Software Engineering (Sessional) Final Report

Course: CSE-356

Smart Agriculture Platform

Team Members

Shuva Dey
ID: 2104001

Abir Dey
ID: 2104005

Rathijit Aich
ID: 2104014

Department of Computer Science and Engineering (CSE)
Chittagong University of Engineering & Technology (CUET)
Chattogram- 4349, Bangladesh

Team Members

Photo	Name	ID	Email	Contact NO.	Total Credit Passed	CGPA acquired
	Shuva Dey	2104001	u2104001@ student.cuet.ac.bd	01402495768	101.75	3.74
	Abir Dey	2104005	u2104005@ student.cuet.ac.bd	01766559199	101.75	3.71
	Rathijit Aich	2104014	u2104014@ student.cuet.ac.bd	01852462164	101.75	3.76

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Goals and Objectives of the project

The objectives of this project are:

- To Enable farmers to monitor crop health and farm activities independently.
- To Provide timely guidance for crop care, disease prevention, and irrigation management.
- To Integrate DL technologies to detect crop diseases and optimize farming practices.
- To Establish a sustainable, easy-to-use system that supports small-scale farmers and adapts to future agricultural innovations.

1.2 Scope of the work

This project focuses on the development of an agriculture platform that has all the important features in it. The project scope includes:

- Design and development of a platform application React.
- Integration of DL algorithms for leaf disease prediction.
- Ensuring user authentication, data privacy, and secure operations throughout the system.

The project primarily focuses on the functional implementation and demonstration of the concept rather than large-scale commercial deployment.

1.2.1 Current situation and context

Many farmers in Bangladesh struggle with limited access to modern farming techniques, awareness of crop diseases, and optimal resource usage. This platform provides an all-in-one digital solution to assist farmers in decision-making, reduce losses, and increase crop yield. It also aligns with Bangladesh's goal of modernizing agriculture and supporting rural development.

1.2.2 Competing products (available in market)

Some of the existing competing platforms include:

- **Krishibid.com** — Provides agricultural information, expert consultations, and farming solutions for Bangladeshi farmers [1].
- **iFarmer Bangladesh** — Offers digital financial services, farm management tools, and agricultural advisory support [2].
- **AmarKrishi** — Connects farmers with agricultural resources, training, and expert guidance through mobile technology [3].
- **AgroStar (India)** — Provides crop advisory services, input recommendations, and farmer support through a digital platform [4].

However, most of these platforms focus on specific aspects such as financing or advisory support and lack a fully integrated, data-driven solution. This project differentiates itself by offering a comprehensive DL-enabled agricultural assistant that combines farm management, crop monitoring, expert communication, and predictive analytics in a single, user-friendly system.

1.3 System overview

The system is designed as a full-stack web platform that connects farmers, agricultural experts, and data-driven services within a unified ecosystem. The platform features a user-friendly web interface for managing farm activities and monitoring crop conditions, supported by a robust backend that handles data storage, analytics, and communication.

The backend integrates Deep Learning components, including:

- **Crop Health Detection Module**: Analyzes uploaded crop images to identify diseases, nutrient deficiencies, and pest infestations.
- **Expert Advice System**: Farmers can get advice from expert and can send them to review results.
- **Activities Logging**: Farmer can log their daily activities, giving them a platform to keep track of important details.

Additional platform features include secure user authentication, role-based dashboards, real-time weather and irrigation alert and market price updates.

1.4 Structure of the document

The report is organized as follows:

- **Chapter 1: Introduction** — Presents the project background, goals, objectives, scope, and system overview.
- **Chapter 2: Project Management Plan** — Describes the project management framework, including planning, scheduling, resource allocation, and risk management.

- **Chapter 3: Requirement Specification** — Defines the functional and non-functional requirements, system features, and performance expectations.

1.5 Terms, Acronyms, and Abbreviations Used

- **DL** — Deep Learning
- **ML** — Machine Learning
- **UI** — User Interface
- **API** — Application Programming Interface Environment

Chapter 2

Project Management Plan

2.1 Project Organization

The project team consists of three members working collaboratively with defined responsibilities and regular coordination meetings. The team follows an agile-inspired approach with iterative development cycles and continuous stakeholder feedback.

2.1.1 Individual Contribution to the project

Individual Contribution to the project are shown in Table-2.1.The table illustrates each member's role in major project phases such as requirement specification, planning, designing, model building, user interface development, testing and deployment.

Table 2.1: Individual Contribution to the project

Member Name	Requirement Specification	Planning	Designing	Model Building	User Interface	Testing	Deployment
Shuva Dey	✓	✓		✓	✓		
Abir Dey	✓		✓		✓		✓
Rathijit Aich	✓			✓		✓	✓

2.2 Process Model Used

The Prototyping Model is especially suitable for projects where requirements are not fully clear at the beginning. By developing an early working prototype, both developers and stakeholders can visualize the core functionalities of the system at an early stage. This approach helps in identifying missing features, refining requirements, and correcting design flaws much earlier in the development cycle. The iterative nature of prototyping also supports continuous user feedback, enabling the team to improve the interface, workflow, and system behavior incrementally. As a result, this model reduces the overall project risk and ensures that the final product closely aligns with user needs and expectations.

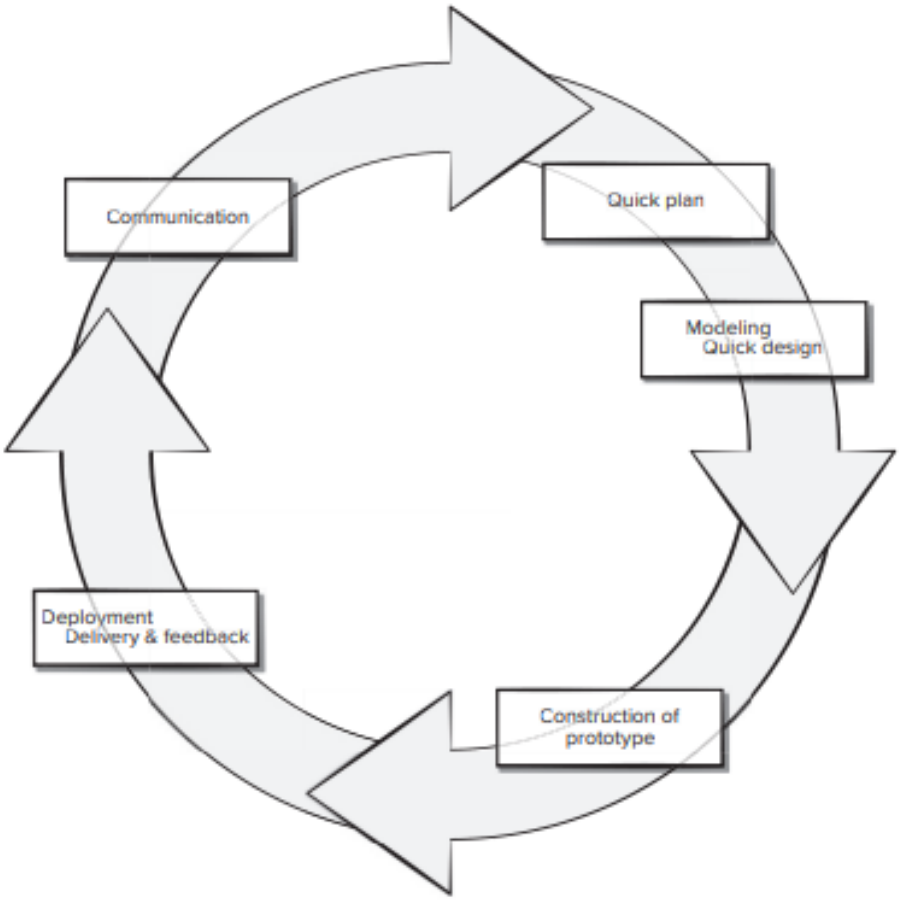

Figure 2.1: Prototyping Process Model

2.2.1 Rationale for choosing lifecycle model

The Prototyping Model was selected because it aligns well with the nature of this project and supports iterative refinement. The key reasons for choosing this lifecycle model are:

- **Clear Requirements Through Testing:** Prototyping enables refining and validating requirements by building and testing early versions of the platform, ensuring that the final product meets user needs.
- **Focus on Users:** By presenting prototypes to users and stakeholders, feedback can be gathered early and incorporated to make the system more user-friendly.
- **Better Communication:** Visual prototypes improve communication between the team and stakeholders by clearly demonstrating system features and progress.
- **Reduces Risks:** Early testing and feedback help identify and address issues before deeper development begins, reducing overall risk.
- **Ideal for Design Work:** Since UI/UX is important, prototyping allows the team to design an intuitive and visually appealing interface early.
- **Quick Adjustments:** Changes can be made rapidly based on feedback, keeping

the project aligned with user expectations.

- **Fits Academic Projects:** The prototyping model is simple, flexible, and suitable for small teams with limited time, making it ideal for academic work.

2.3 Risk Analysis

Risk management is essential to identify, evaluate, and mitigate challenges during the development of the Smart Agriculture Platform. The major risks and mitigation strategies are:

- **Technical Risk:** Issues related to deep learning models and overall system integration.
  Mitigation: Use reliable frameworks, verified dependencies, and modular testing to ensure stable DL model deployment.
- **Data Risk:** Limited or inconsistent agricultural data may reduce model accuracy.
  Mitigation: Collect additional sensor data and apply data augmentation techniques to strengthen model performance.
- **Operational Risk:** Poor network connectivity in rural areas may disrupt real-time updates.
  Mitigation: Implement offline data storage with automatic synchronization when connectivity is restored.
- **Schedule Risk:** DL model training and debugging may extend development time.
  Mitigation: Allocate buffer time and run non-dependent tasks in parallel to maintain project progress.

2.4 Constraints to project implementation

The project faces several constraints related to resources and development limitations:

- **Schedule Constraints:** Limited development time requires effective task prioritization and parallel execution.
- **Budget Constraints:** Financial limitations restrict project scope and available tools.
- **Software Constraints:** The system depends on stable versions of deep learning libraries and compatible development environments.
- **Hardware Constraints:** Training DL models requires hardware with sufficient processing power and memory.
- **User Capability Constraints:** Farmers in rural regions, have limited digital proficiency and minimal prior exposure to modern technology. The system must therefore offer a highly intuitive, language-accessible, and simplified interface to ensure effective adoption and usability.

2.5 Hardware and Software Resource (Tools/Language) Requirements

The tools, technologies, and resources required for developing the Smart Agriculture Platform are as follows:

Hardware Requirements

- Intel i5 or higher processor
- Minimum 8 GB RAM
- At least 20 GB available storage
- Android device or web browser for application testing

Software Requirements

- **Operating System:** Windows 10 or Linux
- **Frontend Framework:** React.js
- **Backend Framework:** Spring Boot
- **Development Environments:** Visual Studio Code, IntelliJ IDEA, or Eclipse
- **Programming Languages:** Java, JavaScript, Python
- **Deep Learning Tools:** Google Colab, Kaggle, TensorFlow/Keras
- **Database:** MySQL or Firebase (for real-time data)
- **Version Control:** Git and GitHub
- **External API:** Open-source weather API

2.6 Project Timeline and Schedule

Table 2.2: Project Timeline and Schedule

Phase	**Tasks Covered**	**Duration**
Requirement Analysis	Requirement gathering, documentation, stakeholder identification.	Week 1–2
Planning and Design	System architecture, database schema, UI layout, DL workflow planning.	Week 3–4
Development Phase 1	Frontend (React), backend (Spring Boot), REST APIs, DB connectivity.	Week 5–7
Development Phase 2	DL model training and integration; Kaggle/Colab data processing.	Week 8–9
Testing & Evaluation	Testing, debugging, collecting user feedback.	Week 10–11
Finalization	Documentation, deployment optimization, final presentation.	Week 12

2.7 Estimated Budget

Table 2.3: Estimated Budget

Category	**Details**	**Cost (BDT)**
Infrastructure (6 months)	MySQL, 50GB storage, 1TB traffic, domain + SSL.	22,000
External APIs & Notifications	Weather, Market API, SMS package.	12,000
AI & Data	GPU credits, labeling set (≈500 images).	30,000
Development & Testing	Test device, accessories, field testing.	20,000
Training	Onboarding sessions, materials, honorarium.	15,000
Total		**104,000**

2.8 Social/Cultural/Environmental impact of the project

The Smart Agriculture Platform is expected to create significant positive impacts across social, cultural, and environmental dimensions.

Social Impact

- Empowers small and medium-scale farmers with knowledge and digital tools to manage their farms effectively.
- Reduces crop losses, thereby improving farmers' income and economic stability.
- Enhances community engagement by enabling communication with agricultural experts and knowledge sharing among farmers.

Cultural Impact

- Encourages the adoption of modern agricultural practices while respecting traditional farming methods.
- Helps preserve agricultural heritage by incorporating local crop varieties and culturally relevant farming practices.

Environmental Impact

- Supports sustainable farming through optimized irrigation, fertilizer usage, and pest management.
- Reduces excessive chemical use, lowering soil and water pollution.
- Enhances climate resilience by providing timely alerts on weather conditions and potential crop disease risks.

Overall, the platform is designed to improve the quality of life for farmers, promote sustainable agriculture, and contribute to rural development in Bangladesh.

Chapter 3

Requirement Specifications

3.1 Stakeholders for the system

[5] is used for this chapters writings. The stakeholders for the Smart Agriculture Platform include all individuals and entities directly or indirectly affected by its development, deployment, and usage. Each stakeholder group has distinct expectations and interactions with the system:

- **Farmers (Primary Users):** Small and medium-scale farmers who log farm activities, monitor crop health, receive alerts, and access expert advice to optimize their farming practices.
- **Agricultural Experts:** Provide guidance, recommendations, and real-time notifications to farmers. They help ensure accurate advice regarding crop care, disease prevention, and resource management.
- **System Administrator:** Manages user accounts, ensures database integrity, monitors performance, and maintains the platform's overall operational stability.
- **Developers:** Responsible for designing, developing, and maintaining the frontend, backend, AI/ML modules, and database components of the platform.
- **Government & NGOs:** Optional stakeholders who may utilize aggregated data for agricultural programs, policymaking, and farmer training initiatives.

3.2 Use case diagram with Graphical and Textual Description

3.2.1 Preliminary Use Case Diagram for Smart Agriculture Platform

The Smart Agriculture Platform connects three main actors: Farmer, Expert, and Machine. Farmers can perform several key actions through the platform, including viewing crop health reports, getting expert advice, logging farm activities, receiving weather information, and checking market price information. Experts can publish vlogs and also contribute to crop health reports accessed by farmers. The machine represents external

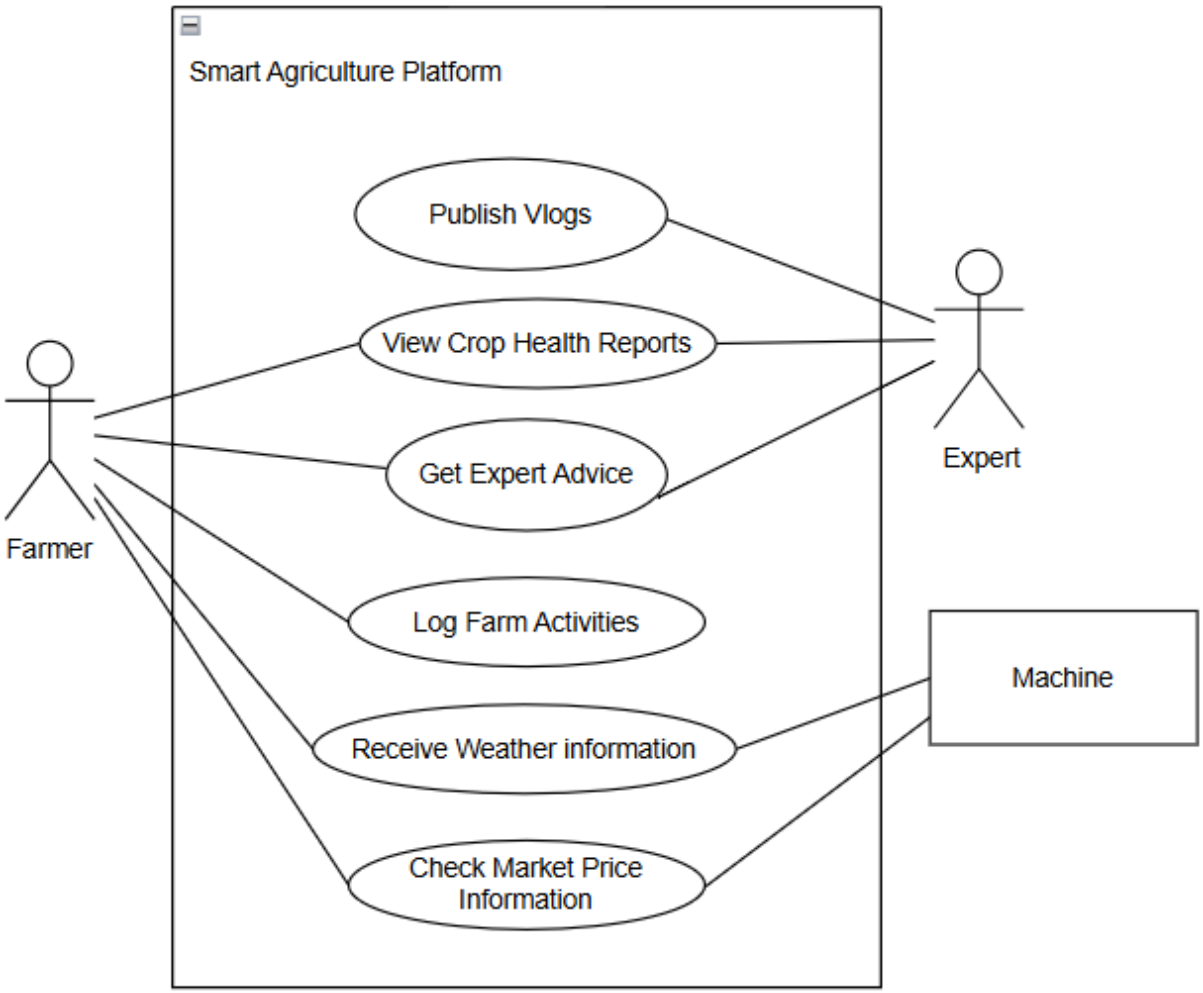

Figure 3.1: Preliminary Use Case Diagram for Smart Agriculture Platform

api that supports backend operations such as providing weather updates and market price information to farmers. Overall, the platform facilitates communication between farmers and experts while integrating essential agricultural information services.

3.2.2 Expert Advisory & Communication System

The Farmer can initiate communication by sending queries to the system. These queries are processed through the Analyze Queries use case, which is performed by the Expert. The system then enables the expert to provide responses through the Receive Expert Advice use case, which the farmer accesses.

In situations requiring field assistance, the Farmer can request an expert visit, which includes the Approve Visit use case. The Expert evaluates and approves the visit request before it is carried out.

Overall, the system supports seamless two-way communication, enabling farmers to receive expert guidance and request on-site support when necessary.

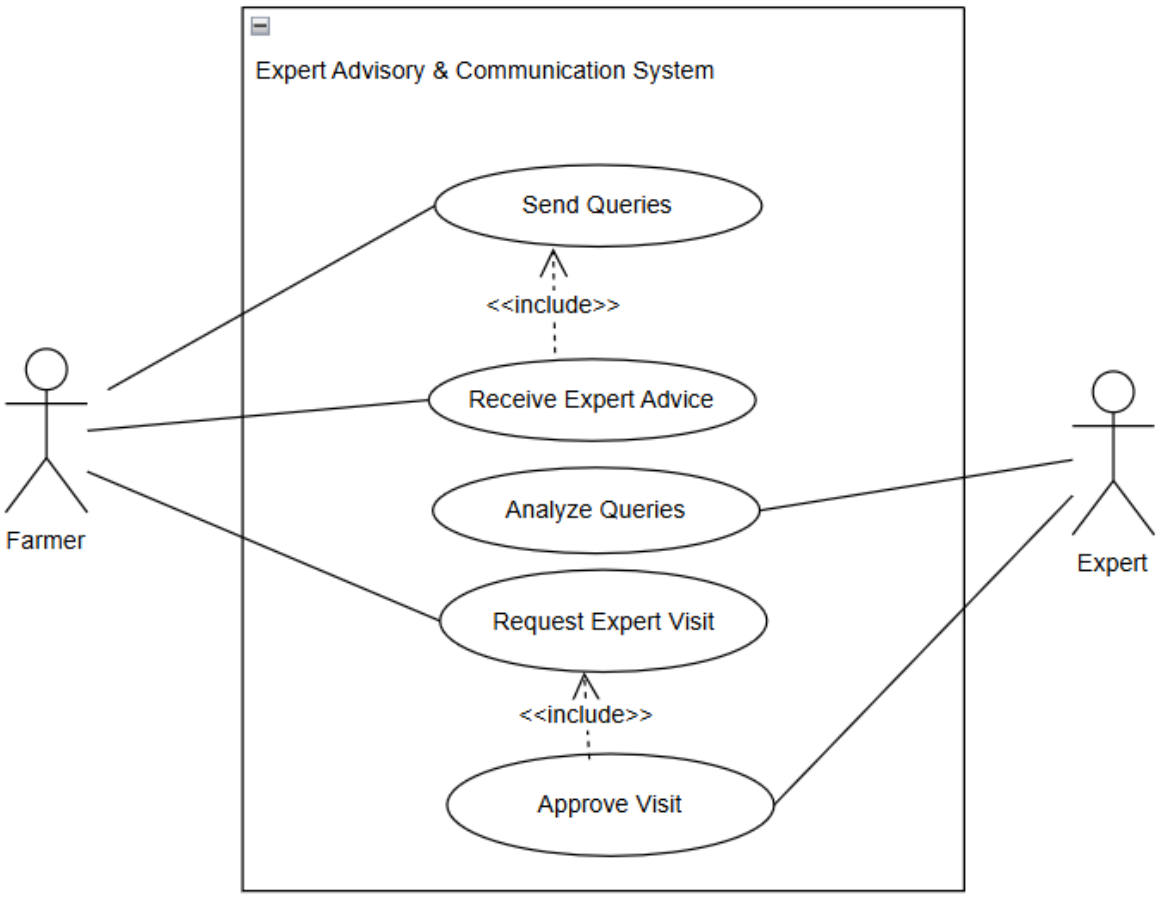

Figure 3.2: Use Case Diagram for Expert Advisory & Communication System

3.2.3 Crop Disease Detection

This use-case diagram illustrates the Crop Disease Detection module, involving three actors: the Farmer, expert and Machine.

The Farmer initiates the process by uploading a crop image. The Machine that represents the trained DL model analyzes the uploaded image and identifies potential diseases. The expert gets to see the detection and review the detection. Based on the detection results, the system proceeds to recommend preventive measures to the farmer. Finally, the Farmer can provide feedback regarding the recommendations or detection accuracy.

Overall, the module enables automated disease detection and guidance based on image analysis, supported by server-side processing.

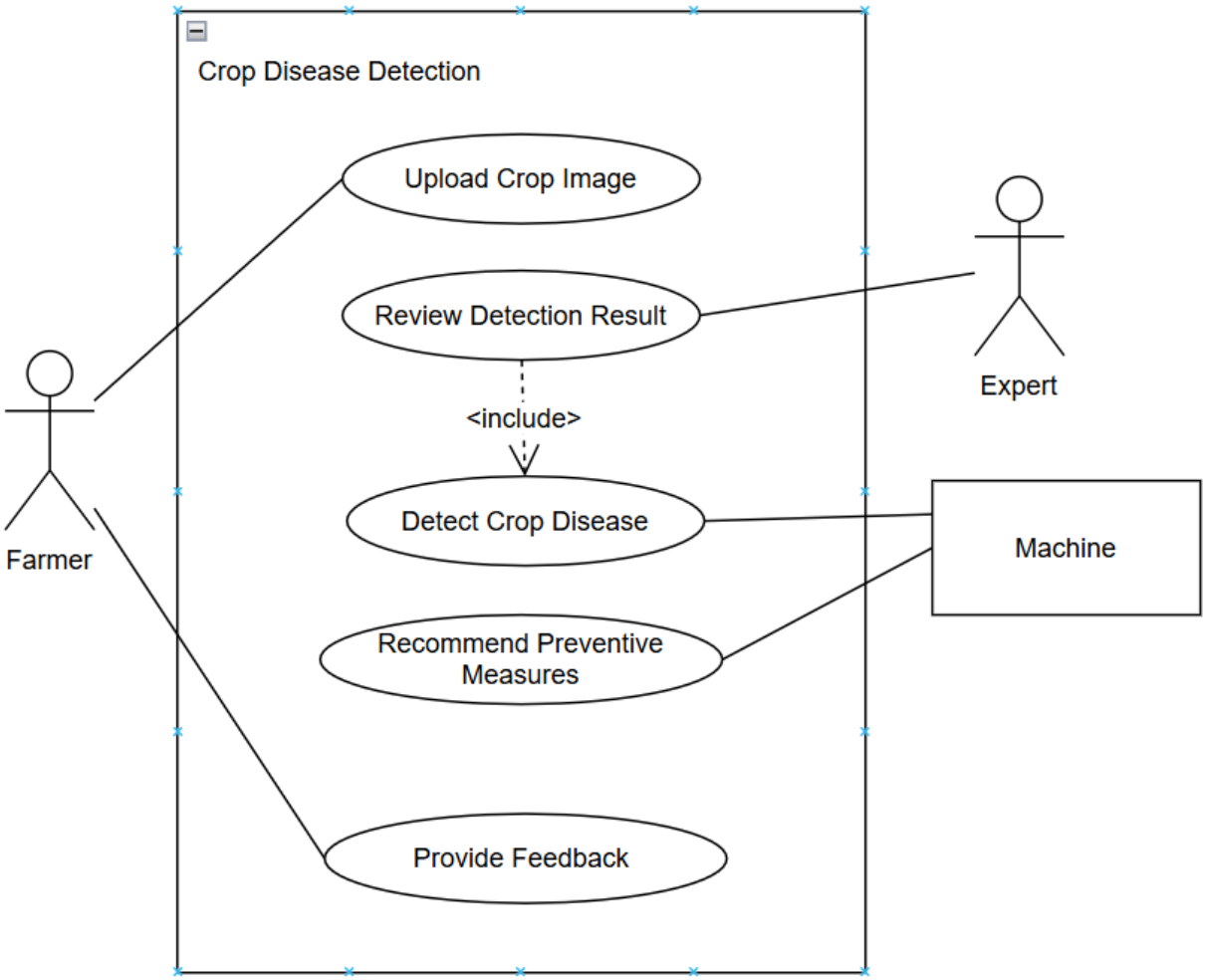

Figure 3.3: Use Case Diagram for Crop Disease Detection

3.3 Development of Software Requirement Specification (SRS)

3.3.1 Purpose

The purpose of this Software Requirement Specification (SRS) is to define the functional and non-functional requirements of the Smart Agriculture Platform. The system helps farmers monitor crop health, detect diseases, communicate with experts, access weather and market information, and manage farm activities through a unified digital platform.

3.3.2 Scope

The Smart Agriculture Platform assists farmers by providing:

- AI-based crop disease detection and preventive recommendations.
- Expert advisory communication and query management.
- Weather updates and market price information.

- Farm activity logging and viewing crop health reports.
- Educational and engagement features such as vlog publishing.

Actors:

- **Farmer:** The main end-user who interacts with all major system functionalities.
- **Expert:** Provides agricultural advice, validates detections, and responds to farmer queries.
- **External Systems:** Includes DL model inference services, weather APIs, and market data providers that supply information to the platform.

Use Cases: Publish Vlogs, View Crop Health Reports, Get Expert Advice, Log Farm Activities, Receive Weather Information, Check Market Prices, Upload Crop Image, Detect Crop Disease, Recommend Preventive Measures, Provide Feedback.

Outputs: Disease diagnosis (type, severity, confidence), recommended actions, weather reports, market prices, expert responses, historical activity insights.

3.3.3 Overall Description

Product Perspective

The system consists of interconnected modules:

- **Upload Crop Image:** Enables farmers to capture or upload crop photos along with metadata.
- **Detect Crop Disease:** Uses machine learning to identify diseases and severity.
- **Recommend Preventive Measures:** Maps disease results to actionable recommendations.
- **Expert Advisory Module:** Handles communication between farmers and experts.
- **Weather and Market Module:** Provides real-time weather conditions and crop market prices.
- **Farm Activity Log:** Allows farmers to maintain records and observations.
- **Vlog Publishing:** Lets expert upload knowledge-sharing articles.

User Classes and Characteristics

- **Farmer:** Non-technical users requiring simplicity; interacts with all modules.
- **Expert:** Technical agricultural specialists validating problems and recommending solutions.
- **Machine:** Maintains external APIs, ML models.

Operating Environment

- Web and mobile web browsers (Chrome, Firefox, Edge).
- Backend server with REST APIs and background processing.

- Relational database for structured data; object storage for images.
- Integrations: AI inference service, weather API, market rate APIs.

3.3.4 Functional Requirements

1. Upload Crop Image

- FR-1.1: The farmer shall upload or capture an image and provide crop/field metadata.
- FR-1.2: The system shall validate file type, size, and quality.
- FR-1.3: The system shall store the image and create a corresponding record.

2. Detect Crop Disease

- FR-2.1: The system shall send the image to an AI model based on crop type and version.
- FR-2.2: The AI model shall detect disease and return confidence and severity.
- FR-2.3: The system shall store results with timestamps and model version.
- FR-2.4: On failure, the system shall retry and mark the status as "pending".

3. Recommend Preventive Measures

- FR-3.1: The system shall recommend preventive actions based on crop, disease, and severity.
- FR-3.2: The system shall include safety notes and chemical restrictions.
- FR-3.3: Low-confidence cases shall be flagged as "review suggested".

4. View Analytical Reports

- FR-4.1: Timeline view of images, diagnoses, and actions.
- FR-4.2: Charts showing disease frequency and severity trends.
- FR-4.3: Filtering by date, crop, and severity.

5. Expert Advisory & Communication

- FR-5.1: Farmers shall send queries with descriptions and images.
- FR-5.2: Experts shall review and respond with recommendations.
- FR-5.3: Farmers shall request field visits when required.
- FR-5.4: Experts shall approve or decline visit requests.

6. Weather and Market Information

- FR-6.1: The system shall display real-time weather updates.
- FR-6.2: The system shall fetch market prices from external APIs.

7. Notifications

- FR-7.1: Notify farmers when analyses or expert responses are available.
- FR-7.2: Notify users when external data (weather/market) is outdated.

8. Access Control and Audit

- FR-8.1: Farmers shall access only their own records.
- FR-8.2: All create/update events shall be logged.

3.3.5 Non-Functional Requirements

Performance

- P-1: Image upload shall complete within 3 seconds (excluding network delay).
- P-2: Disease detection turnaround time 15 seconds.
- P-3: Dashboard and reports shall load under 3 seconds.

Reliability

- R-1: Background tasks shall persist after system failures.
- R-2: Image and inference data shall remain consistent.

Usability

- U-1: User interface must be simple and mobile-friendly.
- U-2: Diagnosis and recommendations shall be presented clearly.

Security and Privacy

- S-1: All communication must use HTTPS.
- S-2: User images and crop data must be securely stored.

Availability

- A-1: The system shall maintain $\geq 99.5\%$ uptime.

Model Governance

- G-1: Each prediction must store model ID and version.
- G-2: System shall display an AI advisory disclaimer.

3.3.6 Constraints

Technical

- Requires stable Internet connection.
- Uses relational DB + object storage for images.
- ML inference depends on external GPU servers.

Operational

- Weather and market data accuracy depends on external APIs.
- Large images may be compressed.

Time

- Weather data refreshed every 60 minutes.
- Analysis jobs start within 1 minute under normal load.

3.3.7 Assumptions and Dependencies

Assumptions:

- Farmers upload clear images with correct metadata.
- Organization maintains API keys and ML model registry.

Dependencies:

- AI inference engine availability.
- External weather and market APIs.
- Database, storage, and notification services.

3.3.8 Future Enhancements

- Offline lightweight inference for fast local feedback.
- Expert-verified feedback loops to improve ModeL accuracy.
- Multi-image segmentation for disease localization.
- Multilingual recommendations and localized crop product databases.

3.3.9 Conclusion

The Smart Agriculture Platform integrates AI-based disease detection, expert communication, weather and market data services, and logging features into a single digital ecosystem. It enhances decision-making for farmers while ensuring scalability, reliability, and security.

3.4 Development of Use Case Template

3.4.1 Use Case Template for Expert Advisory & Communication System

Use Case: Send Queries, Receive Expert Advice, Analyze Queries, Request Expert Visit, Approve Visit

Iteration: 1 **Last Modified:** 10 Nov 2025
By: Rathijit Aich

Primary Actor: Farmer

Goal: Allow farmers to send agricultural queries, obtain expert recommendations, and request approved expert visits when remote advice is insufficient.

Preconditions:

- Farmer account exists & is authenticated.
- Expert accounts are active and available for query handling.
- Farmer has valid crop/field details for context.

Trigger:

- Farmer decides to seek expert assistance for a crop-related issue.

Scenario:

1. Farmer logs into the platform.
2. Farmer navigates to the *Expert Advisory & Communication System*.
3. Farmer selects "Send Query" and enters problem description with optional images.
4. System forwards the query to an available expert.
5. Expert reviews the query under *Analyze Queries*.
6. Expert sends recommendations or clarification under *Receive Expert Advice*.
7. Farmer reviews the advice.
8. Farmer selects "Request Expert Visit" if physical inspection is needed.

Exceptions:

- Missing information → system prompts farmer to complete required fields.
- No expert available → query is queued and farmer is shown expected wait time.
- Incorrect login credentials → show authentication error.

Priority: High (core advisory module)

Availability: Available after user authentication

Frequency: Moderately frequent.

Channel to Actor: Web interface with active Internet connection

Secondary Actors:

- **Expert:** Reviews farmer queries, provides advice, and approves field visits.

Channel to Secondary Actors: Expert portal for query management and visit approval workflows

Open Issues:

- How should queries be prioritized during high load periods?
- Should experts be required to justify declined visit requests?
- What criteria determine when an issue warrants on-site inspection?
- Should farmers be allowed to rate expert advice for quality tracking?

3.4.2 Use Case Template for Crop Disease Detection System

Use Case: Upload Crop Image, Detect Crop Disease, Recommend Preventive Measures, Provide Feedback

Iteration: 1 **Last Modified:** 10 Nov 2025
By: Abir Dey

Primary Actor: Farmer

Goal: Enable the farmer to upload crop images, obtain automated disease detection results, and receive preventive agricultural recommendations.

Preconditions:

- Farmer account exists and is authenticated.
- Farmer device has crop images.
- Disease detection module is operational and model endpoints are accessible.

Trigger:

- Farmer decides to check whether a crop is infected or showing unusual symptoms.

Scenario:

1. Farmer logs into the platform.
2. Farmer navigates to the *Crop Disease Detection* module.
3. Farmer selects "Upload Crop Image" and chooses or captures an image.
4. System validates the image and sends it to the detection engine.
5. System processes the image under *Detect Crop Disease*.
6. System displays detected disease (if any) with confidence level.
7. Expert gets to see the prediction result and review it.
8. System provides suggested preventive measures or treatment steps.
9. Farmer may submit "Provide Feedback" regarding accuracy or usefulness.

Exceptions:

- Poor image quality → system requests a clearer image.
- No disease detected → system suggests continued monitoring.
- Detection engine unavailable → system displays last stored analysis or advises retry later.

Priority: High (critical diagnostic module)

Availability: Released after integration with the AI detection model

Frequency: Moderate; increases during disease-prone seasons

Channel to Actor: Web with Internet connectivity

Secondary Actors:

- **Expert:** Validates model predictions when requested or escalated.
- **Machine:** Provides model prediction.

Channel to Secondary Actors: Expert portal for validating difficult or uncertain detection results. Machine provides the model result through the backend.

Open Issues:

- Should the system store images for future model retraining?
- Is the model trained with the latest images to provide best functionality?
- What safeguards are needed to ensure farmer privacy for uploaded images?

3.5 Activity Diagram

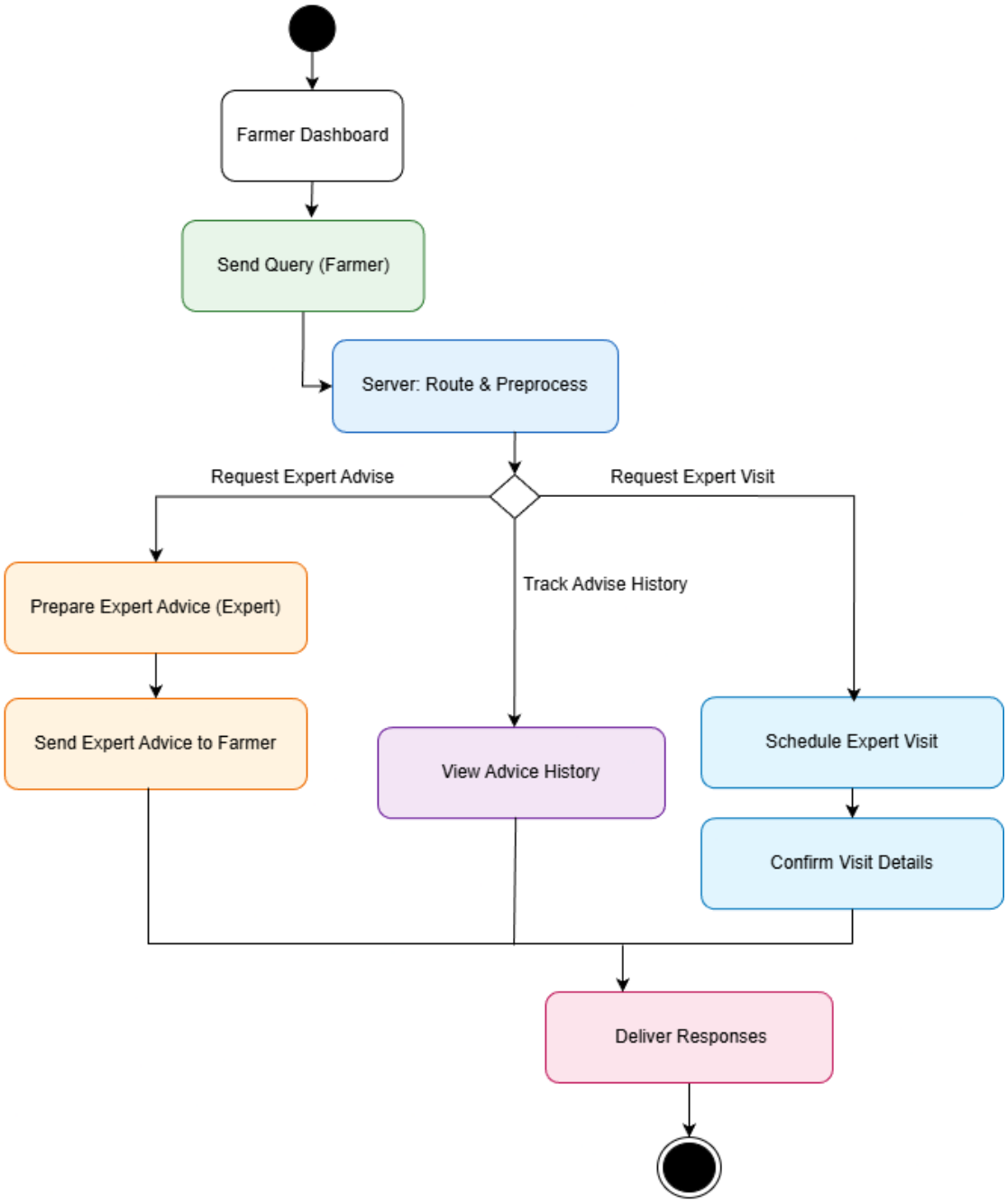

Figure 3.4: Activity Diagram Of Expert Communication System

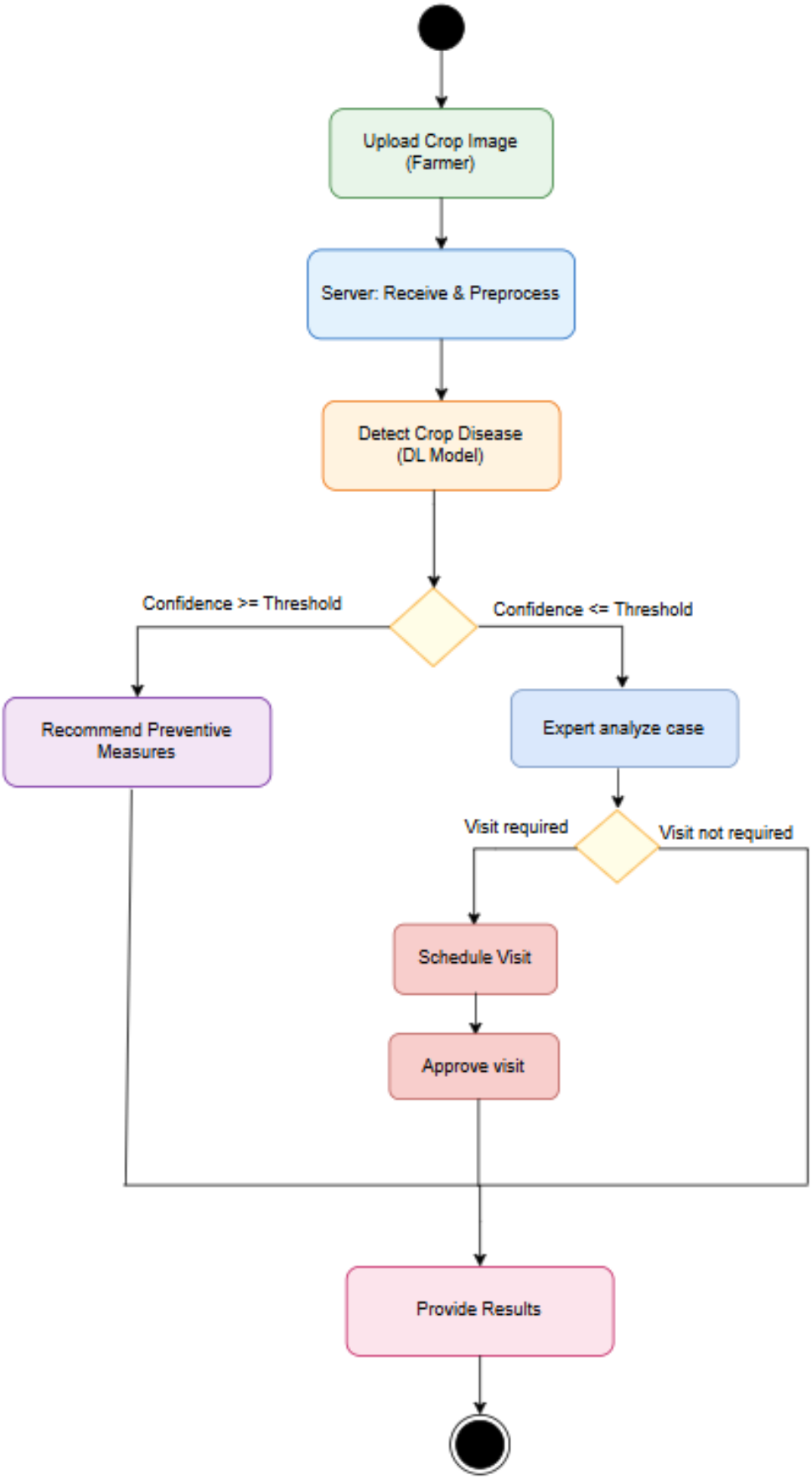

Figure 3.5: Activity Diagram of Crop disease Detection

3.6 Static model – class diagram

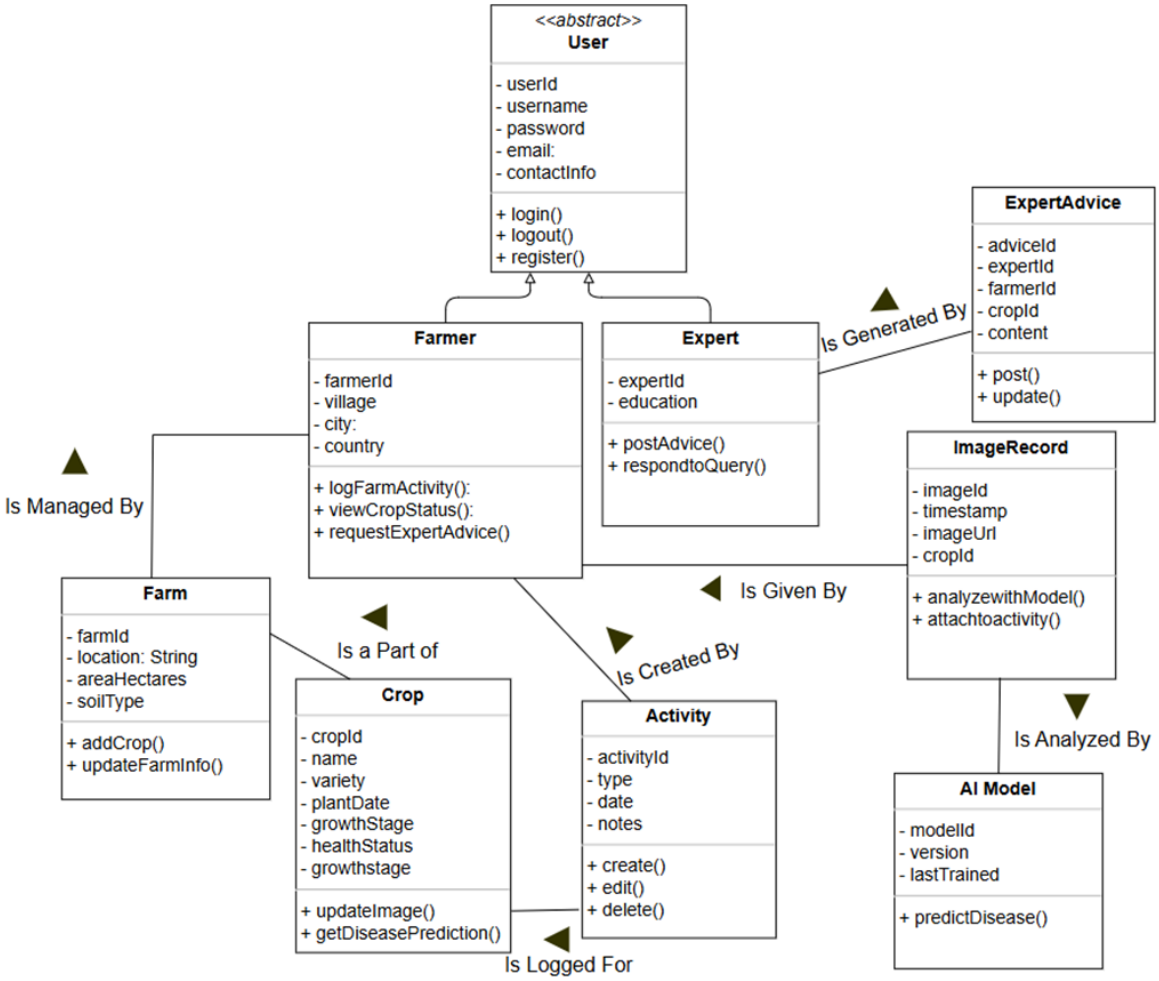

Figure 3.6: Class Diagram of Smart Agriculture Platform

3.7 Dynamic model – sequence diagram

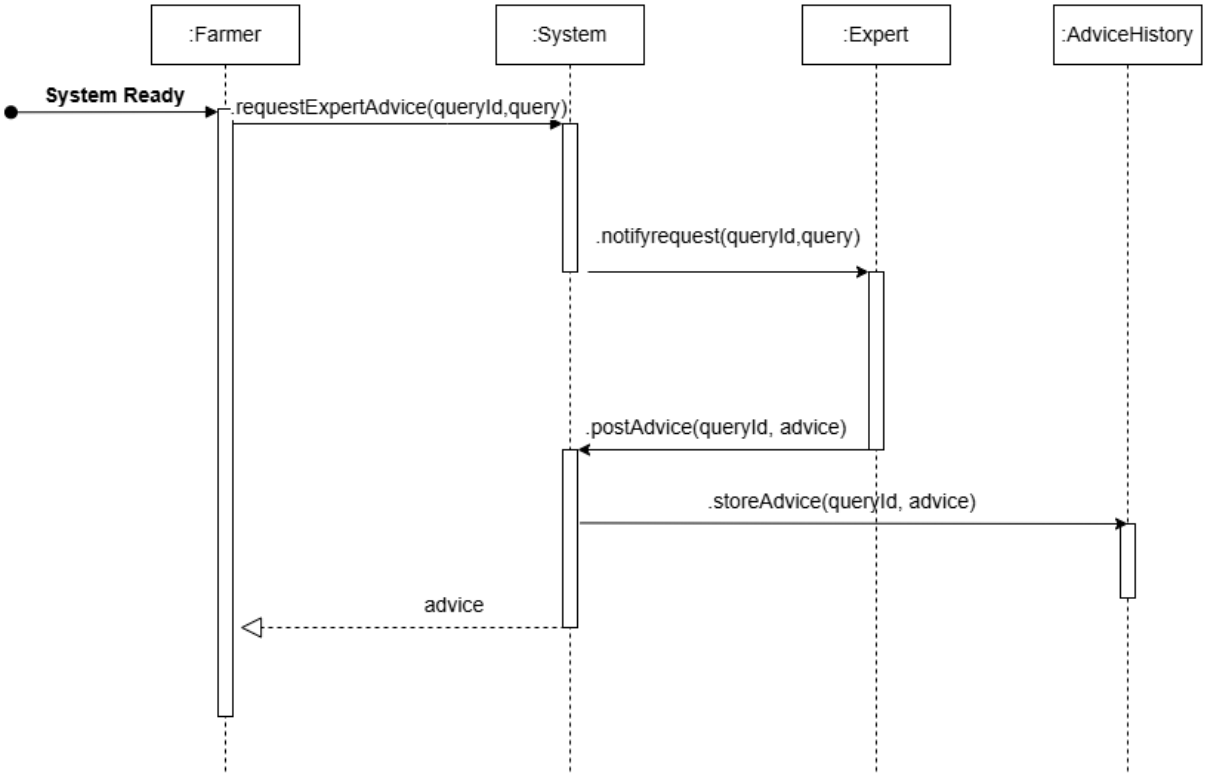

Figure 3.7: Sequence Diagram of Expert Advice

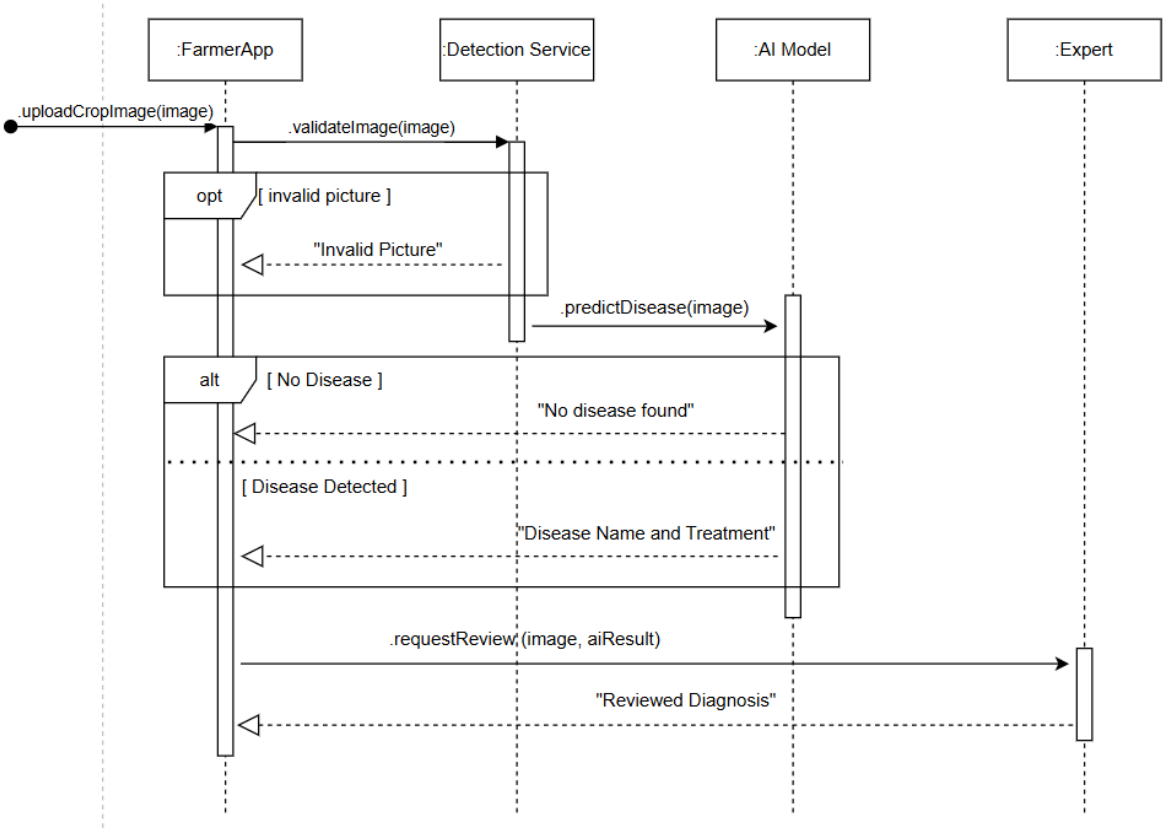

Figure 3.8: Sequence Diagram of Crop Disease Detection

3.8 Safety and Security requirements

3.8.1 Access Requirements

Ensuring security and reliability is critical for a system that manages sensitive user and farm data. The major access-related safety and security requirements are outlined below:

- Only registered and authenticated users (farmers and experts) can access the platform's main functionalities.
- Farmers can log farm activities, view crop health data, and receive alerts, but cannot modify other users' information.
- Agricultural experts can provide recommendations and notifications but cannot access data belonging to unrelated farmers.
- The system administrator has full access for moderation, maintenance, and database management.
- Access control is enforced through secure authentication and authorization mechanisms using encrypted credentials.

3.8.2 Integrity Requirements

- All farm, crop, and user information must remain accurate, consistent, and protected from unauthorized changes.

- Data validation and integrity checks are implemented at both frontend and backend levels.
- All database transactions adhere to ACID (Atomicity, Consistency, Isolation, Durability) principles to ensure data reliability.
- Regular backups and versioning processes are maintained to prevent accidental data loss.

3.8.3 Privacy Requirements

- The platform follows standard data protection practices to safeguard user information.
- Sensitive information such as email, phone number, and farm details is stored in encrypted form.
- User data is not shared with third parties without explicit user consent.
- Secure communication channels (HTTPS, SSL/TLS) are used for all data transmission.
- Users may request deletion or removal of their data from the system at any time.

ed.)

References

[1] "Krishibid.com: Agricultural information, expert consultations, and farming solutions for bangladeshi farmers," https://www.krishibid.com, accessed: 2024.

[2] "ifarmer bangladesh: Digital financial services, farm management tools, and agricultural advisory support," https://www.ifarmer.com.bd, accessed: 2024.

[3] "Amarkrishi: Connecting farmers with agricultural resources, training, and expert guidance," https://www.amarkrishi.com, accessed: 2024.

[4] "Agrostar (india): Crop advisory services, input recommendations, and digital farmer support," https://www.agrostar.in, accessed: 2024.

[5] R. S. Pressman and B. R. Maxim, *Software Engineering: A Practitioner's Approach*, 9th ed. New York, NY, USA: McGraw-Hill Education, 2019.