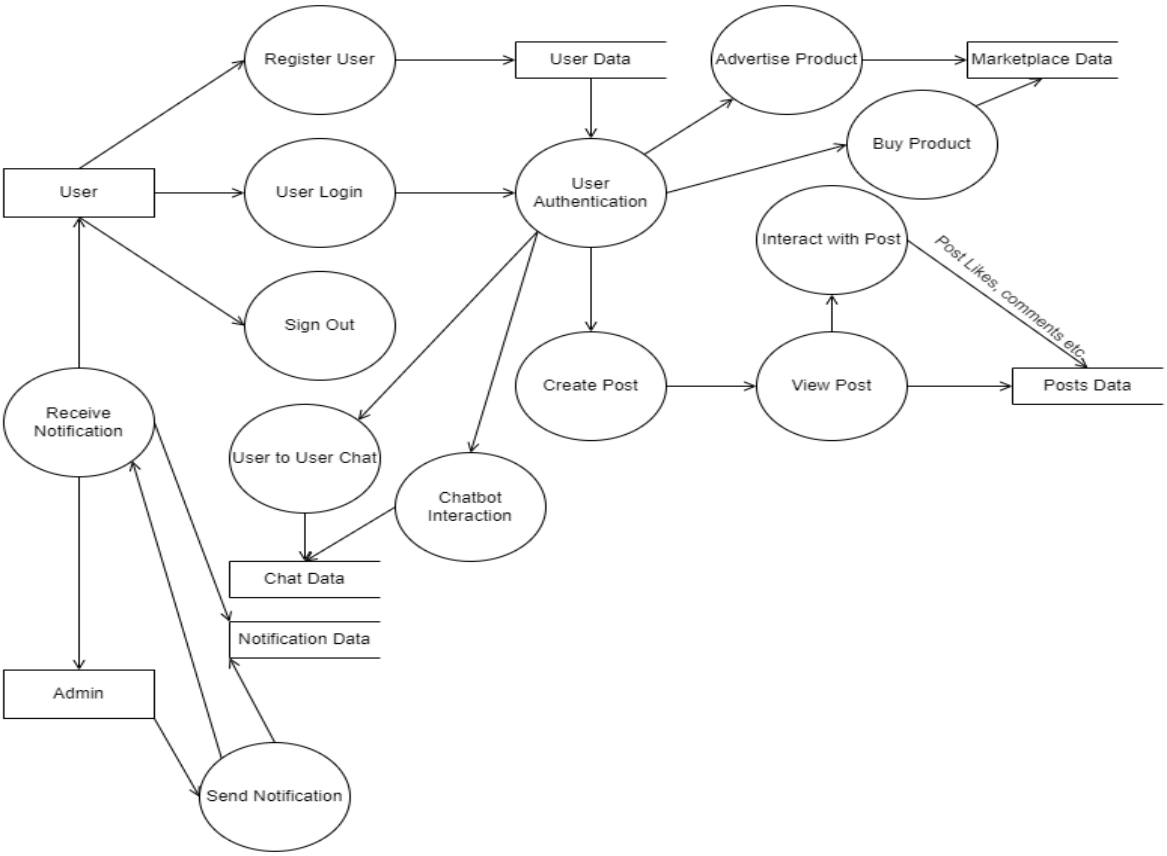


*Figure 3.5.3 DFD Level 2*

## 3.6. Block Diagram:

Here is block diagram for FarmConnect:

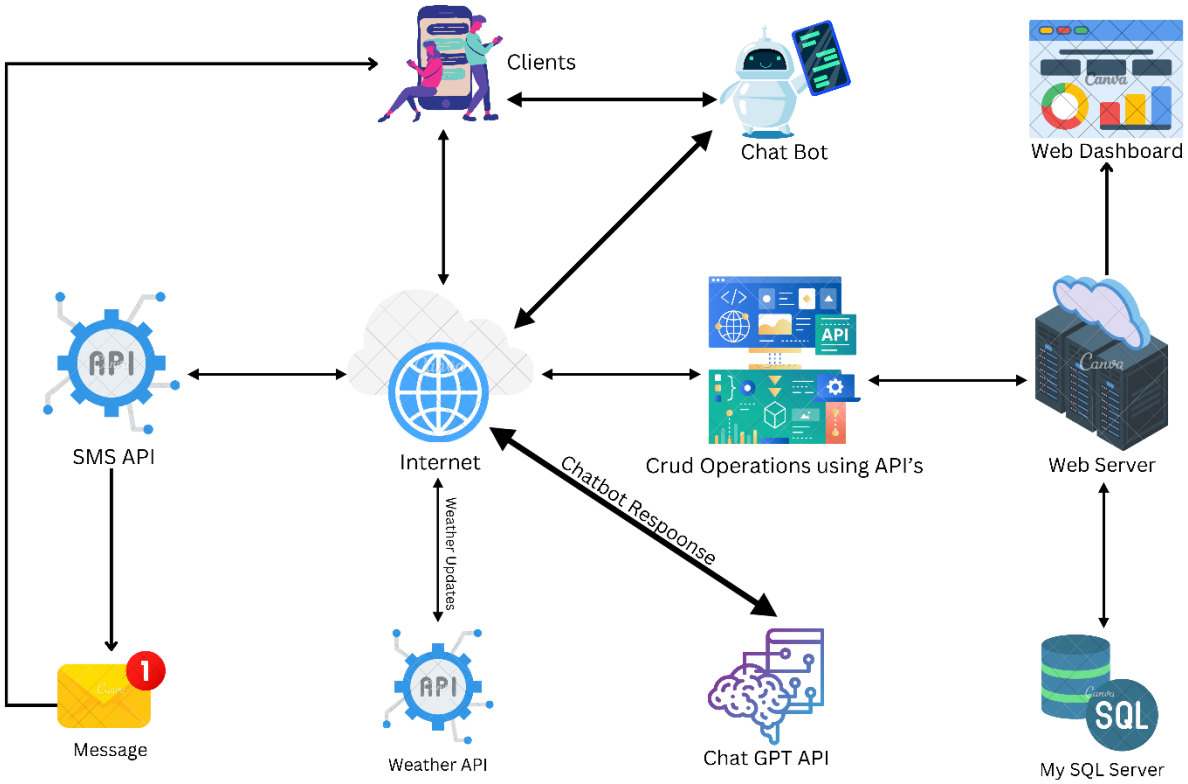


*Figure 3.6. Block Diagram*

A block diagram for FarmConnect provides a high-level overview of the system architecture and its main components. At the core is the Application Server, which handles business logic and user interactions. It connects to several key modules: the User Interface for user inputs and display, the Newsfeed Module for aggregating and presenting updates, the Marketplace Module for managing product listings and transactions, and the Community Module for facilitating group discussions and interactions.

The Expert Consultation Module connects users with agricultural specialists for personalized advice. All data, including user profiles, market information, and transaction history, is managed by the Database, ensuring secure storage and retrieval. The Service handles user login and security. External integrations, such as weather APIs or market data sources, provide real-time information to the system. The block diagram visually organizes these components, highlighting how they interact to deliver a cohesive and integrated experience for users.

# Chapter 4: Design and Architecture

This chapter will discuss the design and architecture of your system.

## 4.1. UML Structural Diagrams

Unified Modeling Language (UML) structural diagrams are graphical representations that showcase the static structure of a system. These diagrams provide insights into the components, relationships, and organization of the elements within a system. The main UML structural diagrams include:

### 4.1.1. ERD Diagram

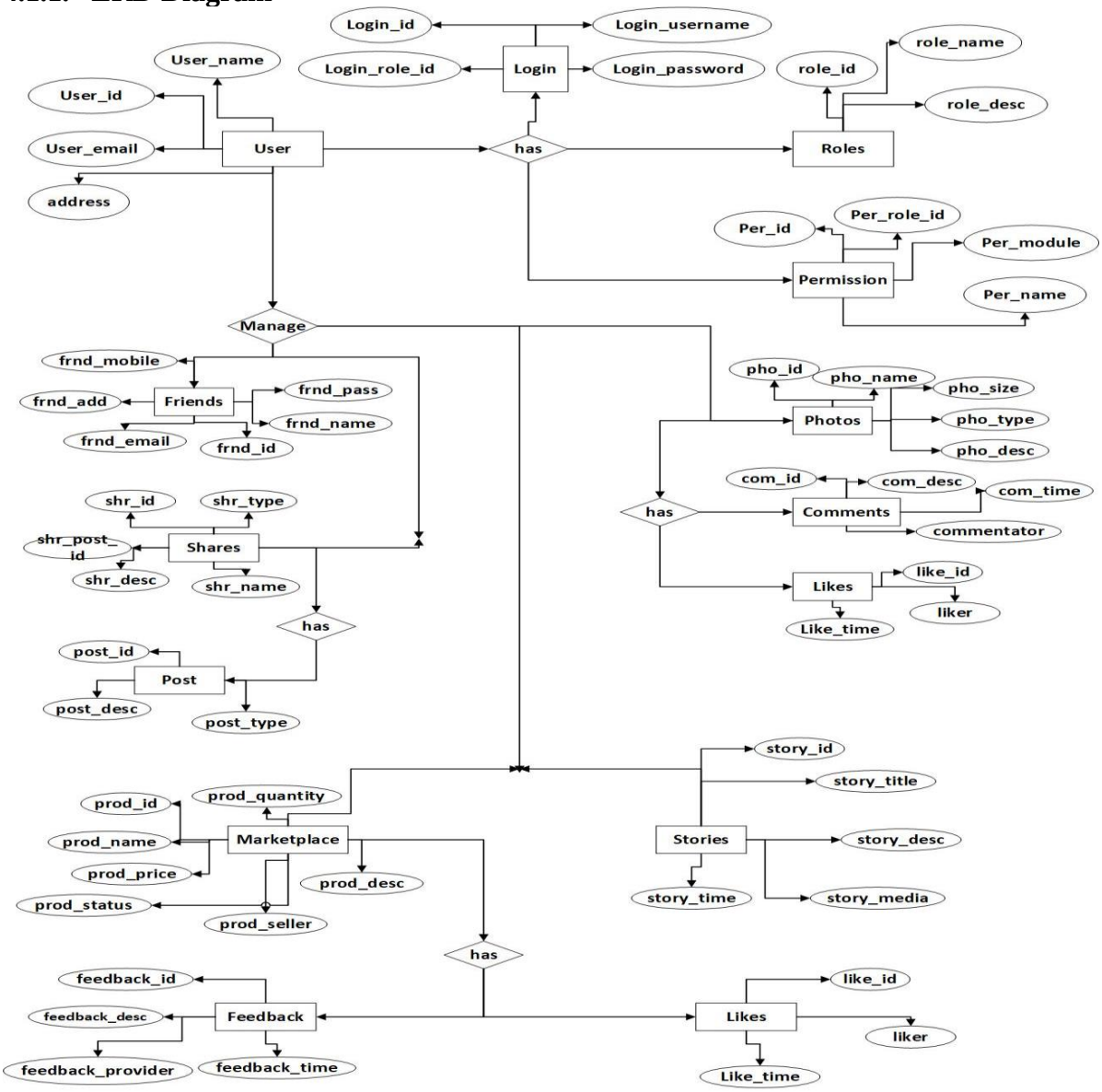


*Figure 4.1 ERD Diagram*

### 4.1.2. Class Diagram

Displayed below is the class diagram illustrating FarmConnect.

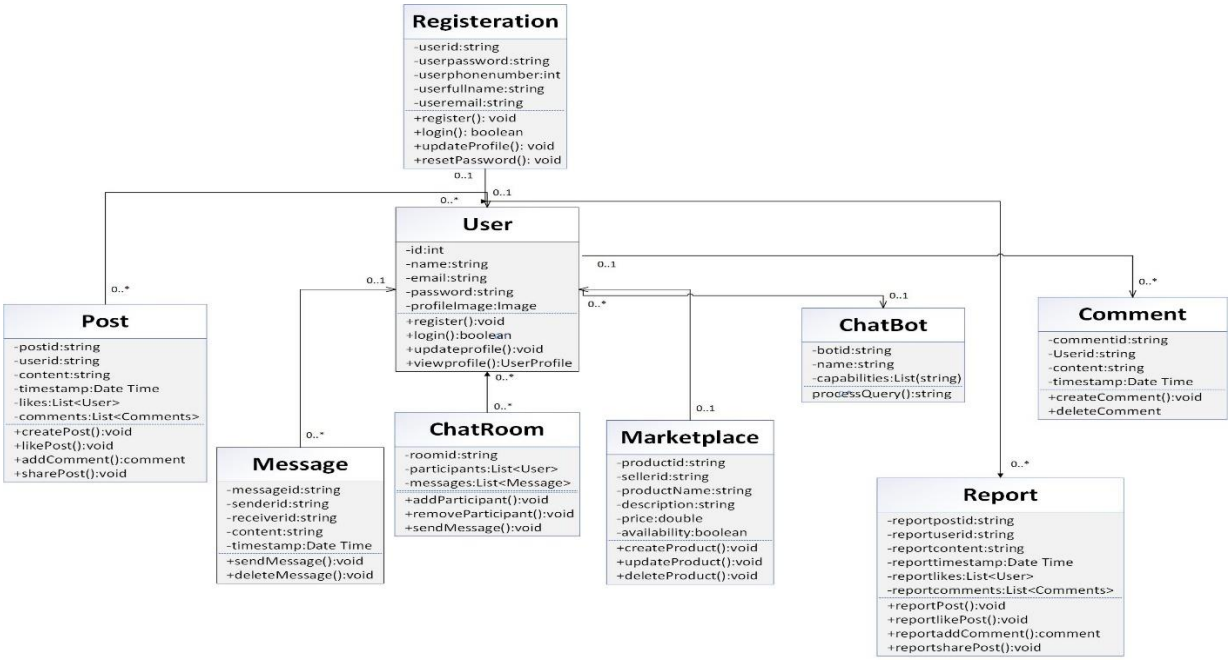


*Figure 4.1.2 Class Diagram*

### 4.1.3. Component Diagram

Displayed below is the component diagram illustrating FarmConnect.

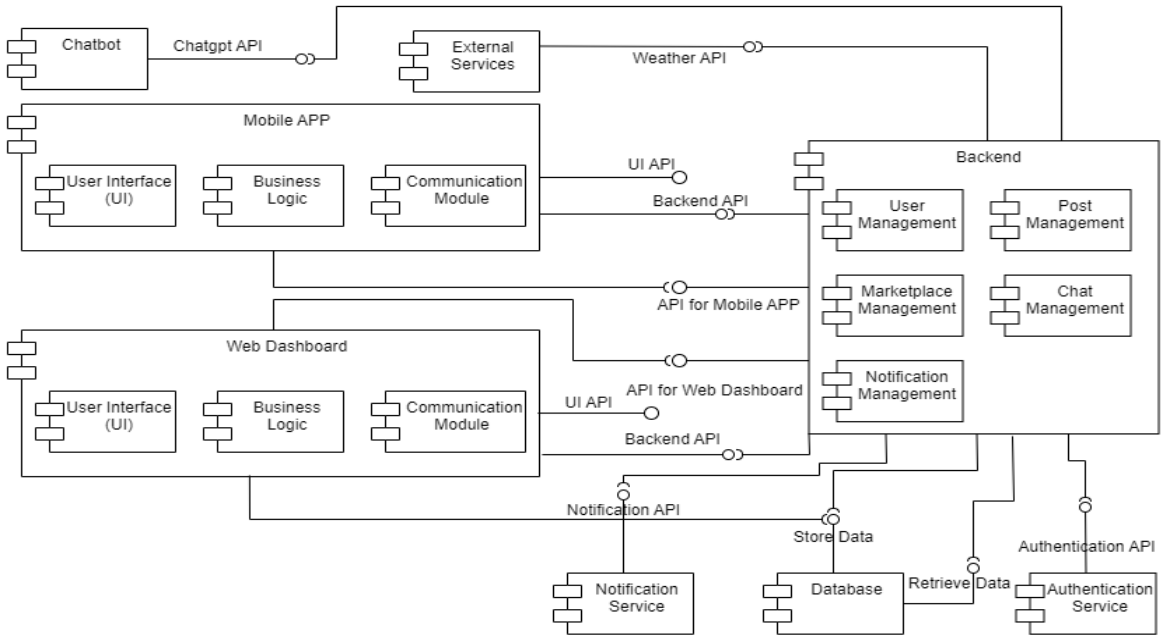


*Figure 4.1.3. Component Diagram*

### 4.1.4. System Component Diagram

Displayed below is the system component diagram illustrating FarmConnect.

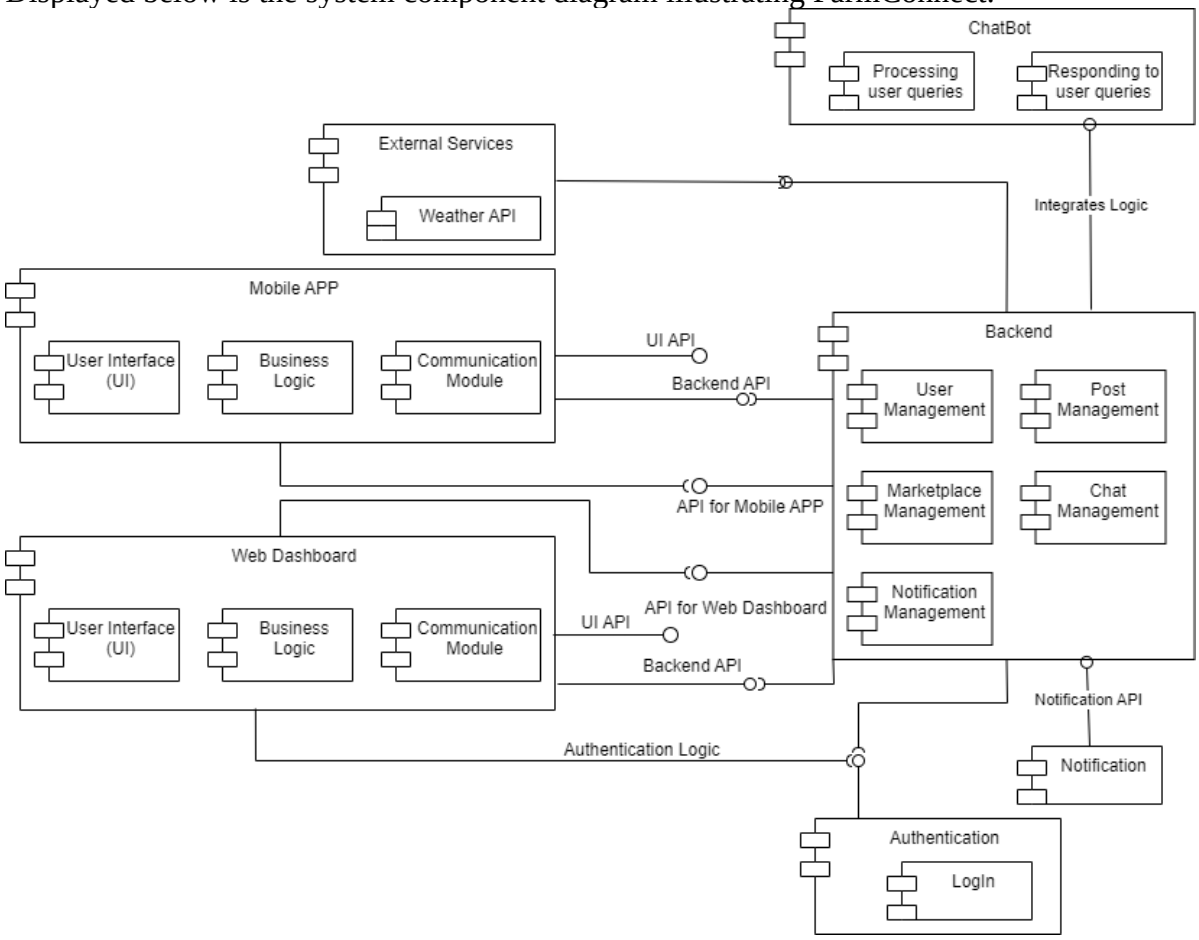


*Figure 4.1.4 System Component Diagram*

A system component diagram for FarmConnect outlines the key software components and their interactions within the application. At the center is the Application Server, which coordinates the core functions of the system. It interfaces with several major components: the User Interface (UI), responsible for user interactions and displaying information; the Newsfeed Component, which aggregates and delivers updates on agricultural trends; and the Marketplace Component, handling product listings and transactions.

The Community Component manages group discussions and user interactions, while the Expert Consultation Component connects users with agricultural specialists for personalized advice. Data is stored and managed by the Database Component, which includes user profiles, market data, and transaction histories. The Authentication Service ensures secure user login and access control. Additionally, external integrations like weather and market data APIs feed real-time information into the system. This diagram visually represents how these components work together to deliver a seamless and functional user experience in FarmConnect.

### 4.1.5. Package Diagram

Displayed below is the system package diagram illustrating FarmConnect. A package diagram for FarmConnect organizes the system into logical groupings of related functionalities, enhancing modularity and manageability. The diagram includes several key packages:

1. User Management: Handles user profiles, authentication, and authorization.
2. Newsfeed: Manages the aggregation and display of agricultural updates and trends.
3. Marketplace: Facilitates product listings, transactions, and marketplace interactions.
4. Community: Supports group discussions, user interactions, and community engagement.
5. Expert Consultation: Connects users with agricultural experts for personalized advice.
6. Data Management: Oversees database interactions, including storage and retrieval of user and transaction data.
7. Integration Services: Interfaces with external APIs for real-time weather and market information.

Each package encapsulates related functions and data, streamlining development and maintenance while promoting a modular architecture that enhances scalability and flexibility.

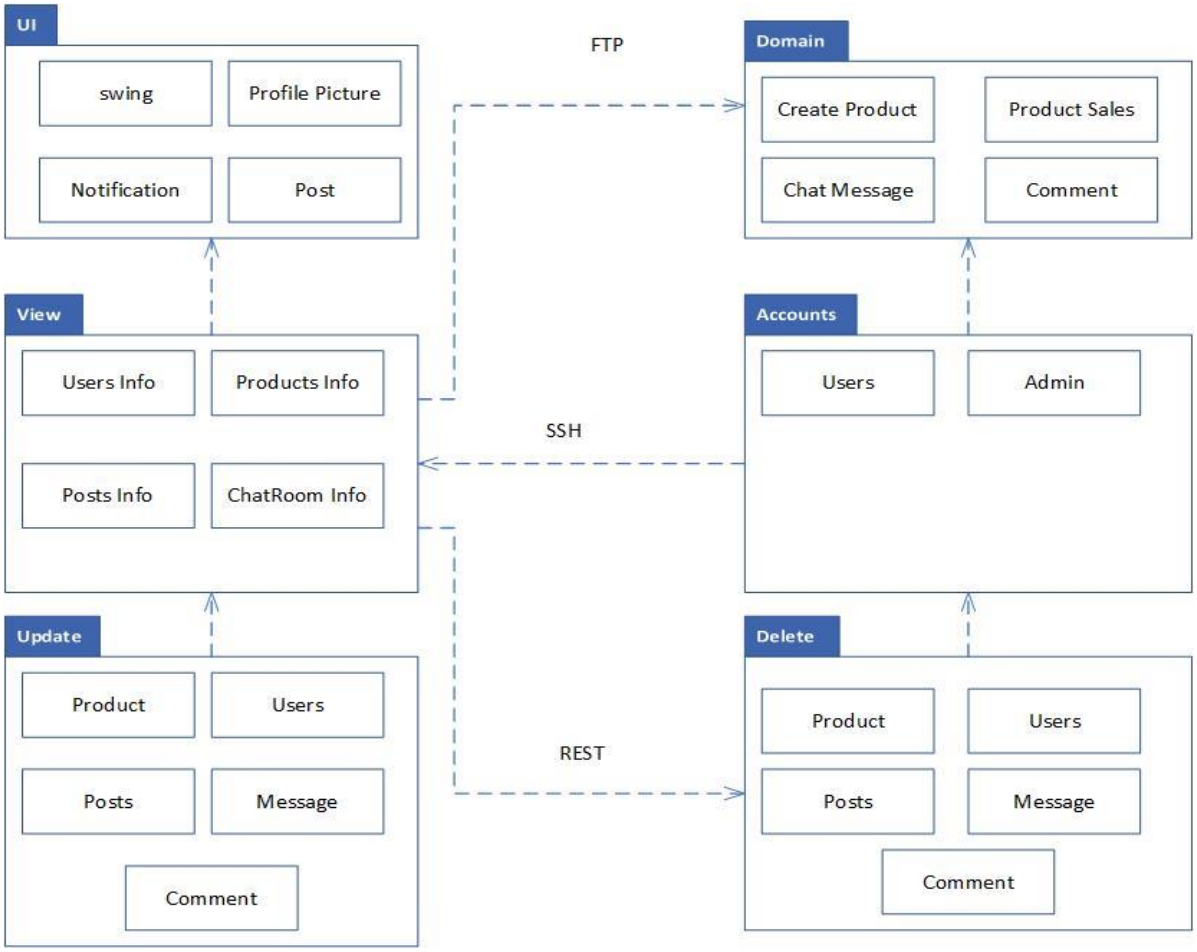


*Figure 4.1.5 Package Diagram*

### 4.1.6. Deployment Diagram

Displayed below is the system deployment diagram illustrating FarmConnect.

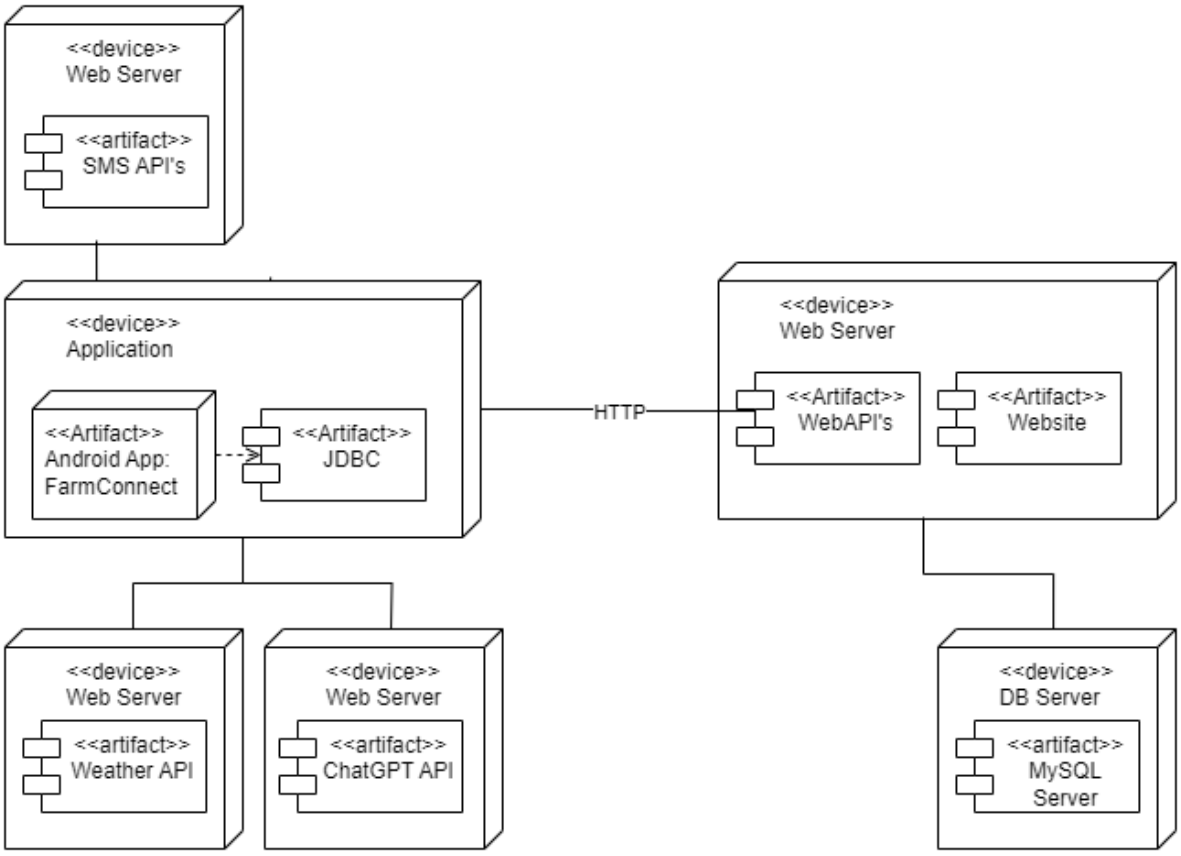


*Figure 4.1.6. Deployment Diagram*

## 4.2. UML Behavioral Diagrams

A behavioral diagram typically includes UML diagrams like Use Case Diagrams, Activity Diagrams, and State Chart Diagrams. A UML behavioral diagram for FarmConnect illustrates the dynamic interactions and processes within the system. It typically includes **Use Case Diagrams** that depict how users interact with the app's functionalities, such as accessing the newsfeed, participating in community discussions, or using the marketplace. **Sequence Diagrams** show the sequence of events for key processes, like user login or purchasing items, detailing the flow of messages between objects. **Activity Diagrams** outline the workflow for various activities, such as profile creation or expert consultations, highlighting decision points and concurrent processes. **State Diagrams** illustrate the states of different components, such as user status or marketplace listings, and transitions between these states. These diagrams collectively provide a comprehensive view of how FarmConnect's features are used and how the system responds to user actions and events.

These diagrams include:

### 4.2.1. Activity Diagrams

Displayed below are the activity diagrams illustrating FarmConnect.

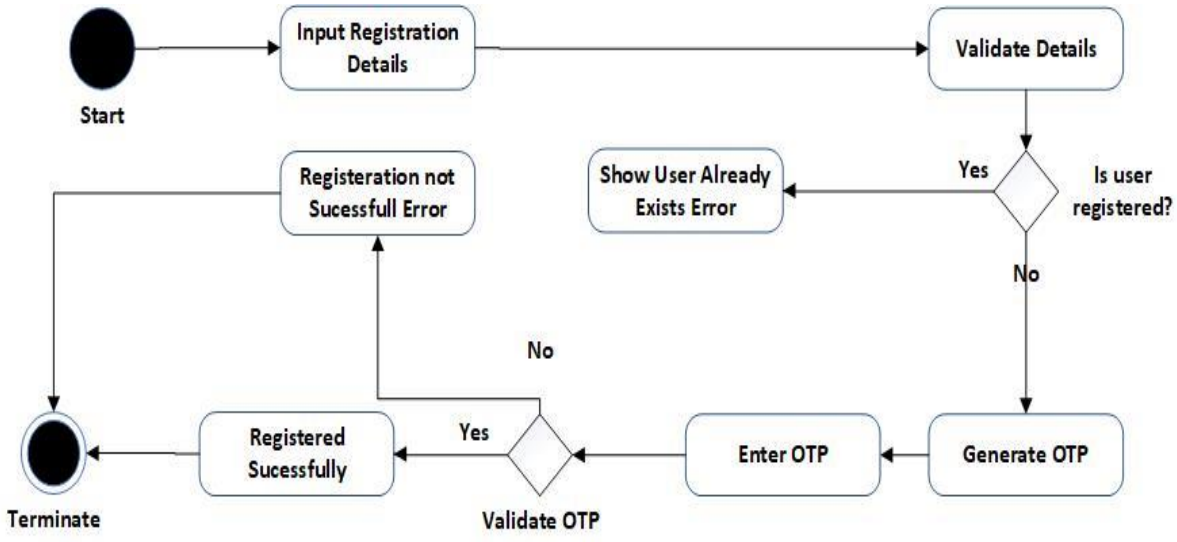


*Figure 4.2.1.1. Registration*

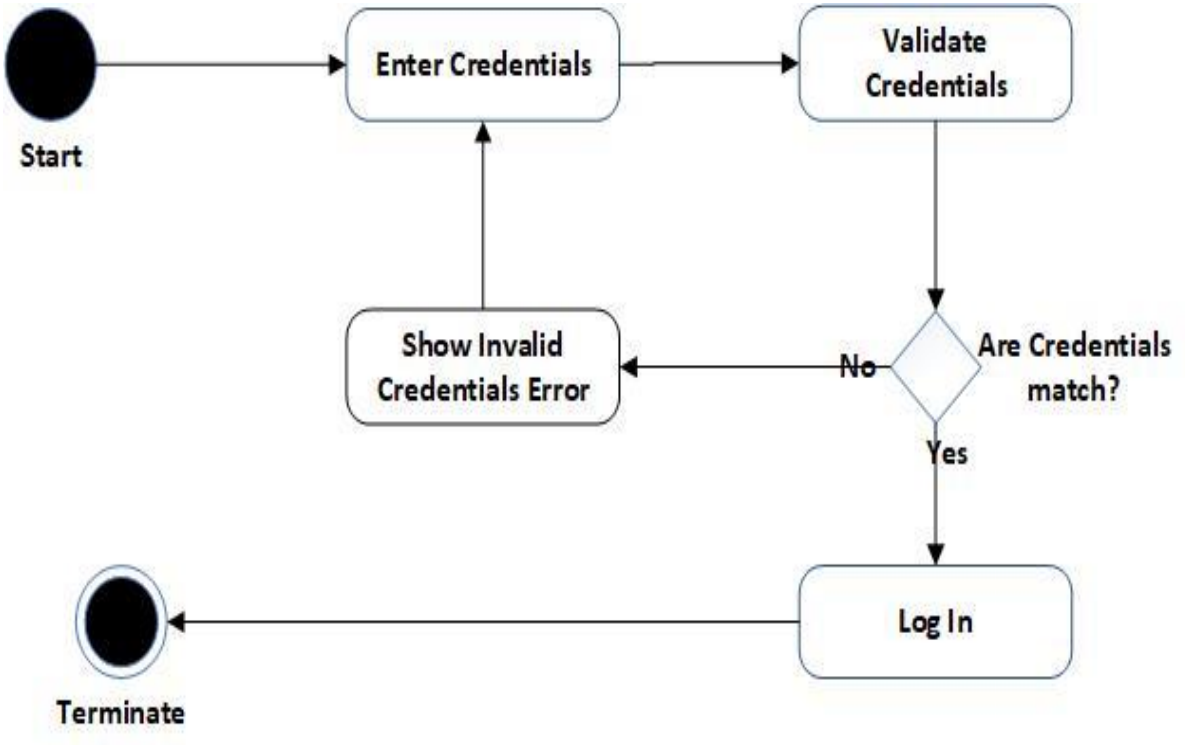


*Figure 4.2.1.2. Login*

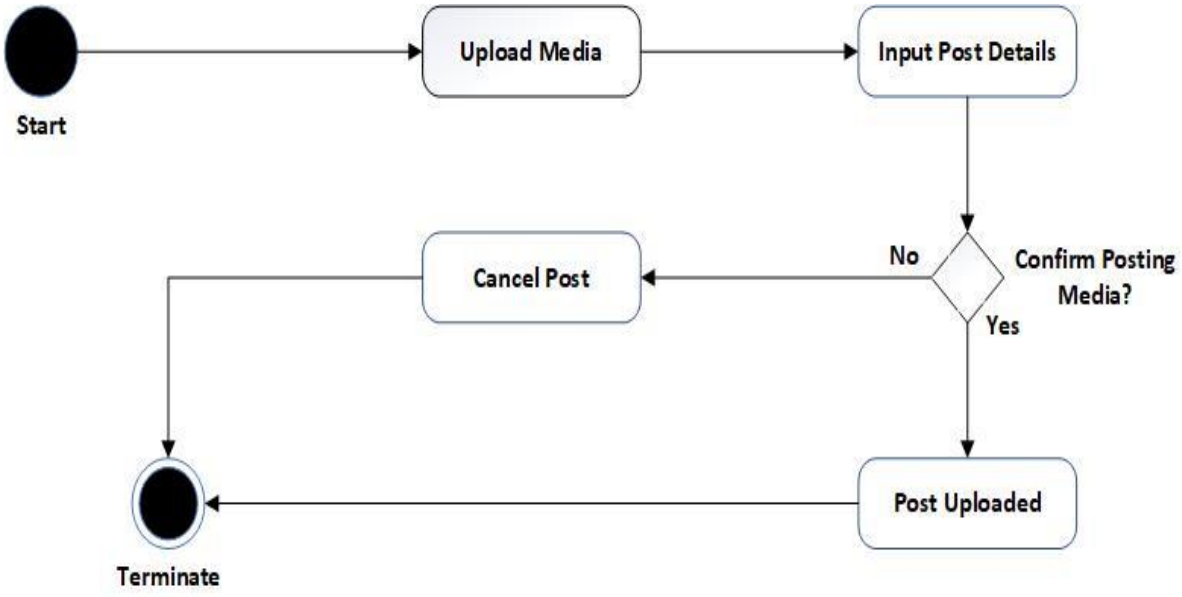


*Figure 4.2.1.3. Post*

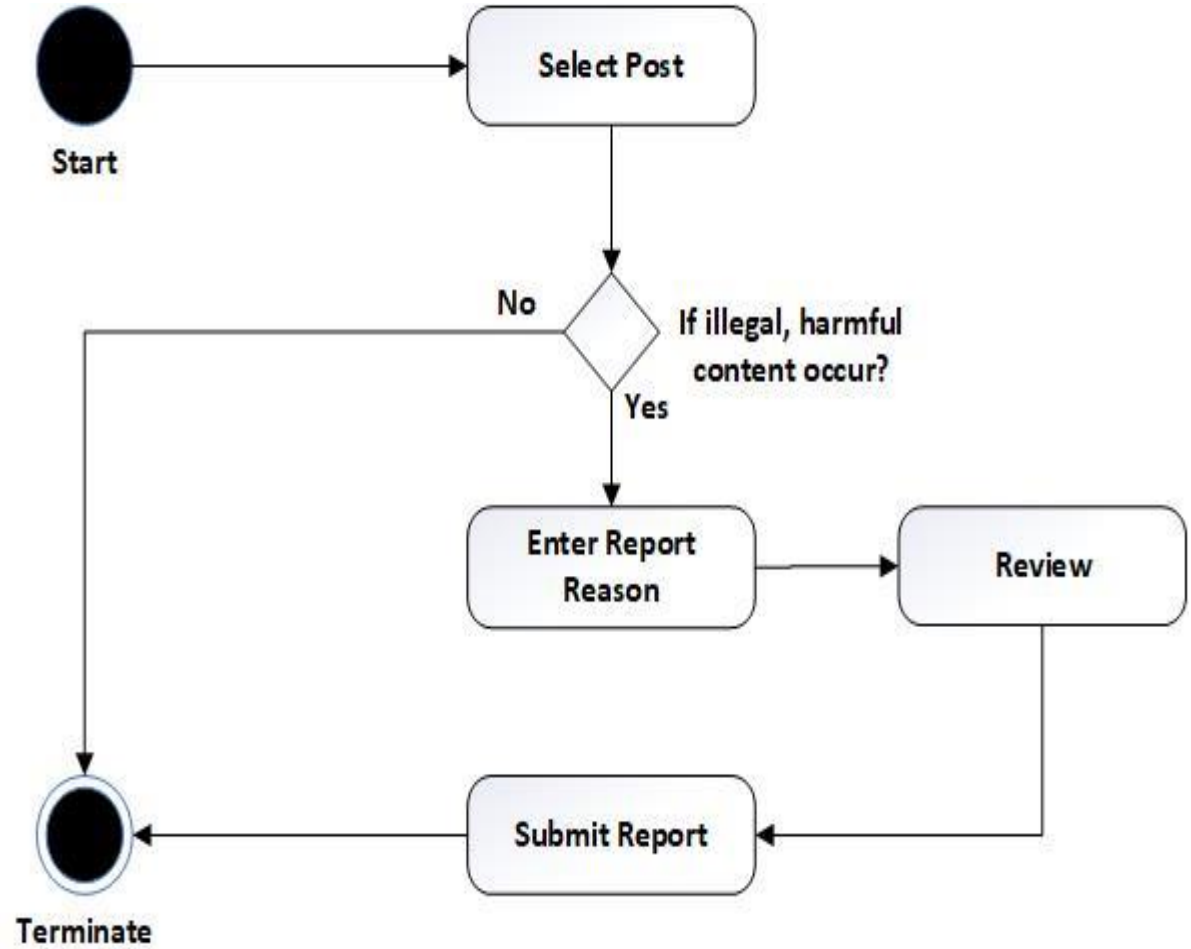


*Figure 4.2.1.4. Report Post*

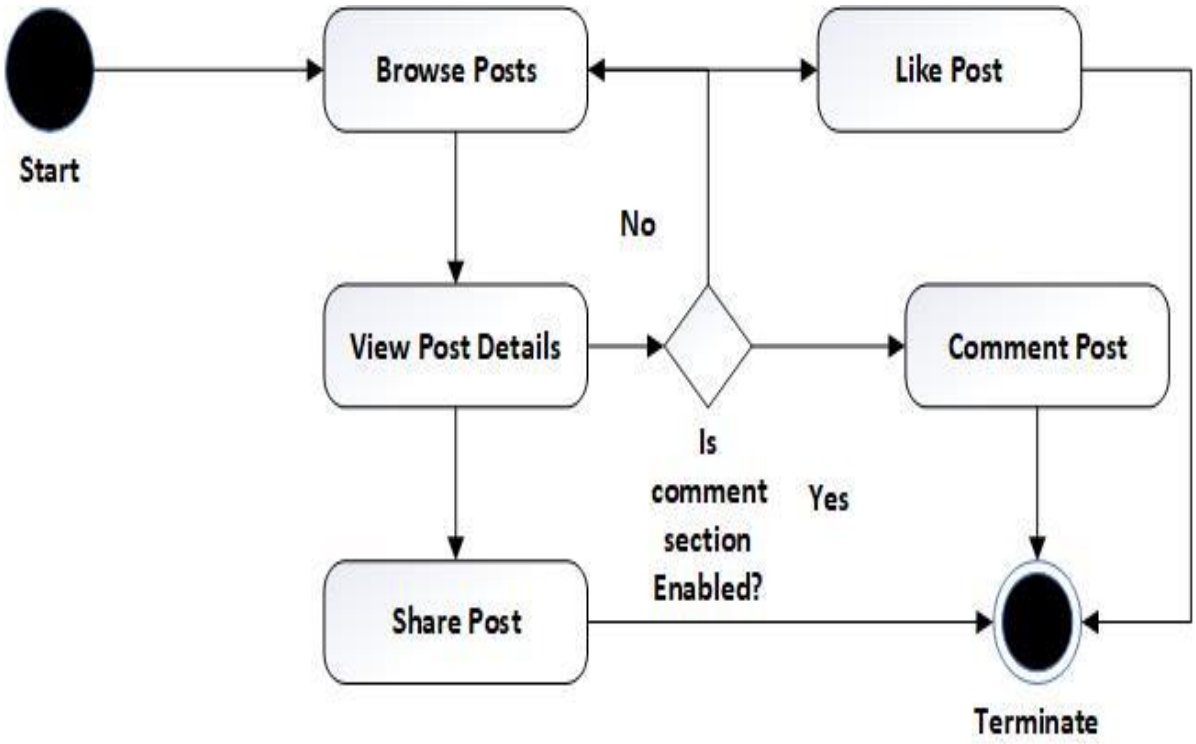


*Figure 4.2.1.5. Interact with Post*

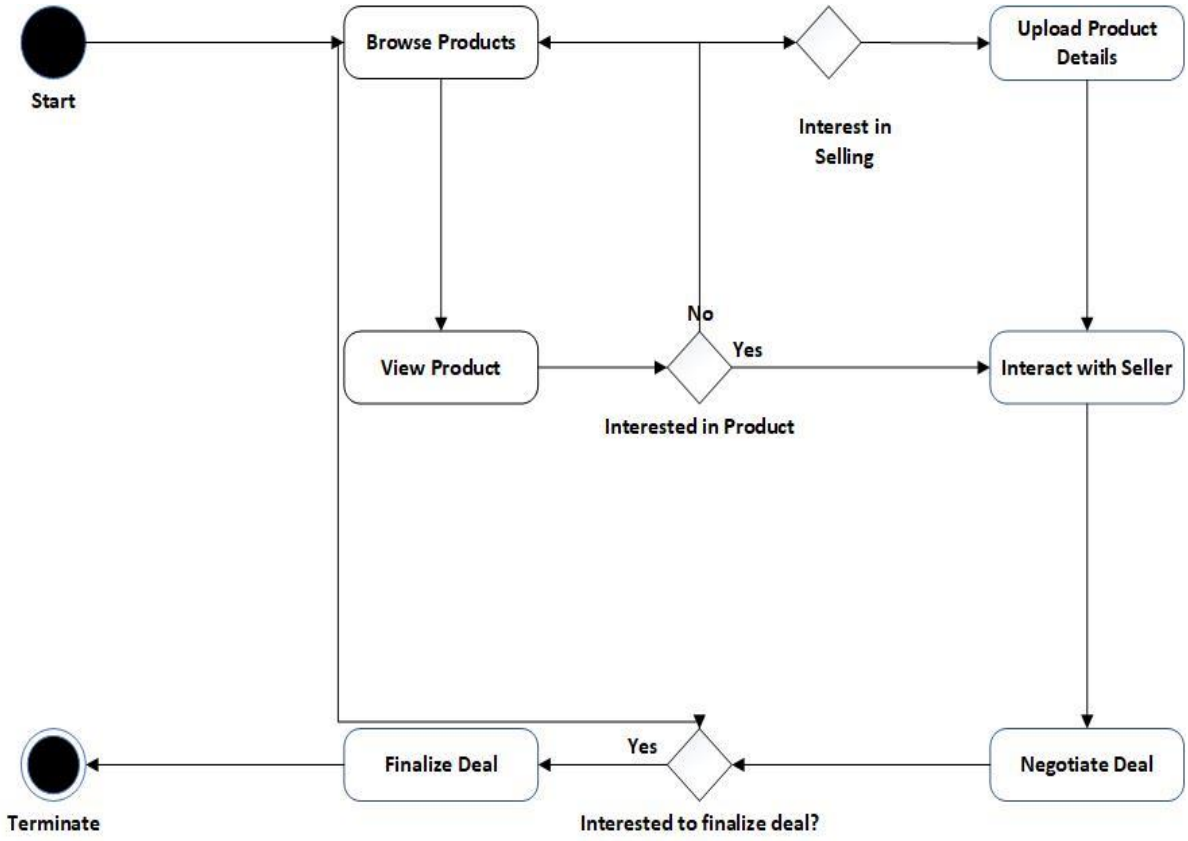


*Figure 4.2.1.6. Marketplace*

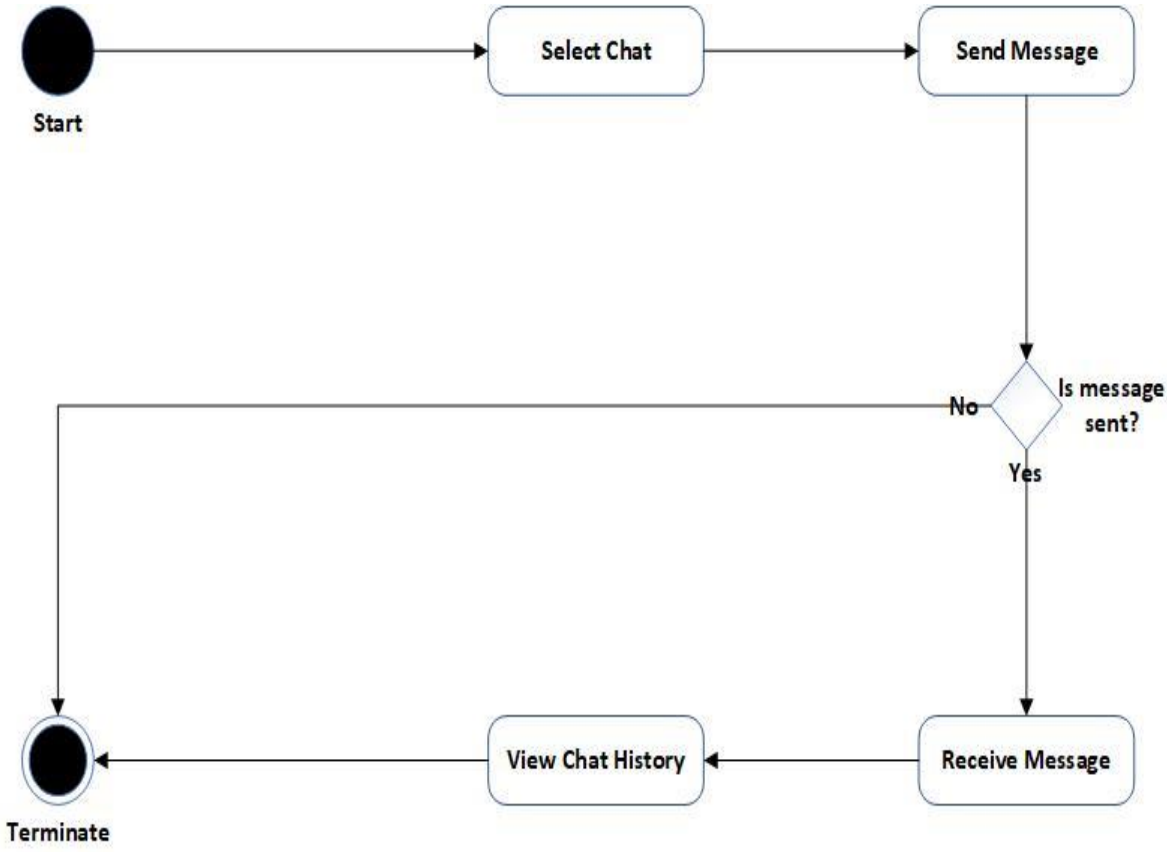


*Figure 4.2.1.7. Chatting*

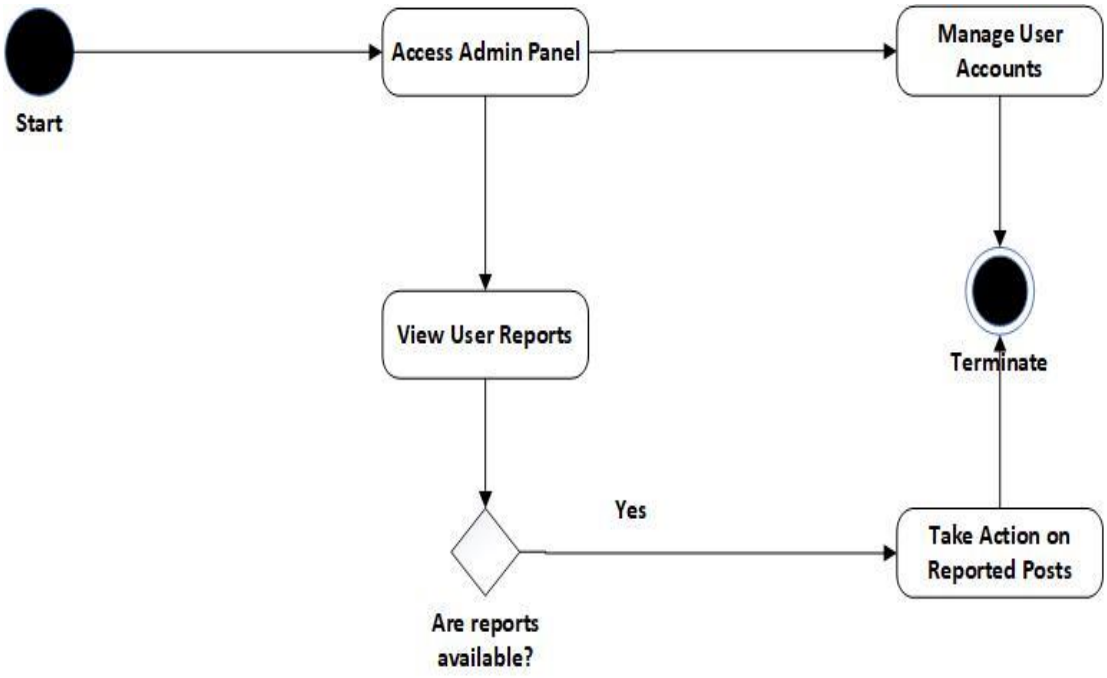


*Figure 4.2.1.8. Manage FarmConnect*

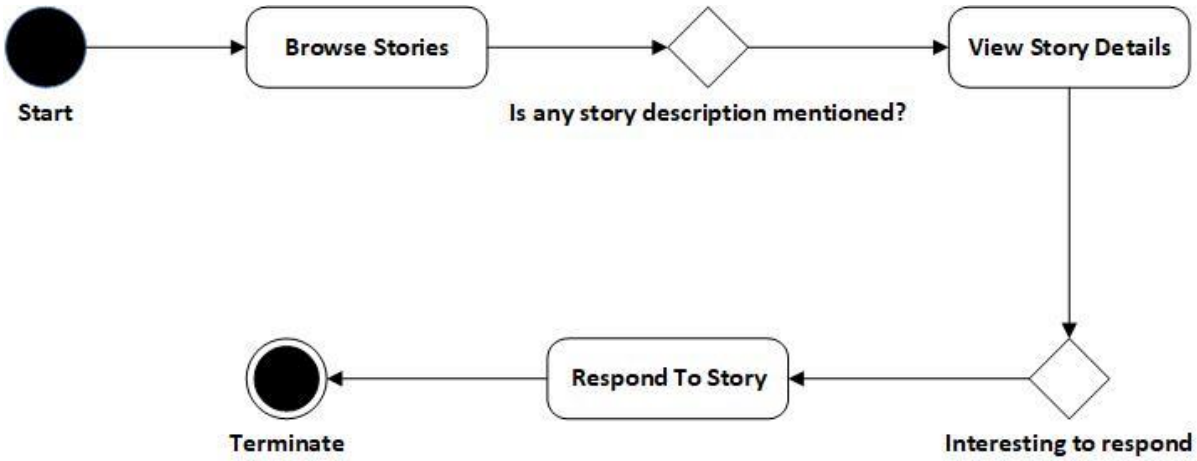


*Figure 4.2.1.9. Stories Response*

### 4.2.2. State Machine Diagrams

Displayed below are the state machine diagrams illustrating FarmConnect.

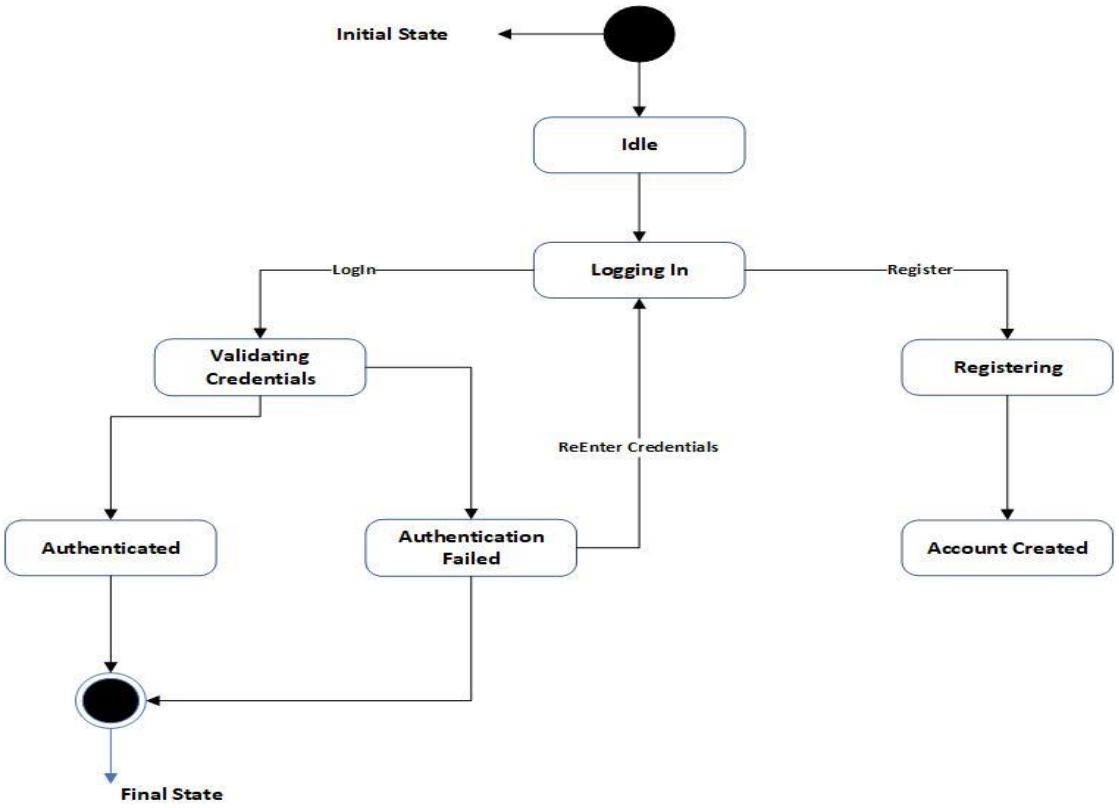


*Figure 4.2.2.1. Authentication*

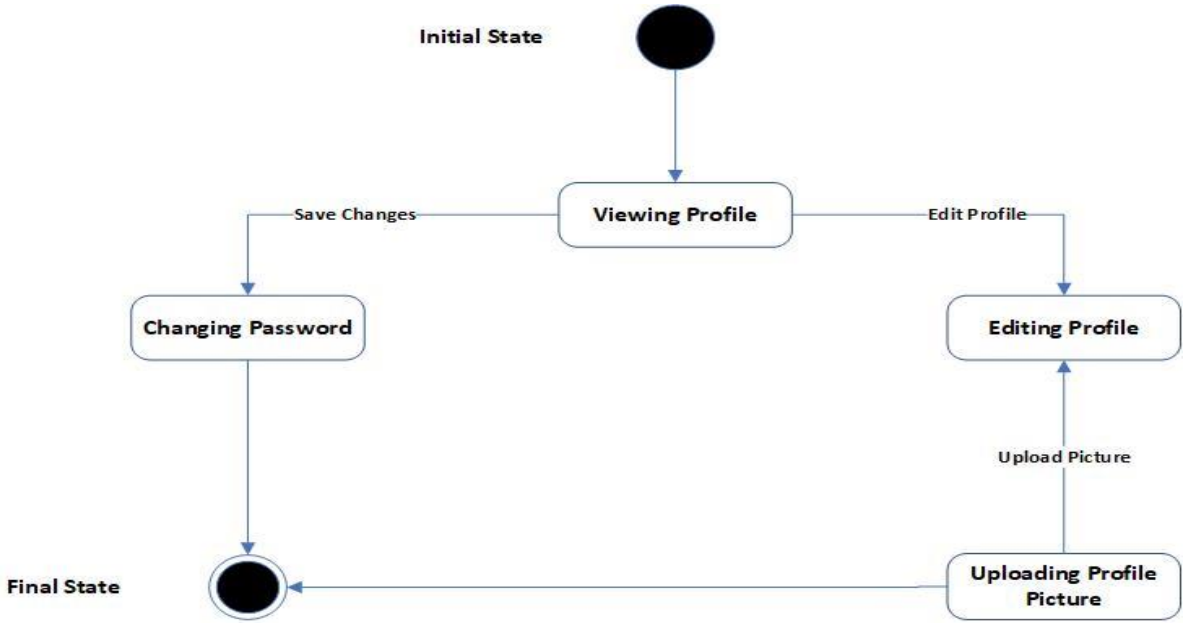


*Figure 4.2.2.2. Manage Profile*

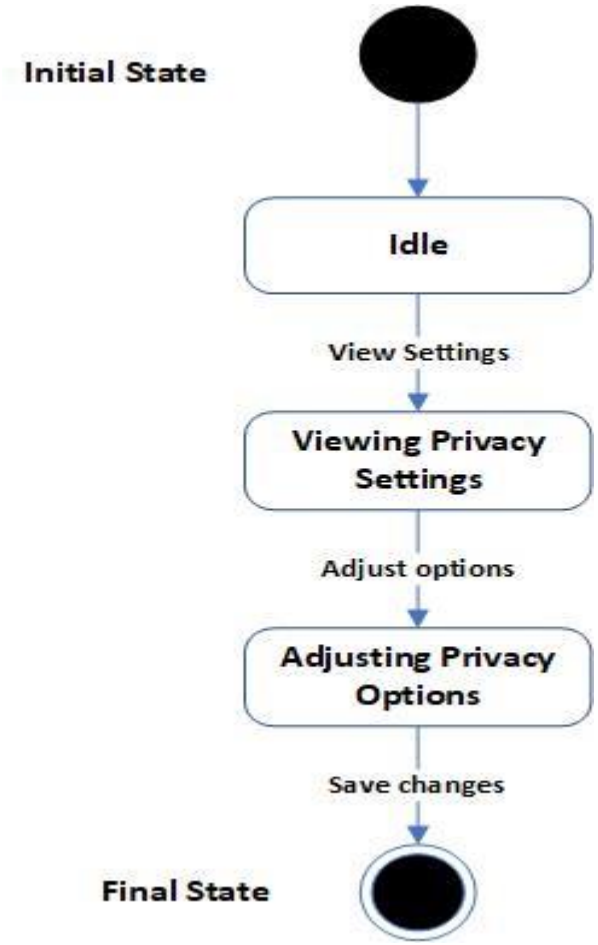


*Figure 4.2.2.3. Manage Privacy*

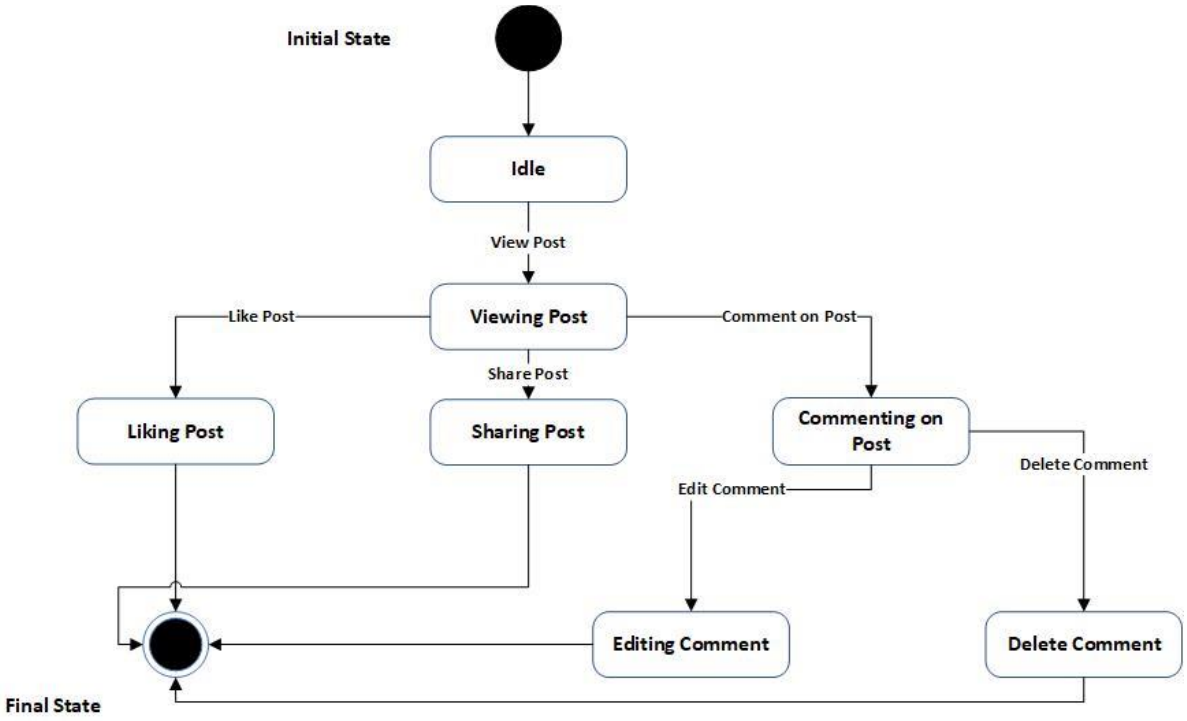


*Figure 4.2.2.4. Interaction with Post*

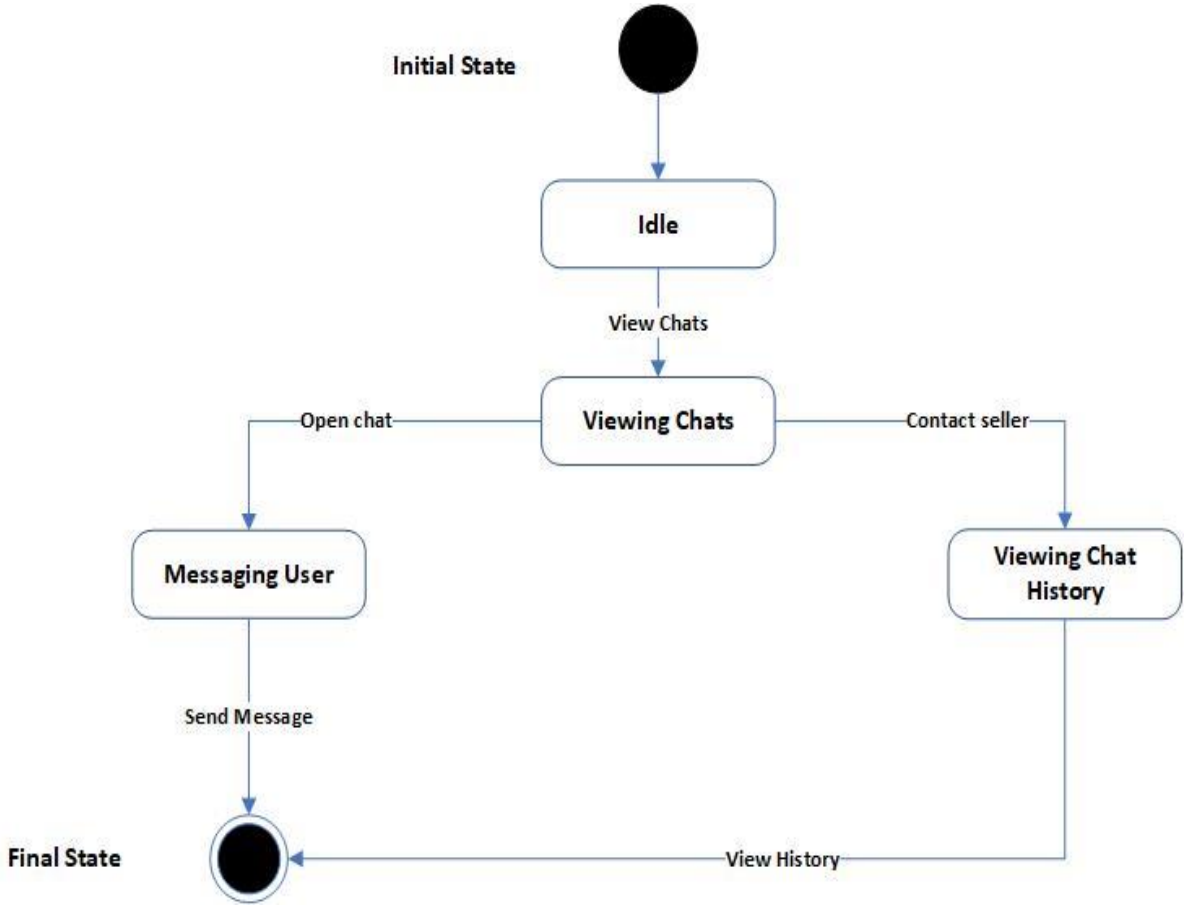


*Figure 4.2.2.5. Chatting*

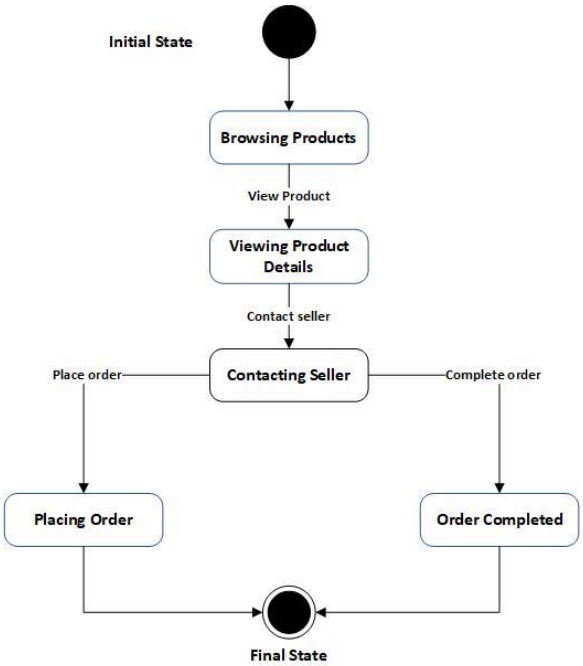


*Figure 4.2.2.6. Marketplace*

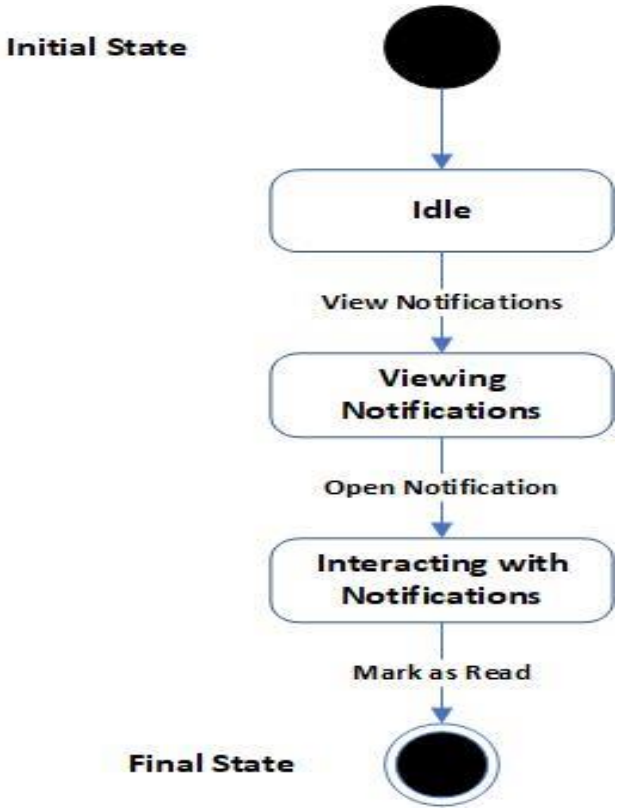


*Figure 4.2.2.7. Notifications*

## 4.3. UML Interaction Diagrams

Interaction diagrams are a category of UML (Unified Modeling Language) diagrams that emphasize the dynamic aspects of a system by illustrating how objects collaborate and interact with each other over time. The two main types of interaction diagrams in UML are Sequence Diagrams and Communication Diagrams.

### 4.3.1. Sequence Diagrams

Here is the Sequence Diagram for FarmConnect:

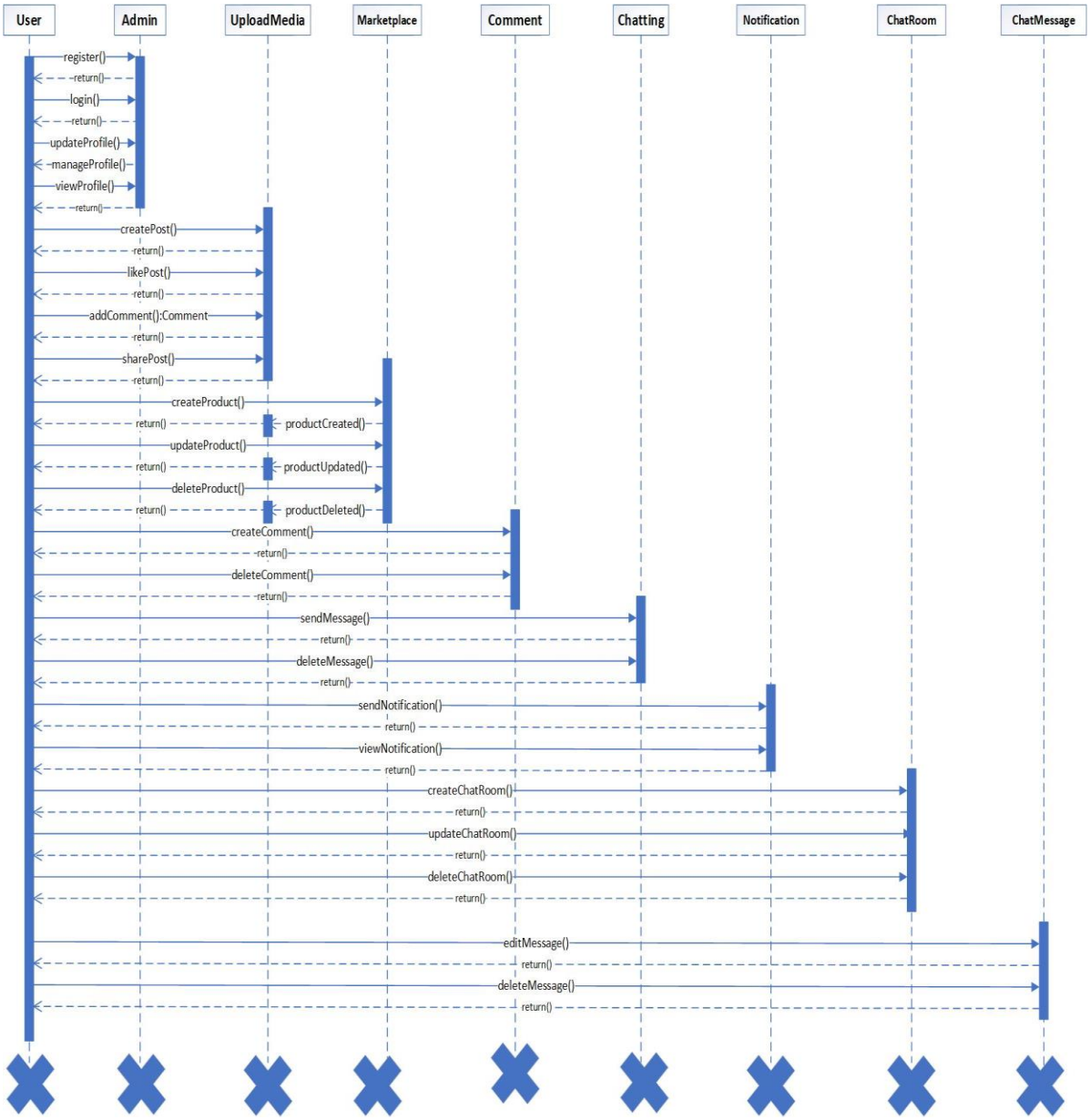


*Figure 4.3.1. Sequence Diagram*

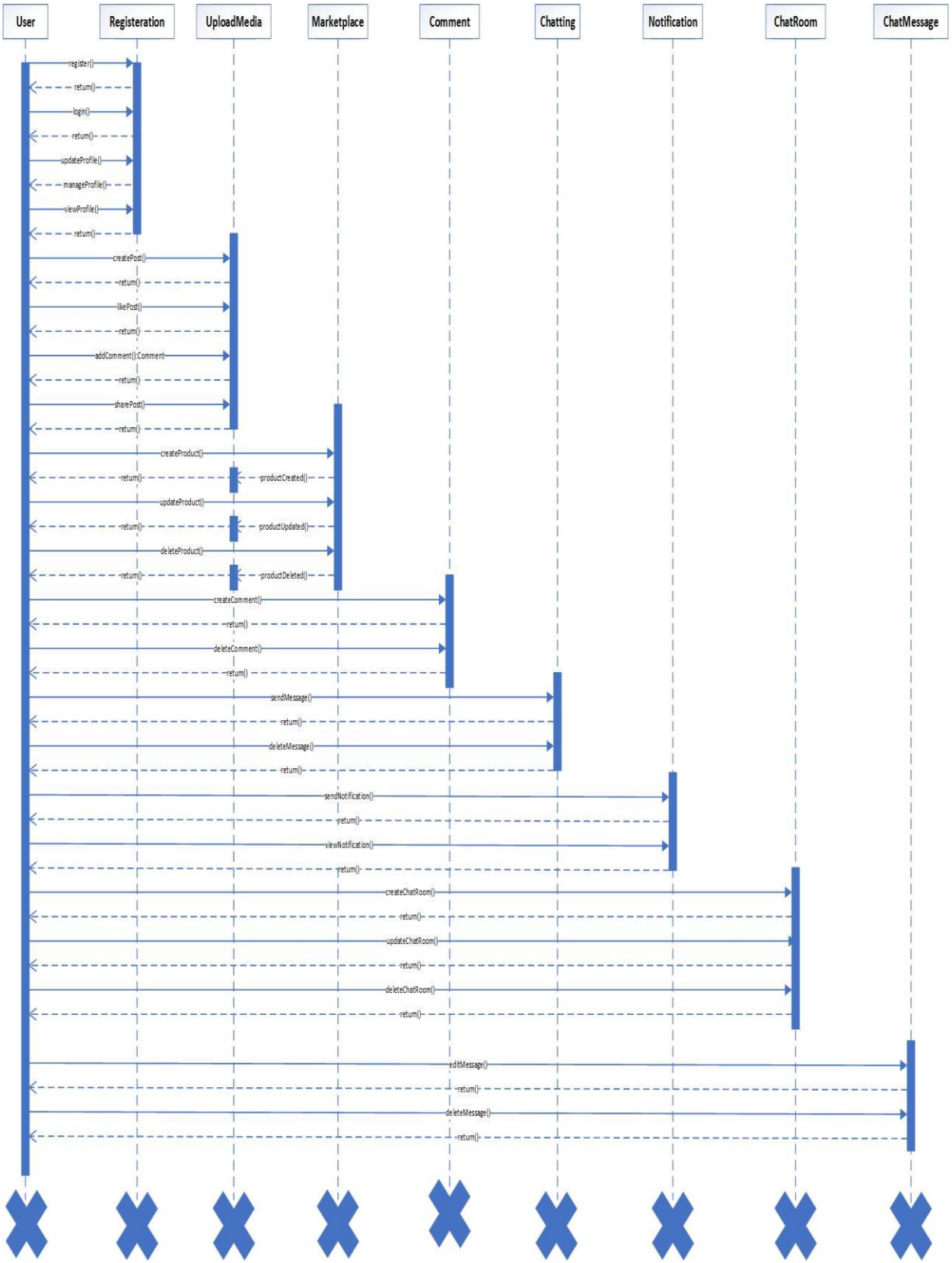


*Figure 4.3.1.1 User Panel Dashboard*

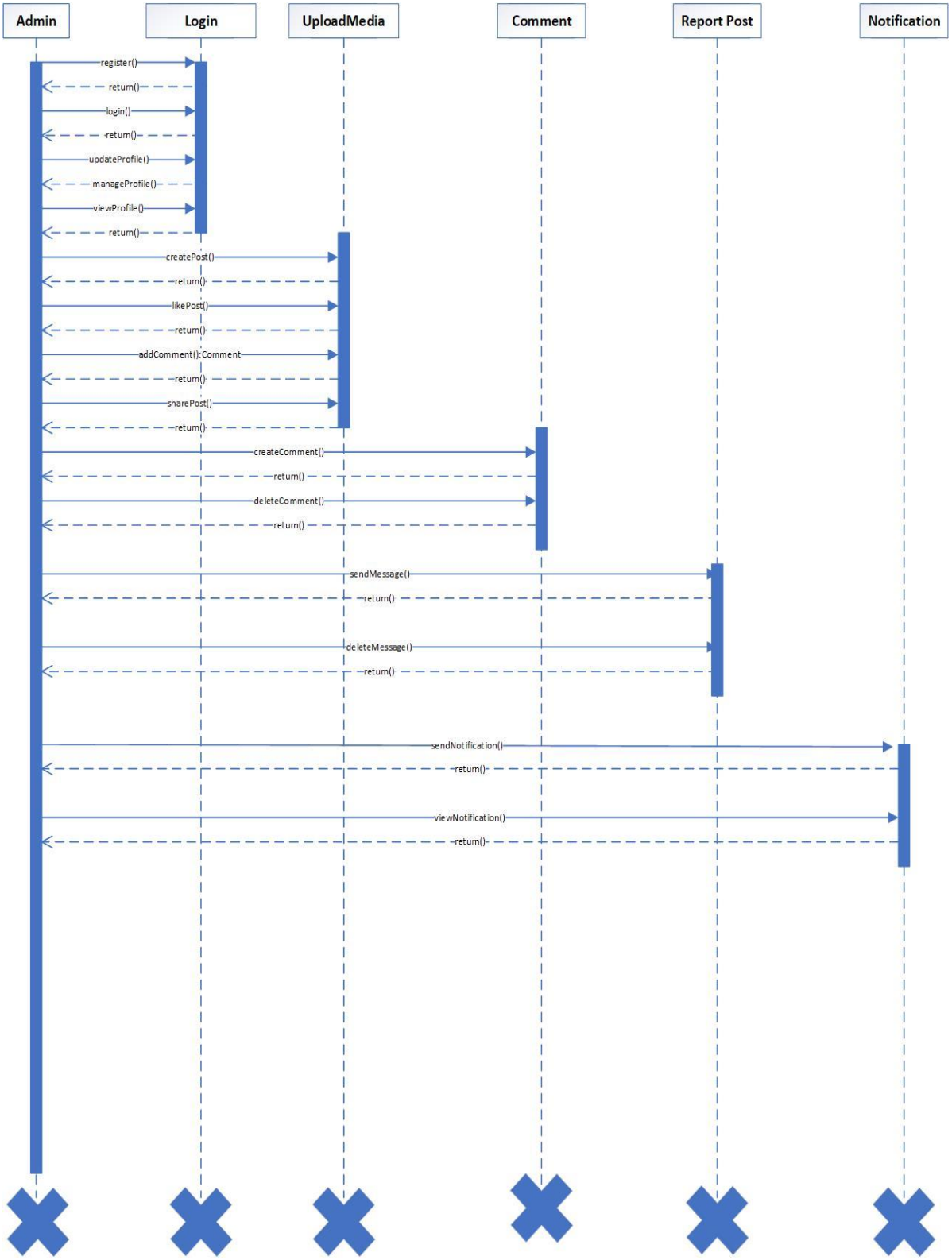


*Figure 4.3.1.2 Admin Panel Dashboard*

### 4.3.2. Collaboration Diagrams

Here is the Collaboration Diagram for FarmConnect:

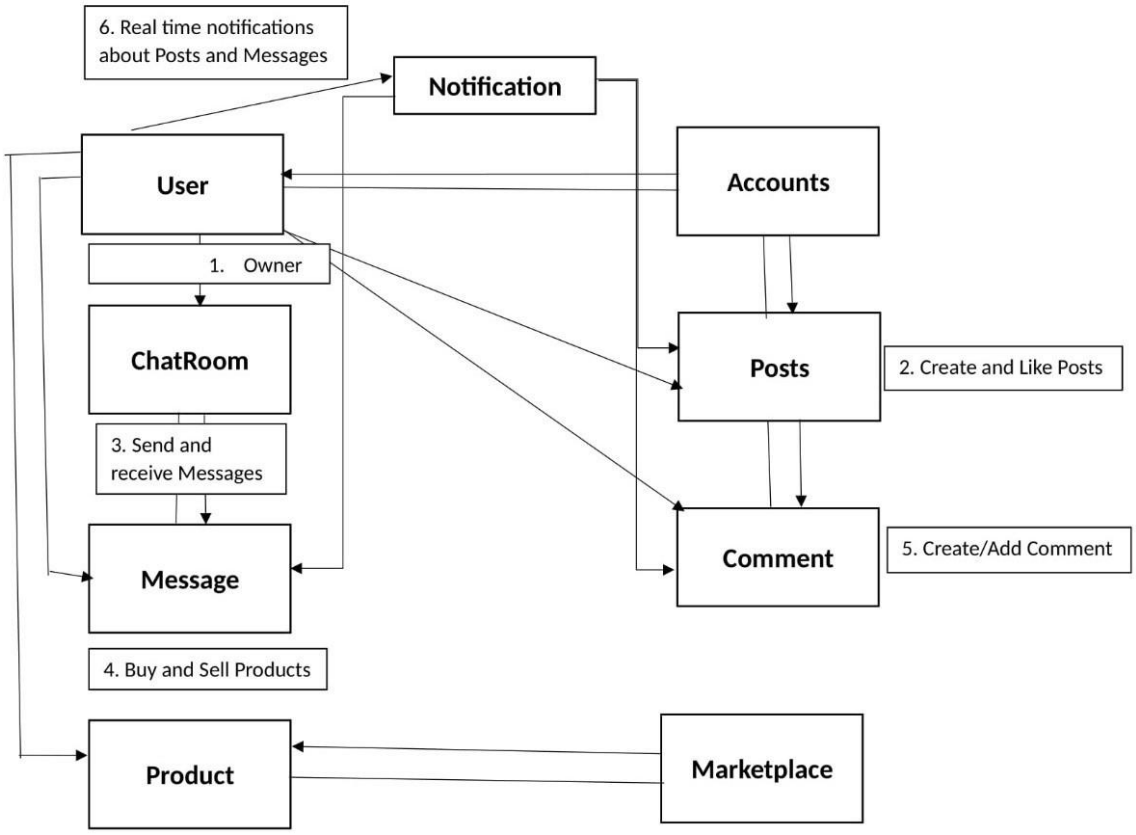


*Figure 4.3.2. Collaboration Diagram*

## 4.4. Node Structure

In the context of FarmConnect, the term "Node Structure" encompasses the organized arrangement of key components within the system, contributing to its overall functionality. FarmConnect's node structure involves various aspects, reflecting its comprehensive design and architecture. Here are key interpretations:

**i. Network Nodes**

**Definition:** Network nodes represent essential elements within the FarmConnect network, including servers, databases, and user devices.
**Role:** They facilitate seamless communication, data storage, and user interactions.

**ii. Graphical User Interface (GUI) Nodes**

**Definition:** GUI nodes refer to graphical elements within the FarmConnect application or web dashboard, such as buttons, forms, and UI components.
**Role:** They contribute to a user-friendly interface, enhancing user experience and accessibility.

**iii. Distributed System Nodes**

**Definition:** Nodes in a distributed system context signify organizational units responsible for specific functionalities, like user authentication, data storage, and communication.
**Role:** They ensure efficient system operation, scalability, and responsiveness.

**iv. Hierarchical Node Structure**

**Definition:** FarmConnect employs a hierarchical structure for elements like user profiles, posts, comments, etc.
**Role:** This structure aids in categorization, organization, and management of diverse data within the system.

FarmConnect's node structure is meticulously designed to optimize performance, promote scalability, and provide a robust framework for seamless user interactions. This structured approach enhances the system's overall reliability, ensuring a cohesive and efficient agricultural networking platform.

## 4.5. Communication Design Protocol

The communication design protocol for FarmConnect is a crucial aspect that governs the exchange of information between various components of the system, ensuring seamless and secure interactions. The protocol encompasses the following key elements:

**i. RESTful API Architecture:**

**Description:** FarmConnect adopts a Representational State Transfer (REST) architecture for its APIs.
**Purpose:** This design choice enables stateless communication, scalability, and standardization in data exchange between the frontend and backend components.

**ii. HTTPS (Hypertext Transfer Protocol Secure):**

**Description:** All communication between clients (mobile app, web dashboard) and the server occurs over the HTTPS protocol.
**Purpose:** HTTPS ensures secure and encrypted data transmission, safeguarding user information and preventing unauthorized access.

**iii. JSON (JavaScript Object Notation) Data Format:**

Description: Data exchanged between FarmConnect components is formatted in JSON.
Purpose: JSON offers a lightweight and human-readable format, enhancing data interchange efficiency and compatibility across different platforms.

**iv. WebSocket's for Real-Time Communication:**

**Description:** FarmConnect utilizes WebSocket's for real-time communication features such as chat.
**Purpose:** WebSocket's enable low-latency, bidirectional communication, supporting instant updates and notifications for users.

**v. OAuth 2.0 for Authentication:**

**Description:** Authentication is implemented using OAuth 2.0.

**Purpose:** OAuth 2.0 provides a secure and standardized framework for user authentication, ensuring only authorized users access FarmConnect services.

vi. **Custom API Tokens for Authorization:**

**Description:** Authorization is managed through custom API tokens.

**Purpose:** API tokens enhance security by allowing controlled access to specific resources, preventing unauthorized actions within the system.

**vii. Message Queues for Asynchronous Processing:**

**Description:** FarmConnect incorporates message queues for asynchronous processing of tasks.

**Purpose:** Message queues enhance system responsiveness, scalability, and resilience by decoupling components and processing tasks asynchronously.

The communication design protocol ensures that FarmConnect maintains a secure, efficient, and scalable framework for exchanging information, promoting a reliable and responsive agricultural networking platform.

## 4.6. System Architecture

The system architecture for FarmConnect aligns with a combination of the Client-Server Architecture and the Layered Architecture Pattern. Here's an explanation and justification for this choice:

**Explanation:**

**Client-Server Architecture:**

1. **Client-Side (Mobile and Web):**
   - The client-side includes both mobile and web interfaces, allowing users to interact with the system using diverse devices.
   - **Mobile App:** Developed in Java for Android.
   - **Web Interface:** Utilizes HTML, CSS, Bootstrap and JavaScript.

2. **Server-Side:**
   - The server-side is responsible for processing requests, managing data, and handling business logic.
   - **Backend:** Developed using the Laravel PHP framework for API development.
   - **Database:** MySQL is employed for data storage and retrieval.

**Layered Architecture Pattern:**

- The system follows a layered architecture pattern to organize and separate concerns within the application.
- Layers include Presentation, Application, Business, and Data layers.
- Separation of concerns enhances maintainability, scalability, and flexibility.

**Justification:**

**Scalability:**

1. **Modular Design:** The architecture supports a modular design, enabling scalability by adding or updating specific components without affecting the entire system.
2. **Server-Side Scalability**: Allows horizontal scaling by adding more servers or leveraging cloud services as the user base grows.

**Security and Performance:**

1. **Laravel Security Features:** Laravel provides built-in security features, including authentication, data encryption, and protection against common web vulnerabilities.
2. **MySQL Security:** Utilizing MySQL for the database ensures robust security measures for data protection.
3. **Performance Optimization:** The system incorporates caching mechanisms and query optimizations for improved performance.

**Ease of Maintenance:**

1. **Separation of Concerns:** The layered architecture separates different concerns, making it easier to maintain and update specific components independently.
2. **Code Modularity:** Modular components allow developers to focus on specific parts of the system, simplifying maintenance efforts.

**Flexibility:**

**Framework Flexibility:** Leveraging versatile frameworks like Laravel and mobile development languages allows for easy adaptation to evolving technology trends.

**Reliability and Availability:**

1. **Use of Established Technologies:** Laravel and MySQL are well-established technologies known for their reliability and stability.
2. **Potential Cloud Integration:** Integration with cloud services offers high availability and disaster recovery capabilities.

**Community Support and Resources:**

**Laravel and MySQL:** Both technologies enjoy extensive community support, abundant resources, and a wide range of plugins/extensions, contributing to the overall reliability of the system.

In summary, the chosen architecture of FarmConnect aligns with a Client-Server model with a Layered Architecture Pattern, emphasizing scalability, security, ease of maintenance, and compatibility to meet the unique requirements of a social platform tailored for the agricultural community.

## 4.7. System Design

Design patterns are reusable solutions to common problems in software design. Choosing the appropriate design pattern depends on the specific requirements and characteristics of the system. In the case of FarmConnect, a social networking and marketplace platform, various design patterns can be applied based on the system's architecture and functionality. Below are some potential design patterns that could be considered:

1. **MVC (Model-View-Controller) Design Pattern:**

**Description:**

- MVC separates the application into three interconnected components: Model, View, and Controller.
- **Model:** Represents the data and business logic.
- **View:** Displays the data and user interface.
- **Controller**: Handles user input and updates the Model and View accordingly.

**Application:**

- The MVC pattern is suitable for separating the user interface, business logic, and data handling in FarmConnect.
- The Model can represent user profiles, posts, and marketplace data.
- Views can handle the presentation of user profiles, posts, and marketplace listings.
- Controllers manage user interactions, such as posting, chatting, and marketplace interactions.

2. **Observer Design Pattern:**

**Description:**

Observer defines a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically.

**Application:**

- The Observer pattern can be used for real-time updates and notifications within FarmConnect.
- For example, when a user posts something or updates their profile, followers or interested users are notified.
- This pattern is useful for handling events like new messages, post likes, and marketplace updates.

3. **Strategy Design Pattern:**

**Description:**

- Strategy defines a family of algorithms, encapsulates each algorithm, and makes them interchangeable.
- It allows the client to choose the appropriate algorithm at runtime.

**Application:**

- In FarmConnect, the Strategy pattern can be applied to handle different algorithms for sorting posts, searching for users, or filtering marketplace listings.
- For example, different strategies can be employed for sorting posts by date, popularity, or relevance.

**4. Singleton Design Pattern:**

**Description:**
- Ensures a class has only one instance and provides a global point to this instance.
- Useful when only a single instance of a class should control actions.

**Application:**
- The Singleton pattern can be applied to manage a centralized notification system or a logging service within FarmConnect.
- Ensures that there's a single point for managing system-wide operations or resources.

# Chapter 5: Implementation

.

## 5.1. Component Diagram

1. **Chatbot Module (ChatGPT API):**

This module represents the integration of a chatbot powered by the ChatGPT API. The chatbot allows users to interact with the system in a conversational manner, providing assistance, answering questions, and performing tasks based on user input. Interactions with the chatbot module occur through API requests, where user messages are sent to the ChatGPT API, and responses are received and displayed to the user.

2. **Weather Module (External Services):**

This module connects to external weather services to provide real-time weather information to users. It may make use of APIs provided by weather service providers to fetch weather data based on the user's location or specified criteria. The weather module periodically fetches weather updates and may store them in the system's database for future reference.

3. **Backend Modules (Web API/Web Dashboard APIs):**

These modules encompass various backend functionalities of Farm Connect, including user management, data storage, and business logic. They expose APIs that the mobile app interacts with to perform operations such as user authentication, data retrieval, and content creation. The backend modules handle requests from the mobile app and respond with the appropriate data or perform the necessary actions. Data managed by the backend modules includes user profiles, newsfeed content, marketplace listings, group/community information, and more. The web dashboard APIs specifically cater to administrative tasks, allowing authorized users to manage system settings, content moderation, and other administrative functions through a web-based interface. Database: This component represents the database where all system data is stored. It includes tables/entities for storing user information, chatbot interactions, weather data, content posts, marketplace listings, and other relevant data. The database serves as the central repository for FarmConnect's information, allowing efficient data retrieval and management by the backend modules.

## 5.2. Network and Protocol Choice

Network and protocol choices for FarmConnect play a crucial role in ensuring efficient communication, data security, and scalability. Here's an explanation of the network and protocol choices:

1) **Network Choice:**

**Internet:** FarmConnect operates over the internet, allowing users to access the platform from anywhere with an internet connection. This choice provides ubiquitous access to the application, enabling farmers to stay connected regardless of their location.

**Mobile Networks:** Given that FarmConnect is likely to be accessed via mobile devices by farmers who may not always have access to Wi-Fi, supporting mobile networks (2G, 3G, 4G, and potentially 5G) is essential. This ensures that users can access the platform even in rural areas with limited connectivity.

2) **Protocol Choice:**

**HTTP/HTTPS:** Hypertext Transfer Protocol (HTTP) and its secure counterpart HTTPS are fundamental protocols for communication between clients (such as mobile apps or web browsers) and servers over the internet. FarmConnect utilizes HTTPS to ensure secure communication between the mobile app and backend servers, protecting sensitive user data and preventing unauthorized access.

**RESTful API:** Representational State Transfer (REST) is an architectural style for designing networked applications. FarmConnect's backend exposes RESTful APIs, allowing clients to interact with the server using standard HTTP methods (GET, POST, PUT, DELETE). This choice promotes interoperability, scalability, and simplicity in communication between the mobile app and backend services.

**WebSocket (Optional):** While traditional HTTP-based communication is suitable for most interactions between the mobile app and backend, FarmConnect may optionally implement WebSocket protocol for real-time features such as chat functionality. WebSocket enables bidirectional communication channels over a single TCP connection, facilitating instant updates and notifications without the overhead of HTTP polling.

3) **Data Transfer Formats:**

**JSON (JavaScript Object Notation):** JSON is a lightweight data interchange format that is easy for both humans and machines to read and write. FarmConnect utilizes JSON for data transfer between the mobile app and backend APIs, ensuring compatibility and ease of parsing on various platforms and programming languages.

**Protocol Buffers (Optional):** Protocol Buffers is a binary serialization format developed by Google. While JSON is widely used for its simplicity and human readability, Protocol Buffers offer more compact and efficient serialization, which can be beneficial for reducing network bandwidth and improving performance in data-intensive scenarios.

## 5.3. Choice of Object Middleware

**Remote Method Invocation (RMI):**

- RMI is a Java-specific middleware technology that allows Java objects to invoke methods on remote objects running on different Java Virtual Machines (JVMs).
- In the context of FarmConnect, RMI could be used if the system is predominantly built using Java technologies and requires seamless communication between Java components across distributed environments.
- RMI simplifies distributed object communication within a Java ecosystem but may have limitations when integrating with non-Java components or when interoperability with other platforms is required.

**Common Object Request Broker Architecture (CORBA):**

- CORBA is a platform-independent, language-independent middleware standard for distributed computing. It enables communication between objects written in different programming languages.
- FarmConnect could utilize CORBA if it involves components written in multiple languages (not limited to Java) and requires interoperability between them.
- CORBA provides a standardized approach to distributed object communication but may involve more complexity in setup and configuration compared to language-specific middleware like RMI.

**Distributed Component Object Model (DCOM):**

- DCOM is a Microsoft-specific middleware technology for communication between software components distributed across networked computers in a heterogeneous environment.
- If FarmConnect predominantly utilizes Microsoft technologies (such as .NET) and requires seamless communication between components running on Windows-based systems, DCOM could be considered.
- DCOM provides tight integration with Windows platforms and tools but may have limited support for interoperability with non-Microsoft technologies.

## 5.4. User Interface

### 1. Authentication Module:

- **Login Screen:** Users can enter their credentials (username/email and password) to log in to their FarmConnect account.
- **Registration Screen:** New users can sign up for an account by providing required information such as name, email, password, and optionally, profile information like location and farm size.
- **Forgot Password Screen:** Users can reset their password by providing their email address and following the password reset instructions sent to their email.

### 2. Dashboard Module:

- **Main Dashboard**: Upon successful login, users are presented with a main dashboard displaying personalized information such as weather updates, newsfeed posts (Stories etc.), upcoming events, and quick access to key features.
- **Weather Widget:** A widget displaying real-time weather information based on the user's location or a location of their choice.
- **Newsfeed:** A feed displaying posts, articles, and updates relevant to the user's interests and subscriptions.

### 3. Communication Module:

- **Chat Interface:** Users can engage in real-time chat conversations with other users, groups, or automated chatbots for assistance.
- **Messaging Inbox:** Users can access their message inbox to view, reply to, and manage conversations with other users.

### 4. Community Module:

- **Group/Community Directory:** Users can browse and search for different farming communities or groups based on interests, locations, or specific topics.
- **Group/Community Page:** Upon joining a group or community, users can access a dedicated page displaying group information, discussions, events, and member profiles.
- **Event Calendar:** A calendar interface displaying upcoming farming-related events, workshops, and meetings organized by communities or groups.

5. **Marketplace Module:**

- **Product Listings:** Users can browse and search for agricultural products and services listed by sellers in the marketplace.
- **Product Details Page:** Users can view detailed information about a specific product or service, including pricing, description, seller information, and customer reviews.
- **Shopping Cart:** Users can add products to their cart for purchase and proceed to checkout to complete the transaction securely.

6. **Profile Module:**

- **User Profile Page:** Users can view and edit their profile information, including personal details, farm information, interests, and preferences.
- **Settings:** Users can access settings to customize their FarmConnect experience, including notification preferences, privacy settings, and account management options.

7. **Admin Module (Web Dashboard):**

- **Admin Dashboard:** Administrators can access a web-based dashboard to manage system settings, user accounts, content moderation, analytics, and other administrative tasks.
- **User Management:** Administrators can view and manage user accounts, including creating, editing, and suspending user accounts as needed.
- **Content Moderation:** Administrators can review and moderate user-generated content, including posts, comments, and marketplace listings, to ensure compliance with community guidelines and policies.

# Chapter 6: Testing and Evaluation

## 6.1. Verification

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_01 | |
| **Associated Use Case:** | | | UC_1.1.1 | |
| **Functionality:** | | | User Registration | |
| **Description:** | | | Verify that users can register successfully with valid data and OTP verification. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is not logged into the application.<br>2. Registration page is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the registration page. | ➢ Enter a valid username.<br>➢ Enter a valid phone number.<br>➢ Enter a valid password.<br>➢ Confirm the password.<br>➢ Click "Register".<br>➢ Enter the received OTP.<br>➢ Click "Verify". | User is registered successfully, OTP is verified, and user is redirected to the login page or dashboard. | Pass |
| **Post-condition:** | | | User account is created in the system, and user receives a welcome message. | |

*Table 6.1.1*

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_1.1 | |
| **Associated Use Case:** | | | UC_1.1.1 | |
| **Functionality:** | | | User Registration | |
| **Description:** | | | Verify registration fails with an incorrect OTP. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is not logged into the application.<br>2. Registration page is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the registration page. | ➢ Enter a valid username.<br>➢ Enter a valid phone number.<br>➢ Enter a valid password.<br>➢ Confirm the password.<br>➢ Click "Register".<br>➢ Enter an incorrect OTP.<br>➢ Click "Verify". | System displays an error message indicating the OTP is incorrect | Pass |
| **Post-condition:** | | | User account is not created, and user is prompted to enter the correct OTP. | |

*Table 6.1.2*

| **Test Case ID:** | | | TC_1.2 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC_1.1.1 | |
| **Functionality:** | | | User Registration | |
| **Description:** | | | Verify registration fails with an already registered phone number. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is not logged into the application.<br>2. Registration page is accessible.<br>3. Another user is already registered with the same phone number. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Registration page. | ➢ Enter a valid username.<br>➢ Enter the phone number already in use.<br>➢ Enter a valid password.<br>➢ Confirm the password.<br>➢ Click "Register". | System detects duplicate phone number and displays an error message. | Pass |
| **Post-condition:** | | | User account is not created, and system prompts user with an error. | |

*Table 6.1.3*

| **Test Case ID:** | | | TC_1.3 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC_1.1.2 | |
| **Functionality:** | | | User Login | |
| **Description:** | | | Verify that users can log in with valid credentials. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is not logged into the application.<br>2. Login page is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the login page. | ➢ Enter the registered phone number.<br>➢ Enter the correct password.<br>➢ Click "Login". | User is logged in successfully and redirected to the dashboard. | Pass |
| **Post-condition:** | | | User session is active and user has access to account functionalities. | |

*Table 6.1.4*

| **Test Case ID:** | | | TC_1.4 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC_1.1.2 | |
| **Functionality:** | | | User Login | |
| **Description:** | | | Verify login fails with incorrect credentials. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 3. User is not logged into the application.<br>4. Login page is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the login page. | ➢ Enter the registered phone number.<br>➢ Enter an incorrect password.<br>➢ Click "Login". | System displays an error message indicating the login credentials are incorrect. | Pass |
| **Post-condition:** | | | User session is not started, and user is prompted to enter correct credentials | |

*Table 6.1.5*

| **Test Case ID:** | TC_2 |
|---|---|
| **Associated Use Case:** | UC_2.2.1 |
| **Functionality:** | Post Creation in Newsfeed |

| **Description:** | | | Verify that users can create a post in the newsfeed. | |
|---|---|---|---|---|
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is logged into the application.<br>2. User is on the newsfeed page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the newsfeed and open the post creation form | ➢ Click on the post creation button/form.<br>➢ Enter post content.<br>➢ Click the "Submit" button. | Post is successfully created and displayed in the newsfeed. | Pass |
| **Post-condition:** | | | Post is stored in the database.<br>Post appears in the user's newsfeed and that of their connections/followers. | |

*Table 6.1.6*

| **Test Case ID:** | | | TC_2.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC_2.2.1 | |
| **Functionality:** | | | Post Creation in Newsfeed | |
| **Description:** | | | Verify post creation fails with empty content. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is logged into the application.<br>2. User is on the newsfeed page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the newsfeed and open the post creation form. | ➢ Click on the post creation button/form.<br>➢ Leave the post content field empty.<br>➢ Click the "Submit" button. | System displays an error message indicating the post content cannot be empty | Pass |
| **Post-condition:** | | | Post is not created.<br>User is prompted to enter content. | |

*Table 6.1.7*

| **Test Case ID:** | | | TC_3 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-3.1.1 | |
| **Functionality:** | | | Chatting | |
| **Description:** | | | Verify that users can send a private message. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is logged into the application.<br>2. User is on the messaging page/conversation window. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the chatting page. | ➢ Open a conversation with another user.<br>➢ Type a message.<br>➢ Click "Send. | Message is sent successfully and appears in the conversation thread. | Pass |
| **Post-condition:** | | | Message is saved in the database. Recipient sees the message in their conversation thread. | |

*Table 6.1.8*

| **Test Case ID:** | | | TC_3.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-3.1.1 | |
| **Functionality:** | | | Chatting | |
| **Description:** | | | Verify that users cannot send empty messages. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is logged into the application.<br>2. User is on the messaging page/conversation window. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the chatting page. | ➢ Open a conversation with another user.<br>➢ Leave the message field empty.<br>➢ Click "Send." | System displays an error message indicating the message content cannot be empty. | Pass |
| **Post-condition:** | | | Empty message is not sent. User is prompted to enter a message. | |

*Table 6.1.9*

| **Test Case ID:** | | | TC_3.2 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-3.1.1 | |
| **Functionality:** | | | Chatting | |
| **Description:** | | | Verify that users can create a new group. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is logged into the application.<br>2. User is in the groups section. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the chatting page. | ➢ Click on the group creation button/form.<br>➢ Enter group name, description, and other required details.<br>➢ Click "Create." | Group is successfully created and appears in the user's groups list. | Pass |
| **Post-condition:** | | | Group is saved in the database. Group appears in the user's group list and other relevant sections. User is prompted to complete the required fields. | |

*Table 6.1.10*

| **Test Case ID:** | | | TC_3.3 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-3.1.1 | |
| **Functionality:** | | | Chatting | |
| **Description:** | | | Verify group creation fails with missing or invalid data. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is logged into the application.<br>2. User is in the groups section. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the chatting page. | ➢ Click on the group creation button/form.<br>➢ Leave the required | System displays an error message indicating the data is missing or invalid. | Pass |

| | | | | |
|---|---|---|---|---|
| | | fields empty or fill them with invalid data.<br>➢ Click "Create." | | |
| **Post-condition:** | | | Group is not created.<br>User is prompted to enter valid data. | |

*Table 6.1.11*

| **Test Case ID:** | | | TC_4 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-4.1.1 | |
| **Functionality:** | | | Explore Marketplace | |
| **Description:** | | | Verify that users can create a product listing in the marketplace. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | 1. User is logged into the application.<br>2. User is on the marketplace page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the marketplace page. | ➢ Click on the product listing button/form.<br>➢ Enter valid product details.<br>➢ Click "Submit." | Product listing is successfully created and visible in the marketplace. | Pass |
| **Post-condition:** | | | Product listing is saved in the database.<br>Product appears in the marketplace for other users to see. | |

*Table 6.1.12*

| **Test Case ID:** | TC_4.1 |
|---|---|
| **Associated Use Case:** | UC-4.1.1 |
| **Functionality:** | Explore Marketplace |
| **Description:** | Verify product listing fails with incomplete details. |
| **Actor:** | User |
| **Pre-Conditions** | 3. User is logged into the application. |

| | | | | |
|---|---|---|---|---|
| | | | 4. User is on the marketplace page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the marketplace page. | ➢ Click on the product listing button/form.<br>➢ Enter incomplete product details.<br>➢ Click "Submit." | System displays an error message indicating required fields are missing. | Pass |
| **Post-condition:** | | | Product listing is not created.<br>User is prompted to complete the required fields. | |

*Table 6.1.13*

## 6.2. Validation

**User Registration Validation Test Cases: -**

| **Test Case ID:** | | | TC_1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1 | |
| **Functionality:** | | | User Registration | |
| **Description:** | | | Verify successful user account creation with valid data. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User has accessed the registration page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the registration page. | ➢ Enter valid user details including name, email, password, and phone number.<br>➢ Submit the registration form. | User account is successfully created and a success message is displayed. | Pass |
| **Post-condition:** | | | User is able to log into the system using the newly created credentials. | |

*Table 6.2.1*

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_1.1 | |
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | User Registration | |
| **Description:** | | | Verify that registration fails with invalid data. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User has accessed the registration page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the registration page. | ➢ Enter invalid user details (e.g., incorrect email format, password less than 8 characters).<br>➢ Submit the registration form. | Error message "User registration failed" is displayed. | Pass |
| **Post-condition:** | | | User remains on the registration page or is prompted to correct the invalid information. | |

*Table 6.2.2*

**Password Validation Test Cases: -**

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_2 | |
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | Password Compliance | |
| **Description:** | | | Verify that the password meets complexity requirements. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is on the registration or password change page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the registration or login page. | ➢ Enter a password that meets required standards (minimum 8 characters, includes | Password is accepted and user proceeds to the next step. | Pass |

| | | | | |
|---|---|---|---|---|
| | | numbers, and special characters).<br>➢ Submit the form. | | |
| **Post-condition:** | | | User successfully registers or updates their password. | |

*Table 6.2.3*

| **Test Case ID:** | | | TC_2.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | Password Compliance | |
| **Description:** | | | Verify system rejects non-compliant passwords. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is on the registration or password change page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the registration or login page. | ➢ Enter a password that does not meet the complexity requirements.<br>➢ Submit the form. | Error message "Invalid Credentials" or "Password doesn't match" is displayed. | Pass |
| **Post-condition:** | | | User is prompted to enter a compliant password. | |

*Table 6.2.4*

**User Number Validation Test Cases: -**

| **Test Case ID:** | | | TC_3 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | Phone Number Validation | |
| **Description:** | | | Verify the phone number is validated correctly. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is on the registration or profile update page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the registration | ➢ Enter a valid 11-digit phone number | Phone number is accepted and user can proceed. | Pass |

| | | | | |
|---|---|---|---|---|
| | or login page. | including the country code.<br>➢ Submit the form. | | |
| **Post-condition:** | | | User successfully completes the registration or profile update. | |

*Table 6.2.5*

| **Test Case ID:** | | | TC_3.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | Phone Number Validation | |
| **Description:** | | | Verify error on invalid phone number input. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is on the registration or profile update page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the registration or login page. | ➢ Enter an invalid phone number (incorrect digits, missing country code).<br>➢ Submit the form. | Error message "Phone number is invalid" is displayed. | Pass |
| **Post-condition:** | | | User is prompted to correct the phone number. | |

*Table 6.2.6*

**OTP Validation Test Cases: -**

| **Test Case ID:** | | | TC_4 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | OTP Verification | |
| **Description:** | | | Ensure OTP is correctly validated during user verification. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User has requested an OTP during registration or login. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the OTP page. | ➢ Enter the received 6-character OTP correctly.<br>➢ Submit for verification. | OTP is accepted and user is authenticated successfully. | Pass |
| **Post-condition:** | | | User proceeds to the main GUI of the system. | |

*Table 6.2.7*

| **Test Case ID:** | | | TC_4.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | OTP Verification | |
| **Description:** | | | Verify system rejects incorrect OTP. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User has received an OTP. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the OTP page. | ➢ Enter an incorrect OTP.<br>➢ Submit for verification. | Error message "OTP verification failed" is displayed. | Pass |
| **Post-condition:** | | | User is either prompted to retry or request a new OTP. | |

*Table 6.2.8*

**Post Module Validation Test Cases: -**

| **Test Case ID:** | | | TC_5 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-2.2.1, UC-2.2.2, UC-2.2.3, UC-2.2.4, UC-2.2.5, UC-2.2.6, UC-2.2.7 | |
| **Functionality:** | | | Creating a Post | |
| **Description:** | | | Verify that users can successfully create posts with valid content. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is logged into the FarmConnect app.<br>User navigates to the post creation page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the post creation page. | ➢ Enter valid post content including text, images | Post is successfully created and visible on the user's profile or feed. | Pass |

| | | | | |
|---|---|---|---|---|
| | | (if applicable), and tags.<br>➢ Submit the post | | |
| **Post-condition:** | | | Post appears in relevant sections of the app, like the news feed. | |

*Table 6.2.9*

| **Test Case ID:** | | | TC_5.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-2.2.1, UC-2.2.2, UC-2.2.3, UC-2.2.4, UC-2.2.5, UC-2.2.6, UC-2.2.7 | |
| **Functionality:** | | | Post Validation | |
| **Description:** | | | Verify that creating a post with invalid or prohibited content fails. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is logged into the FarmConnect app.<br>User navigates to the post creation page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the post creation page. | ➢ Enter invalid content such as unsupported media formats or exceed character limits.<br>➢ Attempt to submit the post. | Error message is displayed, and post is not created. | Pass |
| **Post-condition:** | | | User remains on the post creation page to correct the input or is informed of restrictions. | |

*Table 6.2.10*

**Marketplace Module Validation Test Cases: -**

| **Test Case ID:** | TC_6 |
|---|---|
| **Associated Use Case:** | UC-4.2, UC-4.2.1, UC-4.2.2 |
| **Functionality:** | Listing a Product |
| **Description:** | Verify that users can list a product with all required details accurately. |

| | | | | |
|---|---|---|---|---|
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is logged into the FarmConnect app.<br>User navigates to the marketplace section to list a new product. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the marketplace page. | ➢ Enter all required product details (name, description, price, photos).<br>➢ Submit the product listing. | Product is successfully listed in the marketplace. | Pass |
| **Post-condition:** | | | Product appears in the marketplace for other users to view and purchase. | |

*Table 6.2.11*

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_6.1 | |
| **Associated Use Case:** | | | UC-4.2, UC-4.2.1, UC-4.2.2 | |
| **Functionality:** | | | Product Listing Validation | |
| **Description:** | | | Verify that listing a product without mandatory fields fails. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is logged into the FarmConnect app.<br>User navigates to the marketplace section to list a new product. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the marketplace page. | ➢ Leave mandatory fields empty (like price or description).<br>➢ Attempt to submit the product listing. | Error message is displayed, and product is not listed. | Pass |
| **Post-condition:** | | | User is prompted to fill in all required fields before proceeding. | |

*Table 6.2.12*

**ChatBot Module Validation Test Cases: -**

| **Test Case ID:** | | | TC_7 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-3.1.1 | |
| **Functionality:** | | | ChatBot Interaction | |
| **Description:** | | | Verify that the ChatBot responds accurately to user queries. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is logged into the FarmConnect app.<br>User accesses the ChatBot feature. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the ChatBot page. | ➢ Enter various user queries related to FarmConnect services (like "How do I list a product?" or "Help with my account").<br>➢ Review the responses from ChatBot. | ChatBot provides relevant and correct responses to the queries. | Pass |
| **Post-condition:** | | | User gains useful information or guidance from the ChatBot. | |

*Table 6.2.13*

| **Test Case ID:** | | | TC_7.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-3.1.1 | |
| **Functionality:** | | | ChatBot Error Handling | |
| **Description:** | | | Verify that the ChatBot handles unrecognized queries gracefully. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is logged into the FarmConnect app.<br>User accesses the ChatBot feature. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the ChatBot page. | ➢ Enter queries that the ChatBot is | ChatBot displays a helpful error message like "Sorry, no response found". | Pass |

| | | | | |
|---|---|---|---|---|
| | | unlikely to understand.<br>➢ Observe the responses. | | |
| **Post-condition:** | | | User is either given an option to rephrase the query or directed to contact human support. | |

*Table 6.2.14*

**API Key Validation Test Cases: -**

| **Test Case ID:** | | | TC_8 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-2.1.1, UC-2.2.8, UC-5.1.3 | |
| **Functionality:** | | | API Key Generation | |
| **Description:** | | | Verify that API keys are correctly generated for authorized users. | |
| **Actor:** | | | Admin | |
| **Pre-Conditions** | | | User has administrative privileges necessary to generate API keys.<br>User is logged into the admin dashboard of the FarmConnect app. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the API management section. | ➢ Request a new API key generation.<br>➢ Confirm the generation process. | A new API key is generated and displayed or sent to the user securely. | Pass |
| **Post-condition:** | | | API key is added to the user's account or specific application configuration. | |

*Table 6.2.15*

| **Test Case ID:** | | | TC_8.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-2.1.1, UC-2.2.8, UC-5.1.3 | |
| **Functionality:** | | | API Key Validation | |
| **Description:** | | | Verify that the system accepts only valid API keys for accessing protected resources. | |
| **Actor:** | | | Admin | |
| **Pre-Conditions** | | | API key has been generated and is ready for use.<br>User attempts to access a resource requiring API key authentication. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the API management section. | ➢ Enter a valid API key along with the request to the protected resource.<br>➢ Submit the request. | Access is granted to the resource. | Pass |
| **Post-condition:** | | | User successfully accesses the resource. | |

*Table 6.2.16*

| **Test Case ID:** | | | TC_8.2 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-2.1.1, UC-2.2.8, UC-5.1.3 | |
| **Functionality:** | | | API Key Invalidity Handling | |
| **Description:** | | | Verify that the system correctly handles invalid or expired API keys. | |
| **Actor:** | | | Admin | |
| **Pre-Conditions** | | | User possesses an invalid or expired API key.<br>User attempts to access a resource requiring API key authentication. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the API management section. | ➢ Enter an invalid or expired API key along with the request to the protected resource.<br>➢ Submit the request. | Access to the resource is denied, and a clear error message is displayed. | Pass |
| **Post-condition:** | | | User is informed about the invalid API key and possibly guided on how to obtain a valid one. | |

*Table 6.2.17*

| **Test Case ID:** | TC_8.3 |
|---|---|
| **Associated Use Case:** | UC-2.1.1, UC-2.2.8, UC-5.1.3 |
| **Functionality:** | API Key Revocation |
| **Description:** | Verify that revoked API keys are no longer able to access protected resources. |
| **Actor:** | Admin |

| Pre-Conditions | | | An API key has been revoked by an administrator.<br>User attempts to use the revoked API key to access a resource. | |
|---|---|---|---|---|
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the API management section. | ➢ Use a revoked API key to access a protected resource.<br>➢ Submit the request. | The request is denied and accompanied by a message indicating the key's revocation. | Pass |
| **Post-condition:** | | | The system correctly prevents access with the revoked API key. | |

*Table 6.2.18*

## 6.3. Usability Testing

**User Registration/Login: -**

| Test Case ID: | | | TC_1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | User Registration | |
| **Description:** | | | Ensure that users can register successfully without encountering any errors. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is not logged into the application.<br>Registration page is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the registration page. | ➢ Enter a valid username.<br>➢ Enter a valid phone number.<br>➢ Enter a valid password.<br>➢ Confirm the password.<br>➢ Click "Register".<br>➢ Enter the received OTP. | User is registered successfully, and clear error messages are displayed for any invalid input. | Pass |

| | | | | |
|---|---|---|---|---|
| | | ➢ Click "Verify". | | |
| **Post-condition:** | | | User account is created, and user is redirected to the login page. | |

*Table 6.3.1*

| **Test Case ID:** | | | TC_1.1 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | User Login | |
| **Description:** | | | Ensure that users can log in successfully with valid credentials. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is not logged into the application. Login page is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the login page. | ➢ Enter the registered phone number.<br>➢ Enter the correct password.<br>➢ Click "Login". | User is logged in successfully, and clear error messages are displayed for invalid credentials. | Pass |
| **Post-condition:** | | | User session is active, and user is redirected to the dashboard. | |

*Table 6.3.2*

**User Number: -**

| **Test Case ID:** | | | TC_2 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | User Number Validation | |
| **Description:** | | | Ensure that users can enter their phone number correctly. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is not logged into the application.<br>Registration page is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the login/registration page. | ➢ Enter a phone number. | User can enter the phone number correctly, and clear error messages are displayed for | Pass |

| | | | | |
|---|---|---|---|---|
| | | | incorrect or invalid phone numbers. | |
| **Post-condition:** | | | Phone number is validated, and user can proceed with the registration. | |

*Table 6.3.3*

**OTP: -**

| **Test Case ID:** | | | TC_3 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-1.1.1, UC-1.1.2 | |
| **Functionality:** | | | OTP Verification | |
| **Description:** | | | Ensure that OTP is delivered promptly and successfully verified. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User has initiated the registration process.<br>User has entered their phone number and received an OTP. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the login/registration page. | ➢ Enter the received OTP.<br>➢ Click "Verify". | OTP is verified successfully, and clear error messages are displayed for any issues with OTP delivery or verification. | Pass |
| **Post-condition:** | | | User proceeds with the registration process upon successful OTP verification. | |

*Table 6.3.4*

**API Key: -**

| **Test Case ID:** | | | TC_4 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-2.1.1, UC-2.2.8, UC-5.1.3 | |
| **Functionality:** | | | API Key Management | |
| **Description:** | | | Ensure API keys facilitate secure and authorized communication. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User has administrative privileges.<br>User is logged into the admin dashboard. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the API management section. | ➢ Request a new API key. | API key is generated successfully, and clear instructions are provided on its use. | Pass |
| **Post-condition:** | | | API key is available for use, ensuring secure communication between components. | |

*Table 6.3.5*

**ChatBot: -**

| **Test Case ID:** | | | TC_5 | |
|---|---|---|---|---|
| **Associated Use Case:** | | | UC-3.1.1 | |
| **Functionality:** | | | ChatBot Response | |
| **Description:** | | | Ensure the ChatBot responds effectively to user queries. | |
| **Actor:** | | | User | |
| **Pre-Conditions** | | | User is logged into the application. ChatBot is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the ChatBot section. | ➢ Interact with the ChatBot by asking a query. | ChatBot provides useful information or guidance, and clear error messages are displayed if it fails to respond appropriately. | Pass |
| **Post-condition:** | | | User receives appropriate responses from the ChatBot, enhancing user engagement. | |

*Table 6.3.6*

## 6.4. Module / Unit Testing

**User Profiles and Networking: -**

| **Test Case ID:** | | | TC_1 | |
|---|---|---|---|---|
| **Functionality:** | | | Profile Creation | |
| **Testing Approach: -** | | | Verify that user profiles can be created with all required information. | |
| **Pre-Conditions** | | | User is not logged in, and the registration page is accessible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the registration page | Profile is created successfully. | Fill in all required fields, and submit the form. | Pass |
| **Post-condition:** | | | User account is created in the system. | |

*Table 6.4.1*

| **Test Case ID:** | | | TC_1.1 | |
|---|---|---|---|---|
| **Functionality:** | | | Profile Editing | |
| **Testing Approach: -** | | | Test profile editing features to ensure changes are saved correctly. | |
| **Pre-Conditions** | | | User is logged in and on their profile page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the update profile section. | Edit profile details and save changes. | Changes are saved and displayed correctly. | Pass |
| **Post-condition:** | | | Updated profile information is stored in the system. | |

*Table 6.4.2*

| **Test Case ID:** | | | TC_1.2 | |
|---|---|---|---|---|
| **Functionality:** | | | Networking | |
| **Testing Approach: -** | | | Check if networking features work as expected, such as sending connection requests and viewing connections. | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Group communities/Chatting/ChatBot/Respond to Story or Post/Comment section. | Send a connection request to another user, accept a connection request, view connections list. | Connection actions perform as expected, connections are listed correctly. | Pass |

| | |
|---|---|
| **Post-condition:** | Connection relationships are updated in the system. |

*Table 6.4.3*

**Groups and Communities: -**

| **Test Case ID:** | | | TC_2 | |
|---|---|---|---|---|
| **Functionality:** | | | Group Creation | |
| **Testing Approach: -** | | | Verify that groups can be created with appropriate settings (privacy, visibility). | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group creation page. | Enter group details and select privacy settings, and create the group. | Group is created with specified settings. | Pass |
| **Post-condition:** | | | Group is saved in the database. | |

*Table 6.4.4*

| **Test Case ID:** | | | TC_2.1 | |
|---|---|---|---|---|
| **Functionality:** | | | Joining/Leaving Groups | |
| **Testing Approach: -** | | | Test joining and leaving groups. | |
| **Pre-Conditions** | | | User is logged in and group exists. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group community's page. | Join an existing group, leave the group. | User can join and leave groups successfully. | Pass |
| **Post-condition:** | | | Membership status is updated. | |

*Table 6.4.5*

| **Test Case ID:** | | | TC_2.2 | |
|---|---|---|---|---|
| **Functionality:** | | | Group Interaction | |
| **Testing Approach: -** | | | Ensure group interaction features like posting, commenting, and liking work as intended. | |
| **Pre-Conditions** | | | User is logged in and is a group member. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the group community's page. | Post a message in the group, comment on a post, like a post/comment. | Posts, comments, and likes are processed and displayed correctly. | Pass |
| **Post-condition:** | | | Interactions are recorded in the database. | |

*Table 6.4.6*

**Messaging and Collaboration: -**

| **Test Case ID:** | | | TC_3 | |
|---|---|---|---|---|
| **Functionality:** | | | Individual Messaging | |
| **Testing Approach: -** | | | Verify that messages can be sent and received between users | |
| **Pre-Conditions** | | | Both users are logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group community's / chatting page. | Open a chat with another user, send a message. | Messages are sent and received correctly. | Pass |
| **Post-condition:** | | | Messages are stored in the database. | |

*Table 6.4.7*

| **Test Case ID:** | | | TC_3.1 | |
|---|---|---|---|---|
| **Functionality:** | | | Group Messaging | |
| **Testing Approach: -** | | | Test group messaging functionality, including adding/removing participants and message delivery. | |
| **Pre-Conditions** | | | User is logged in and has a group chat | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group community's / chatting page. | Add participants, send messages, remove participants. | Group chat functionalities work as expected. | Pass |
| **Post-condition:** | | | Group chat data is updated in the database. | |

*Table 6.4.8*

**Search Functionality, Notifications, and Alerts: -**

| **Test Case ID:** | TC_4 |
|---|---|
| **Functionality:** | Notifications |

| | | | | |
|---|---|---|---|---|
| **Testing Approach: -** | | | Verify that notifications are triggered for relevant events (e.g., new messages, group invitations). | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the notification's section. | Perform actions that trigger notifications (e.g., receive a new message). | Notifications are accurate and timely. | Pass |
| **Post-condition:** | | | Notifications are recorded in the system. | |

*Table 6.4.9*

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_4.1 | |
| **Functionality:** | | | Search Functionality | |
| **Testing Approach: -** | | | Test search functionality for finding products/services, users, and posts. | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the any section/page of FarmConnect. | Use the search feature to find products/services, users, and posts. | Search results are relevant and accurate. | Pass |
| **Post-condition:** | | | Search results are displayed. | |

*Table 6.4.10*

**Weather Updates and Crop Monitoring: -**

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_5 | |
| **Functionality:** | | | Weather Data | |
| **Testing Approach: -** | | | Verify that weather data is accurately fetched and displayed. | |
| **Pre-Conditions** | | | User is logged in and, on the weather, (icon) updates page. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the weather icon. | Access the weather updates section, verify the accuracy of weather data. | Weather data is accurate and timely. | Pass |
| **Post-condition:** | | | Weather data is displayed correctly. | |

*Table 6.4.11*

| Test Case ID: | | | TC_5.1 | |
|---|---|---|---|---|
| **Functionality:** | | | Crop Monitoring | |
| **Testing Approach: -** | | | Test crop monitoring features like setting up alerts for specific conditions. | |
| **Pre-Conditions** | | | User is logged in and on the crop monitoring page (main landing posts, stories page). | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the landing page. | Set up alerts for specific crop conditions. | Alerts are set up correctly and triggered as expected. | Pass |
| **Post-condition:** | | | Alerts are stored in the database. | |

*Table 6.4.12*

**FarmConnect Web Dashboard: -**

| Test Case ID: | | | TC_6 | |
|---|---|---|---|---|
| **Functionality:** | | | Admin Functionalities | |
| **Testing Approach: -** | | | Verify that admin functionalities like user management, content moderation, and analytics work as expected. | |
| **Pre-Conditions** | | | Admin user is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the dashboard page. | Access user management, content moderation, and analytics sections, perform actions like adding, editing, and deleting users, and moderate content. | Admin functionalities work correctly. | Pass |
| **Post-condition:** | | | Changes are reflected in the database. | |

*Table 6.4.13*

| Test Case ID: | | | TC_6.1 | |
|---|---|---|---|---|
| **Functionality:** | | | User Features | |
| **Testing Approach: -** | | | Test user features accessible through the web dashboard. | |
| **Pre-Conditions** | | | Regular user is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the dashboard page. | Access various features available to regular users through the web dashboard. | User features work seamlessly. | Pass |
| **Post-condition:** | | | User interactions are recorded. | |

*Table 6.4.14*

## 6.5. Integration Testing

**Web Dashboard API Key Integration: -**

| **Test Case ID:** | | | TC_1 | |
|---|---|---|---|---|
| **Functionality:** | | | Admin Dashboard | |
| **Testing Approach: -** | | | Integrate API keys into the web dashboard for secure communication between the frontend and backend systems. | |
| **Pre-Conditions** | | | Admin is logged into the Admin Dashboard. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the dashboard page. | Perform actions like managing users, accessing data, and modifying user activities. | Actions are performed correctly, data is securely communicated and updated in the backend. | Pass |
| **Post-condition:** | | | Database reflects changes made through the Admin Dashboard. | |

*Table 6.5.1*

| **Test Case ID:** | | | TC_1.1 | |
|---|---|---|---|---|
| **Functionality:** | | | User Dashboard | |
| **Testing Approach: -** | | | Ensure both Admin Dashboard and User Dashboard functionalities are correctly integrated. | |
| **Pre-Conditions** | | | User is logged into the User Dashboard. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the dashboard page. | View data, insert, update, or delete personal information | Data is displayed correctly, and CRUD operations are securely | Pass |

| | | | | |
|---|---|---|---|---|
| | | | communicated and processed. | |
| **Post-condition:** | | | User data is correctly reflected in the app and database. | |

*Table 6.5.2*

**ChatBot Integration: -**

| **Test Case ID:** | | | TC_2 | |
|---|---|---|---|---|
| **Functionality:** | | | ChatBot Integration | |
| **Testing Approach: -** | | | Integrate the ChatBot through URL API key to provide communication between the user and the system. | |
| **Pre-Conditions** | | | User is logged into the system. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the ChatBot page. | Initiate a conversation with the ChatBot, ask questions, and request assistance. | ChatBot responds accurately and provides relevant information. | Pass |
| **Post-condition:** | | | User receives correct responses, and interaction data is stored. | |

*Table 6.5.3*

**Weather API Key Integration: -**

| **Test Case ID:** | | | TC_3 | |
|---|---|---|---|---|
| **Functionality:** | | | Weather API Key Integration | |
| **Testing Approach: -** | | | Incorporate weather data APIs to provide real-time weather updates and crop monitoring information. | |
| **Pre-Conditions** | | | User is logged into the platform. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the weather icon. | Access the weather updates section. | Accurate weather data is fetched and displayed in real-time. | Pass |
| **Post-condition:** | | | Users view up-to-date weather information for their location. | |

*Table 6.5.4*

**Notifications and Alerts: -**

| **Test Case ID:** | | | TC_4 | |
|---|---|---|---|---|
| **Functionality:** | | | Messaging Notifications: | |
| **Testing Approach: -** | | | Connect the notification system with various modules like messaging, marketplace, and user profiles. | |
| **Pre-Conditions** | | | User has messaging enabled. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the messaging, marketplace, and user profiles section. | Send and receive messages. | User receives notifications for new messages promptly. | Pass |
| **Post-condition:** | | | Notifications are logged and displayed. | |

*Table 6.5.5*

| **Test Case ID:** | | | TC_4.1 | |
|---|---|---|---|---|
| **Functionality:** | | | Marketplace Notifications | |
| **Testing Approach: -** | | | Connect the notification system with various modules like messaging, marketplace, and user profiles. | |
| **Pre-Conditions** | | | User is active in the marketplace. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the messaging, marketplace, and user profiles section. | List a new product or receive an invitation. | User receives relevant notifications. | Pass |
| **Post-condition:** | | | Marketplace activities are notified. | |

*Table 6.5.6*

| **Test Case ID:** | | | TC_4.2 | |
|---|---|---|---|---|
| **Functionality:** | | | Profile Updates | |
| **Testing Approach: -** | | | Connect the notification system with various modules like messaging, marketplace, and user profiles. | |
| **Pre-Conditions** | | | User is logged into their profile. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the messaging, | Update profile information. | Notification is sent for profile updates. | Pass |

| | | | | |
|---|---|---|---|---|
| | marketplace, and user profiles section. | | | |
| **Post-condition:** | | | Profile changes are notified. | |

*Table 6.5.7*

**Search Functionality Integration: -**

| **Test Case ID:** | | | TC_5 | |
|---|---|---|---|---|
| **Functionality:** | | | 1. Marketplace Search<br>2. Group Search<br>3. User Profile Search | |
| **Testing Approach: -** | | | Integrate search capabilities into modules like marketplace, groups, and user profiles. | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the marketplace search section. | Search for products or services. | Relevant search results are displayed. | Pass |
| 2. | Navigate to the group community search section. | Search for groups | Groups matching the search criteria are displayed. | Pass |
| 3. | Navigate to the user profile search section. | Search for other users. | Relevant user profiles are displayed. | Pass |
| **Post-condition:** | | | 1. Search logs are updated.<br>2. Group search results are shown.<br>3. User search results are shown. | |

*Table 6.5.8*

**Chatting Integration: -**

| **Test Case ID:** | TC_6 |
|---|---|
| **Functionality:** | 1. User Chatting<br>2. ChatBot Interaction<br>3. Chat Notifications |
| **Testing Approach: -** | Connect the chatting module with user profiles, messaging systems, and notification systems. |

| **Pre-Conditions** | | | 1. Users are logged in.<br>2. User initiates a chat with ChatBot.<br>3. Chatting module is enabled. | |
|---|---|---|---|---|
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the chatting section. | Start a chat with another user | Messages are sent and received correctly. | Pass |
| 2. | Navigate to the ChatBot section. | Ask queries and request assistance. | ChatBot provides accurate and helpful responses. | Pass |
| 3. | Navigate to the chat notifications section. | Receive new messages. | Notifications are triggered for new messages. | Pass |
| **Post-condition:** | | | 1. Chat history is updated.<br>2. Interaction is logged.<br>3. Users are notified of new messages. | |

*Table 6.5.9*

## 6.6. System Testing

**User Profiles and Networking: -**

| **Test Case ID:** | | | TC_1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Create User Profile | |
| **Pre-Conditions** | | | User is registered and logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the profile creation page. | Fill in all required fields (name, email, phone number, etc.). Submit the profile. | User profile is created successfully, and confirmation message is displayed. | Pass |
| **Post-condition:** | | | User profile data is saved in the database. | |

*Table 6.6.1*

| **Test Case ID:** | | | TC_1.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Edit User Profile | |
| **Pre-Conditions** | | | User profile is already created. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the profile editing page. | Make changes to the profile information. Save the changes. | Changes are saved successfully, and updated information is displayed. | Pass |
| **Post-condition:** | | | Updated profile data is saved in the database. | |

*Table 6.6.2*

| **Test Case ID:** | | | TC_1.2 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Send Connection Request | |
| **Pre-Conditions** | | | User is logged in and another user's profile is visible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to another user's profile. | Click on the "Connect" button. | Connection request is sent successfully, and confirmation message is displayed. | Pass |
| **Post-condition:** | | | Connection request is recorded in the database. | |

*Table 6.6.3*

**Groups and Communities: -**

| **Test Case ID:** | | | TC_2 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Create Group | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group creation page. | Fill in the group name and settings (privacy, visibility). Submit the group creation form. | Group is created successfully, and confirmation message is displayed. | Pass |
| **Post-condition:** | | | Group data is saved in the database. | |

*Table 6.6.4*

| **Test Case ID:** | | | TC_2.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Join Group | |
| **Pre-Conditions** | | | Group is created and visible to the user. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group page. | Click on the "Join" button. | User joins the group successfully, and confirmation message is displayed. | Pass |

| | |
|---|---|
| **Post-condition:** | User is added to the group members list in the database. |

*Table 6.6.5*

| **Test Case ID:** | | | TC_2.2 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Post in Group | |
| **Pre-Conditions** | | | User is a member of the group. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group page. | Create a new post. Submit the post. | Post is created successfully, and visible in the group feed. | Pass |
| **Post-condition:** | | | Post data is saved in the database. | |

*Table 6.6.6*

**Messaging and Collaboration: -**

| **Test Case ID:** | | | TC_3 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Send Message | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the chatting page. | Select a contact. Type a message and send it. | Message is sent successfully, and visible in the chat history. | Pass |
| **Post-condition:** | | | Message data is saved in the database. | |

*Table 6.6.7*

| **Test Case ID:** | | | TC_3.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Group Messaging | |
| **Pre-Conditions** | | | User is part of a group. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group messaging section. | Type a message and send it to the group. | Message is sent successfully, and visible in the group chat history | Pass |
| **Post-condition:** | | | Group message data is saved in the database. | |

*Table 6.6.8*

**Search Functionality, Notifications, and Alerts: -**

| | |
|---|---|
| **Test Case ID:** | TC_4 |
| **Test Case Name:** | Search for Products |

| **Pre-Conditions** | | | User is logged in. | |
|---|---|---|---|---|
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the search bar. | Type in the product name.<br>Execute the search. | Relevant products are displayed in the search results. | Pass |
| **Post-condition:** | | | Search results are displayed. | |

*Table 6.6.9*

| **Test Case ID:** | | | TC_4.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Receive Notifications | |
| **Pre-Conditions** | | | User has notifications enabled. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the search bar. | Trigger an event (e.g., receive a new message, group invitation). | User receives a notification for the event. | Pass |
| **Post-condition:** | | | Notification is displayed and logged. | |

*Table 6.6.10*

**Weather Updates: -**

| **Test Case ID:** | | | TC_5 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Fetch Weather Data | |
| **Pre-Conditions** | | | User is logged in | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the weather updates section. | Updated weather will be displayed according to region/place. | Accurate and real-time weather data is displayed. | Pass |
| **Post-condition:** | | | Weather data is visible to the user | |

*Table 6.6.11*

**FarmConnect Web Dashboard: -**

| **Test Case ID:** | | | TC_6 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Admin User Management | |
| **Pre-Conditions** | | | Admin is logged into the dashboard. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the user management section. | Add, update, or delete a user. | User management actions are performed successfully. | Pass |
| **Post-condition:** | | | User data is updated in the database. | |

*Table 6.6.12*

| **Test Case ID:** | | | TC_6.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | User Access to Features | |
| **Pre-Conditions** | | | User is logged into the web dashboard. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the user management section. | Navigate to various user features (e.g., profile, marketplace). | User can access and use features successfully. | Pass |
| **Post-condition:** | | | User actions are reflected in the app and database. | |

*Table 6.6.13*

## 6.7. Acceptance Testing

**User Profiles and Networking: -**

| **Test Case ID:** | | | TC_1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Create User Profile | |
| **Pre-Conditions** | | | User is registered and logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the profile creation page. | Fill in all required fields (name, email, phone number, etc.). Submit the profile. | User profile is created successfully with a confirmation message. | Pass |
| **Acceptance Criteria:** | | | User profile data is correctly saved in the database, and the profile is displayed accurately. | |

*Table 6.7.1*

| **Test Case ID:** | | | TC_1.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Edit User Profile | |
| **Pre-Conditions** | | | User profile is already created. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the profile editing page. | Make changes to the profile information. Save the changes. | Changes are saved successfully and displayed correctly. | Pass |
| **Acceptance Criteria:** | | | Updated profile data is saved in the database and displayed accurately on the user's profile page. | |

*Table 6.7.2*

| **Test Case ID:** | | | TC_1.2 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Send Connection Request | |
| **Pre-Conditions** | | | User is logged in and another user's profile is visible. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to another user's profile. | Click on the "Connect" button. | Connection request is sent successfully with a confirmation message. | Pass |
| **Acceptance Criteria:** | | | Connection request is recorded in the database, and the receiving user is notified. | |

*Table 6.7.3*

**Groups and Communities: -**

| **Test Case ID:** | | | TC_2 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Create Group | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to group creation page. | Fill in the group name and settings (privacy, visibility). Submit the group creation form. | Group is created successfully with a confirmation message. | Pass |
| **Acceptance Criteria:** | | | Group data is saved in the database, and the group is visible to the creator and eligible users. | |

*Table 6.7.4*

| **Test Case ID:** | | | TC_2.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Join Group | |
| **Pre-Conditions** | | | Group is created and visible to the user. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to group page. | Click on the "Join" button. | User joins the group successfully with a confirmation message. | Pass |
| **Acceptance Criteria:** | | | User is added to the group members list in the database, and the group appears in the user's list of joined groups. | |

*Table 6.7.5*

| **Test Case ID:** | | | TC_2.2 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Post in Group | |
| **Pre-Conditions** | | | User is a member of the group. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to group page. | Create a new post. Submit the post. | Post is created successfully and visible in the group feed. | Pass |
| **Acceptance Criteria:** | | | Post data is saved in the database, and the post is displayed correctly in the group feed. | |

*Table 6.7.6*

**Messaging and Collaboration: -**

| **Test Case ID:** | | | TC_3 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Send Message | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the messaging page. | Select a contact. Type a message and send it. | Message is sent successfully and visible in the chat history. | Pass |
| **Acceptance Criteria:** | | | Message data is saved in the database, and the message is displayed correctly in the chat history. | |

*Table 6.7.7*

| **Test Case ID:** | | | TC_3.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Group Messaging | |
| **Pre-Conditions** | | | User is part of a group. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group messaging page. | Type a message and send it to the group. | Message is sent successfully and visible in the group chat history. | Pass |

| | |
|---|---|
| **Acceptance Criteria:** | Group message data is saved in the database, and the message is displayed correctly in the group chat history. |

*Table 6.7.8*

**Search Functionality, Notifications, and Alerts: -**

| **Test Case ID:** | | | TC_4 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Search for Products | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the search bar. | Type in the product name.<br>Execute the search. | Relevant products are displayed in the search results. | Pass |
| **Acceptance Criteria:** | | | Search results are accurate and relevant to the search query. | |

*Table 6.7.9*

| **Test Case ID:** | | | TC_4.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Receive Notifications | |
| **Pre-Conditions** | | | User has notifications enabled. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the search bar. | Trigger an event (e.g., receive a new message, group invitation). | User receives a notification for the event. | Pass |
| **Acceptance Criteria:** | | | Notifications are timely and accurately reflect the events. | |

*Table 6.7.10*

**Weather Updates: -**

| **Test Case ID:** | | | TC_5 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Receive Notifications | |
| **Pre-Conditions** | | | User is logged in. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the weather updates section. | Accurate and real-time weather data is displayed. | User receives a notification for the event. | Pass |
| **Acceptance Criteria:** | | | Notifications are timely and accurately reflect the events. | |

*Table 6.7.11*

**FarmConnect Web Dashboard: -**

| **Test Case ID:** | | | TC_6 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Admin User Management | |
| **Pre-Conditions** | | | Admin is logged into the dashboard. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the user management section. | Add, update, or delete a user. | User management actions are performed successfully. | Pass |
| **Acceptance Criteria:** | | | User data is updated correctly in the database. | |

*Table 6.7.12*

| **Test Case ID:** | | | TC_6.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | User Access to Features | |
| **Pre-Conditions** | | | User is logged into the web dashboard. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the user features. | Navigate to the various user features (e.g., profile, marketplace). | User can access and use features successfully. | Pass |
| **Acceptance Criteria:** | | | User features are accessible, functional, and data is displayed correctly. | |

*Table 6.7.13*

## 6.8. Stress Testing

**User Profiles and Networking: -**

| **Test Case ID:** | | | TC_1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Simultaneous User Profile Creation | |
| **Pre-Conditions** | | | System is running normally. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the user profile section. | Simulate 1000 users simultaneously creating profiles. Monitor system performance and response time. | System handles the load without crashing, and profiles are created successfully. | Pass |
| **Post Condition:** | | | Profiles are saved in the database, and the system remains responsive. | |

*Table 6.8.1*

| **Test Case ID:** | | | TC_1.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | High Volume Connection Requests | |
| **Pre-Conditions** | | | Multiple user profiles exist. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the user profile section. | Simulate 1000 users sending connection requests simultaneously. Monitor system performance and response time. | System handles the load without crashing, and connection requests are processed. | Pass |
| **Post Condition:** | | | Connection requests are recorded in the database, and the system remains responsive. | |

*Table 6.8.2*

**Groups and Communities: -**

| **Test Case ID:** | | | TC_2 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Mass Group Creation | |
| **Pre-Conditions** | | | System is running normally. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group and community section. | Simulate 500 users creating groups simultaneously. Monitor system performance and response time. | System handles the load without crashing, and groups are created successfully. | Pass |
| **Post Condition:** | | | Group data is saved in the database, and the system remains responsive. | |

*Table 6.8.3*

| **Test Case ID:** | | | TC_2.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | High Volume Group Interactions | |
| **Pre-Conditions** | | | Multiple groups and users exist. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the group and community section. | Simulate 1000 users posting, commenting, and liking in groups simultaneously. Monitor system performance and response time. | System handles the load without crashing, and group interactions are processed. | Pass |

| | |
|---|---|
| **Post Condition:** | Interaction data is saved in the database, and the system remains responsive. |

*Table 6.8.4*

**Messaging and Collaboration: -**

| **Test Case ID:** | | | TC_3 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Mass Messaging | |
| **Pre-Conditions** | | | System is running normally. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the chatting section. | Simulate 1000 users sending messages simultaneously. Monitor system performance and response time. | System handles the load without crashing, and messages are delivered. | Pass |
| **Post Condition:** | | | Message data is saved in the database, and the system remains responsive. | |

*Table 6.8.5*

| **Test Case ID:** | | | TC_3.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Group Messaging Spike | |
| **Pre-Conditions** | | | Multiple groups and users exist. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the chatting section. | Simulate 1000 users sending messages in group chats simultaneously. Monitor system performance and response time. | System handles the load without crashing, and group messages are delivered. | Pass |
| **Post Condition:** | | | Message data is saved in the database, and the system remains responsive. | |

*Table 6.8.6*

**Search Functionality, Notifications, and Alerts: -**

| **Test Case ID:** | | | TC_4 | |
|---|---|---|---|---|
| **Test Case Name:** | | | High Volume Searches | |
| **Pre-Conditions** | | | System is running normally. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the search bar section. | Simulate 1000 users performing searches simultaneously. Monitor system performance and response time. | System handles the load without crashing, and search results are returned. | Pass |
| **Post Condition:** | | | Search operations are logged, and the system remains responsive. | |

*Table 6.8.7*

| **Test Case ID:** | | | TC_4.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Mass Notifications | |
| **Pre-Conditions** | | | Multiple events are set up to trigger notifications. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the search bar section. | Simulate 1000 events triggering notifications simultaneously. Monitor system performance and response time. | System handles the load without crashing, and notifications are delivered. | Pass |
| **Post Condition:** | | | Notification data is logged, and the system remains responsive. | |

*Table 6.8.8*

**Weather Updates: -**

| **Test Case ID:** | | | TC_5 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Frequent Weather Data Fetch | |
| **Pre-Conditions** | | | System is running normally. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the weather icon section. | Simulate frequent weather data fetch requests (e.g., every second) from 1000 users. Monitor system performance and response time. | System handles the load without crashing, and weather data is fetched. | Pass |
| **Post Condition:** | | | Weather data is updated correctly, and the system remains responsive. | |

*Table 6.8.9*

**FarmConnect Web Dashboard: -**

| **Test Case ID:** | | | TC_6 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Admin Bulk User Management | |
| **Pre-Conditions** | | | Admin is logged into the dashboard. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Admin panel dashboard. | Simulate the admin performing bulk user management actions (e.g., updating 1000 users simultaneously). Monitor system performance and response time. | System handles the load without crashing, and user management actions are performed. | Pass |
| **Post Condition:** | | | User data is updated correctly, and the system remains responsive. | |

*Table 6.8.10*

| **Test Case ID:** | | | TC_6.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | High Volume User Access | |
| **Pre-Conditions** | | | System is running normally. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the User panel dashboard. | Simulate 1000 users accessing the web dashboard simultaneously. Monitor system performance and response time. | System handles the load without crashing, and users can access dashboard features. | Pass |
| **Post Condition:** | | | Dashboard remains accessible, and the system remains responsive. | |

*Table 6.8.11*

## 6.9. Hardware Configuration for Testing

**User Profiles and Networking: -**

| **Test Case ID:** | TC_1 |
|---|---|
| **Test Case Name:** | Profile Creation on Low-End Hardware |

| **Pre-Conditions** | | | FarmConnect is installed on a low-end device (e.g., 2GB RAM, dual-core processor). | |
|---|---|---|---|---|
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the user profile section. | Attempt to create a new user profile. Monitor the time taken and system performance. | Profile is created successfully without significant lag or crashes. | Pass |
| **Post Condition:** | | | Profile data is saved, and the system remains responsive. | |

*Table 6.9 1*

| **Test Case ID:** | | | TC_1.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Networking Features on High-End Hardware | |
| **Pre-Conditions** | | | FarmConnect is installed on a high-end device (e.g., 16GB RAM, quad-core processor). | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the user profile section. | Test sending connection requests and viewing connections. Monitor the time taken and system performance. | Networking features perform seamlessly with fast response times. | Pass |
| **Post Condition:** | | | Connections are made, and the system remains highly responsive. | |

*Table 6.9 2*

**Groups and Communities: -**

| **Test Case ID:** | | | TC_2 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Group Creation on Different Hardware | |
| **Pre-Conditions** | | | FarmConnect is installed on both low-end and high-end devices | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Groups and Communities section. | Create groups on both types of devices. Monitor the time taken and system performance. | Groups are created successfully on both types of devices, with acceptable performance levels. | Pass |

| | |
|---|---|
| **Post Condition:** | Group data is saved, and the system remains responsive on both devices. |

*Table 6.9 3*

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_2.1 | |
| **Test Case Name:** | | | Group Interactions on Low-End Hardware | |
| **Pre-Conditions** | | | FarmConnect is installed on a low-end device. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Groups and Communities section. | Post, comment, and like within groups. Monitor the time taken and system performance. | Group interactions perform adequately with minor delays, if any. | Pass |
| **Post Condition:** | | | Interaction data is saved, and the system remains responsive. | |

*Table 6.9 4*

**Messaging and Collaboration: -**

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_3 | |
| **Test Case Name:** | | | Message Sending on Low-End Hardware | |
| **Pre-Conditions** | | | FarmConnect is installed on a low-end device. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Chatting section. | Send messages to other users. Monitor the time taken and system performance. | Messages are sent and received with minimal delays. | Pass |
| **Post Condition:** | | | Message data is saved, and the system remains responsive. | |

*Table 6.9 5*

| | | | | |
|---|---|---|---|---|
| **Test Case ID:** | | | TC_3.1 | |
| **Test Case Name:** | | | Group Messaging on High-End Hardware | |
| **Pre-Conditions** | | | FarmConnect is installed on a high-end device. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the Chatting section. | Test group messaging functionality. Monitor the time taken and system performance. | Group messages are sent and received instantly with no delays. | Pass |
| **Post Condition:** | | | Message data is saved, and the system remains highly responsive. | |

*Table 6.9 6*

**Search Functionality, Notifications, and Alerts: -**

| **Test Case ID:** | | | TC_4 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Search Performance on Low-End Hardware | |
| **Pre-Conditions** | | | FarmConnect is installed on a low-end device. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Chatting section. | Perform various search queries. Monitor the time taken and system performance. | Search results are returned within an acceptable time frame. | Pass |
| **Post Condition:** | | | Search data is processed, and the system remains responsive. | |

*Table 6.9 7*

| **Test Case ID:** | | | TC_4.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Notification Delivery on Different Hardware Configurations | |
| **Pre-Conditions** | | | FarmConnect is installed on both low-end and high-end devices. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Chatting section. | Trigger notifications for various events. Monitor the time taken and system performance on both types of devices. | Notifications are delivered promptly on both types of devices. | Pass |
| **Post Condition:** | | | Notification data is logged, and the system remains responsive on both devices. | |

*Table 6.9 8*

**Weather Updates: -**

| **Test Case ID:** | | | TC_5 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Weather Data Fetching on Low-End Hardware | |
| **Pre-Conditions** | | | FarmConnect is installed on a low-end device. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the weather icon section. | Fetch real-time weather updates. Monitor the time taken and system performance. | Weather data is fetched and displayed within an acceptable time frame. | Pass |
| **Post Condition:** | | | Weather data is updated, and the system remains responsive. | |

*Table 6.9 9*

**FarmConnect Web Dashboard: -**

| **Test Case ID:** | | | TC_6 | |
|---|---|---|---|---|
| **Test Case Name:** | | | Admin Dashboard Operations on Low-End Hardware | |
| **Pre-Conditions** | | | Admin dashboard is accessed on a low-end device. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| 1. | Navigate to the Admin Dashboard section. | Perform various admin operations (e.g., user management, content moderation). Monitor the time taken and system performance. | Admin operations are performed with minimal delays. | Pass |
| **Post Condition:** | | | Admin data is processed, and the system remains responsive. | |

*Table 6.9 10*

| **Test Case ID:** | | | TC_6.1 | |
|---|---|---|---|---|
| **Test Case Name:** | | | User Dashboard Access on High-End Hardware | |
| **Pre-Conditions** | | | User dashboard is accessed on a high-end device. | |
| **Sr. No.** | **Action** | **Expected Results** | **Actual Results** | **Pass/Fail** |

| | | | | |
|---|---|---|---|---|
| 1. | Navigate to the User Dashboard section. | Test all features accessible through the user dashboard. Monitor the time taken and system performance. | User dashboard features are accessed and used seamlessly. | Pass |
| **Post Condition:** | | | User data is processed, and the system remains highly responsive. | |

*Table 6.9 11*

## 6.10. Evaluation

The provided text effectively describes the evaluation process of the FarmConnect project. Here's a breakdown of the key points:

- **Rigorous Evaluation:** It emphasizes that FarmConnect has been evaluated throughout the development lifecycle, not just at the end.
- **Testing Methods:** Different testing methods are mentioned, including unit testing, integration testing, system testing, acceptance testing, stress testing, and hardware configuration assessments. This indicates a comprehensive approach to evaluation.
- **Focus Areas:** The text highlights that all modules, from core functionalities to the web dashboard, have been evaluated for functionality, integration, and performance.
- **Positive Outcomes:** The evaluation demonstrates FarmConnect's ability to handle data volume, user interactions, and deliver accurate information while being user-friendly. This suggests the platform is ready for real-world use.
- **Overall Impact:** FarmConnect is presented as a well-tested and dependable solution for the agricultural community.

Here are some **improvements** you can consider:

- ✓ **Specific Examples:** Providing specific examples of issues identified and resolved during testing would strengthen the evaluation's credibility.
- ✓ **Evaluation Metrics:** Mentioning any metrics used to measure performance (e.g., response time, data accuracy) would add a quantitative element.
- ✓ **User Feedback:** Highlighting how user acceptance testing or usability testing contributed to the evaluation could emphasize the user-centric approach.

Overall, the text effectively conveys the thoroughness of FarmConnect's evaluation process and its positive outcomes. With minor additions, it can be even more impactful.

## 6.11. Deployment

Deploying FarmConnect involves transitioning the system from the testing and evaluation phase to a production environment where it can be used by farmers and agricultural experts. Here's how deployment can be approached with a focus on testing and evaluation:

## Pre-Deployment

I. **Evaluation Review:** Conduct a final review of the evaluation results, ensuring all critical functionality, performance, and security aspects are addressed.

II. **User Training Materials:** Develop comprehensive user training materials (video tutorials, user manuals) to guide farmers on using FarmConnect effectively.

III. **Pilot Testing (Optional):** Consider a pilot deployment in a limited region with a smaller group of farmers to gather real-world usage data and identify any unforeseen issues.

IV. **Server and Infrastructure Setup:** Provision the necessary server infrastructure (cloud or on-premise) based on scalability requirements and non-functional requirements (security, uptime). This may involve configuring load balancers, firewalls, and databases.

V. **App Store Deployment:** Prepare the FarmConnect mobile app for deployment on the Google Play Store (Android). Ensure compliance with store guidelines and provide clear app descriptions, screenshots, and keywords for discoverability.

VI. **Web Dashboard Deployment:** Deploy the FarmConnect web dashboard on the chosen web server. Configure security measures like SSL certificates and user authentication.

VII. **Data Migration (if applicable):** If pilot testing involved data collection, migrate that data to the production environment securely.

VIII. **Marketing and User Acquisition Strategy:** Develop a marketing strategy to target farmers and promote FarmConnect's benefits. This could involve social media campaigns, agricultural community outreach, and partnerships with agricultural organizations.

## Deployment Procedure

I. **Cloud Hosting:** The document mentions cloud infrastructure as a requirement. Exclamation This suggests FarmConnect might be deployed on a cloud platform like Google Cloud Platform (GCP), Amazon Web Services (AWS), or Microsoft Azure. Cloud deployment offers scalability, reliability, and easier maintenance compared to on-premise servers.

II. **Mobile App Store Deployment:** The FarmConnect mobile application will likely be deployed on app stores like Google Play Store (for Android) and potentially the Apple App Store (if iOS is supported). This ensures users can easily discover and download the app.

III. **Web Dashboard Deployment:** The Laravel-based web dashboard can potentially be deployed on the same cloud platform where the backend logic resides. This would allow for centralized management and easier integration between the mobile app and web platform functionalities.

Here are some additional details that would be helpful to know about FarmConnect's deployment:

I. **Specific Cloud Platform:** Knowing the chosen cloud platform (GCP, AWS, Azure) would provide a clearer picture of the deployment environment.

II. **Security Measures:** The document mentions security as a non-functional requirement. It would be beneficial to understand the specific security measures implemented during deployment to protect user data and ensure platform security.

III. **Load Balancing:** For a platform expecting high user volumes, information on load balancing strategies would be relevant. Load balancing distributes incoming traffic across multiple servers, ensuring smooth performance even under peak loads.

## Post-Deployment

I. **Iterative Improvement:** Based on user feedback, evaluation data, and identified issues, release new versions of the app and web dashboard with bug fixes, enhancements, and new features. This emphasizes the Agile development approach mentioned earlier.

II. **Content Moderation:** Implement a content moderation strategy to ensure the quality and appropriateness of user-generated content within FarmConnect.

III. **Community Building:** Foster a sense of community within FarmConnect by encouraging user interactions, discussions, and knowledge sharing. This could involve promoting successful user stories and organizing online events.

By following a structured approach to deployment with a focus on testing and evaluation, FarmConnect can ensure a successful transition to production while maintaining high standards of quality, reliability, and user satisfaction.

## 6.12. Maintenance

Maintaining FarmConnect involves ongoing testing and evaluation activities to ensure that the system remains functional, reliable, and secure over time. Here's how maintenance can be approached with a focus on testing and evaluation.

### I. Continuous Monitoring:

Implement monitoring tools to track key performance indicators (KPIs) like uptime, response times, error rates, and resource utilization (CPU, memory) for both the mobile app and web dashboard.
Monitor user activity logs to identify usage patterns and potential areas for improvement.
Regularly review server logs for errors, security threats, or any anomalies that might require investigation.

### II. Proactive Testing:

Conduct regular automated testing (unit testing, integration testing) to ensure existing functionalities continue to work as expected after code updates or bug fixes.
Perform periodic regression testing to verify that new features haven't introduced unintended regressions in core functionalities.
Conduct regular penetration testing to identify and address potential security vulnerabilities.

### III. Performance Testing:

Schedule load testing to simulate high user traffic scenarios and ensure the platform can handle peak loads without performance degradation.
Conduct stress testing to push the system beyond normal usage limits and identify bottlenecks or breaking points that need to be addressed.

## IV. User Feedback Integration:

Encourage user feedback through surveys, app store reviews, and in-app feedback mechanisms.
Analyze user feedback to identify usability issues, feature requests, or areas for improvement.
Conduct user acceptance testing (UAT) with new releases to ensure new features meet user expectations and address their needs.

## V. Regular Security Updates:

Stay updated on the latest security patches and vulnerabilities for the underlying technologies used in FarmConnect (e.g., Android OS, Laravel framework).
Regularly update third-party libraries and dependencies used in the project to address any known security issues.
Conduct regular security audits to identify and mitigate potential security risks.

## VI. Documentation and Knowledge Transfer:

Maintain comprehensive documentation for the codebase, deployment process, and testing procedures.
Regularly update the documentation to reflect any changes made to the system.
Conduct knowledge transfer sessions within the development team to ensure everyone is familiar with the testing and evaluation processes.

By prioritizing testing and evaluation activities as part of the maintenance process, FarmConnect can ensure that it continues to deliver value to users, maintain high standards of quality and reliability, and adapt to evolving user needs and industry trends over time.

# Chapter 7: Conclusion and Future Work

This chapter concludes the project and highlights future work.

## 7.1. Conclusion

The conclusion of the FarmConnect project marks the culmination of efforts to develop a comprehensive platform tailored to the needs of the farming community. Here are the key points to highlight in the conclusion:

1. **Achievement of Objectives:**

FarmConnect successfully addresses the challenge of fostering collaboration and knowledge-sharing within the farming community. It provides a holistic solution encompassing networking, information exchange, collaboration tools, marketplace, and secure communication.

2. **User-Centric Design:**

The system's user-centric design ensures ease of use, intuitive navigation, and personalized user experiences. Through features like personalized user profiles, advanced search, newsfeed, group communities, and marketplace, FarmConnect enhances user engagement and satisfaction.

3. **Functionalities and Features:**

FarmConnect offers a wide range of functionalities, including user profiles and networking, newsfeed and updates, group communities, messaging and collaboration, marketplace for products and services, knowledge sharing and education, events and forums, notifications and alerts, weather updates and crop monitoring, user-generated content sharing, and expert consultation.

4. **Refinement Based on Feedback:**

Throughout the development process, FarmConnect underwent refinement and improvement based on user feedback and iterative testing. This iterative approach ensured that the system evolves to meet the evolving needs of the farming community.

5. **Data Security and Scalability:**

Emphasis on data security protocols ensures secure communication and data exchange within the platform. Additionally, the system's architecture prioritizes scalability to accommodate the growing user base and increasing data volume over time.

6. **Contribution to Farming Community:**

FarmConnect eliminates fragmentation in the agriculture sector by providing a centralized platform for farmers to connect, collaborate, and access resources. By facilitating communication, knowledge-sharing, and marketplace activities, FarmConnect contributes to the advancement and empowerment of the farming community.

In conclusion, FarmConnect represents a significant step forward in leveraging technology to address the unique challenges faced by farmers. By providing a comprehensive and user-friendly platform, FarmConnect aims to enhance productivity, efficiency, and sustainability in agriculture.

## 7.2. Future Work

As FarmConnect concludes its initial development phase, there are several avenues for future work and enhancements to further improve the platform and better serve the farming community. Here are some areas of potential future work for FarmConnect:

1. **Integration of AI and Machine Learning:**

Incorporate AI and machine learning algorithms to provide personalized recommendations, predictive analytics, and decision support tools for farmers. This could include crop yield predictions, pest and disease detection, soil health analysis, and personalized farming recommendations based on historical data and real-time inputs.

2. **Expansion of Marketplace and Services:**

Expand the marketplace functionality to include a wider range of agricultural products, services, and equipment. Enable farmers to buy and sell agricultural inputs, machinery, livestock, and other commodities directly through the platform. Additionally, integrate features for booking agricultural services such as equipment rental, crop spraying, and harvesting assistance.

3. **Enhanced Communication and Collaboration:**

Improve communication and collaboration features to facilitate knowledge-sharing, community building, and peer-to-peer support among farmers. Introduce discussion forums, expert Q&A sessions, and virtual networking events to foster interaction and learning within the farming community.

4. **Geospatial Analysis and Mapping:**

Integrate geospatial analysis and mapping capabilities to provide farmers with insights into land use, crop suitability, and environmental factors affecting agriculture. Enable farmers to visualize and analyze spatial data, such as satellite imagery, soil maps, weather patterns, and

crop performance metrics, to make informed decisions about crop management and land use planning.

5. **Mobile Application Optimization:**

Optimize the FarmConnect mobile application for performance, usability, and compatibility across a wider range of devices and operating systems. Implement offline capabilities, push notifications, and location-based services to enhance the mobile user experience and provide seamless access to critical information and features even in remote or low-connectivity areas.

6. **Community Engagement and Outreach:**

Launch outreach programs and initiatives to promote FarmConnect adoption and engagement among farmers, agricultural organizations, and rural communities. Collaborate with agricultural extension services, farmer cooperatives, and government agencies to raise awareness, provide training, and onboard new users to the platform.

7. **Data Analytics and Insights:**

Invest in data analytics and reporting capabilities to analyze trends, patterns, and insights derived from user interactions and platform usage. Provide farmers with actionable insights, benchmarking data, and performance metrics to optimize their farming practices, improve decision-making, and achieve better outcomes.

8. **Continuous Improvement and Feedback Loop:**

Establish a robust feedback loop with users to gather feedback, prioritize feature requests, and address issues or concerns in a timely manner. Embrace an agile development approach to iterate on FarmConnect's features and functionalities based on user needs, market trends, and technological advancements.

By focusing on these areas of future work, FarmConnect can continue to evolve as a valuable and indispensable tool for farmers, empowering them with knowledge, resources, and connections to thrive in an ever-changing agricultural landscape.

# References

References to any book, journal paper or website should properly be acknowledged. Please consistently follow the style. The following are few examples of different resources i.e. journal article, book, and website.


1. Chrismanto, A. R., Delima, R., Santoso, H. B., Wibowo, A., & Kristiawan, R. A. (2019). Developing agriculture land mapping using Rapid Application Development (RAD): A case study from Indonesia. *International Journal of Advanced Computer Science and Applications*, *10*(10), 232–241. https://doi.org/10.14569/ijacsa.2019.0101033

2. Dewi, L. J. E., Wijaya, I. N. S. W., & Seputra, K. A. (2021). Web-based Buleleng regency agriculture product information system development. *Journal of Physics: Conference Series*, *1810*(1). https://doi.org/10.1088/1742-6596/1810/1/012029

3. F. M., J. M. S. (2020). A Web Based Application for Agriculture : "Smart Farming System." *International Journal of Emerging Trends in Engineering Research*, *8*(6), 2309–2320. https://doi.org/10.30534/ijeter/2020/18862020

4. Xin, J., Zazueta, F. S., Iii, P. V., Mao, X., Kooram, N., & Yang, Y. (2015). *Delivering knowledge and solutions at your fingertips: strategy for mobile app development in agriculture*. http://www.cigrjournal.org

5. Chrismanto, A. R., Delima, R., Santoso, H. B., Wibowo, A., & Kristiawan, R. A. (2019). Developing agriculture land mapping using Rapid Application Development (RAD): A case study from Indonesia. *International Journal of Advanced Computer Science and Applications*, *10*(10), 232–241. https://doi.org/10.14569/ijacsa.2019.0101033
6. Dewi, L. J. E., Wijaya, I. N. S. W., & Seputra, K. A. (2021). Web-based Buleleng regency agriculture product information system development. *Journal of Physics: Conference Series*, *1810*(1). https://doi.org/10.1088/1742-6596/1810/1/012029
7. F. M., J. M. S. (2020). A Web Based Application for Agriculture : "Smart Farming System." *International Journal of Emerging Trends in Engineering Research*, *8*(6), 2309–2320. https://doi.org/10.30534/ijeter/2020/18862020
8. Xin, J., Zazueta, F. S., Iii, P. V., Mao, X., Kooram, N., & Yang, Y. (2015). *Delivering knowledge and solutions at your fingertips: strategy for mobile app development in agriculture*. http://www.cigrjournal.org