Software Requirements Specifications

AI Assistant for Plant Disease Diagnosis

Project Code:

FYP2226G

Internal Advisor:

Dr. Hussam Ali

External Advisor:

Nill

Project Manager:

Dr. Muhammad Iliyas

Project Team:

Taha Mukhtar (Team Lead)
Hammad Ahmad
Usama Farooq

Submission Date:

20th OCT 2025

Project Manager's Signature

Table of Contents

1. Introduction

1.1 Purpose of Document

The purpose of this document is to describe the functionality, objectives, and scope of the ***Plant Disease Detection and Management System****. It outlines how the system operates, its intended users, and the benefits it aims to deliver.*
This document is intended for:

- ***Developers:*** *To understand the system's architecture, functionality, and technical requirements.*
- ***Project Supervisors/Stakeholders:*** *To evaluate the project's objectives, scope, and expected outcomes.*

End Users (Farmers): *To understand how the system assists in diagnosing plant diseases and managing crop health efficiently.*

1.2 Project Overview

The ***Plant Disease Detection*** *is an AI-powered mobile application developed using* ***Kotlin*** *and* ***Firebase****. The system is designed to assist farmers in identifying plant diseases through image recognition and provide effective management recommendations.*

Key Features

1. ***AI-Based Disease Detection:*** *Farmers can upload images of their plants, and the AI model analyzes them to detect potential diseases accurately.*
2. ***Disease Information and Treatment Suggestions:*** *The system provides detailed information about the detected disease along with preventive and treatment guidelines.*
3. ***Weather Updates:*** *Offers live weather information to help farmers make better decisions about planting, irrigation, and crop care.*
4. ***User-Friendly Interface:*** *The application provides a simple, intuitive interface that makes disease diagnosis easy for all types of users, including those with limited technical experience.*

Goals and Benefits

- *Enhance* ***crop health*** *through early and accurate disease detection.*
- *Reduce* ***crop losses*** *caused by unidentified or untreated diseases.*

- *Minimize **dependency on agricultural experts** by providing AI-driven recommendations.*

Empower farmers with accessible, reliable, and quick diagnostic support.

1.3 Scope

Included Features

- *Image-based detection of plant diseases using Artificial Intelligence.*
- *Display of disease-related information and suitable treatment or preventive measures.*
- *Real-time weather updates relevant to agricultural activities.*
- *Secure user registration and data management via Firebase.*

Not Included

- *Advanced agricultural automation tools such as drones for field monitoring.*
- *Automated irrigation, pest control, or fertilizer application systems.*
- *Marketplace or product-selling functionalities*

2. Overall System Description

Describe the environment, in which the system will be developed and used, the anticipated users of the system and the known constraints, assumptions and dependencies.

2.1 User characteristics

*The **Plant Disease Detection and Management System** is mainly designed for **farmers** and people involved in agriculture who want an easy way to identify plant diseases and get useful guidance.*

User Types

Farmers

Farmers are the main users of this app. Most of them may not have technical knowledge, so the system is designed with a simple and easy-to-understand interface. They can upload pictures of their plants and get quick feedback about possible diseases and treatments.

Agricultural Experts (optional users):

Experts or agricultural officers may use the system to verify results, provide recommendations, or use it as a support tool in research or advisory work.

*The app focuses on being **simple, reliable, and helpful** for all users, even those with minimal technical experience.*

2.2 Operating environment

*The system is developed as a **mobile application** that runs on Android devices.*
Operating Details:

- ***Platform:** Android smartphones and tablets.*
- ***Programming Language:** Kotlin.*
- ***Database and Backend:** Firebase Realtime Database and Firebase Authentication for storing user data and managing accounts.*
- ***Internet Requirement:** The app requires a stable internet connection to upload images, fetch weather updates, and display results.*
- ***AI Model:** The disease detection model processes images on a cloud server or integrated AI module connected with Firebase.*

The app is designed to work smoothly on most Android devices with moderate specifications and a reliable internet connection.

2.3 System constraints

The following constraints and limitations may affect how the system operates:
Software Constraints

- *The app is currently developed for Android only; it is not available for iOS or desktop platforms.*
- *Requires active internet access for image uploads and data retrieval from Firebase.*
- *The AI model's accuracy depends on the quality and clarity of the uploaded plant images.*

Hardware Constraints:

- *Users need a smartphone with a camera and internet connectivity.*

- *The device should have enough memory and storage to run the app and handle image files.*

Cultural Constraints:

- *The app's language is English, which may be difficult for users who are not familiar with it. Future updates may include local language support.*

Legal Constraints:

- *Users must only upload images they own or have permission to use.*
- *Data is stored according to Firebase's data privacy and security policies.*

Environmental Constraints:

- *The app is designed for outdoor use, but performance may be affected by poor network signals in rural areas.*

User Constraints:

- *The app is made for adults involved in farming; therefore, it avoids complex terms and provides simple text and visuals.*

Performance Constraint:

*The app aims to detect plant diseases within **10 seconds** of image upload under normal internet conditions.*

3. External Interface Requirements.

3.1 Hardware Interfaces

Mobile Device Requirements

- *Platform: Android smartphones and tablets*
- *Minimum Android Version: Android 8.0 (API Level 26) or higher*
- *Recommended Android Version: Android 10.0 (API Level 29) or above*
- *Processor: Quad-core 1.5 GHz or higher*
- *RAM: Minimum 2 GB, Recommended 4 GB or higher*
- *Storage: Minimum 100 MB free space for app installation and data caching*
- *Camera: Rear camera with minimum 8 MP resolution for plant disease image capture*
- *Network: Wi-Fi or mobile data connectivity (3G/4G/5G) for real-time data synchronization*

Camera Interface

- *The application interfaces with the device's camera hardware to capture images of diseased plants*
- *Camera permissions must be granted by the user for disease detection functionality*
- *The app utilizes the native Android Camera API for image capture*
- *Supports autofocus and flash control for optimal image quality*

GPS/Location Services

- *The application accesses the device's GPS hardware to obtain location data*
- *Location information is used to provide localized weather updates*
- *Required for marketplace proximity-based filtering (optional feature)*

Network Interface

- *The application requires active internet connectivity through Wi-Fi or cellular data.*
- *Recommended bandwidth: 2 Mbps or higher for optimal performance including image upload.*

3.2 Software Interfaces

Mobile Application Platform

- ***Development Framework**: Kotlin programming language*
- ***Target Platform**: Android OS*
- ***IDE**: Android Studio*
- ***Minimum SDK**: API Level 26 (Android 8.0)*
- ***Target SDK**: API Level 33 (Android 13) or latest stable version*
- ***UI Framework**: Jetpack Compose or XML-based layouts for user interface design*

Backend Service - Firebase

The application integrates with Google Firebase platform for backend services:

Firebase Authentication

- *Manages user registration and login*
- *Supports email/password authentication*
- *Handles user session management and token-based authentication*
- *Interface: Firebase Authentication SDK for Android*

Firebase Cloud Storage

- *Stores uploaded plant disease images*

- *Interface: Firebase Storage SDK for Android*

External APIs

Google Maps API

- ***Purpose**: Display location-based weather information and visualize geographical data*
- ***API Version**: Google Maps SDK for Android v18.0 or later*
- ***Data Exchange Format**: JSON/XML*
- ***Authentication**: API Key-based authentication*
- ***Functions Used**:*
  - *Map display and navigation*
  - *Geolocation services*
  - *Location marker placement*

Weather API (e.g., OpenWeatherMap or similar)

- ***Purpose**: Fetch real-time weather data based on farmer's location*
- ***Data Exchange Format**: JSON*
- ***Authentication**: API Key*
- ***Data Retrieved**:*
  - *Current temperature and conditions*
  - *Humidity and wind speed*
  - *Weather forecasts (3-7 days)*
  - *Agricultural weather alerts*

Machine Learning Model Interface

- ***ML Framework**: TensorFlow Lite for Android*
- ***Model Type**: Convolutional Neural Network (CNN) for image classification*
- ***Input**: Plant disease images captured by farmers*
- ***Output**: Disease classification with confidence score and treatment recommendations*
- ***Model Storage**: Embedded in the app or fetched from Firebase ML*
- ***Inference**: On-device processing using TensorFlow Lite interpreter*

Third-Party Libraries

- ***Image Processing**: Glide or Picasso for image loading and caching*
- ***Networking**: Retrofit or OkHttp for REST API communication*
- ***JSON Parsing**: Gson or Moshi for data serialization*
- ***UI Components**: Material Design Components for Android*

3.3 User Interfaces

Farmer Interface

- *Dashboard with weather updates and quick access to features*
- *Disease detection screen with camera integration*
- *Profile management screen*

Design Guidelines

- *Follows Material Design principles for Android*
- *Responsive design supporting various screen sizes*
- *Intuitive navigation with clear visual hierarchy*
- *Accessibility features for users with disabilities*

3.4 Communication Interfaces

Network Protocols

- *HTTP/HTTPS: For API communication with Firebase and external services*
- *REST: For structured data exchange with backend services*

Data Exchange Formats

- *JSON: Primary format for data exchange between app and backend*
- *Binary: For image upload and download*

Security Protocols

- *TLS/SSL encryption for all network communications*
- *Firebase security rules for database access control*
- *API key protection using Android NDK or ProGuard obfuscation*

4. Functional Requirements

4.1 Weather Information Display

- *The system shall display current weather information on the farmer's dashboard, including temperature, humidity, wind speed, and weather conditions.*
- *The system shall automatically detect the farmer's location using GPS to provide localized weather updates.*
- *The system shall allow farmers to manually enter a location if GPS is unavailable.*
- *The system shall display a 7-day weather forecast relevant to agricultural activities.*
- *The system shall send weather alerts for extreme conditions (heavy rain, frost, heat waves) that may affect crops.*
- *The system shall refresh weather data automatically every 30 minutes or upon user request.*

4.2 Plant Disease Detection

- *The system shall allow farmers to capture or upload images of diseased plants through the camera interface.*
- *The system shall accept images in common formats (JPEG, PNG) with a maximum file size of 10 MB.*
- *The system shall process the uploaded image using a trained machine learning model to identify plant diseases.*
- *The system shall display the disease detection results within 5 seconds of image upload.*
- *The system shall provide the disease name, description, and confidence level (percentage accuracy) of the diagnosis.*
- *The system shall suggest appropriate treatment methods and preventive measures for the identified disease.*
- *The system shall recommend relevant products from the marketplace (fertilizers, pesticides) that can treat the detected disease.*
- *The system shall allow farmers to view their disease detection history with timestamps.*
- *If the disease cannot be identified with high confidence (below 70%), the system shall suggest consulting an agricultural expert.*

4.3 Disease Detection System (Detailed Workflow)

Image Capture and Upload

- *The system shall provide an intuitive camera interface with guidelines for capturing clear plant images.*
- *The system shall allow farmers to choose between capturing a new image or selecting from gallery.*
- *The system shall compress images automatically to optimize upload speed without significant quality loss.*
- *The system shall display a loading indicator during image upload and processing.*

AI Analysis

- *The system shall preprocess the uploaded image (resizing, normalization) before feeding it to the ML model.*
- *The system shall run the preprocessed image through a trained Convolutional Neural Network (CNN) model.*
- *The system shall classify the plant disease from a predefined set of diseases (minimum 10 common crop diseases).*
- *The system shall calculate a confidence score for each prediction.*
- *The system shall handle cases where no disease is detected (healthy plant) by displaying "No Disease Detected."*

Results and Recommendations

- *The system shall display the disease name in both English and local language (optional).*
- *The system shall provide a detailed description of the disease including symptoms and causes.*
- *The system shall list recommended treatment methods including organic and chemical solutions.*
- *The system shall suggest relevant marketplace products with direct links for purchase inquiry.*
- *The system shall allow farmers to save the detection result to their history for future reference.*
- *The system shall provide an option to share the diagnosis report with agricultural experts or other farmers.*

4.4 System Administration (Optional)

- *The system shall support remote updates to the disease detection ML model without requiring app updates.*
- *The system shall maintain version control for ML models to ensure compatibility.*

5. Non-functional Requirements

5.1 Performance Requirements

- *The system should process and return plant disease predictions within 5 seconds of image submission under normal network conditions.*
- *The AI model should maintain an accuracy rate of at least 85% for disease detection across all supported plant types.*
- *The app should be optimized to run efficiently on Android devices running Android 10 or later, with minimal memory and CPU consumption.*
- *The system must be capable of handling at least 1000 concurrent users without significant degradation in response time.*
- *Image preprocessing and model inference should be performed using on-device resources or lightweight cloud APIs to ensure speed and scalability.*

5.2 Safety Requirements

- *The system should ensure that no harmful actions (e.g., system crash, data corruption) occur due to invalid inputs, such as uploading non-image files or corrupted images.*

- *The app should handle all exceptions gracefully, providing clear error messages without exposing internal details.*
- *The system should not make direct agricultural treatment recommendations (e.g., pesticide use) without proper disclaimers, to avoid user harm.*
- *All network operations must include **timeouts and retries** to prevent the application from freezing or becoming unresponsive.*

5.3 Security Requirements

- *All user data (including images) must be transmitted securely using **HTTPS encryption**.*
- *If any data is stored remotely, it should be protected using **secure cloud storage** with proper authentication and access control.*
- *The app should comply with **data privacy laws** (e.g., GDPR if applicable) and should clearly state data usage policies in the user agreement.*
- *User authentication (if implemented) should be protected using **Firebase Authentication** or equivalent secure mechanism.*
- *The AI model and backend endpoints should be protected from unauthorized access or tampering.*

5.4 User Documentation

- *A **User Manual** will be provided, explaining installation, image capture, and interpretation of results.*
- ***Online Help** (FAQ and troubleshooting guide) will be integrated within the app.*
- *A **Quick Start Tutorial** will guide users through the initial steps upon first use.*
- *Technical documentation (for developers) will include API usage, model integration steps, and system design.*

6. Assumptions and Dependencies

Assumptions

- *Users will have a **stable internet connection** to upload images and receive predictions.*
- *Users' devices will have a **functional camera** and **sufficient storage** for temporary image handling.*

- *The AI model will have been **pre-trained and optimized** for the targeted plant species before deployment.*
- *The project will use **publicly available datasets** or **proprietary data collected under consent** for model training.*
- *Weather integration (if implemented) assumes availability of **reliable third-party weather APIs**.*

Dependencies

- *The application depends on **Google ML Kit**, **TensorFlow Lite**, or equivalent libraries for AI inference.*
- *The backend (if used) will depend on **Firebase Cloud Storage** and/or **Firebase Realtime Database** for data handling.*
- *The weather feature depends on integration with APIs such as **OpenWeatherMap** or **WeatherAPI**.*
- *Project success depends on **accurate dataset labeling** and **model training quality**.*
- *Timely delivery of **AI model**, **app interface**, and **testing feedback** are critical dependencies for the project timeline.*

7. References

This section should provide a complete list of all documents referenced at specific point in time. Each document should be identified by title, report number (if applicable), date, and publishing organization. Specify the sources from which the references can be obtained. (This section is like the bibliography in a published book)

Ref. No.	Document Title	Date of Release/ Publication	Source/ Location
FYP2226G	Project Proposal	Oct 13, 2025	https://github.com/mtahamukhtar2-ui/FYP
[1]	PS Venkata Reddy, KS Nandini Prasad, and C. Puttamadappa. Farmer's Friend: Conversational AI Bot for Smart Agriculture. Journal of Positive School Psychology, 6(2):2541–2549	2022	https://journalppw.com/index.php/jpsp/article/view/1833
[2]	Vandana Nayak, NH Sowmya, et al. Agroxpert – Farmer Assistant. Global Transitions Proceedings, 2(2):506–512	2021	https://www.scribd.com/document/716551511/Agroxpert-Farmer-assistant-2021-Global-Transitions-Proceedings

Ref. No.	Document Title	Date of Release/ Publication	Document Source
[3]	Abbas Jafar, Nabila Bibi, Rizwan Ali Naqvi, Abolghasem Sadeghi-Niaraki, and Daesik Jeong. Revolutionising Agriculture with Artificial Intelligence: Plant Disease Detection Methods, Applications, and Their Limitations. Frontiers in Plant Science, 15:1356260	2024	https://www.frontiersin.org/articles/10.3389/fpls.2024.1356260/full
[4]	Tejal Yadav, Pooja Sable, and Dhananjay Kalbande. Smart Kisan: A Mobile App for Farmers' Assistance in Agricultural Activities. 2023 International Conference on Smart Systems for Applications in Electrical Sciences (ICSSES)	2023	IEEE Xplore (Available through conference proceedings)

8. Appendices

8.1 Dataset Description

The AI model for plant disease detection was trained using publicly available and validated datasets to ensure accuracy and diversity in plant types.

Dataset Name	Source	Description	Format
PlantVillage Dataset	Kaggle	Contains over 2,000 labeled images of healthy and diseased plant leaves from 38 plant species.	JPEG
AI Challenger Dataset	AI Challenger Platform	Provides agricultural images for machine vision applications, used for cross-validation.	PNG
Custom-Collected Dataset	Project Team	Locally collected images of regional crops for enhanced model accuracy and real-world testing.	JPEG