Faculty of Computer Science and Information Technology

“Plant Disease Prediction”

Submitted by:

#	ID	Name
1	42010092	Ahmed Shafik
2	42010334	Mohamed Tarek
3	42110420	Mohamed Ashraf
4	42010233	Zeyad Ahmed
5	42010030	Anwar Khaled
6	42010586	Ahmed Eldesoky

A dissertation submitted in partial fulfillment of the requirements for the degree of Bachelor of computer science and information technology

Supervised by:

Dr.

Shady Mohamed

Committee Report

We certify we have read this graduation project report as examining committee, examine the student in its content and that in our opinion it its adequate as a project document for "

."

Chairman:
Name:
Signature:
Date: / /2025

Supervisor:
Name:
Signature:
Date: / /2025

Examiner:
Name:
Signature:
Date: / /2025

Egypt 2025

Intellectual Property Right Declaration

This is to declare that the work under the supervision of Dr. Shady Mohamed having title " Plant Disease Prediction ” Carried out in partial fulfillment of the requirements of Bachelor of Science in Computer Science is the sole property of Ahram Canadian University and the respective supervisor. It is protected under the intellectual property right laws and conventions. It can only be considered/ used for purposes like extension for further enhancement, product development, adoption for commercial/organizational usage, etc. with the permission of the University and respective supervisor. This above statement applies to all students and faculty members.

<u>Names:</u>

Ahmed Shafik
Mohamed Tarek
Mohamed Ashraaf
Zeyad Ahmed
Anwar Khalid
Ahmed Desoky

Supervisor: Dr. Shady Mohamed

Anti-Plagiarism Declaration

This is to declare that the above publication produced under the supervision of Dr. shady Mohamed having title " plant disease prediction" is the sole contribution of the author(s) and no part hereof has been reproduced illegally (cut and paste) which can be considered as Plagiarism. All referenced parts have been used to argue the idea and have been cited properly. We will be responsible and liable for any consequence if violation of this declaration is proven.

Names:

Ahmed Shafik
Mohamed Tarek
Mohamed Ashraf
Zeyad Ahmed
Anwar Khalid
Ahmed Desoky

Table of Contents

List of figures 6

List of abbreviation. 7

Chapter 1 introduction 9

1.1 Overview 9

1.2 Motivation: 10

1.3 Objective: 11

1.4 Aim: 12

1.5 Scope: 12

1.6 General Constraints: 13

Chapter 2 Background and Previous work 14

2.1 Background 14

2.2 Previous Works 16

- 2.2.1 Plant Village 16
- 2.2.2 IBM Watson Decision Platform for Agriculture 17
- 2.2.3 Leaf Snap 18
- 2.2.4 Farm Bot 19

Chapter 3 Planning and Analysis 20

3.1 Planning 20

- 3.1.1 Feasibility study and estimated cost 20
- 3.1.2 Gantt chart 22

3.2 Analysis and limitations of existing system 23

3.3 The existing VC system analysis. 23

3.4 Analysis of new system 24

- 3.4.1 User Requirements 24
- 3.4.2 System requirements 24
- 3.4.3 Domain requirements 25
- 3.4.4 Functional requirements 26
- 3.4.5 Non-functional requirements 32

3.5 Advantages of new system 32

3.6 User characteristics 33

Chapter4 design 34

4.1. Use case diagram: 34

- 4.1.1 Use-case diagrams are helpful in the following situations: 34

4.1.2 The following topics describe model elements in use-case diagrams: 34
4.1.3. The steps to design use case diagram: 35
4.2. Sequence diagram 35
4.2.1. Components of a Sequence Diagram 35
4.2.2. Purpose of Sequence Diagrams 36
4.2.3. Creating a Sequence Diagram 36
4.3. Activity diagram 37
4.3.1. Key Components of an Activity Diagram 37
4.3.2. Purpose of Activity Diagrams 38
4.3.3. Steps to Create an Activity Diagram 38
4.4 UML diagrams 39
References 61

List of figures

Figure 1 plant disease classification 14
Figure 2 Plant Village 16
Figure 3 Watson to Watsonx 17
Figure 4 Leafsnap 18
Figure 5 Farm Bot 19
Figure 6 Gantt chart 1. 22
Figure 7 Gantt chart 2. 22
Figure8 Use case Register 39
Figure9 sequence diagram Register 39
Figure10 Activity diagram Register 40
Figure11 Use case Login 40
Figure12 sequence Diagram Login 41
Figure13 Activity Diagram Login 41
Figure14 Use case Image Upload 42
Figure15 sequence Diagram Image Upload 42
Figure16 Activity Diagram Image Upload 43
Figure17 Use case Image Preprocessing 43
Figure18 sequence Diagram Image Preprocessing 44
Figure19 Activity e Diagram Image Preprocessing 44
Figure20 Use case Receive the information 45
Figure21 Sequence Diagram Receive the information 45
Figure22 Activity Diagram Receive the information 46
Figure23 Use case Give solutions 46
Figure24 Sequence diagram Give solutions 47

Figure25 Activity Diagram Give Solutions ... 47
Figure26 Use case Upload Another Photo ... 48
Figure27 Sequence Diagram Upload Another Photo ... 48
Figure28 Activity Diagram Upload Another Photo ... 49
Figure29 Use case Disease Database and Updates ... 49
Figure30   Sequence Diagram Disease Database and Updates ... 50
Figure31 Activity Diagram Disease Database and Updates ... 50
Figure32 Use case Updates and Maintenance ... 51
Figure33 Sequence Updates and Maintenance ... 51
Figure34 Activity Updates and Maintenance ... 52
Figure35 Use case User Interface ... 53
Figure36 Sequence User Interface ... 53
Figure37 Activity Diagram User Interface ... 54
Figure 38 Use Case System Compatibility ... 54
Figure39 Sequence System Compatibility ... 55
Figure40 Activity diagram System Compatibility ... 55
Figure41 Use Case Analytics and history ... 56
Figure42 Sequence Analytics and history ... 56
Figure43 Activity Analytics and history ... 57
Figure44 Use Case Notifications ... 57
Figure45 Sequence Notifications ... 58
Figure46 Activity Diagram Notifications ... 58
Figure47 Use Case Help and Support ... 59
Figure48 Sequence Help and Support ... 59
Figure49 Activity Help and Support ... 60

List of tables

Table 1 function requirement ... 27
Table 2 function requirement ... 28
Table3 function requirement ... 31

List of abbreviation.

8x	AI	Artificial Intelligence
1x	AND	Logical AND
4x	CNNs	Convolutional Neural Networks

1x	FN	False Negative
2x	FP	False Positive
2x	IBM	International Business Machines
1x	ID	Identification or Identifier
3x	IEEE	Institute of Electrical and Electronics Engineers
2x	IoT	Internet of Things
9x	PDD	Plant Disease Detection
1x	PennState	Pennsylvania State University
1x	PMC10965613	PubMed Central reference ID: 10965613
1x	ResNet	Residual Network
1x	SPE	SPECIFICITY
1x	SVM	support vector machines
3x	TN	True Negative
5x	TP	True Positive
1x	VGG	Visual Geometry Group
1x	XPLORE	IEEE Xplore (digital library for research articles)
1x	UML	Unified Modeling Language
14x	F	Function
14x	FR	Functional Requirement
2x	FAQ	Frequently Asked Questions
3x	PNG	Portable Network Graphics
3x	JPG	Joint Photographic Experts Group

Chapter 1
introduction

1.1 Overview

Plant disease prediction is a critical field within agricultural technology that focuses on identifying and forecasting plant diseases to prevent significant crop losses. As global food demand rises, effective disease management has become essential to ensure food security and sustainable farming practices. Early and accurate prediction of plant diseases can help farmers take preventative measures, limit pesticide use, and increase crop yields.

Purpose and Importance

The primary purpose of plant disease prediction is to detect diseases early, ideally before visible symptoms appear, to enable timely intervention. With predictive tools, farmers can target affected areas with precision, which reduces pesticide use, prevents widespread crop loss, and lowers costs. Ultimately, this approach helps increase agricultural productivity while promoting environmentally responsible practices.

Key Technologies and Approaches

1. **Image Processing and Analysis**
   Many plant diseases present visible signs on leaves, stems, or fruits, such as discoloration, spots, or wilting. By capturing high-resolution images, image processing techniques can identify specific patterns and features associated with diseases.
   - **Feature Extraction**: Techniques like edge detection, texture analysis, and color histograms help isolate diseased areas, highlighting symptoms specific to certain diseases [1].
2. **Machine Learning and Deep Learning Models**
   Machine learning and deep learning algorithms are essential for building reliable prediction models.
   - **Classification Algorithms**: Models like support vector machines (SVM) and decision trees are often trained to classify plants as “healthy” or “diseased” [2].

- **Convolutional Neural Networks (CNNs)**: CNNs are widely used for analyzing plant images due to their effectiveness in identifying complex visual patterns. Pretrained models such as ResNet, Inception, or VGG can be fine-tuned to detect specific diseases [3].

3. **Data Collection and Environmental Sensors**
   Data collection is foundational for effective prediction models [2].
   - **Field Data**: Field-based images and data gathered under varying conditions make models adaptable to real-world scenarios [2].
   - **IoT and Environmental Monitoring**: Sensors track critical environmental variables like temperature, humidity, and soil moisture, all of which influence disease spread. These data points allow predictive models to account for environmental factors [2].
4. **Remote Sensing**
   Remote sensing uses satellite or drone imagery to capture plant conditions over large areas, enabling disease detection and analysis on a larger scale. This technology allows for continuous monitoring of crops, providing data for timely disease prediction [1].

1.2 Motivation:

In recent years, global agriculture has faced significant challenges, including climate change, population growth, and the increasing demand for food production. Plant diseases are a major contributor to crop loss worldwide, causing not only economic damage but also impacting food security. Disease outbreaks can spread rapidly if undetected, leading to severe reductions in yield and quality of crops.

Additionally, the excessive use of pesticides to control diseases poses environmental and health risks, which further emphasizes the need for efficient disease management strategies [1].

Plant disease prediction provides a solution that can help mitigate these issues. By predicting and detecting diseases early, farmers can respond quickly, reducing the need for broad pesticide application and preventing large-scale crop loss. For regions heavily dependent on agriculture, this technology can improve food availability, protect farmers' livelihoods, and contribute to economic stability [4].

1.3 Objective:

The main objective of this project is to develop a reliable and efficient model for predicting plant diseases using advanced technologies such as image processing and machine learning. This objective can be broken down into the following specific goals:

1. **To Detect Disease Symptoms in Plant Images**
   Develop an image processing pipeline that accurately detects and isolates disease symptoms on plant leaves, stems, or fruits. This includes identifying specific visual features like color changes, spots, and lesions that are commonly associated with plant diseases [2].
2. **To Classify Plant Diseases Using Machine Learning Models**
   Train and evaluate machine learning models, such as Conventional Neural Networks (CNNs), to classify images based on various types of plant diseases. This involves collecting and labeling data for training and optimizing model performance to achieve high accuracy [2].
3. **To Integrate Environmental Data for Enhanced Prediction Accuracy**
   Incorporate environmental factors—such as temperature, humidity, and soil conditions—that influence disease spread to improve prediction accuracy. This goal involves integrating sensor data to account for the environmental conditions where the plant is located [4].
4. **To Build a User-Friendly Interface for Disease Detection**
   Design an interface or application that allows users, such as farmers or agricultural consultants, to upload plant images and receive disease predictions along with recommended actions. This interface aims to make the model accessible and useful in real-world agricultural settings.
5. **To Evaluate Model Performance Across Various Conditions**
   Test the model under different conditions, such as various lighting, plant types, and environments, to ensure robustness and adaptability. This includes validating the model's accuracy with field data to confirm its effectiveness in diverse agricultural scenarios.
6. **To Minimize Pesticide Use and Improve Crop Health**
   By accurately predicting diseases, the project aims to support more targeted and timely interventions, reducing excessive pesticide use. This contributes to sustainable farming practices, helping farmers maintain healthier crops with minimal environmental impact.

1.4 Aim:

The aim of this project is to develop an accurate, efficient, and accessible plant disease prediction system that leverages machine learning and image processing to detect and classify plant diseases at an early stage.

By creating a tool that enables farmers and agricultural professionals to identify diseases before they spread, the project aims to reduce crop loss, optimize pesticide usage, and promote sustainable farming practices. This tool ultimately seeks to support food security, improve crop health, and provide farmers with actionable insights to enhance agricultural productivity and resilience.

1.5 Scope:

The scope of this project is to develop an AI-powered system for predicting plant diseases, with a primary focus on essential crops such as wheat, tomatoes, and beans. This project will create a comprehensive model that analyzes images and environmental data to detect disease symptoms, supporting farmers in effective crop management.

Data will be collected from diverse sources, including plant images, environmental records, and agricultural practices, ensuring robust and accurate predictions.

The system will also include a detailed reference library of ideal environmental conditions for each crop, covering soil type, temperature, humidity, and best farming practices. Machine learning algorithms will process and analyze this data, enabling disease identification and providing recommendations on treatment methods.

The project will further evaluate model performance, refining and adjusting it to enhance prediction accuracy and reliability. Ultimately, this system aims to improve crop health by enabling early disease detection, benefiting both local farmers and agricultural productivity [2].

1.6 General Constraints:

1. Hardware Constraints

- Hardware constraints are less critical for websites but can influence user experience on low-end devices.

2. Software Constraints

- React.js and Django are ideal for responsive, cross-browser web applications.

3. Storage and Memory Constraints

- Backend efficiency is crucial, but storage/memory constraints on devices are less critical since websites typically don't store significant data locally.

4. Connectivity Constraints

- Offline functionality is less common for websites, but it can be achieved using technologies like Progressive Web Apps (PWAs) to enable offline browsing and data storage.

Chapter 2
Background and Previous work

2.1 Background

Plant diseases have a global effect on plant production. Therefore, farmers are supposed to attain expert knowledge and thorough training to distinguish early plant pests or viral symptoms and take appropriate action to prevent disease continuity. PDD control can help economic development by reducing hunger and saving the environment through reduced chemical fertilizer utilization.

Environmental factors primarily cause plant diseases, and pathogens (like bacteria, worms, viruses, fungi, and protozoa) are defined as diseases in plant pathology. Several plant diseases commonly occur because of a variety of factors

. For instance, depending on its nature, soil, seed, or air type, it can be caused by a pandemic, epidemic, or endemic. Other factors include symptoms and significant causes such as blight, rots, and viruses, as shown in Fig. 1. Early symptoms are essential in PDD because conventional traits for identifying in-field diseases are based on several factors, including the type of disease, its color, pattern, appearance, and location on the plant. These symptoms on various sections of plants, like the stems, fruits, and leaves, among others, are primarily utilized in PDD.

Farmers and agricultural experts use traditional surveillance to identify disease categories. High-tech agricultural systems that use vision-based learning approaches for PDD can effectively increase crop yields. There has been an improvement in efficiency in identifying plant diseases using AI for early diagnosis and smart inspection automation. Few valid case studies in third-world countries use automatic approaches in agriculture. Despite the contributions of the remarkable endeavors available, a few factors continue to make real-time PDD complex [5].

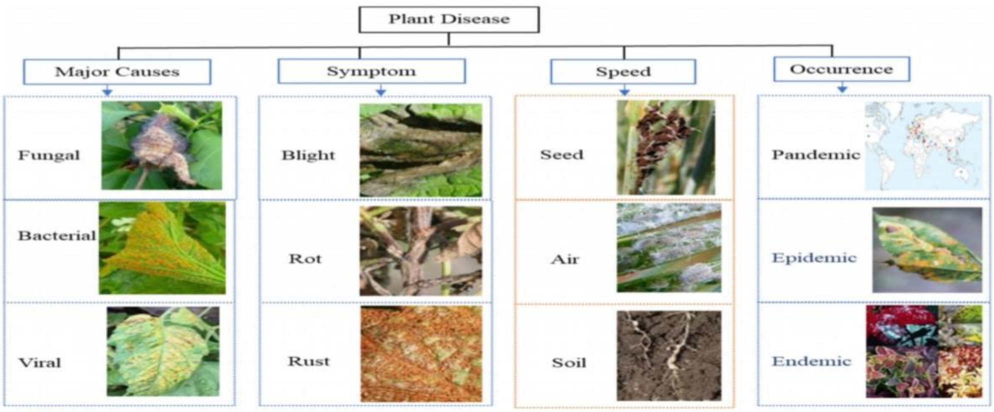


Figure 1 plant disease classification

Farmers and agricultural experts use traditional surveillance to identify disease categories. High-tech agricultural systems that use vision-based learning approaches for PDD can effectively increase crop yields.

There has been an improvement in efficiency in identifying plant diseases using AI for early diagnosis and smart inspection automation. Few valid case studies in third-world countries use automatic approaches in agriculture. Despite the contributions of the remarkable endeavors available, a few factors continue to make real-time PDD complex.

The main objective of this review is to present techniques, available datasets, and challenges in plant disease detection that need to be addressed to develop comprehensive, intelligent agricultural methods for monitoring and diagnosing early plant detection. The following are the main other contributions to this review:

• Based on the current surveys, this review is the first systematic study on image-based PDD approaches covering both localization and disease classification.

• The study shows a complete data collection and prepossessing strategy for PDD used in academia and business.

• The study also looks at three crucial methods: deep learning, machine learning, and image processing relating to plant diseases.

• Deep learning methods and their applicability to Interceded smart farming solutions are also investigate.

• All public datasets that can be used with this research paradigm are examined in the study, and their details are given [5].

ACCURACY

The term ‘‘accuracy’’ is the proportion of correct predictions made compared to the total number of data points-collected (T).

In scientific literature, it is referred to as recognition, correctness, or success rate.
Accuracy = (TN + TP)/T (1) [6]

PRECISION

Precision is described as the proportion of actual positive samples found to the total samples anticipated to be positive.

Precision = TP/ (TP + FP) (2) [6]

RECALL AND SENSITIVITY

The term ‘‘sensitivity’’ or ‘‘recall’’ refers to the proportion of correctly anticipated positives to the total number of actual positive results.

Recall = TP / (FN + TP) (3) [6]

F1-SCORE

The F1-score is a definition that refers to the harmonic mean of both precision and sensitivity (recall).

F1 = 2 ∗ (Recall ∗ Precision) / (Recall + Precision) (4) [6]

SPECIFICITY (SPE)

This is the ratio of precisely anticipated negatives relative to the total number of observed negatives.

Specificity = TN/ (FP + TN) (5) [6]

2.2 Previous Works

2.2.1 Plant Village

Plant Village is an AI-powered platform designed to assist farmers in diagnosing plant diseases using smartphone images. It uses machine learning models, such as convolutional neural networks (CNNs), to identify diseases from leaf images and provide recommendations. However, the system's accuracy can decrease in cases of low-resolution images or poor lighting conditions [7].

Figure 2 Plant Village

Similarities with our project:

1. Provides disease detection based on plant images.
2. Employs machine learning for classification.
3. Aims to improve agricultural productivity.

Differences from our project:

1. Does not integrate environmental data, such as soil moisture or humidity.
2. Limited user interface for localized advice compared to our application.
3. May struggle with accuracy under varied real-world conditions.

2.2.2 IBM Watson Decision Platform for Agriculture

IBM's platform integrates AI, satellite imagery, and IoT to offer precision farming solutions, including disease prediction. It provides predictive insights based on historical weather data and remote sensing. However, its high implementation cost limits accessibility for small-scale farmers [8].

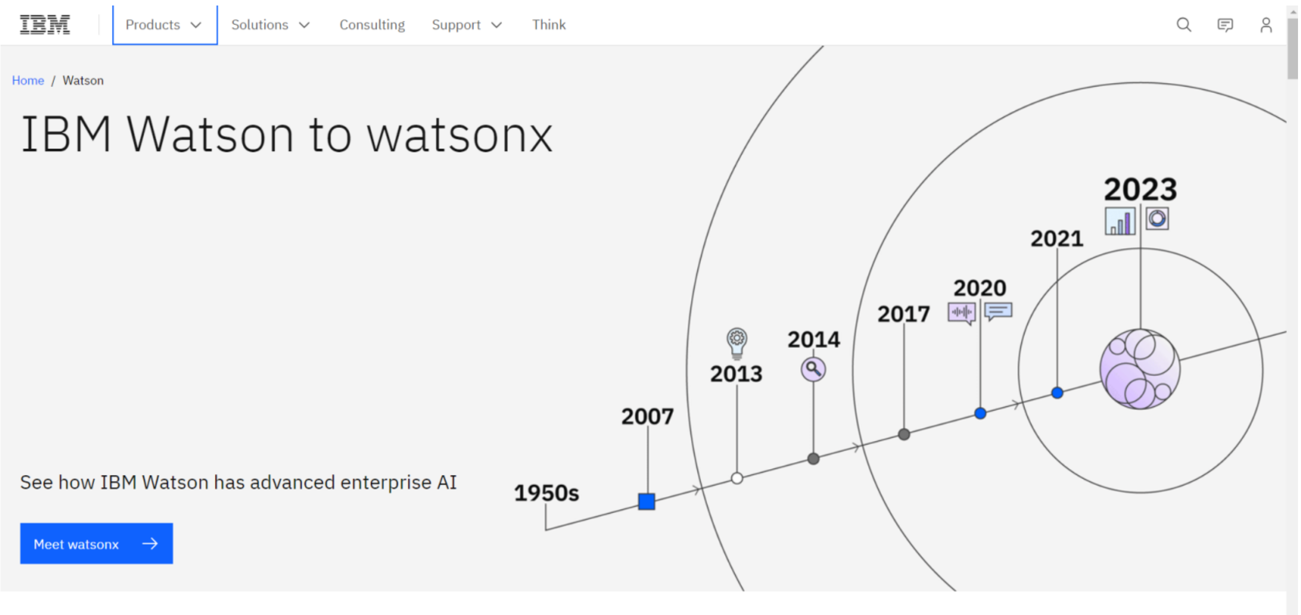


Figure 3 Watson to Watsonx

Similarities with our project:

1. Incorporates AI and machine learning for predictive insights.
2. Utilizes remote sensing for large-scale monitoring.
3. Aims to optimize farming efficiency and reduce losses.

Differences from our project:

1. Focuses on broader agricultural insights rather than disease-specific predictions.
2. High cost compared to our affordable, user-friendly solution.
3. Less emphasis on real-time, on-the-ground image analysis.

2.2.3 Leaf Snap

Leaf Snap is a mobile application that uses image recognition technology to identify plant species based on leaf images. While its primary focus is plant identification rather than disease detection, it has laid the groundwork for integrating AI into agricultural tools. However, its functionality is limited to species identification without extending to disease analysis or prediction [9].

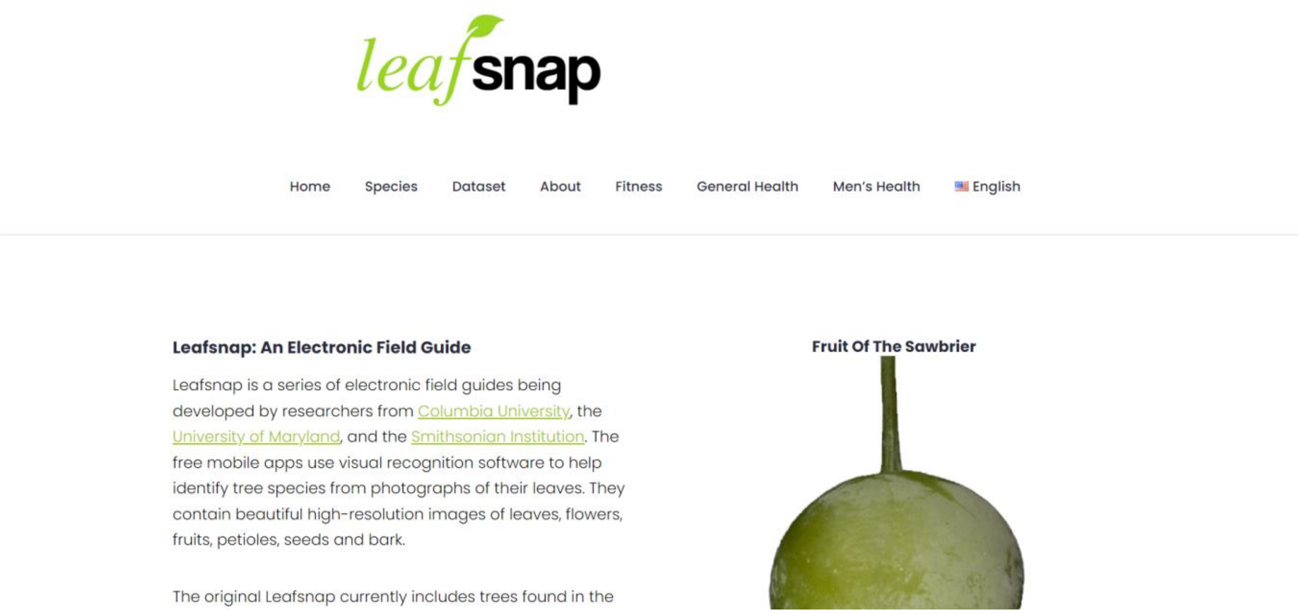


Figure 4 Leafsnap

Similarities with our project:

1. Utilizes image recognition technology.
2. Aims to assist users through mobile-based applications.
3. Focuses on plants as the primary subject of analysis.

Differences from our project:

1. Does not analyse or predict plant diseases.
2. Lacks integration of environmental data and actionable recommendations.
3. Limited application scope compared to our disease-specific tool.

2.2.4 Farm Bot

Farm Bot is a smart farming solution that integrates robotic technology with AI to monitor and manage crops. It includes features for disease detection, watering, and fertilization. However, the high cost and complexity of its setup make it impractical for most farmers [10].

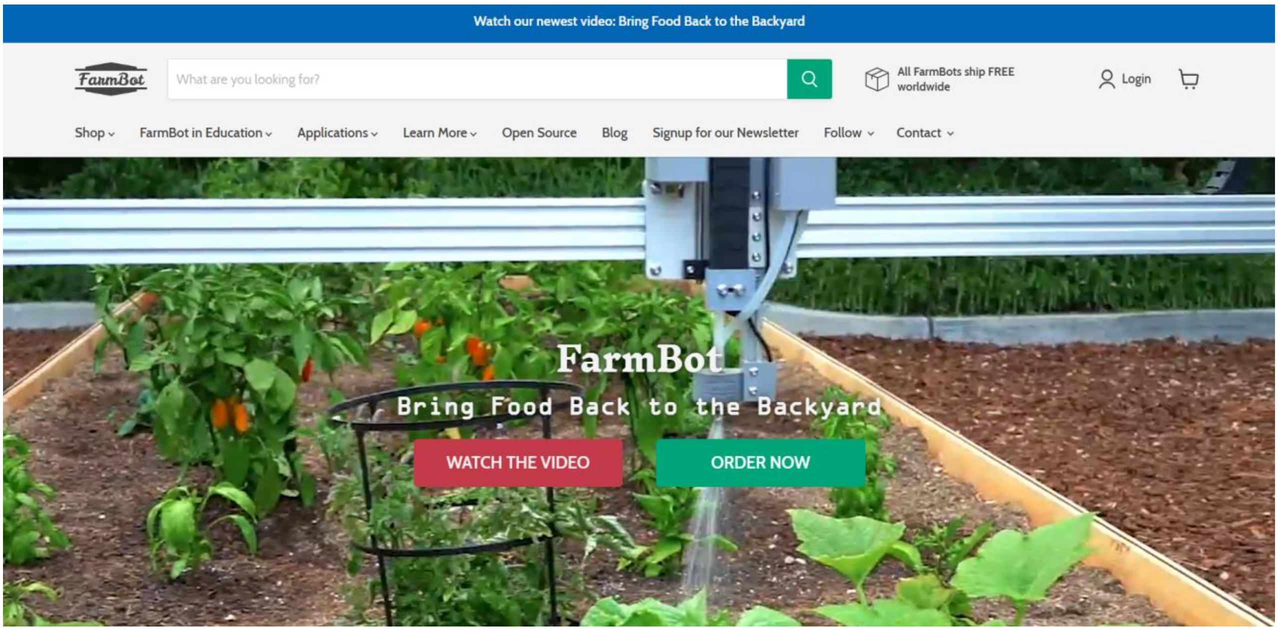


Figure 5 Farm Bot

Similarities with our project:

1. Focuses on improving agricultural efficiency.
2. Includes disease detection as part of its functionality.
3. Leverages AI for precision farming.

Differences from our project:

1. Relies on robotic hardware, increasing costs significantly.
2. Designed for small-scale, automated farms rather than large or traditional ones.
3. Does not prioritize user accessibility for diverse farming scenarios.

Chapter 3
Planning and Analysis

3.1 Planning

3.1.1 Feasibility study and estimated cost

A feasibility study is an initial investigation of the potential benefits and viability of a project. Technical, economic, budgetary, legal, and environmental factors are all examined in an impartial evaluation that is the goal of a feasibility study. Our project involves advanced technologies like image processing and machine learning. Below is the analysis:

Required Skills:

- Knowledge in machine learning.
- Experience in image processing techniques.

Hardware Requirements:

- High-resolution cameras or smartphone cameras for image capture.

Software tools:

- Frameworks for machine learning (TensorFlow, PyTorch).
- A development platform for user interface (e.g., Flutter, React).

Challenges:

- Collecting quality training data.
- Ensuring accuracy under varying environmental conditions.
- Developing a cost-effective and user-friendly application.

In this blog, we will discuss what feasibility analysis is, the various types of feasibilities to evaluate, and the importance of feasibility analysis for app success. Moreover, it will also be clear how to do effective feasibility study for our mobile app project.

The first concern regarding feasibility study is the target of the application, and our application will have a big impact for helping us to know the condition of the planet.

Another concern is features of the application, the app is designed in a user-friendly way that will not let the user bored from using it and the way every tab or widget is placed. We have high hopes and standards for our project.

Economic feasibility

The shift to a website-based system for plant disease prediction reduces development complexities, provides broader accessibility, and aligns with agricultural stakeholders who may prefer web-based solutions over mobile applications. Below is the analysis of its economic feasibility.

Development Costs:

- UI/UX Designers.
- Open-source frameworks:
  - Frontend: React.js, Vue.js.
  - Backend: Node.js, Flask, or Django.
  - AI/ML: TensorFlow, PyTorch.
  - Hosting platform: Cloud services like AWS, Google Cloud, or Azure.

Advertisement Revenue:

- Display targeted ads for agricultural products, fertilizers, or equipment.

  Estimated cost: $5,000–$10,000.

- **Feasibility Assessment.**
  Reduced Costs: Compared to mobile programs with comparable functionalities, websites are typically less expensive to design and maintain.
  Global Reach: Websites may be accessed on any device with an internet connection, which expands their user base.
  Savings for Farmers: The platform attracts end users due to the substantial cost savings in crop loss and pesticide use. [11]

3.1.2 Gantt chart

PROJECT NAME

PLANT DISEASE PREDICTION

2024

				MONTH	September					October					November					December				
				WEEK START DATE	2	9	16	23	30	6	13	20	27	-	4	11	18	25	-	1	8	15	22	29
ACTIVITY	Supervised by	ASSIGNED TO	Chapters	Done	1	2	3	4	5	6	7	8	9	-	10	11	12	13	-	14	15	16	17	18
Meeting	Dr.Shady	Whole team																						
Project Research		Whole team		✓	✓	✓	✓																	
Gathering information		Whole team		✓		✓	✓	✓																
Overview		Mohamed ashraf	1	✓			✓	✓																
Motivation		Ahmed shafik	1	✓			✓	✓																
Objective		Anwar khaled	1	✓			✓	✓	✓															
Aim		Zeyad hendawy	1	✓			✓	✓	✓															
Scope		Mohamed tarek	1	✓			✓	✓	✓															
General constraints		Ahmed eldesoky	1	✓			✓	✓	✓															
Organization the dissertation		Whole team	1	✓			✓	✓	✓															
Doctor meeting		Whole team		✓																				
Start searching for ch2		Whole team	2	✓						✓	✓													
Background		Mohamed Ashraf Anwar Khaled Zeyad Hendawy	2	✓							✓	✓	✓	✓										
Previous work		Ahmed Shafik Mohamed Tarek Ahmed eldesoky	2	✓							✓	✓	✓	✓										
Finish ch2			2	✓											✓									
Group meeting		Whole team		✓											✓	✓								

Figure 6 Gantt chart 1.

PROJECT NAME

PLANT DISEASE PREDICTION

2024

				MONTH	September					October					November					December				
				WEEK START DATE	2	9	16	23	30	6	13	20	27	-	4	11	18	25	-	1	8	15	22	29
ACTIVITY	Supervised by	ASSIGNED TO	Chapters	Done	1	2	3	4	5	6	7	8	9	-	10	11	12	13	-	14	15	16	17	18
Meeting	Dr.Shady	Whole team																						
Chapter 3 assigned		Whole team	3	✓												✓								
Gathering information		Whole team	3	✓												✓	✓							
Feasibility study and estimated cost		Mohamed ashraf	3	✓													✓	✓						
Gantt chart		Ahmed shafik	3	✓													✓	✓						
Analysis and limitations of existing system		Anwar khaled	3	✓													✓	✓						
The existing VC system analysis		Zeyad hendawy	3	✓													✓	✓						
Analysis of new system		Mohamed tarek	3	✓													✓	✓						
Advantages of new system		Ahmed eldesoky	3	✓													✓	✓						
User characteristics		Whole team	3	✓													✓	✓						
Chapter 3 finished		Whole team	3	✓														✓						
Doctor meeting		Whole team																						
Group meeting		Whole team		✓																				

Figure 7 Gantt chart 2.

3.2 Analysis and limitations of existing system

When analyzing existing systems in plant disease prediction, several platforms and technologies provide valuable insights. Below, we review key systems, their features, and their limitations with respect to website development. There are plenty of plant disease prediction mobile applications that provide the same concept but were not accurate enough and had a lot of bugs and issues, and other apps with old version that was not updated for years like (LeafSnap – PlantVillage – Farmbot - etc).

Our plan is to try to build better website than the ones available at the moment in terms of design, performance, features etc

3.3 The existing VC system analysis.

Our website provides techniques using Artificial Intelligence for plant disease prediction. It is essential to address the limitations of existing systems and meet the growing demands of modern agriculture .Our website-based system for plant disease prediction is essential to bridge the gaps in current solutions and provide a user-friendly, efficient, and scalable tool for farmers.

To analyze an existing Visual Computing (VC) system, particularly in the context of plant disease prediction .The goal of the VC system is to use machine learning and image processing methods to recognize and forecast plant illnesses from photos taken by farmers or other agricultural professionals. [12]

Components: usually consists of a user interface for interaction, machine learning models, image processing algorithms, and picture capture devices (such as smartphones).Image Processing Techniques: Critical algorithms include machine learning for classification and feature extraction.

Machine Learning Models: To train models for disease diagnosis, machine learning frequently uses deep learning frameworks that can manage enormous datasets of plant photos.High-resolution plant photos, environmental information (temperature, humidity), and past disease incidence statistics are examples of input data.

Output: Proposed interventions or therapies, confidence scores, and disease projections.

In conclusion

To modernize agriculture, a new website for predicting plant diseases is essential because it offers:

1-Accessible and reasonably priced disease detection.

2-localized and useful information for farmers.

3-Accurate, real-time, and ecologically integrated predictions.

This technology can empower farmers, boost production, and guarantee sustainable agricultural practices by making use of the prevalent use of website.

3.4 Analysis of new system

3.4.1 User Requirements

To ensure website for plant disease prediction meets user needs, a detailed analysis of user requirements is essential. These requirements can be categorized into functional, non-functional aspects, in such a way that they are understandable by system users who do not have detailed technical knowledge.

The app will be used by farmers and individuals too:

- Open the camera and snap a picture for the environment and the planet.
- Give me with facts and specifics regarding any diseases or diagnoses the planet may have after examining the picture. [13]

3.4.2 System requirements

System requirements are the characteristics and specifications that a software application or mobile device must have in order to operate correctly. The defining of system requirements is a crucial stage in the development process that ensures users, developers, and stakeholders are aware of what is needed for the system to function as intended. These requirements cover a widerange of topics, including compatibility requirements, software dependencies, hardware specs, and performance expectations. System requirements provide a roadmap for the whole development lifecycle and form the basis for system design, development, and testing. [14]

System Requirements such as:

- Database: they are the most common way to store and manage data used by applications, through which we can store the data of the app such as the signs and their meaning or translation, and some information about the user such as their name and age.
- Performance Optimization: Optimize system performance to ensure fast loading times, and smooth user experience. use efficient algorithms, and performance testing.
- System Maintenance and Updates: For regular system maintenance, bug fixes, improve performance, and add new features if available.
- Processor: Define the minimum processor requirements, such as ARMv7 or higher for Android or A7 or higher for iOS.
- Memory (RAM): Specify the minimum amount of RAM required for your app to function properly. For example, 2GB or 4GB.
- Storage Space: Indicate the minimum available storage space required for installing the app. Users need to have enough free space on their devices to accommodate the app and any data it generates.
- Display: Specify the minimum screen size and resolution for optimal user
- Experience. Consider different screen sizes and densities for various devices.

3.4.3 Domain requirements

Domain requirements outline the demands and limitations of the plant health and agriculture domain that the application must take into account. These specifications guarantee that the system complies with the operational procedures and expectations of its users in the agriculture sector.

- The website must facilitate the identification and categorization of illnesses in widely grown crops, such as wheat, tomatoes, and apples.
- Future extensions to incorporate more crops and disease kinds ought to be supported.
- Identify visible symptoms on leaves, stems, and fruits, such as browning, spots, lesions, or wilting.

- Employ machine learning models that were trained on a dataset of photos of plants that have been identified as having particular diseases.
- Give guidance on preventative and therapeutic approaches that are in line with accepted farming methods.
- Provide advice on environmentally friendly techniques like integrated pest management or organic cures.
- Different lighting and weather conditions.
- Diverse plant growth stages.
- Regional differences in disease presentation.

3.4.4 Functional requirements

Functional requirements, which outline what features should be included in the system to support a task that the user wants to do, are obtained during the analysis phase. In addition to outlining how the system should operate, functional requirements should also contain details about its services, expected results in a particular setting, and the data required to build the system. The development of functional requirements, which provide specific functionalities that illustrate how user requirements are satisfied, occurs during the planning stage.They must be made clear to both developers and stakeholders.

FR NO	**Functional requirements**	**Description**
Fr1	Register	Users can create an account by providing: • Username. • Email address. • Password**.**
Fr2	Login	• Users can log in using: • Registered email and password. • Third-party login credentials (if implemented) • Implement "Forgot Password" functionality for account recovery.
Fr3	Image upload	• Users can upload images of plant leaves, stems, or fruits in supported formats (JPG, PNG). • Display a progress bar or notification during upload.

Table 1 function requirement

Fr4	Image preprocessing	• Preprocess uploaded images to improve analysis accuracy: • Resize images to a standard resolution. • Enhance contrast and sharpness. • Detect and isolate the region of interest (the affected area).
Fr5	Receive the information	Provide users with the following outputs after analysis: • Whether the plant is healthy or diseased. • If diseased, display: • Disease name. • Affected crop type.

Table 2 function requirement

Fr6	Give solutions	Offer actionable recommendations based on the detected disease: • Treatment options (pesticides, organic remedies). • Preventive measures. • Environmental adjustments (watering, sunlight exposure).
Fr7	Upload another photo	• Allow users to analyze multiple images in a single session. • Display results for each uploaded image separately.
Fr8	Disease Database and Updates	Disease Database • Maintain a comprehensive, searchable database of diseases, including: • Disease name and symptoms. • Affected crops. • Recommended treatments. Updates • Periodically update the database to:Add new diseases and crops.

Fr9	Updates and Maintenance	Regularly update the system for: • Bug fixes and feature enhancements. • Integration of new AI models for better prediction accuracy. • Website security patches to protect user data.
Fr10	User Interface	Provide an intuitive and easyto-navigate interface: • Simple menu structure with clear labels (“Upload Image," "View Results," "Database"). • Interactive elements (buttons, tooltips) to guide users.
Fr11	System Compatibility	• Ensure compatibility with: • All major browsers (Chrome, Firefox, Edge, Safari). • Devices (desktops, tablets, smartphones). • Support responsive design for optimal viewing on all screen sizes

Fr12	Analytics and History	Store and display a history of uploaded images and their results for registered users.
Fr13	Notifications	Notify users about new features, database updates, and maintenance schedules
Fr14	Help and Support	Include a help section with FAQs and contact options for technical support.

Table 3 function requirement

Conclusion:

These functional requirements provide smooth user experience, actionable advice, and precise illness diagnosis from the application. To give farmers and individuals a fast response, they emphasize adaptability, accessibility, and integration with . [15] [16]

3.4.5 Non-functional requirements

A non-functional requirement lists the system's limitations and features. Product requirements are a common requirement that outline how the given product must function, including execution speed, dependability, while organizational requirements are derived from organizational and procedural norms, such as

process standards used for execution, external requirements originate from sources outside the system.
The development process includes requirements for interoperability and legislation.
Next, the following non-functional requirements need to be fulfilled by the system:

- All platforms and device types should have simple designs that are simple to use for all users.
- Allow many people at the same time without any damage or performance to be lowered.
- Availability at any time.
- Reliability using of applications at any time.
- Usability making the application so easy to use.
- The application should recover automatically from crashes or interruptions without losing user data.
- The system must be able to scale horizontally to support a growing number of users without performance degradation.
- The application should be compatible with (Android devices – iPhone devices)

Conclusion:

These non-functional requirements guarantee that the website complies with technical, legal, and environmental requirements while being scalable, secure, and user-friendly. [17]

3.5 Advantages of new system

Compared to current methods, the suggested website for plant disease prediction has several advantages. These advantages make use of cutting-edge technologies to overcome the shortcomings of existing systems and give farmers a more effective, accessible, and user.

The proposed application has these advantages:

- The proposed system helps to eliminate unnecessary pesticide use through targeted disease detection, saving farmers money.
- The proposed system helps the user to know if the planet has disease or no.
- The proposed system supports online functionality, enabling usage in areas with limited or no internet connectivity.
- The proposed system gives the user real results for the planet and environmental conditions.

3.6 User characteristics

There are only one type of user (farmer & individuals) our website is an online website the farmer register and login, the farmer open the place for taking or upload the photo of the plant or environment to detect if it is health or infected. after detecting it shows its condition good or parasite and the way to handle it if it was parasite.

The second hand is the admin that uploads the new dataset and works on the database to enhance the performance of the website.

Chapter4 design

4.1. Use case diagram:

Use-case diagrams illustrate and define the context and requirements of either an entire system or the important parts of the system. You can model a complex system with a single use-case diagram or create many use-case diagrams to model the components of the system. You would typically develop use-case diagrams in the early phases of a project and refer to them throughout the development process.

4.1.1 Use-case diagrams are helpful in the following situations:

Before starting a project, you can create use-case diagrams to model a business so that all participants in the project share an understanding of the workers, customers, and activities of the business.

While gathering requirements, you can create use-case diagrams to capture the system requirements and to present to others what the system should do.

During the analysis and design phases, you can use the use cases and actors from your use-case diagrams to identify the classes that the system requires.

During the testing phase, you can use use-case diagrams to identify tests for the system.

4.1.2 The following topics describe model elements in use-case diagrams:

Use cases.

A use case describes a function that a system performs to achieve the user's goal. A use case must yield an observable result that is of value to the user of the system.

Actors

An actor represents the role of a user that interacts with the system that you are modeling. The user can be a human user, an organization, a machine, or another external system.

Subsystems

In UML models, subsystems are a type of stereotyped component that represent independent, behavioral units in a system. Subsystems are used in class, component, and use-case diagrams to represent large-scale components in the system that you are modeling.

Relationships in use-case diagrams

In UML, a relationship is a connection between model elements. A UML relationship is a type of model element that adds semantics to a model by defining the structure and behavior between the model elements.

4.1.3. The steps to design use case diagram:

A Use Case model can be developed by following the steps below:

1. Identify the Actors (role of users) of the system.
2. For each category of users, identify all roles played by the users relevant to the system.
3. Identify what the users required the system to be performed to achieve these goals.
4. Create use cases for every goal.
5. Structure the use cases.
6. Prioritize, review, estimate and validate the users.

Note that: to make use case approach more "Agile", do not detail all use cases, but prioritize them in your product backlog, you should refine the use case in different level of details according to the development phase with just-in-time and just-enough manner.

4.2. Sequence diagram

A sequence diagram is a form of interaction diagram that demonstrates the sequential interactions of processes with one another. It is utilized within the Unified Modeling Language (UML) to depict the dynamic elements of a system. Sequence diagrams are particularly effective for illustrating and recording the reasoning behind a process or the interaction among various system components.

4.2.1. Components of a Sequence Diagram

- **Actors:** These signify external entities that engage with the system. They can include users, external systems, or devices.
- **Objects:** These represent the components or entities of the system that take part in the interaction. Objects are illustrated as rectangles with their names underscored.
- **Lifelines:** These indicate the presence of an object or actor throughout the interaction. They are shown as vertical dashed lines.
- **Messages:** These denote communication among objects.

- **Synchronous Messages:** Depicted as solid arrows and necessitate a response.
- **Asynchronous Messages:** Illustrated as open arrows and do not require an immediate reply.
- **Return Messages:** Represented as dashed arrows, indicating the feedback to a synchronous message.
- **Activation Bars:** These signify the duration during which an object is engaged in a task or is active.
- **Loops and Conditions:**
  - **Loops:** Employed to illustrate repetitive actions, contained within a rectangular frame labelled "loop".
  - **Conditions:** Indicate decisions, typically marked with "alt" (alternative) or "opt" (optional).

4.2.2. Purpose of Sequence Diagrams

1. **Modelling Scenarios:** Aids in visualizing how various objects and components collaborate over time in particular use cases or scenarios.
2. **Understanding System Behaviour:** Offers clarity on the flow of messages and responses within a system.
3. **Documentation:** Functions as a means of documenting system processes and workflows.
4. **Communication:** Serves as a foundational guide for developers and stakeholders to comprehend system interactions.

4.2.3. Creating a Sequence Diagram

1. **Identify the Scenario:** Specify the exact use case or process to represent.
2. **Determine Participants:** Recognize the actors and objects involved in the interaction.
3. **Map the Interactions:** Outline the sequence of messages exchanged between participants.
4. **Add Conditions and Loops:** Incorporate decision points or recurring actions as necessary.
5. **Validate the Diagram:** Ensure that the flow accurately reflects the intended behavior.

4.3. Activity diagram

An activity diagram is a behavioral diagram within the Unified Modeling Language (UML) that illustrates the progression of activities or processes in a system. It captures the dynamic elements of a system by concentrating on the order of actions and the movement of control or data. Activity diagrams are frequently utilized for workflow evaluation, system development, and the comprehension of intricate operations.

4.3.1. Key Components of an Activity Diagram

1. **Activities:**
- Indicate tasks or actions carried out in the workflow.
- Represented as rounded rectangles.

2. **Transitions (Arrows):**
- Show the flow of control or data among activities.

3. **Initial Node:**
- Signifies the beginning of the workflow.
- Illustrated as a filled black circle.

4. **Final Node:**
- Denotes the conclusion of the workflow.
- Illustrated as a circle containing a filled black circle inside it.

5. **Decision Nodes:**
- Indicate points of decision where the flow diverges based on conditions.
- Illustrated as a diamond shape.

6. **Merge Nodes:**
- Consolidate multiple flows into a single one, typically after a decision.
- Also illustrated as a diamond shape.

7. **Fork and Join Nodes:**
- **Fork:** Divides the flow into several concurrent paths.
- **Join:** Brings together multiple paths into a single one.
- Both are represented as thick horizontal or vertical bars.

8. **Swimlanes:**

- Segment the diagram into parts to depict various actors, roles, or system components tasked with specific activities

9. **Object Flows:**

- Illustrate the transfer of objects or data between activities.
- Represented as arrows connecting activities.

4.3.2. Purpose of Activity Diagrams

- **Visualizing Workflows:** Aids in grasping and documenting the order of activities in a system or process.
- **Analysing Processes:** Pinpoints inefficiencies, bottlenecks, or redundancies in workflows.
- **System Design:** Represents the logic of complex systems or subsystems, such as algorithms or business processes.
- **Communication:** Functions as a clear, visual depiction for stakeholders and developers.

4.3.3. Steps to Create an Activity Diagram

1. Define the Process:

- Determine the workflow or use case to represent.

2.Identify Activities:

- Break the process down into distinct tasks or actions.

3.Determine the Flow:

- Establish the sequence of activities, decisions, and concurrent actions.

4.Add Decision Points:

- Indicate where the process diverges based on conditions.

5.Include Swimlanes:

- Organize activities according to roles or system components if necessary.

6.Validate the Diagram:

- Confirm that the flow accurately reflects the intended process. [18]

4.4 UML diagrams

F1:Register

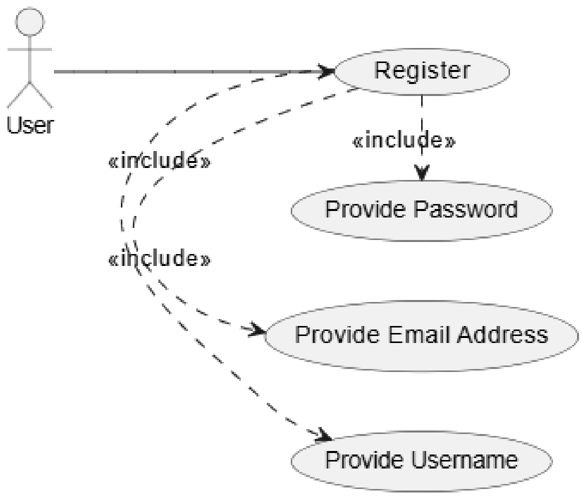


Figure 8 Use case Register

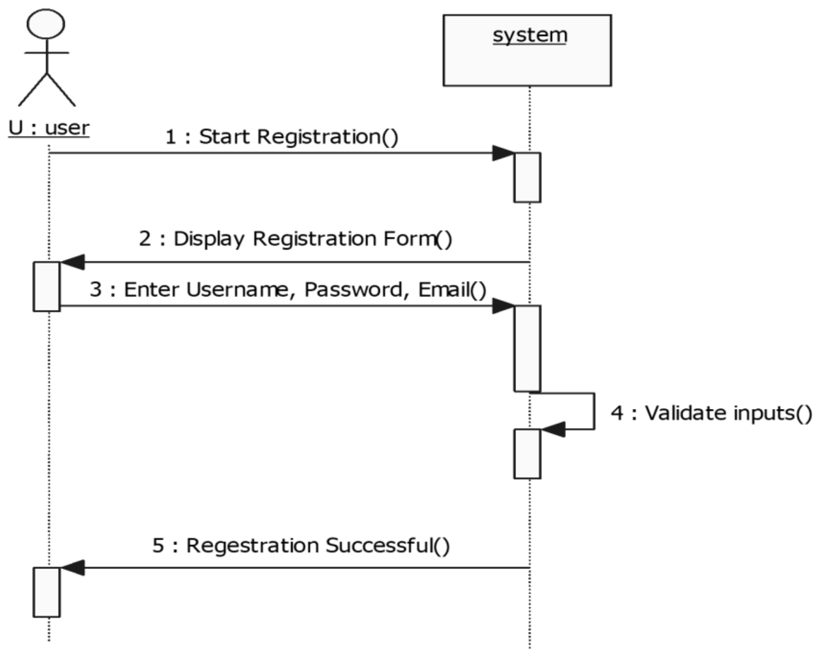


Figure 9 sequence diagram Register

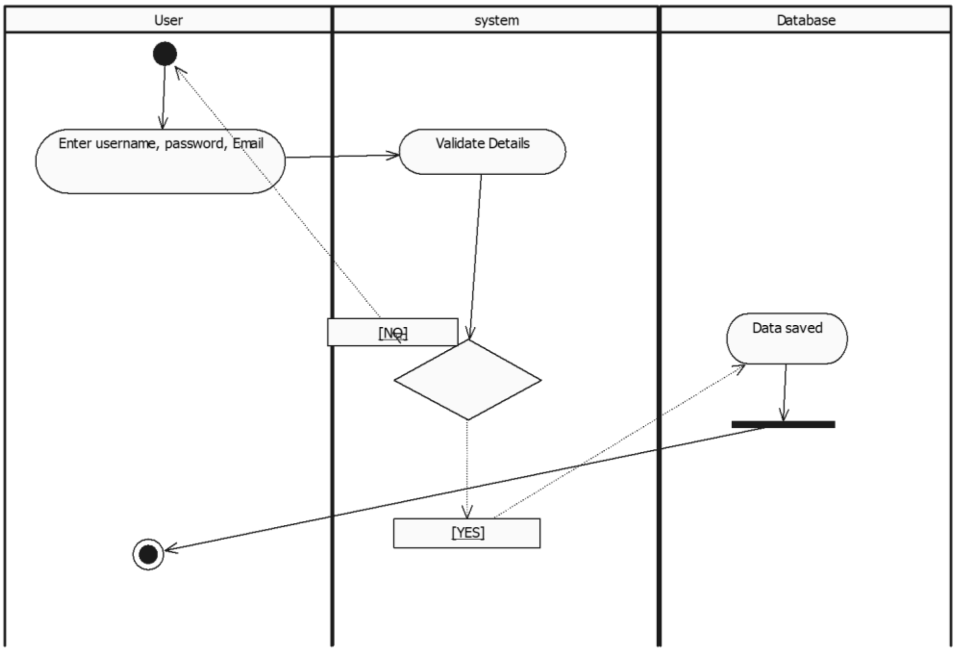


Figure 10 Activity diagram Register

F2:Login

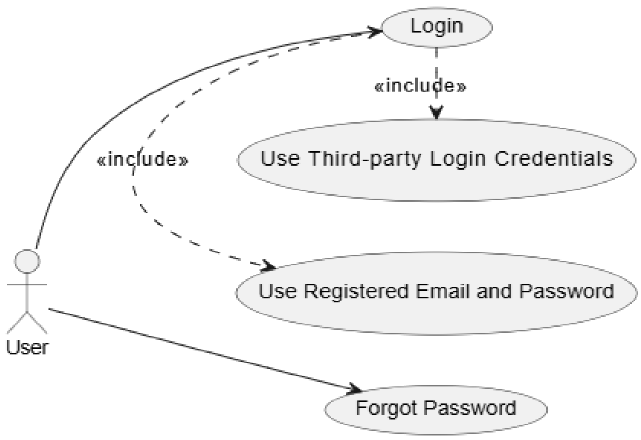


Figure 11 Use case Login

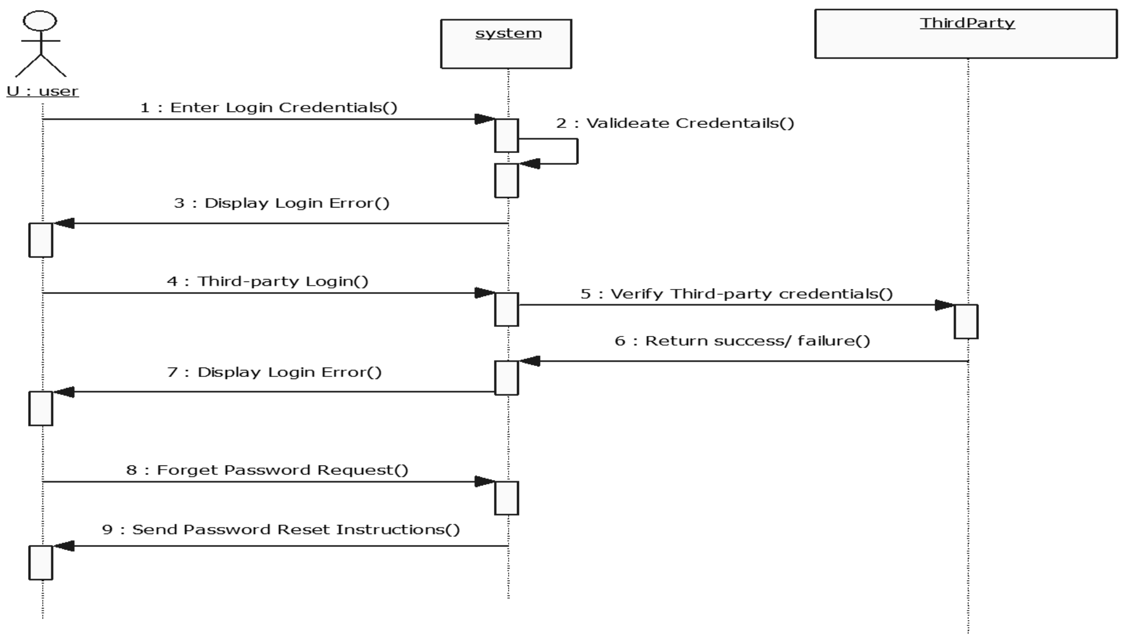


Figure 12 sequence Diagram Login

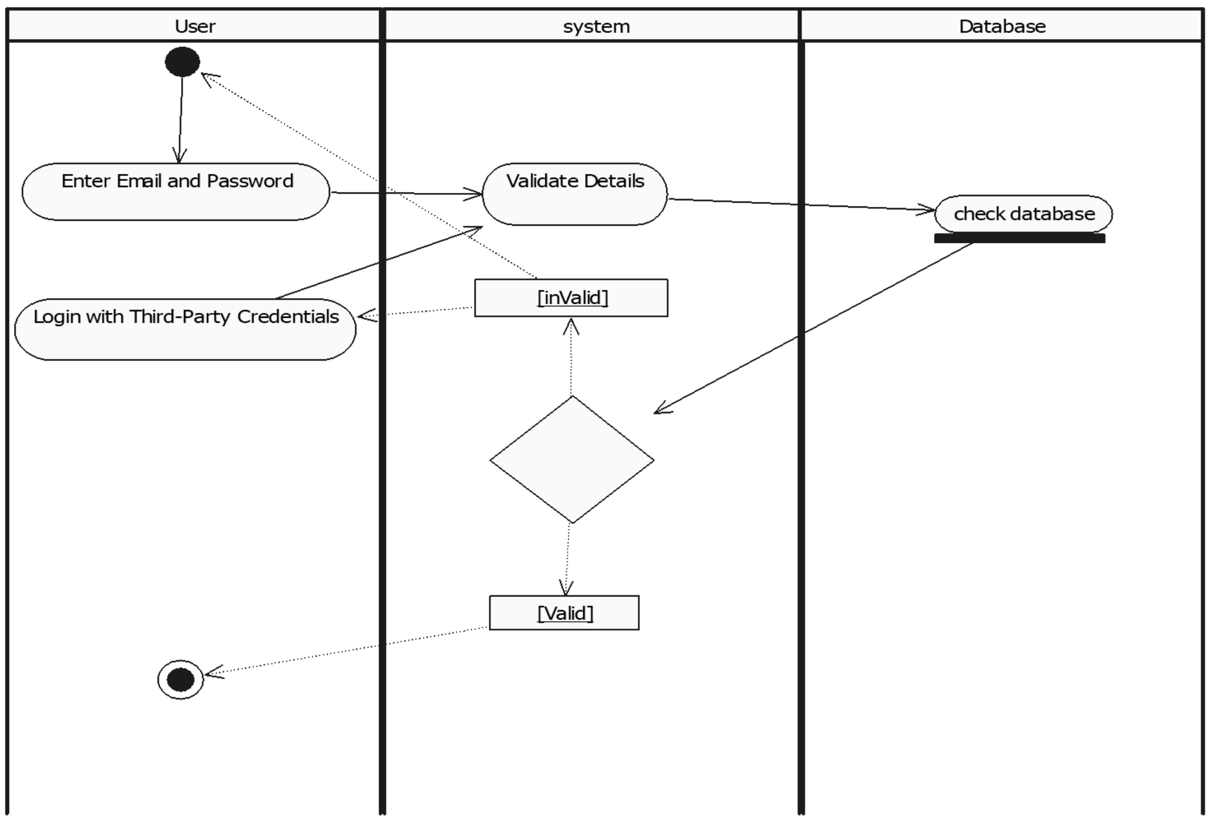


Figure 13 Activity Diagram Login

F3:Image Upload

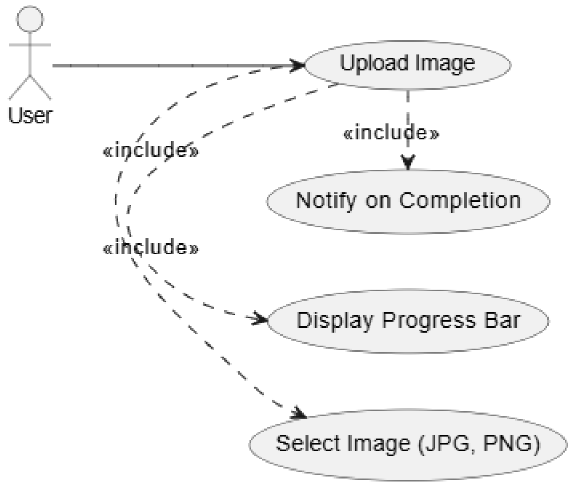


Figure 14 Use case Image Upload

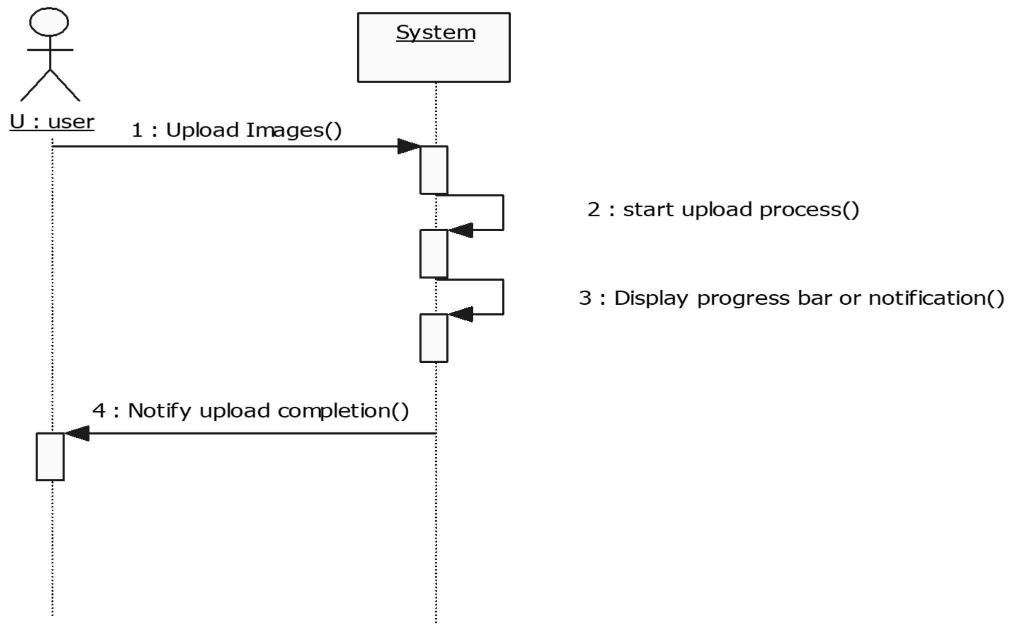


Figure 15 sequence Diagram Image Upload

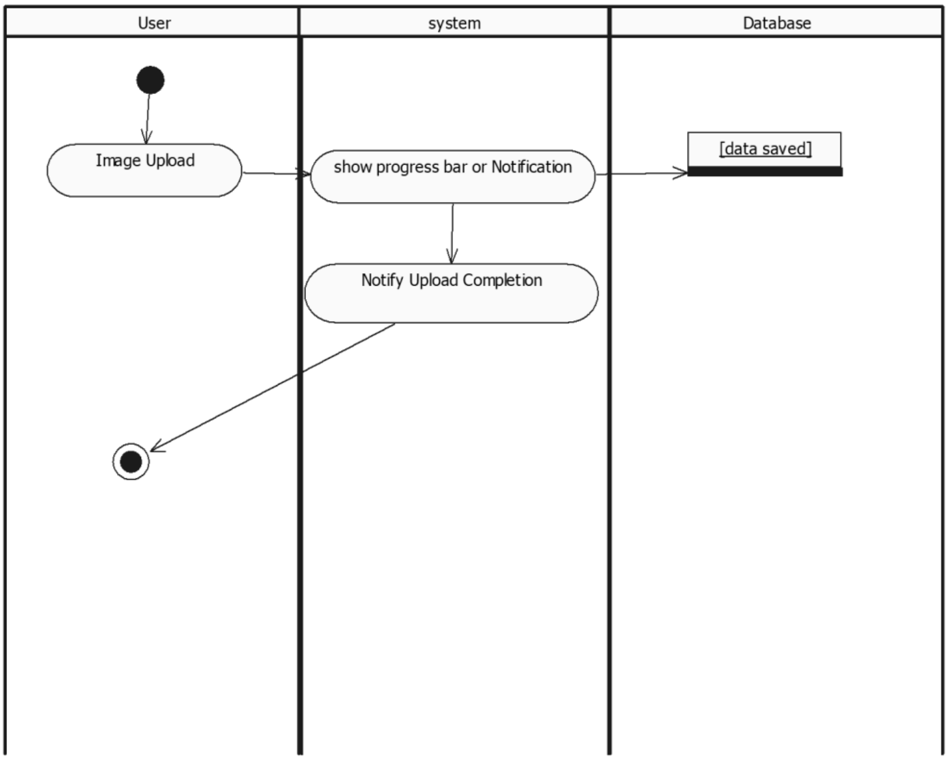


Figure 16 Activity Diagram Image Upload

F4:Image Preprocessing

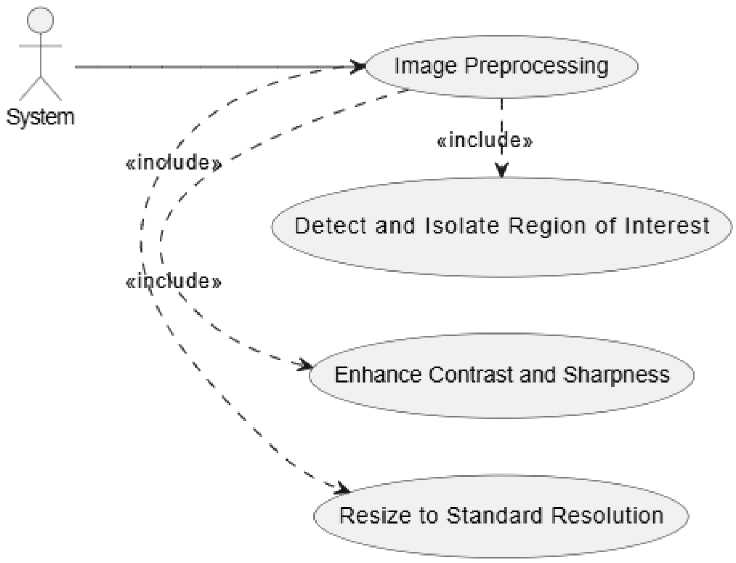


Figure 17 Use case Image Preprocessing

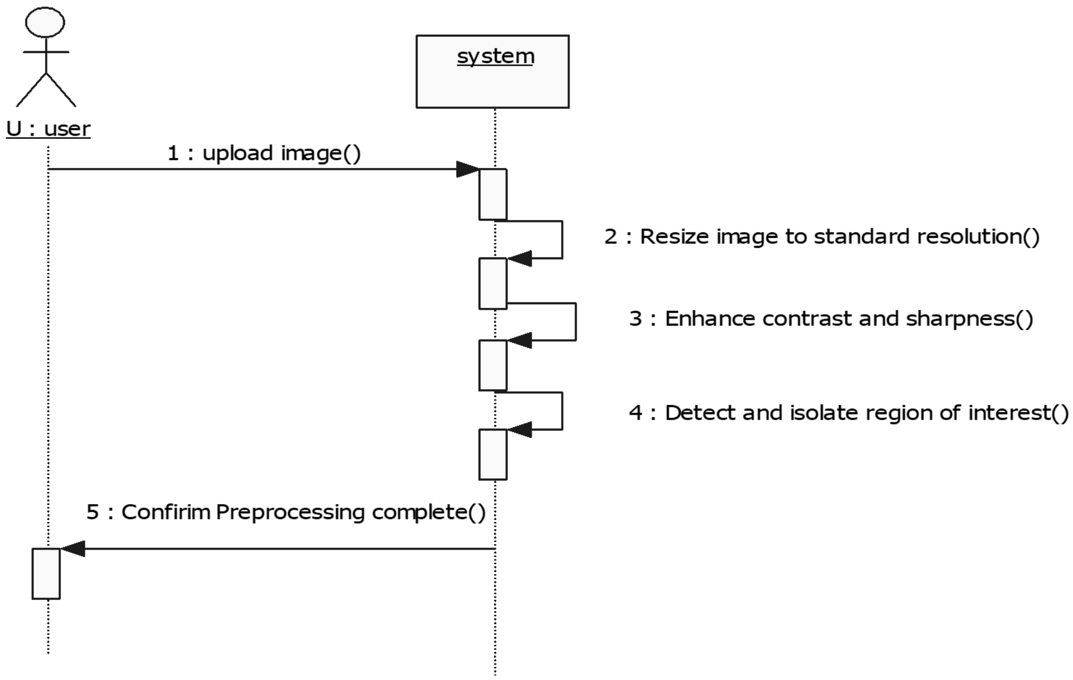


Figure 18 sequence Diagram Image Preprocessing

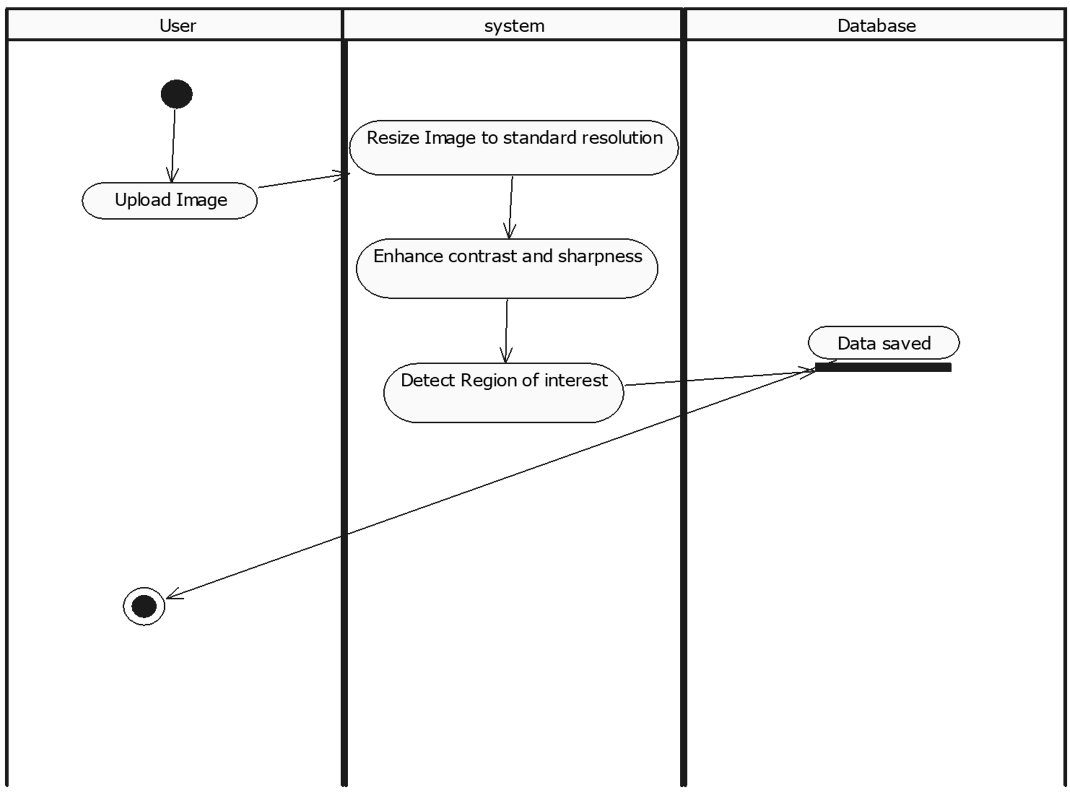


Figure 19 Activity e Diagram Image Preprocessing

F5:Receive The Information

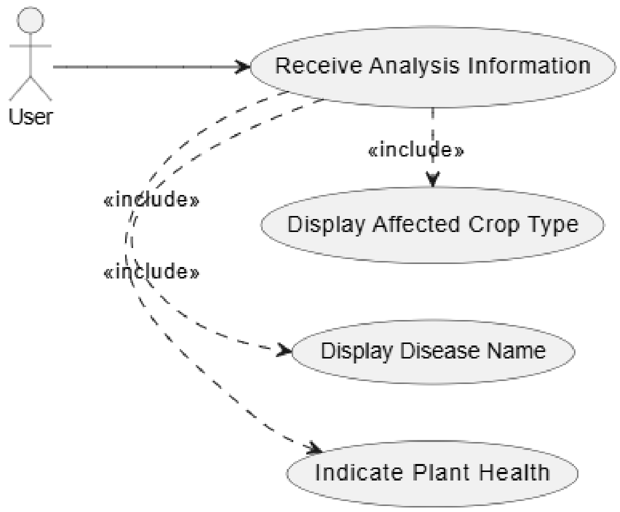


Figure20 Use case Receive the information

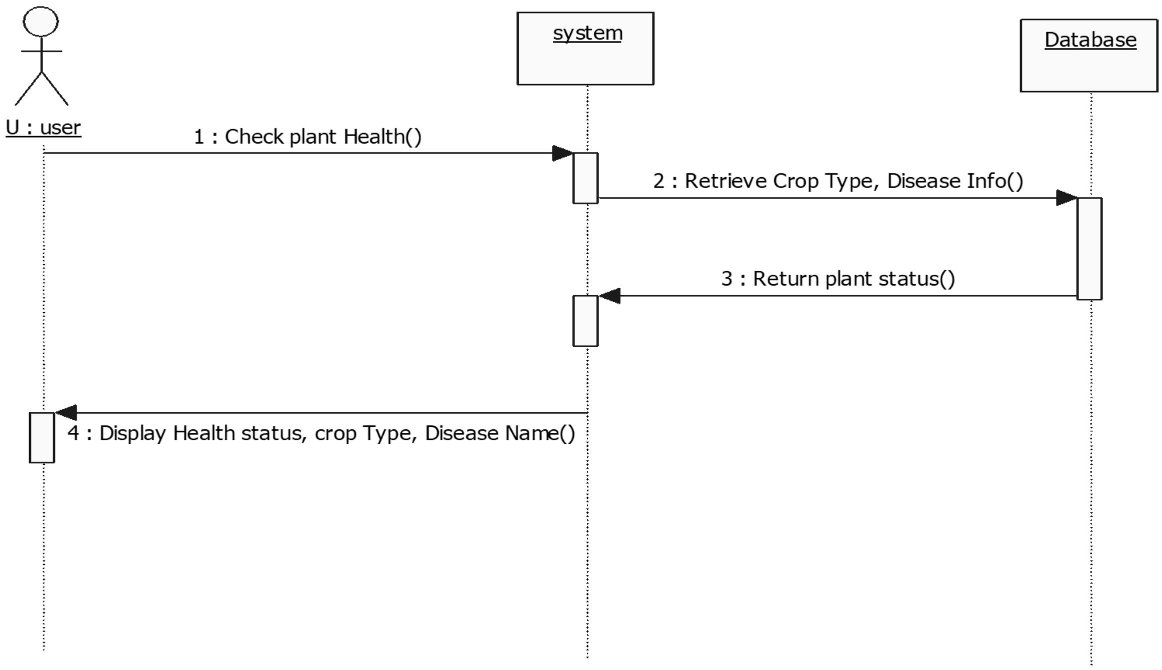


Figure21 Sequence Diagram Receive the information

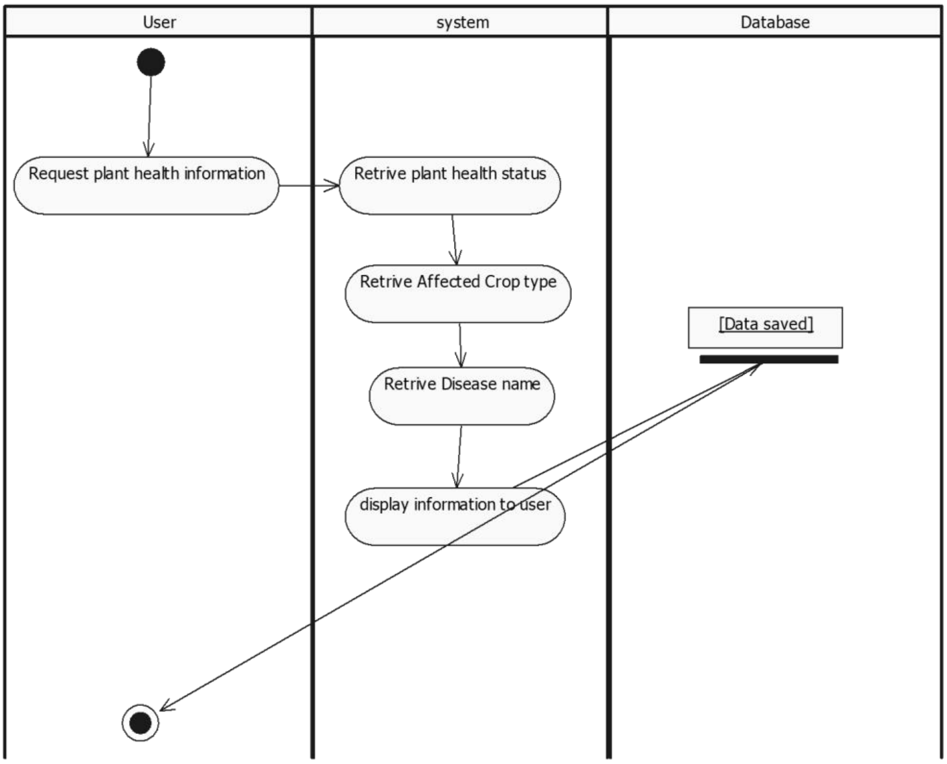


Figure22 Activity Diagram Receive the information

F6:Give Solutions

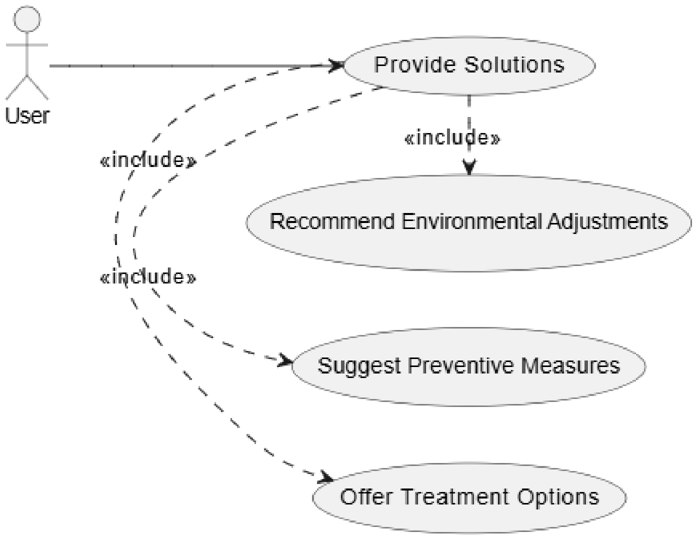


Figure23 Use case Give solutions

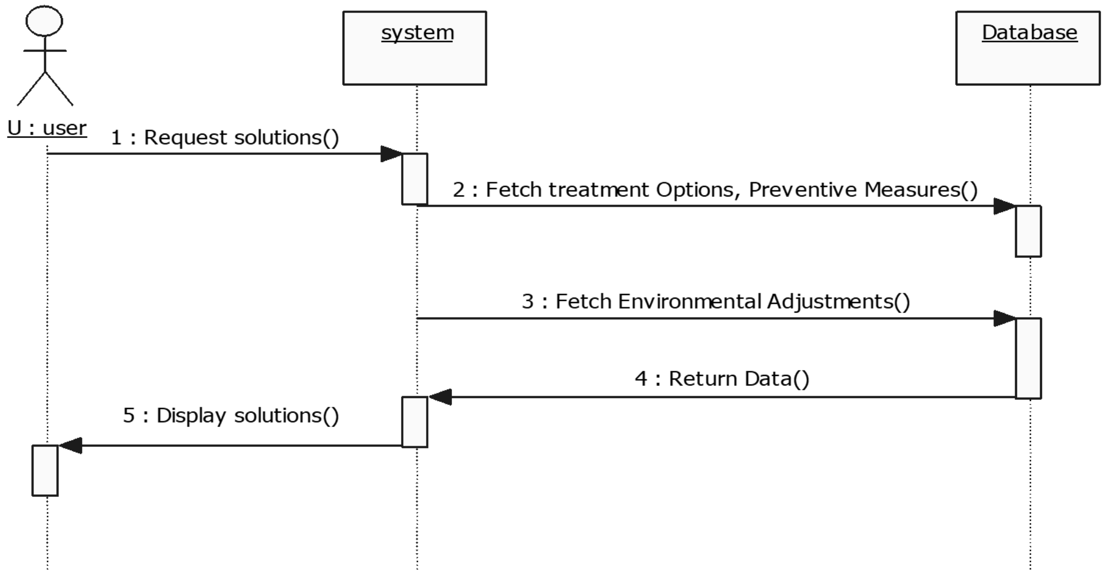


Figure24 Sequence diagram Give solutions

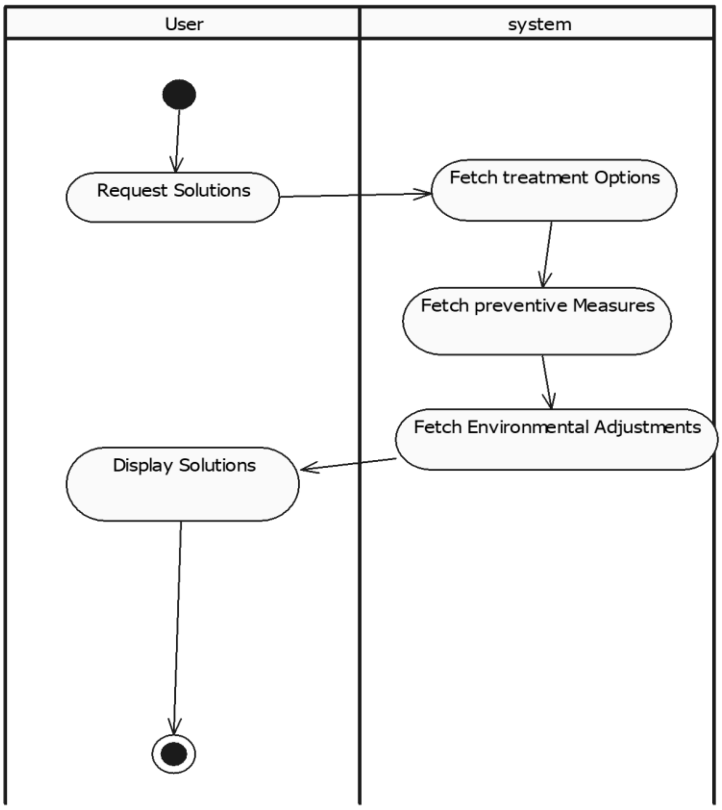


Figure25 Activity Diagram Give Solutions

F7:Upload Another Photo

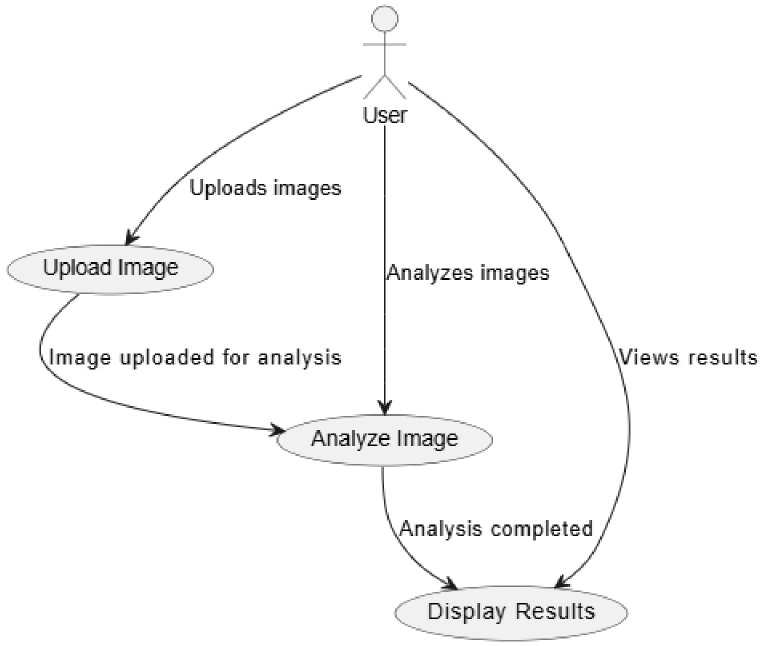


Figure26 Use case Upload Another Photo

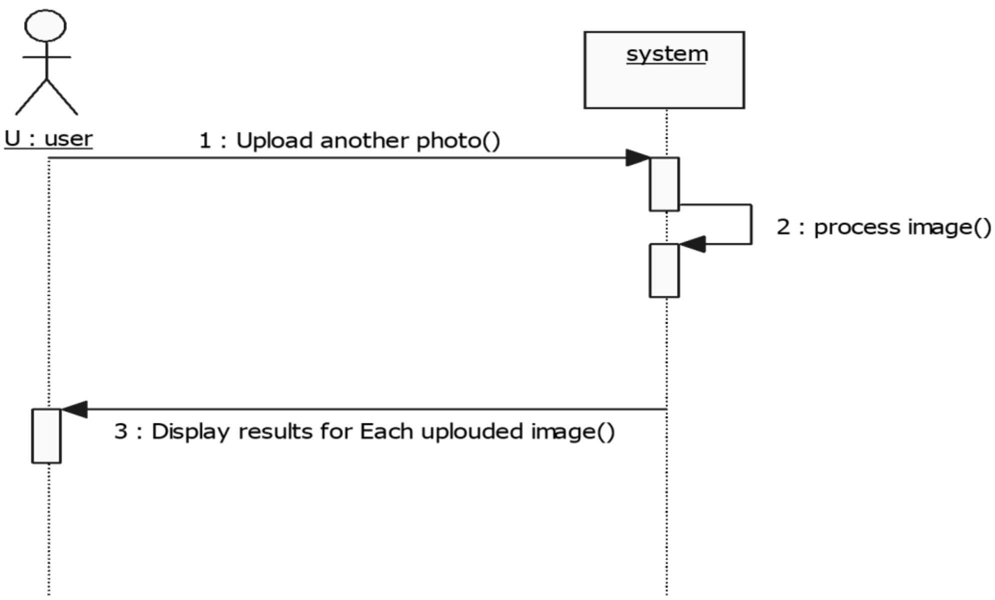


Figure27 Sequence Diagram Upload Another Photo

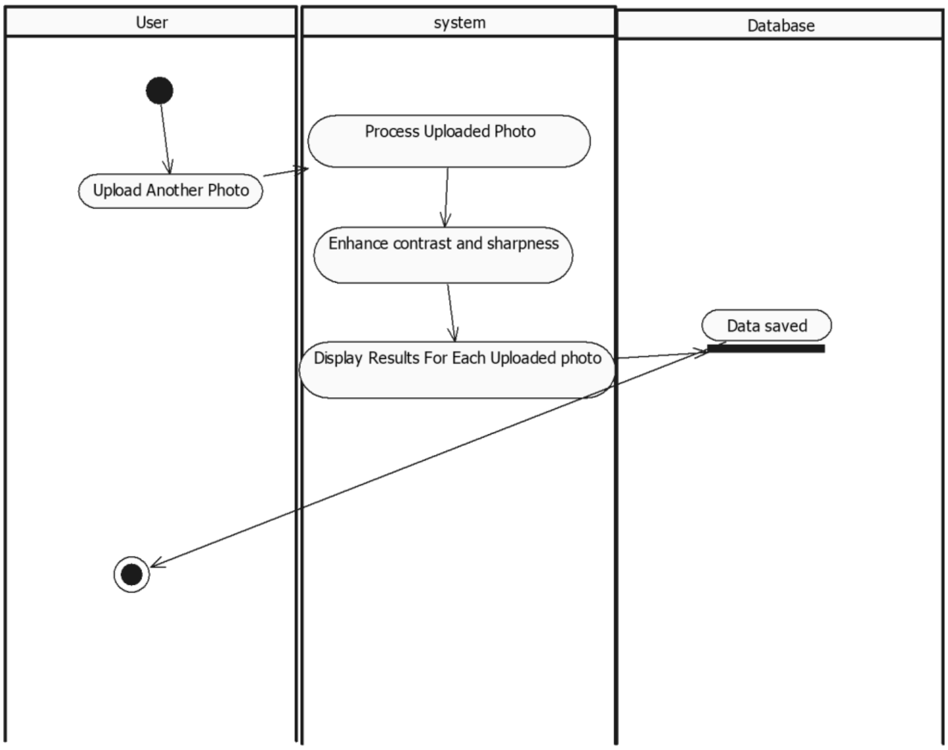


Figure28 Activity Diagram Upload Another Photo

F8: Disease Database and Updates

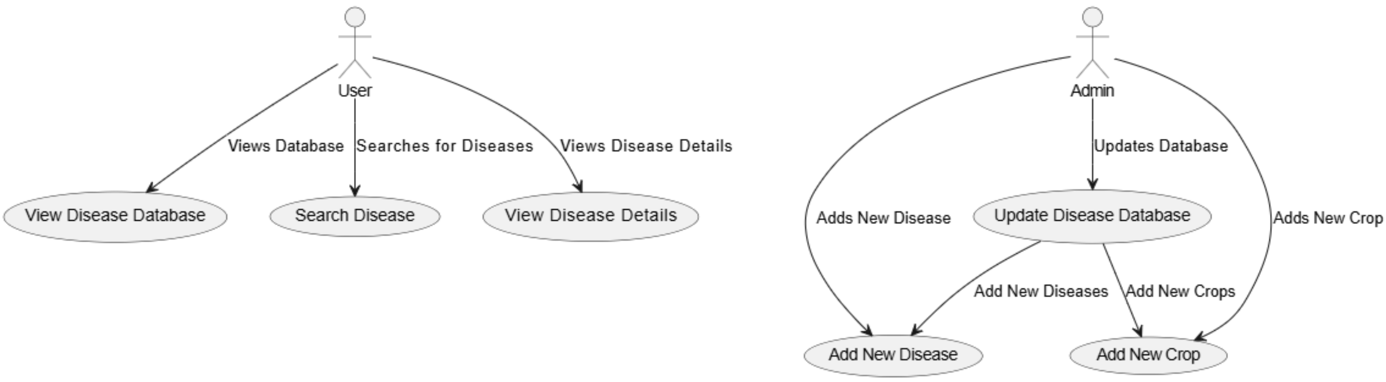


Figure29 Use case Disease Database and Updates

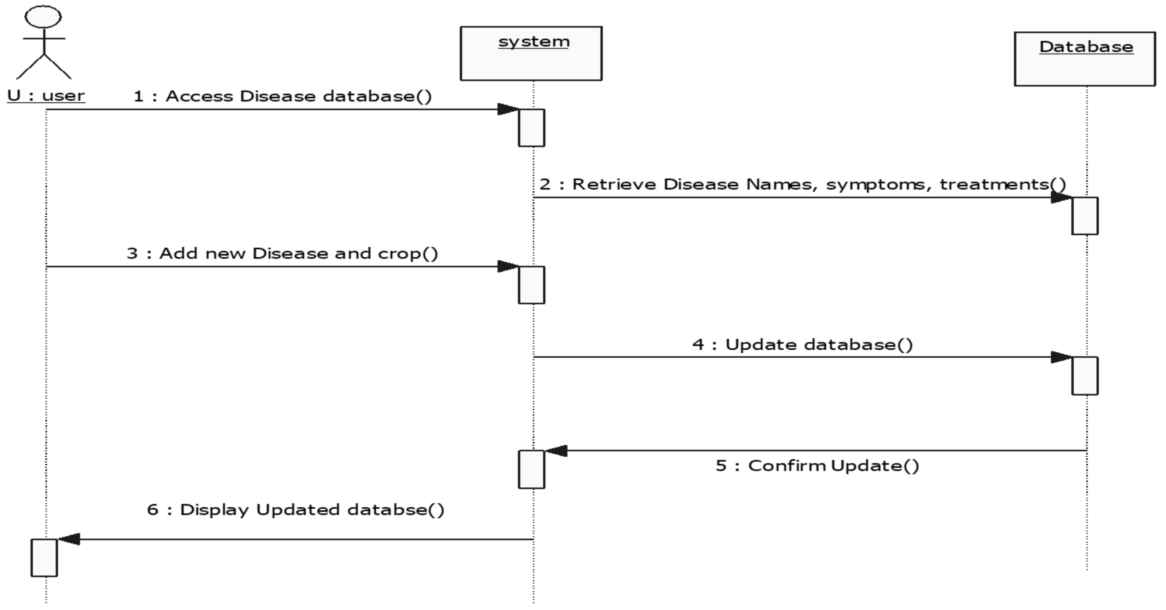


Figure30 Sequence Diagram Disease Database and Updates

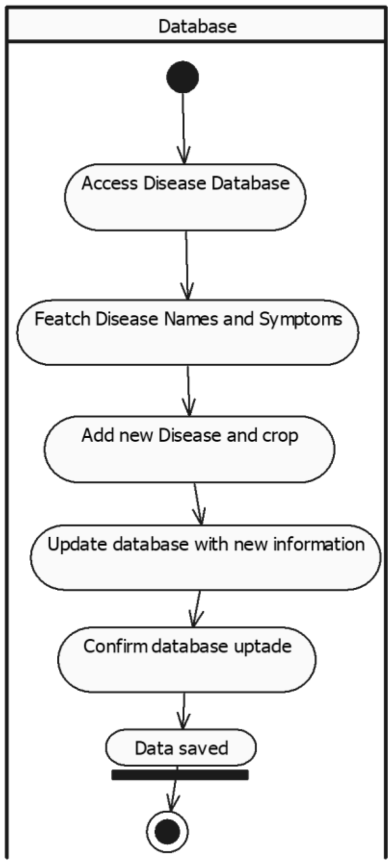


Figure31 Activity Diagram Disease Database and Updates

F9: Updates and Maintenance

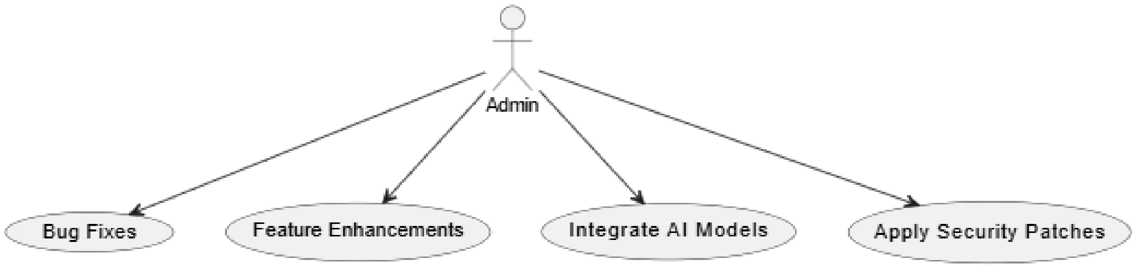


Figure32 Use case Updates and Maintenance

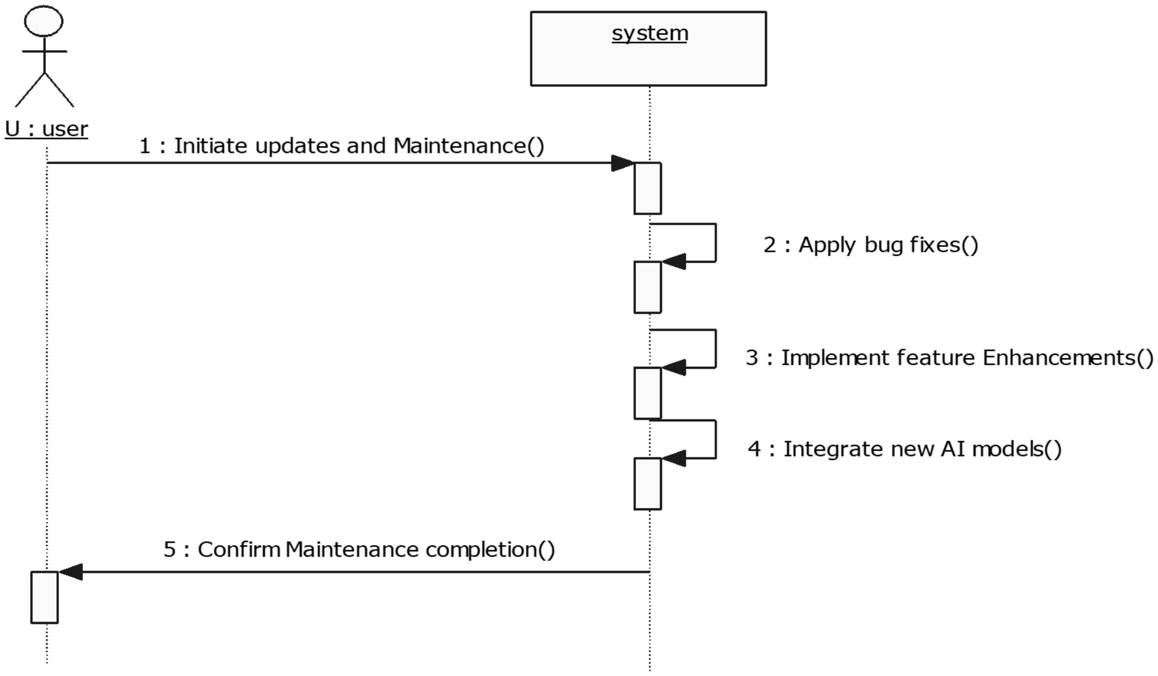


Figure33 Sequence Updates and Maintenance

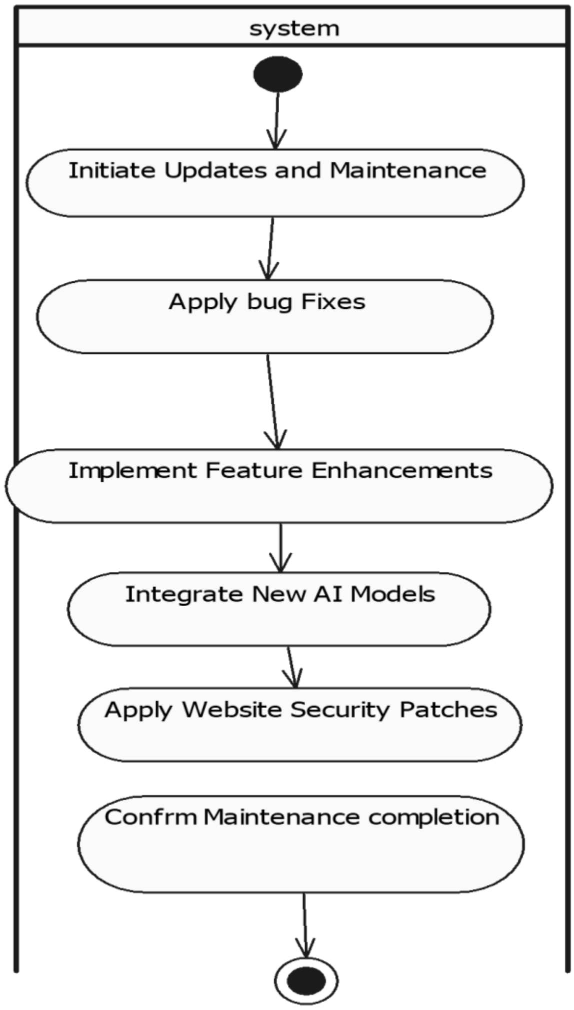


Figure34 Activity Updates and Maintenance

F10: User Interface

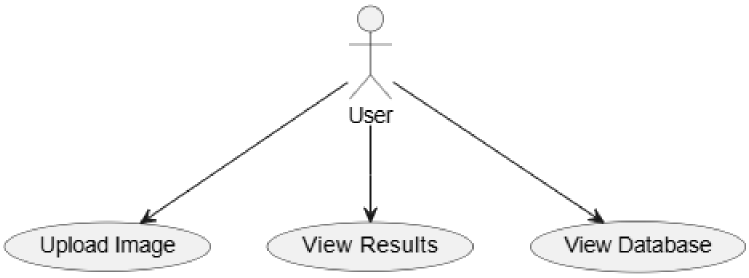


Figure35 Use case User Interface

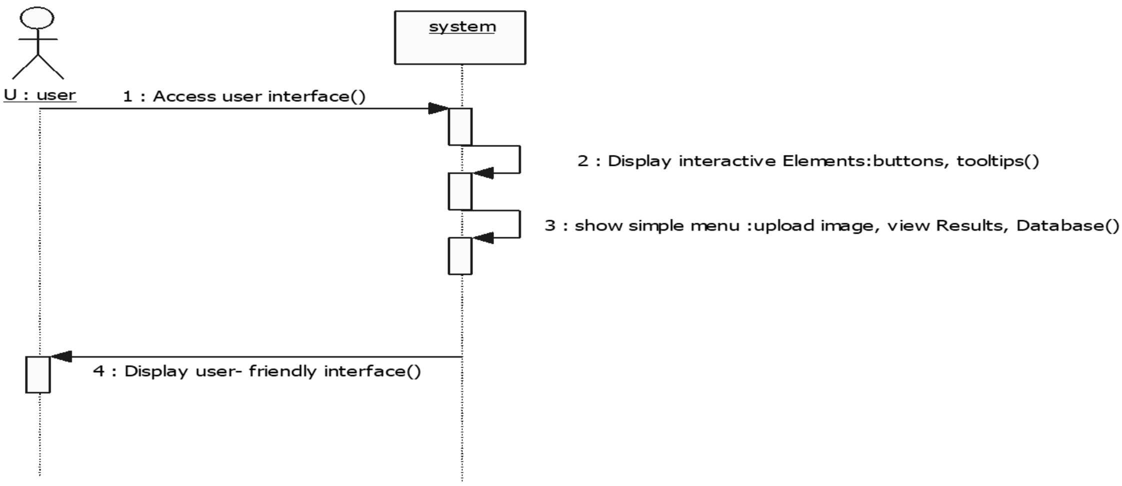


Figure36 Sequence User Interface

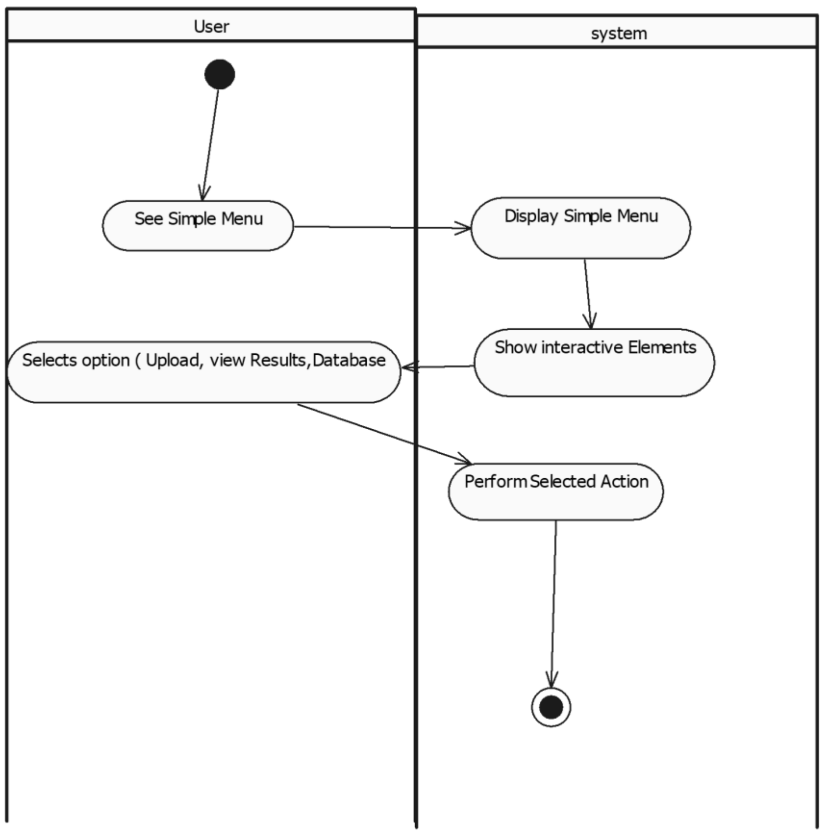


Figure37 Activity Diagram User Interface

F11: System Compatibility

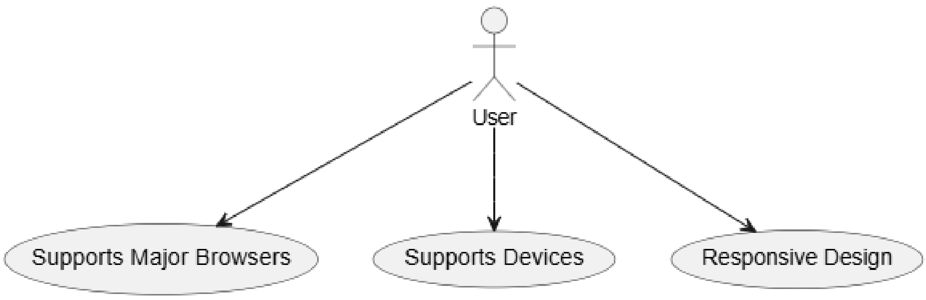


Figure 38 Use Case System Compatibility

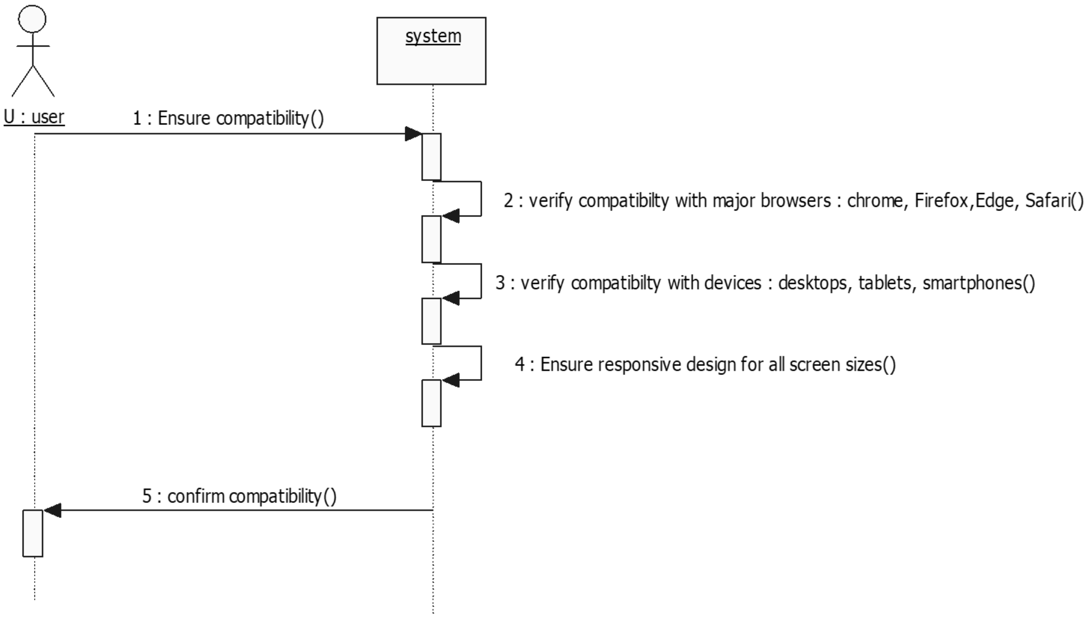


Figure39 Sequence System Compatibility

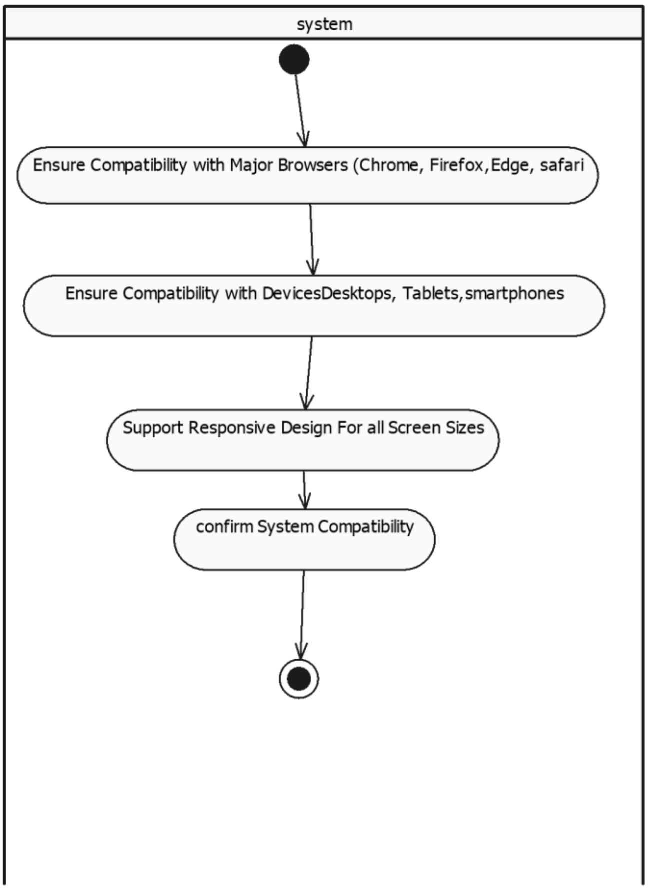


Figure40 Activity diagram System Compatibility

F12: Analytics and History

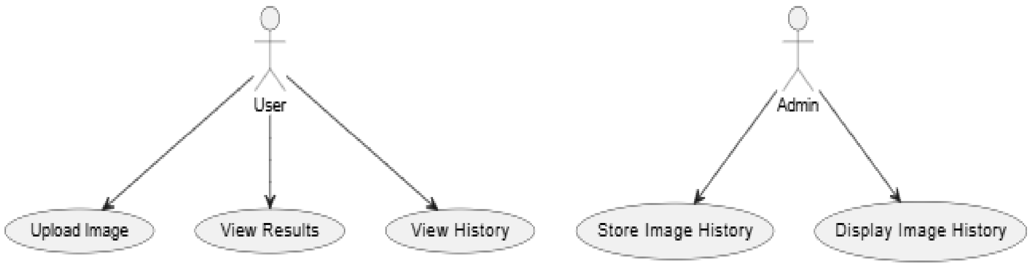


Figure41 Use Case Analytics and history

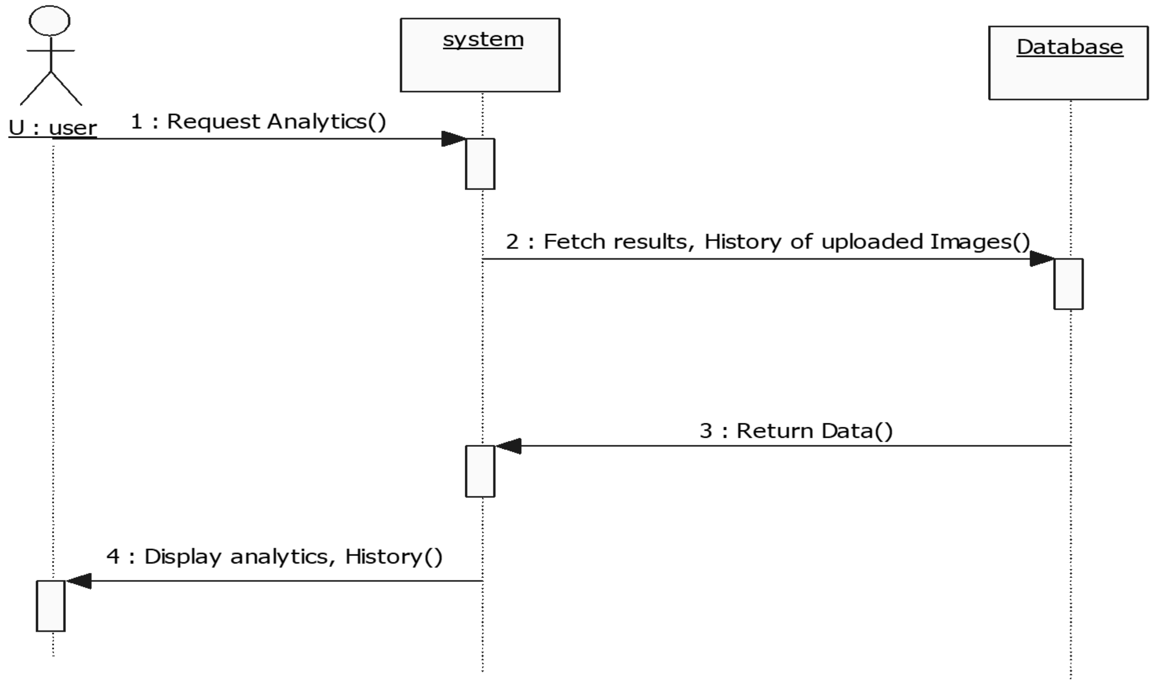


Figure42 Sequence Analytics and history

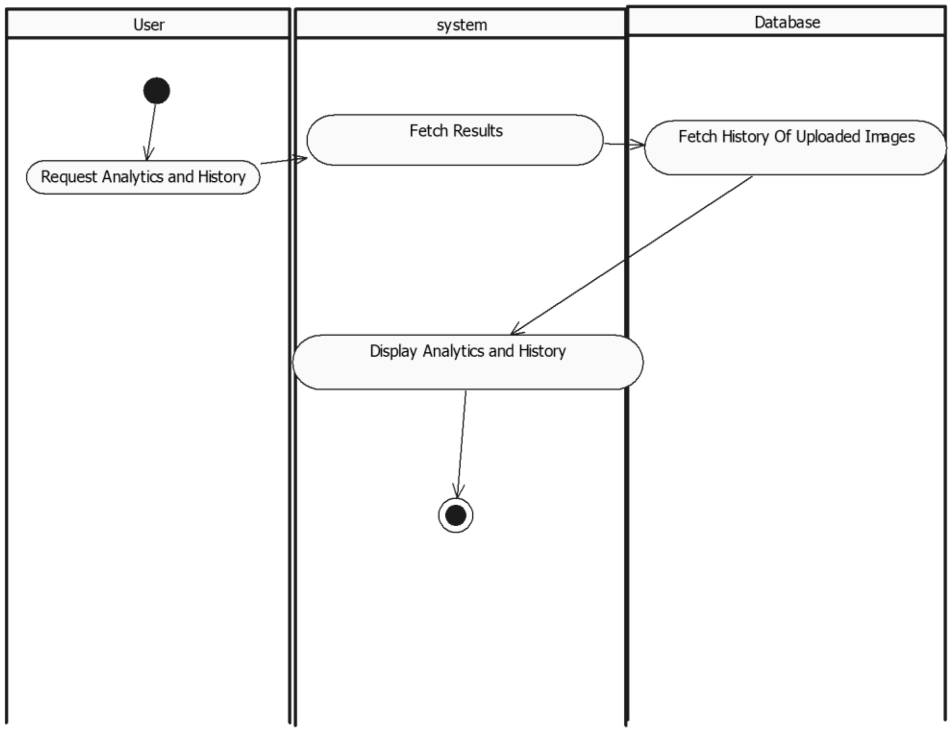


Figure43 Activity Analytics and history

F13:Notifications

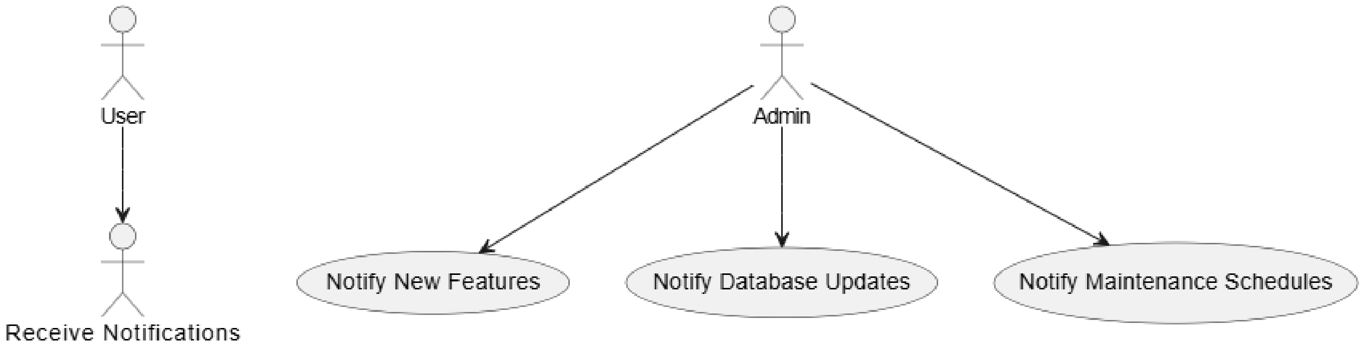


Figure44 Use Case Notifications

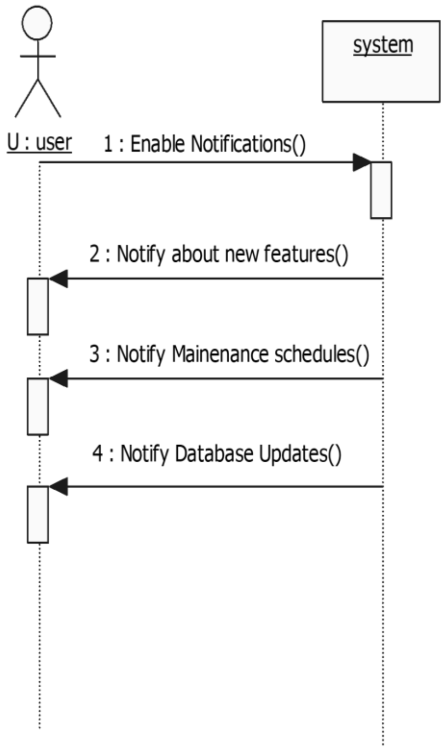


Figure45 Sequence Notifications

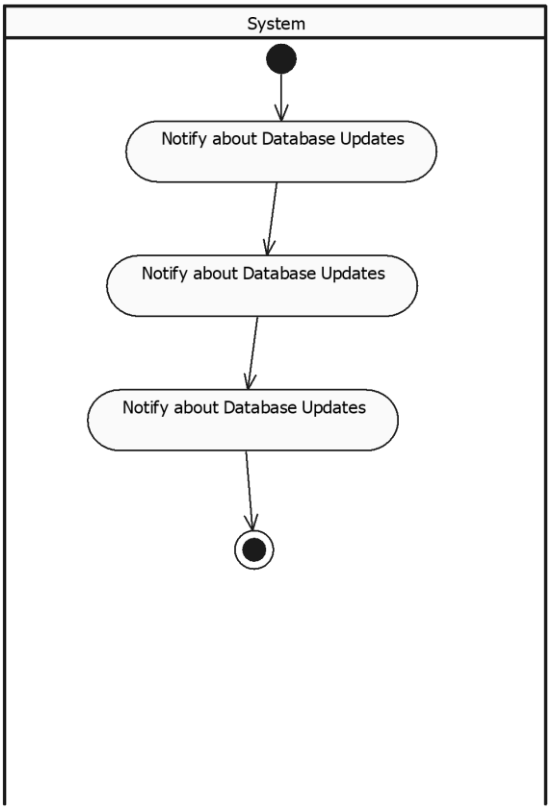


Figure46 Activity Diagram Notifications

F14:Help and Support

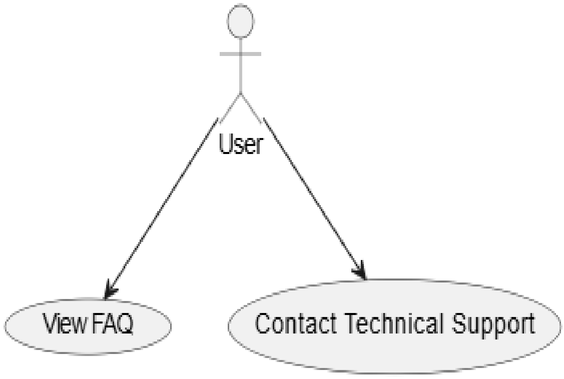


Figure47 Use Case Help and Support

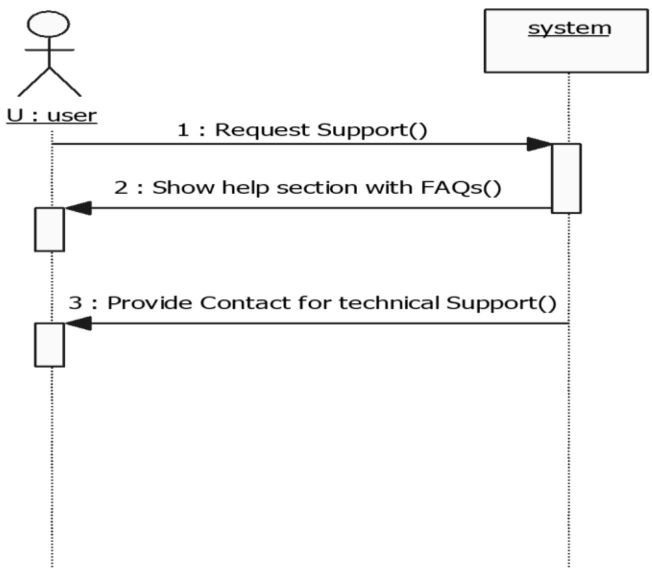
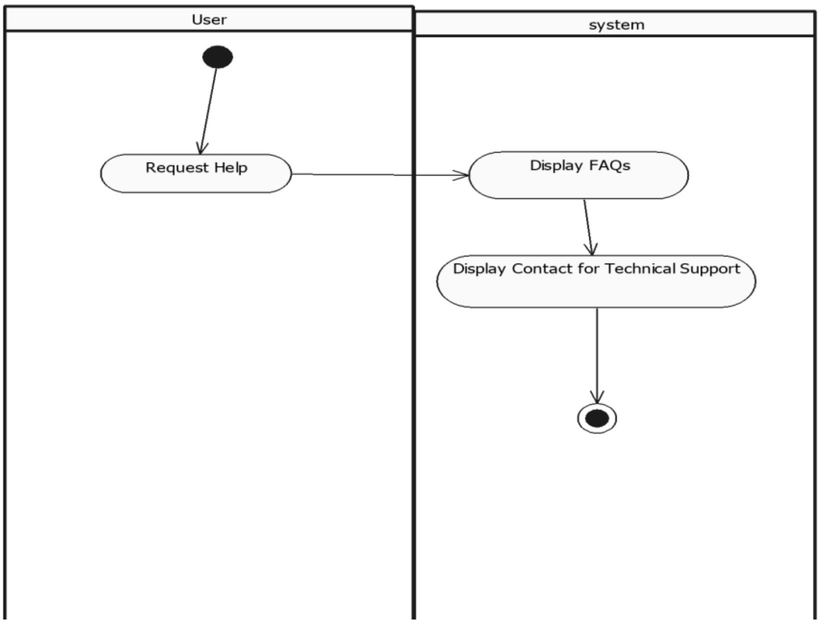


Figure48 Sequence Help and Support



[19] [20] *Figure49 Activity Help and Support*

References

[1] v. k. vishnoi, "ResearchGate," 21 1 2022. [Online]. Available: https://www.researchgate.net/publication/354507540_A_comprehensive_study_of_feature_extraction_techniques_for_plant_leaf_disease_detection.

[2] r. anand, r. a. parray and m. indra, "ScienceDirect," 10 10 2024. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1161030124003058.

[3] l. alzubaidi, o. al-shamma and m. f. mohammed, "journalofbigdata," 31 3 2021. [Online]. Available: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-021-00444-8.

[4] a. m., "National Library of Medicine," 2019. [Online]. Available: https://pmc.ncbi.nlm.nih.gov/articles/PMC10965613/.

[5] w. shafik and t. ali , "IEEE XPLORE," IEEE Access, 19 6 2023. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10147225. [Accessed 18 11 2024].

[6] n. kapoor, "Numpy Ninja," Career-centric Company, 15 1 2021. [Online]. Available: https://www.numpyninja.com/post/recall-specificity-precison-f1-scores-and-accuracy. [Accessed 18 11 2024].

[7] PlantVillage, "PlantVillage," PennState, 2018. [Online]. Available: https://plantvillage.psu.edu/. [Accessed 18 11 2024].

[8] Watson, "watson," Watsonx, 2013. [Online]. Available: https://www.ibm.com/watson. [Accessed 18 11 2024].

[9] leafsnap, "Leafsnap," Columbia University, 2011. [Online]. Available: https://leafsnap.com/. [Accessed 18 11 2024].

[10] R. Aronson, "farmbot," Farmbot, 2016. [Online]. Available: https://farm.bot/. [Accessed 18 11 2024].

[11] J. Martins, "asana," asana design , 13 5 2024. [Online]. Available: https://asana.com/resources/feasibility-study. [Accessed 8 12 2024].

[12] P. S. Kumar, "AIP piblishing," 2 5 2022. [Online]. Available: https://pubs.aip.org/aip/acp/article-abstract/2463/1/020001/2825237/Review-on-disease-detection-of-plants-using-image?redirectedFrom=fulltext. [Accessed 8 12 2024].

[13] S. Kovoor, "guidincode," 28 3 2023. [Online]. Available: https://guidingcode.com/user-and-system-requirements-in-software-engineering/. [Accessed 8 12 2024].

[14] S. Kovoor, "gudincode," 28 3 2023. [Online]. Available: https://guidingcode.com/user-and-system-requirements-in-software-engineering/. [Accessed 8 12 2024].

[1 V. Fulber-Garcia, "Baeldung.cs," 18 3 2024. [Online]. Available: https://www.baeldung.com/cs/requirements-

5] functional-vs-non-functional. [Accessed 8 12 2024].

[16] S. Das, "browser stack," 22 9 2024. [Online]. Available: https://www.browserstack.com/guide/non-functional-requirements-examples. [Accessed 8 12 2024].

[17] S. Cordes, "Business Analysis Toolkit," Business Analysis Toolkit, 13 11 2024. [Online]. Available: https://www.businessanalyststoolkit.com/non-functional-requirements/. [Accessed 8 12 2024].

[18] "miro," miro, [Online]. Available: https://miro.com/diagramming/what-is-a-uml-diagram/. [Accessed 19 1 2025].

[19] M. Fowler, "UML Distilled: A Brief Guide to the Standard Object Modeling Language," 1997.

[20] "Lucidchart UML Guide," Lucidchart, 2025. [Online]. Available: https://www.lucidchart.com/pages. [Accessed 19 1 2025].