Chapter Three

System Analysis

3.1 Introduction

In this chapter, we explore the System Analysis phase, which involves a comprehensive examination of client requirements, system functionalities, and limitations. Systems analysis is a process of collecting factual data, understand the processes involved, identifying problems and recommending feasible suggestions for improving the system functioning.

The objective System Analysis is to ensure the effective planning and utilization of the proposed solution. This chapter outlines user roles, and interactions through various UML diagrams such as ***use case diagrams, sequence diagrams, activity diagrams, and class diagrams***. Additionally, it presents a ***user interface prototype*** to demonstrate the system's usability and accessibility.

3.2 System Requirement Specifications (SRS)

The System Requirement Specification (SRS) serves as a blueprint for system developers, outlining what the smart farming advisory system should do, how it should behave, and the constraints it must operate within. An SRS typically includes use cases, system interactions, performance criteria, security requirements, and design constraints.

For our project, the SRS defines the technical and functional requirements needed to develop a centralized, automated advisory system that leverages weather data, and smart analytics to optimize recommendation of better crop management schedules. It ensures that the system meets the needs of farmers, agricultural experts, and the land owners by providing real-time soil monitoring, weather-based irrigation scheduling, data-driven recommendations, and remote-control capabilities.

3.2.1 Use case Diagram

A use case diagram is a visual representation of the interactions between users (actors) and a system. It helps illustrate the functional requirements of the system, showing how different users will interact with it to achieve specific goals.

Use case diagrams help in understanding how ***farmers*** and ***system administrators*** will interact with the smart farming advisory system and what functionalities the system will provide. These diagrams define how users will monitor *soil conditions, provide crop recommendations, provide weather updates and receive alerts,* and *make data-driven decisions* for optimized productivity.

The following are key components of a use case diagram:

Actors: These represent individuals or external systems that interact with the system. Actors can be people, organizations, or external systems that interact with the application or system. They are external entities that generate or consume data.

System*:* This represents a specific sequence of actions and interactions between actors and the system. It can also be referred to as a scenario.

Goals*:* These represent the desired outcomes of most use cases. A well-designed diagram should describe the activities and variations required to achieve the goals.

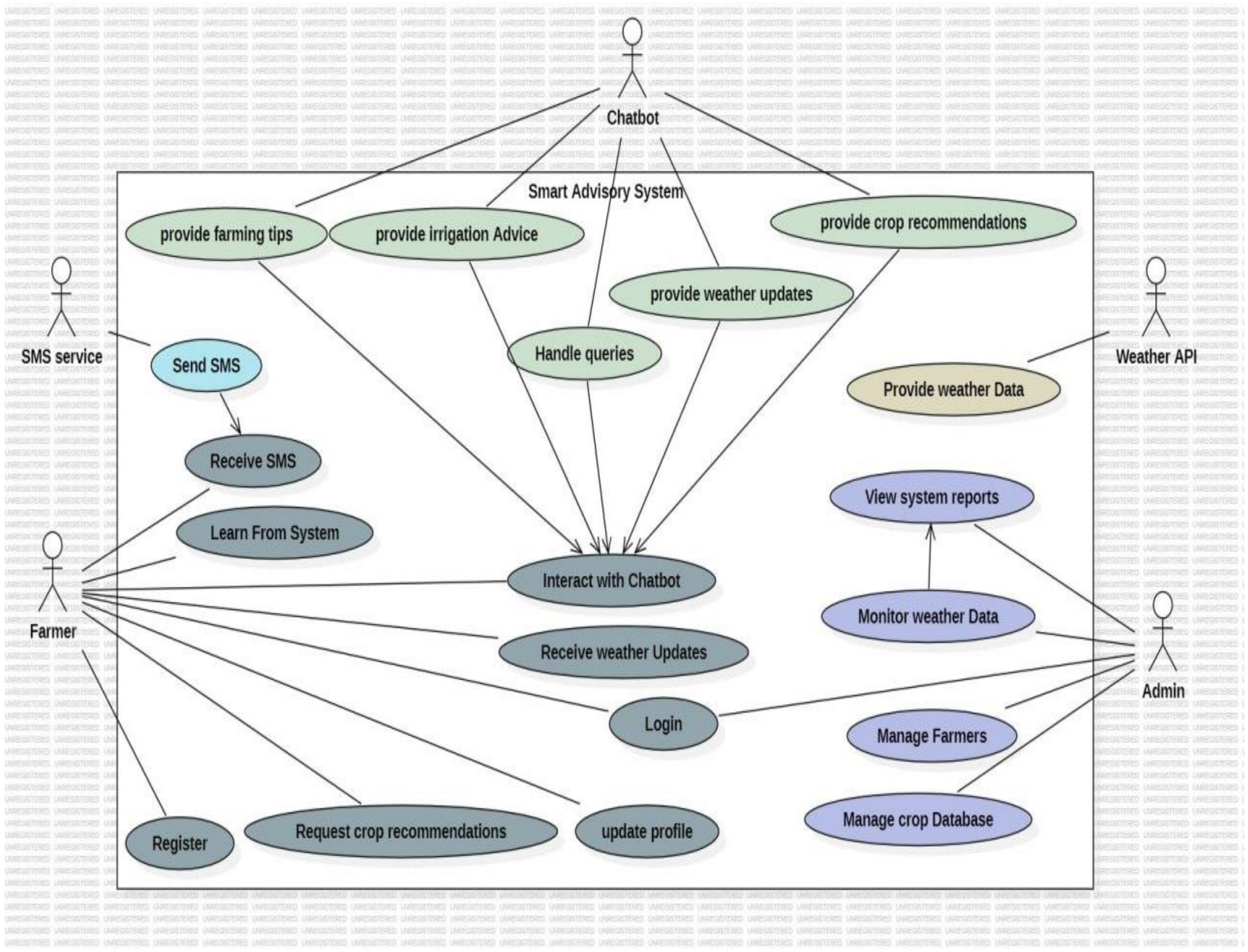


Fig 3.1 Use case diagram 1

Use case id	01
Use case name	Register
Actors	Farmer
Description	The farmer registers in the system by providing personal details such as name, location, and phone number.
Goal	To create a farmer profile in the system.
Precondition	The farmer must have access to the system
The basic flow of action	1. Farmer accesses the registration page. 2. Farmer enters personal details (name, location, phone number). 3. System validates the information. 4. System creates a farmer profile and stores it in the database. 5. System confirms successful registration.
Post condition	Farmer profile is created and stored in the system.
Alternative action	If the farmer provides invalid information, the system displays an error message and prompts the farmer to re-enter details.

Use case id	02
Use case name	Login

Actors	Farmer, Admin
Description	The farmer or admin logs into the system using their credentials (username and password).
Goal	To authenticate the user and grant access to the system.
Precondition	The user must be registered in the system.
The basic flow of action	1. User accesses the login page. 2. User enters their username and password. 3. System validates the credentials. 4. If credentials are valid, the system grants access to the user's dashboard. 5. System confirms successful login.
Post condition	User is logged into the system and can access their dashboard.
Alternative action	If the user enters incorrect credentials, the system displays an error message and prompts the user to re-enter details. If the user forgets their password, the system provides a "Forgot Password" option to reset it.

Use case id	03
Use case name	Request Crop Recommendations

Actors	Farmer
Description	The farmer requests crop recommendations based on their location and current weather conditions.
Goal	To provide the farmer with suitable crop recommendations.
Precondition	The farmer must be registered and logged into the system
The basic flow of action	1. Farmer requests crop recommendations via the chatbot. 2. Chatbot fetches real-time weather data from the Weather API. 3. Chatbot generates crop recommendations based on weather and location. 4. Chatbot sends the recommendations to the farmer.
Post condition	Farmer receives crop recommendations
Alternative action	If the Weather API fails, the chatbot notifies the farmer and suggests manual input of weather data.

Use case id	04
Use case name	Update Profile

Actors	Farmer
Description	The farmer updates their profile information, such as location or phone number.
Goal	To ensure the farmer's profile information is up-to-date.
Precondition	The farmer must be registered and logged into the system.
The basic flow of action	1. Farmer accesses the profile page. 2. Farmer updates the required information 3. System validates the updated information. 4. System saves the updated profile. 5. System confirms the update.
Post condition	Farmer's profile is updated in the system.
Alternative action	If the farmer provides invalid information, the system displays an error message and prompts the farmer to re-enter details.

Use case id	05
Use case name	Receive Weather Updates
Actors	Farmer

Description	The farmer receives real-time weather updates (temperature, humidity, rainfall)
Goal	To keep the farmer informed about current weather conditions.
Precondition	The farmer must be registered and logged into the system.
The basic flow of action	1. Farmer requests weather updates via the chatbot. 2. Chatbot fetches real-time weather data from the Weather API. 3. Chatbot sends the weather updates to the farmer.
Post condition	Farmer receives real-time weather updates.
Alternative action	If the Weather API fails, the chatbot notifies the farmer and suggests manual input of weather data.

Use case id	06
Use case name	Interact with Chatbot
Actors	Farmer, Chatbot
Description	The farmer interacts with the chatbot to get farming tips, crop recommendations, weather updates, and irrigation advice.

Precondition	The farmer must be registered and logged into the system.
The basic flow of action	1. Farmer sends a query to the chatbot (e.g., “What crops should I plant?”). 2. Chatbot processes the query 3. Chatbot retrieves relevant data (e.g., weather, crop rules) from the system. 4. Chatbot generates a response and sends it to the farmer.
Post condition	Farmer receives the requested information
Alternative action	. If the chatbot cannot understand the query, it asks the farmer to rephrase the question.

Use case id	07
Use case name	Manage Farmers
Actors	Admin
Description	The admin manages farmer accounts, including adding and, deleting accounts.
Goal	To maintain an up-to-date database of farmers.
Precondition	The admin must be logged into the system.
	1. Admin accesses the farmer management page.

The basic flow of action	2. Admin performs actions (add, or delete farmer profiles). 3. System validates the actions. 4. System updates the farmer database. 5. System confirms the action.
Post condition	Farmer database is updated.
Alternative action	If the admin provides invalid information, the system displays an error message and prompts the admin to re-enter details.

Use case id	08
Use case name	Monitor Weather Data
Actors	Admin
Description	The admin monitors real-time and historical weather data fetched from the Weather API.
Goal	To analyze weather trends
Precondition	The admin must be logged into the system.
The basic flow of action	1. Admin accesses the weather monitoring page. 2. System fetches real-time and historical weather data from the Weather API. 3. System displays the weather data to the admin.

Post condition	Admin can view and analyze weather data.
Alternative action	.If the Weather API fails, the system notifies the admin

Use case id	09
Use case name	Manage Crop Database
Actors	Admin
Description	The admin manages the crop database, including adding, updating, or deleting crop rules and recommendations.
Goal	To ensure the crop database is accurate and up-to-date.
Precondition	The admin must be logged into the system
The basic flow of action	1. Admin accesses the crop database management page. 2. Admin performs actions (add, update, or delete crop rules). 3. System validates the actions. 4. System updates the crop database. 5. System confirms the action.
Post condition	Crop database is updated.

Alternative action	If the admin provides invalid information, the system displays an error message and prompts the admin to re-enter details.

Use case id	10
Use case name	Send SMS Notifications
Actors	SMS Service
Description	The system sends SMS notifications to farmers with important updates, such as weather alerts, farming tips, or crop recommendations. Notifications can be sent manually by the admin or automatically based on a schedule (daily or weekly).
Goal	To keep farmers informed via SMS.
Precondition	The SMS service must be configured and operational.
The basic flow of action	1. Admin Configures Schedule: (e.g., daily at 8 AM or weekly on Mondays). 2. System Fetches Data: (e.g., weather data, crop recommendations) 3. System Generates SMS Content: The system generates the SMS content based on the fetched data.

	4. System Sends SMS: The system sends the SMS notifications to the farmers via the SMS Service
Post condition	Farmers receive SMS notifications as per the schedule.
Alternative action	If the SMS Service fails, the system notifies the admin. If no data is available for the scheduled notification, the system skips sending the SMS.

Use case id	11
Use case name	Provide Farming Tips
Actors	Chatbot
Description	The chatbot provides farming tips and best practices to the farmer.
Goal	To educate the farmer on best farming practices.
Precondition	The farmer must be registered and logged into the system.
The basic flow of action	1. Farmer requests farming tips via the chatbot. 2. Chatbot retrieves relevant farming tips from the database.

	3. Chatbot sends the farming tips to the farmer.
Post condition	Farmer receives farming tips
Alternative action	.If the chatbot cannot find relevant tips, it notifies the farmer and suggests contacting an expert.

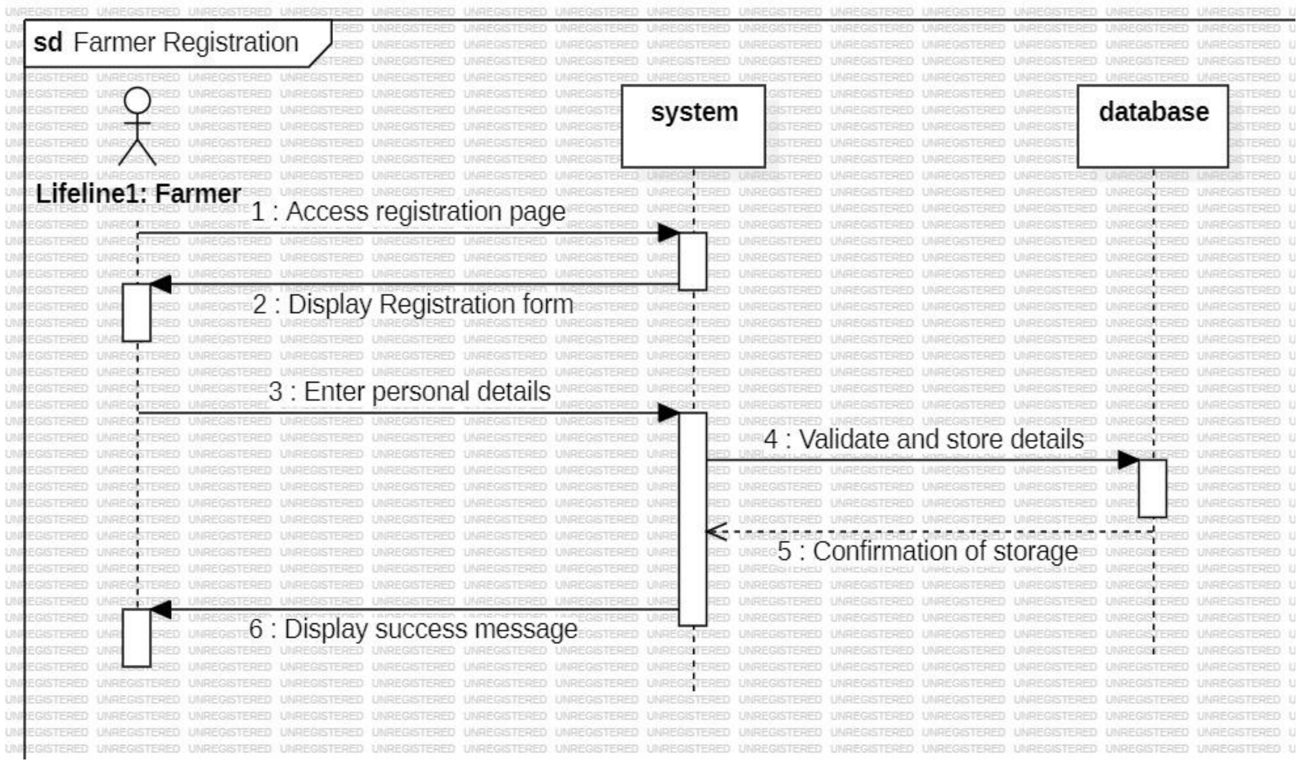
Use case id	12
Use case name	Provide Irrigation Advice
Actors	Chatbot,
Description	The chatbot provides irrigation advice based on weather conditions and crop type.
Goal	To help the farmer optimize water usage.
Precondition	The farmer must be registered and logged into the system.
The basic flow of action	1. Farmer requests irrigation advice via the chatbot. 2. Chatbot retrieves weather data and crop type 3. Chatbot generates irrigation advice based on the data. 4. Chatbot sends the irrigation advice to the farmer.

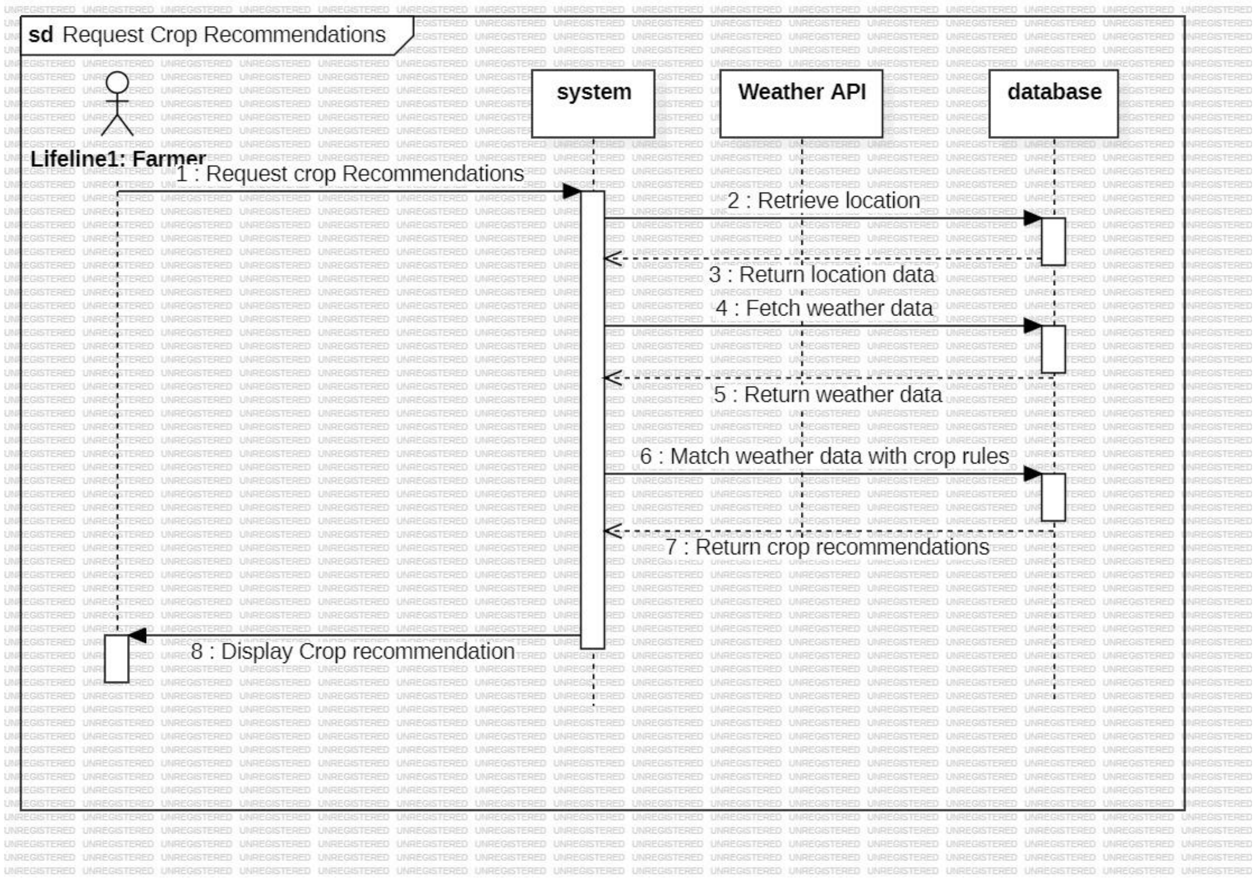
Post condition	Farmer receives irrigation advice.
Alternative action	If the chatbot cannot generate advice, it notifies the farmer and suggests manual input of data.

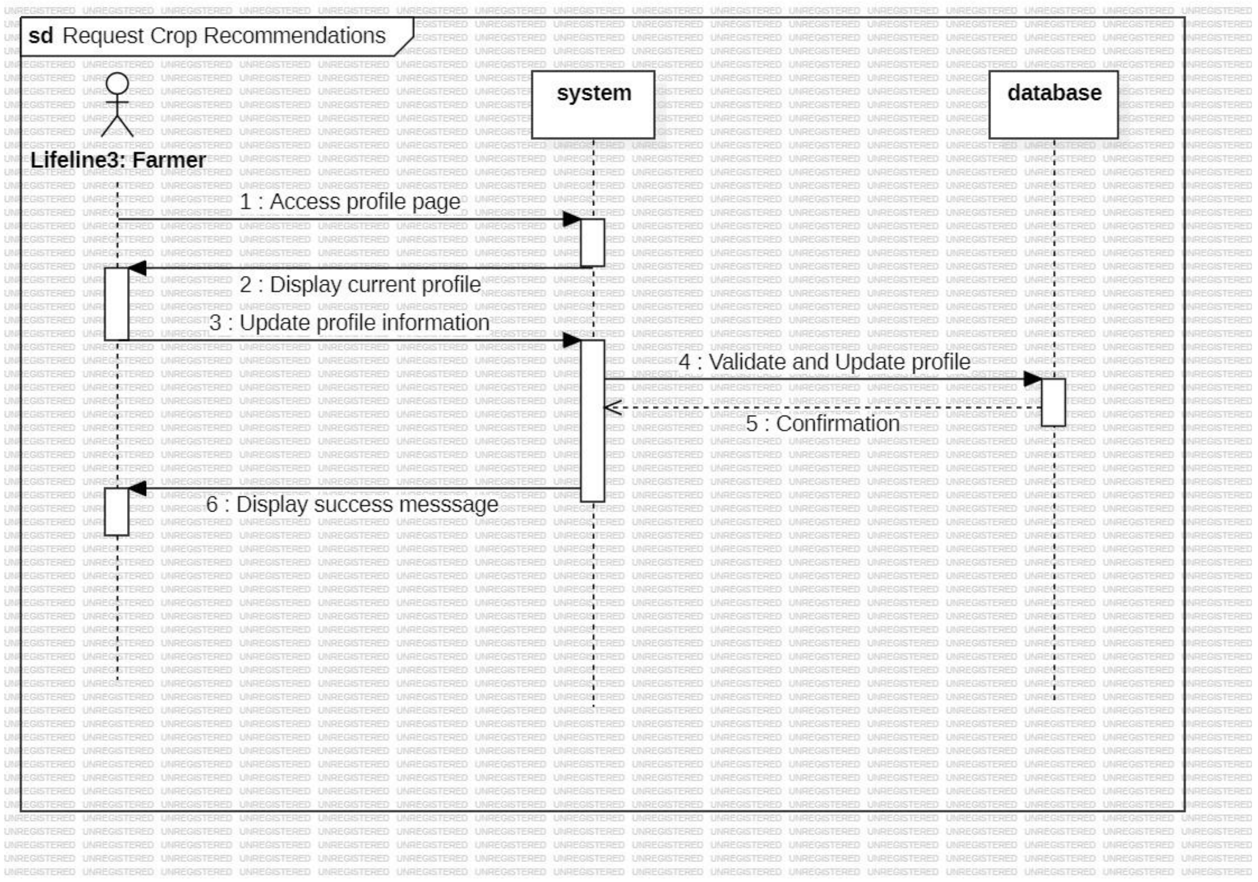
3.2.3 Sequence diagram

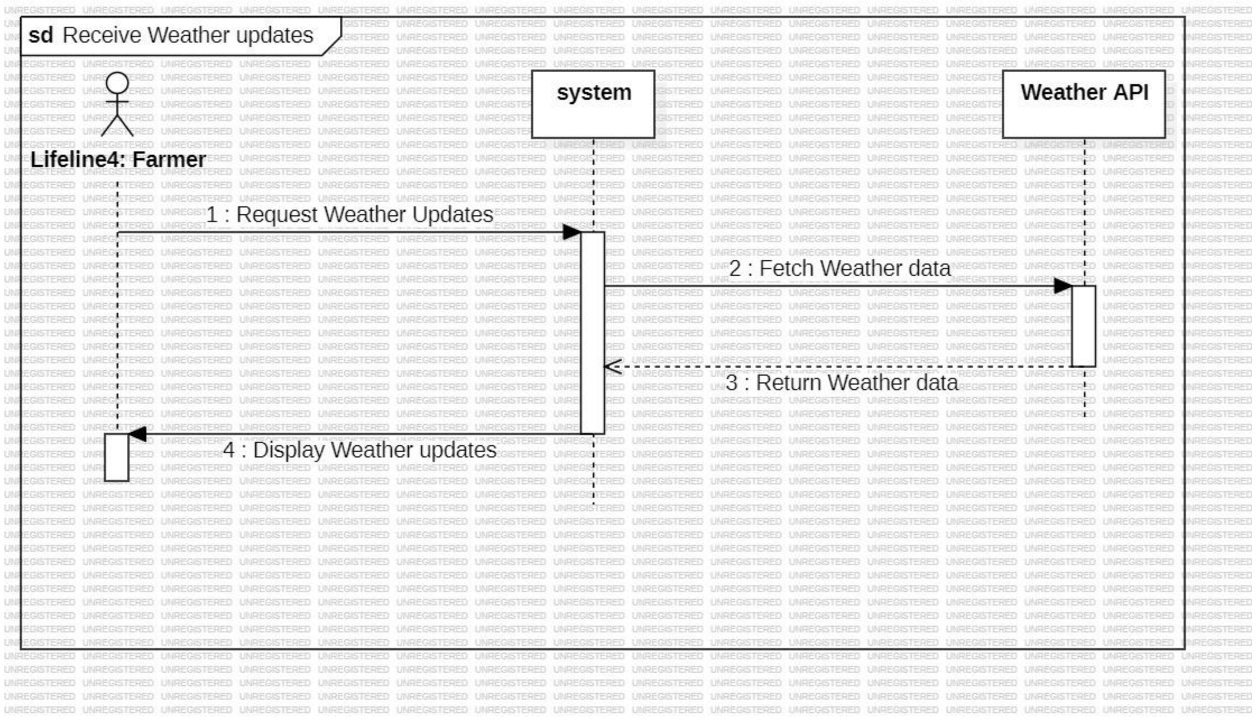
A sequence diagram is an essential tool in the design and development of the *Smart Farming Advisory System for Crop Management in Arbaminch*, as it visually represents the interactions between system components and actors over time. This type of diagram focuses on the flow of messages and processes, illustrating how farmers, administrators, and external systems (such as weather APIs or soil sensors) collaborate to achieve specific functionalities, such as generating crop recommendations or analyzing soil conditions. By mapping out these interactions, the sequence diagram provides a clear and detailed understanding of the system's dynamic behavior, ensuring that all stakeholders have a shared vision of how the system will operate.

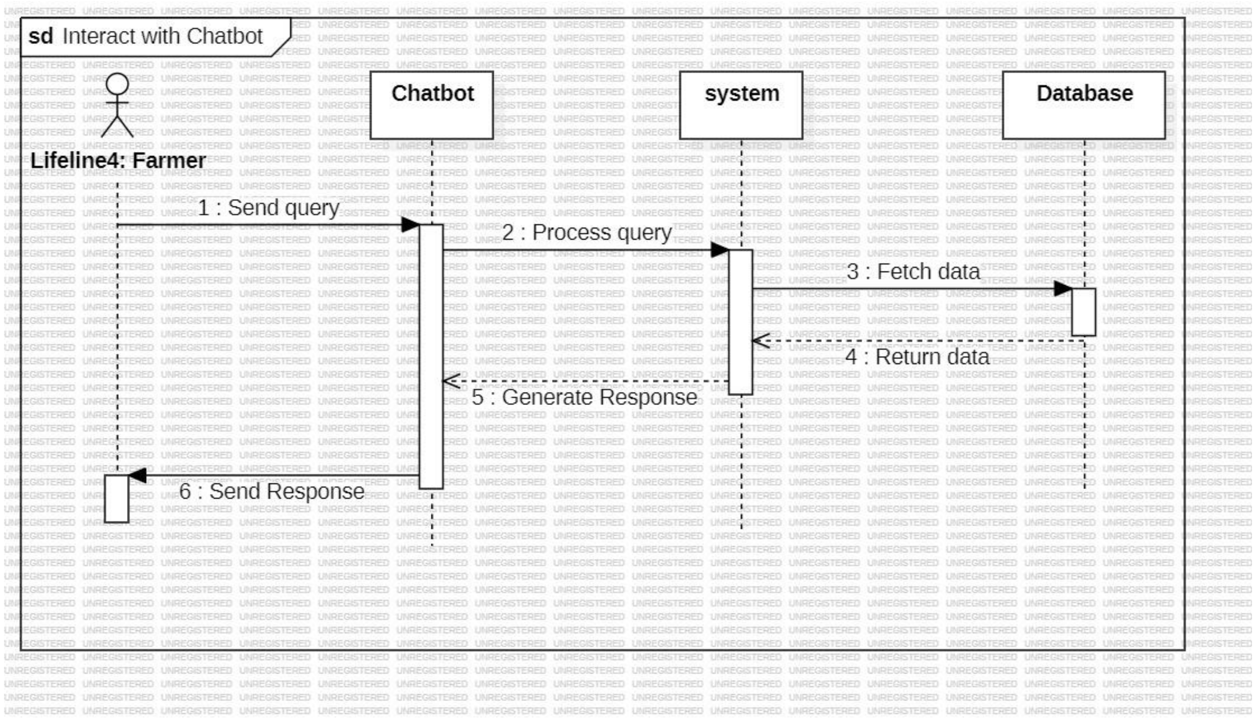
It serves as a critical communication tool between developers, designers, and end-users, helping to identify potential bottlenecks, optimize processes, and ensure the system meets the needs of farmers in Arbaminch for efficient and data-driven crop management.


sd Farmer Registration
Lifeline1: Farmer
system
database
1 : Access registration page
2 : Display Registration form
3 : Enter personal details
4 : Validate and store details
5 : Confirmation of storage
6 : Display success message

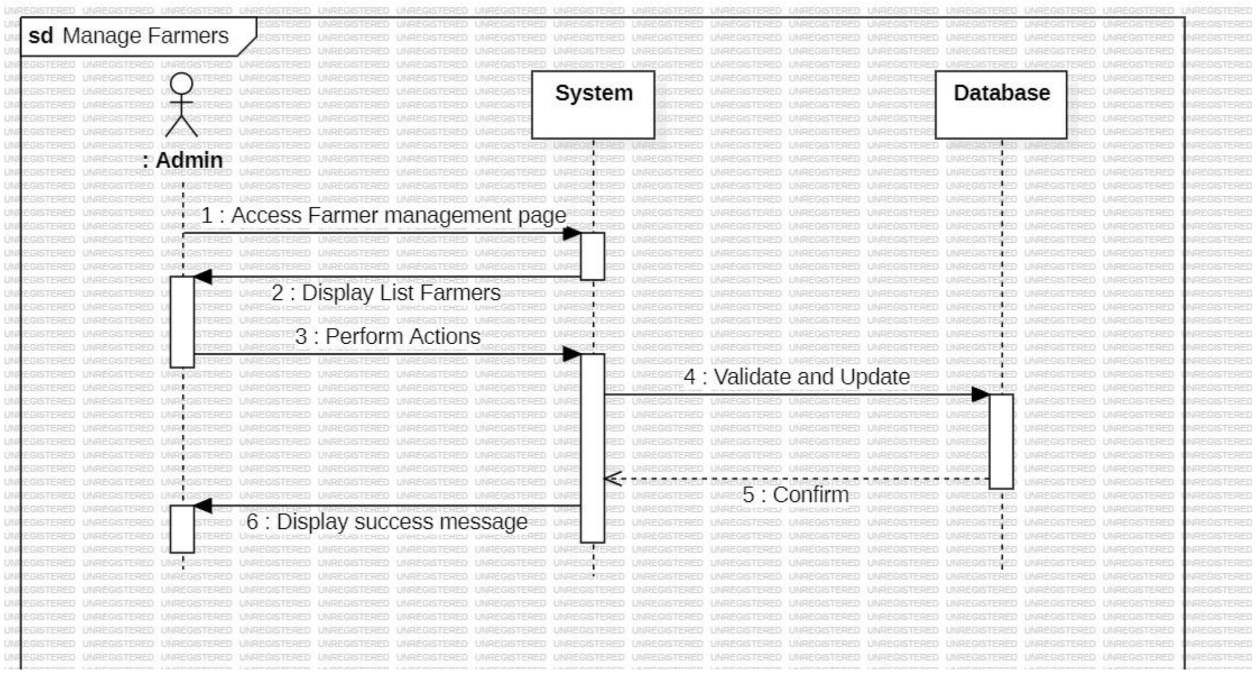

sd Request Crop Recommendations
Lifeline1: Farmer
system
Weather API
database
1 : Request crop Recommendations
2 : Retrieve location
3 : Return location data
4 : Fetch weather data
5 : Return weather data
6 : Match weather data with crop rules
7 : Return crop recommendations
8 : Display Crop recommendation


sd Request Crop Recommendations
system
database
Lifeline3: Farmer
1 : Access profile page
2 : Display current profile
3 : Update profile information
4 : Validate and Update profile
5 : Confirmation
6 : Display success messsage

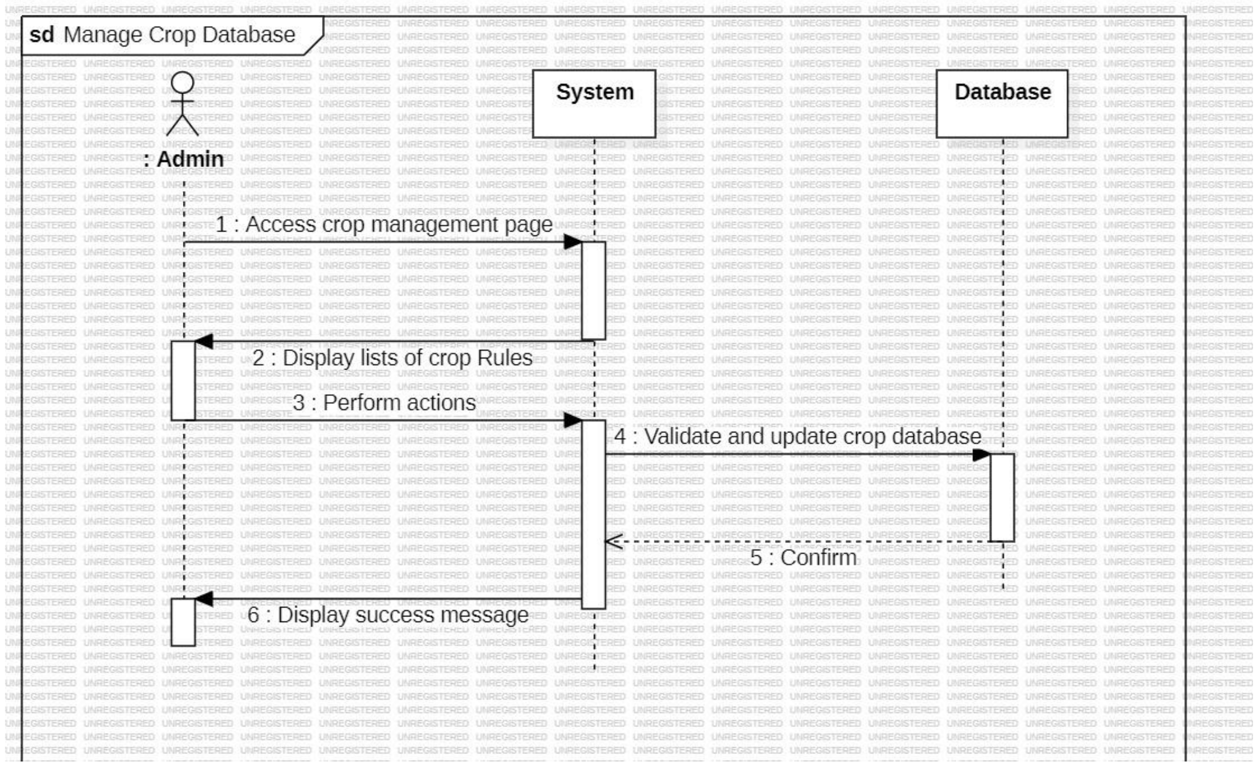

sd Receive Weather updates
system
Weather API
Lifeline4: Farmer
1 : Request Weather Updates
2 : Fetch Weather data
3 : Return Weather data
4 : Display Weather updates



sd Interact with Chatbot
Lifeline4: Farmer
Chatbot
system
Database
1 : Send query
2 : Process query
3 : Fetch data
4 : Return data
5 : Generate Response
6 : Send Response



sd Manage Farmers
: Admin
System
Database
1 : Access Farmer management page
2 : Display List Farmers
3 : Perform Actions
4 : Validate and Update
5 : Confirm
6 : Display success message


sd Manage Crop Database
: Admin
System
Database
1 : Access crop management page
2 : Display lists of crop Rules
3 : Perform actions
4 : Validate and update crop database
5 : Confirm
6 : Display success message

3.2.4 Activity Diagram

It is an essential aspect of designing and developing the *Smart Farming Advisory System for Crop Management in Arbaminch*; it helps in visualizing the workflow and processes contained in the system. The degree of detailing of the activity diagram captures the sequence of activities, decision points, and interactions among different entities- farmers, administrators, and external systems like weather services or soil sensors. For example, the diagram can represent the flow of activities where a farmer asks a question about crop conditions and the system processes and analyzes soil and weather data and the generation of advisory recommendations made accordingly.

Activity diagrams help in streamlining such workflows by helping to identify inefficiencies, clarify complex processes, and ease the running of the system in smooth and efficient manner. It is a very important document for both developer and stakeholder doing visualization and refinement of the functionality of the system to reflect the farmers' specific needs of Arbaminch, thereby improving their capacities in efficient and sustainable crop management.

3.2.5 Analysis level class diagram

The analysis-level class diagram is a crucial component in the design and development of the *Smart Farming Advisory System for Crop Management in Arbaminch*, as it provides a high-level representation of the system’s structure, key entities, and their relationships. At this stage, the class diagram focuses on identifying the essential classes, their attributes, and the interactions between them, without delving into implementation details.

For this project, the diagram will capture core entities such as *Farmers*, *Crops*, *Soil Data*, *Weather Conditions*, and *Advisory Recommendations*, along with their associations and dependencies. By modeling these elements, the analysis-level class diagram serves as a blueprint for understanding how the system will function, ensuring that it aligns with the requirements of farmers in Arbaminch to optimize crop management through data-driven insights and intelligent recommendations. This foundational step lays the groundwork for subsequent design and development phases, enabling the creation of a robust and user-centric smart farming solution.

The key components from the use case diagram include:

- **Chatbot**: Handles queries, provides advice, and interacts with users.
- **Farmer**: Registers, interacts with the chatbot, receives updates.
- **Admin**: Manages farmers and the crop database.
- **Weather API**: Provides weather data.
- **SMS Service**: Sends and receives SMS.
- **Smart Advisory System**: Central system that handles farming tips, irrigation advice, and weather updates.

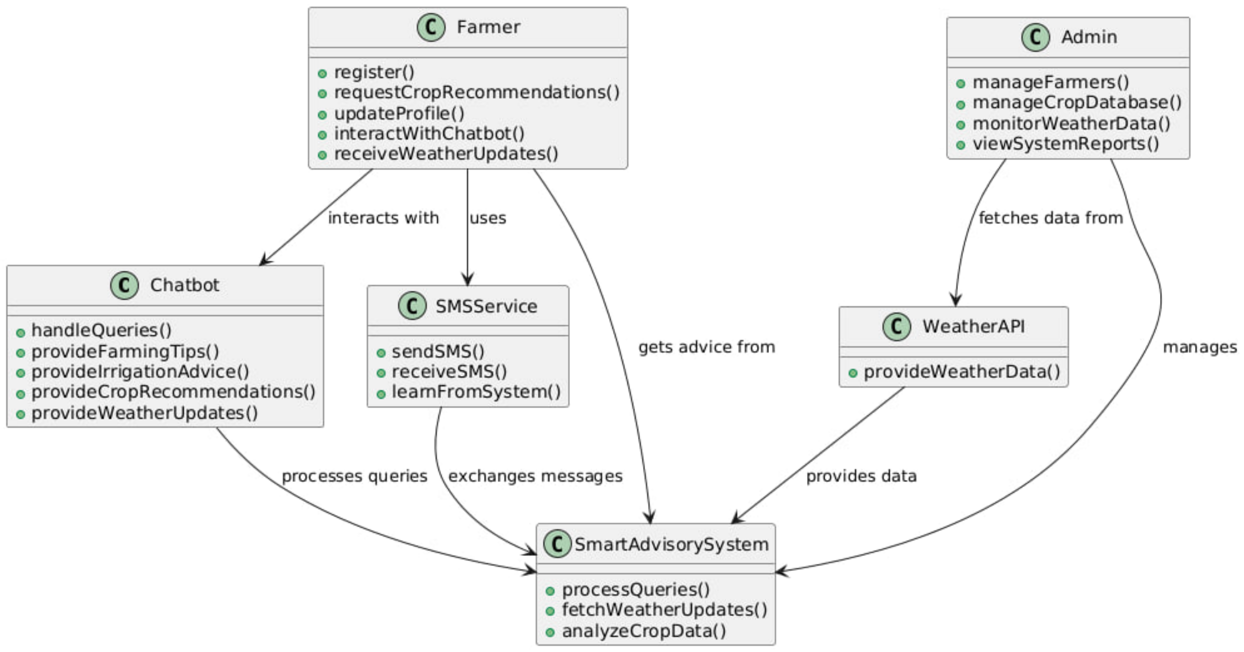


Fig Analysis Level Class Diagram

3.2.6 User Interface Prototyping

User interface (UI) prototyping plays a pivotal role in the development of the “Smart Farming Advisory System for Crop Management in Arbaminch”, as it enables the creation of an intuitive and user-friendly interface tailored to the needs of farmers. This process involves designing interactive mockups that visually and functionally represent how farmers will interact with the system to access critical information such as crop recommendations, soil data, weather forecasts, and pest management advice.

By developing low-fidelity wireframes and evolving them into high-fidelity, clickable prototypes, the design team can gather valuable feedback from end-users and stakeholders, ensuring the interface is accessible, efficient, and easy to navigate, even for users with limited technical expertise. UI prototyping not only bridges the gap between design and functionality but also ensures that the final system aligns with the real-world requirements of farmers in Arbaminch, ultimately enhancing their ability to make informed decisions and improve crop productivity.