Hamdard University

Department of Computing

Final Year Project

Project Title

Dr. Crop Android Application

Project Code

FYP-002/SP25

Software Requirements Specifications

Submitted by

Name	Roll Number	Project Role
Muhammad Ibrahim	(1895-2021)	(Team Lead)
Zeeshan Ali	(1505-2021)	(Team Member 2)
Wali Muhammad	(1894-2021)	(Team Member 3)

Supervisor

Teacher: Sir Saifullah Adnan

Fall 2024

Document Sign off Sheet

Document Information

Project Title	Dr.Crop Android Application
Project Code	FYP-002/SP25
Document Name	Software Requirements Specifications (SRS)
Document Version	<1.0>
Document Identifier	FYP-002/SP25-SRS
Document Status	Final
Author(s)	M.Ibrahim , Zeeshan Ali , Wali Muhammad
Approver(s)	Saifullah Adnan
Issue Date	

Name	**Role**	**Signature**	**Date**
Muhammad Ibrahim	Team Lead		02-07-2025
Zeeshan Ali	Team Member 2		02-07-2025
Wali Muhammad	Team Member 3		02-07-2025
Saifullah Adnan	Supervisor		
	Co-Supervisor		
	Project Coordinator		

Revision History

Date	Version	Description	Author
04-07-2025	1.0	Initial version of the document created, including system functions, non-functional requirements, use cases, and other essential sections.	Muhammad Ibrahim Zeesham ali Wali Muhammad

Definition of Terms, Acronyms, and Abbreviations

In the **Definition of Terms, Acronyms, and Abbreviations** section, you'll list and define all relevant terms, acronyms, and abbreviations used throughout the document. This ensures clarity and helps the reader understand the specialized language or abbreviations. Here's a general template for this section.

Term	Description
Actor	An entity that interacts with the system, e.g., user, external systems.
CNN	Convolutional Neural Network, a type of deep learning algorithm used for image recognition.
API	Application Programming Interface, a set of routines and tools for building software applications.
GUI	Graphical User Interface, a system that allows users to interact with software through graphical elements.
Camera Input	Input method where visual data is captured using a camera for processing.

Table of Contents

1. Introduction

1.1 Purpose of Document

This document defines the software requirements for a prototype system focused on Wheat Disease Detection using CNN and TensorFlow Lite. The long-term goal is to expand this system to support multiple crops such as cotton.

1.2 Intended Audience

This document is intended for developers, agricultural researchers, academic supervisors, and future contributors who plan to extend the project scope to other crops.

2. Overall System Description

2.1 Project Background

Crop diseases reduce yields and economic returns. This project began as a multi-crop disease detection solution (cotton and wheat), but in this prototype phase, we focus only on wheat diseases to test core functionality and feasibility.

2.2 Problem Statement

Manual detection of wheat diseases is time-consuming and requires expert knowledge. Farmers need an automated solution to detect diseases early with minimal effort.

2.3 Project Scope

This prototype will classify wheat leaf images into four categories: Black Rust, Brown Rust, Yellow Rust, or Healthy. Future iterations will expand the model to include cotton diseases.

2.4 Not In Scope

This version does not include:

- Cotton disease detection
- Disease treatment suggestions
- Web interface or cloud support

2.5 Project Objective

- Train and test a CNN model for wheat diseases
- Convert it to TensorFlow Lite
- Build a prototype Android app
- Evaluate accuracy and usability for future crop support

2.6 Stakeholder & Affected Groups

• Farmers
• Researchers
• Agriculture Extension Officers
• Future developers of cotton integration

2.7 Operating Environment

Android smartphones with TensorFlow Lite compatibility. No internet access is required for core functionality.

2.8 System Constraints

• Prototype limited to wheat
• Works only on Android 8+
• Requires sufficient lighting in images

2.9 Assumptions & Dependencies

• Future crops (e.g., cotton) will require additional datasets and model tuning
• Mobile devices are available with camera and TensorFlow Lite
• Dataset quality impacts classification performance

3. External Interface Requirements

3.1 Hardware Interface

The mobile device must have a functioning camera and sufficient RAM (minimum 2GB) to process TensorFlow Lite inference.

3.2 Software Interfaces

The application will use TensorFlow Lite runtime, Android OS 8.0 or higher, and support standard image processing libraries.

3.3 Communications Interfaces

The application is designed to work offline and does not require internet connectivity. No data transmission is expected.

4. System Functions / Functional Requirements

4.1 System Functions

- Capture or select an image of a wheat leaf.
- Preprocess and normalize the image.
- Classify the image using TFLite model.
- Display the predicted class and confidence.
- Support expansion to other crops in future.

4.2 Use Cases

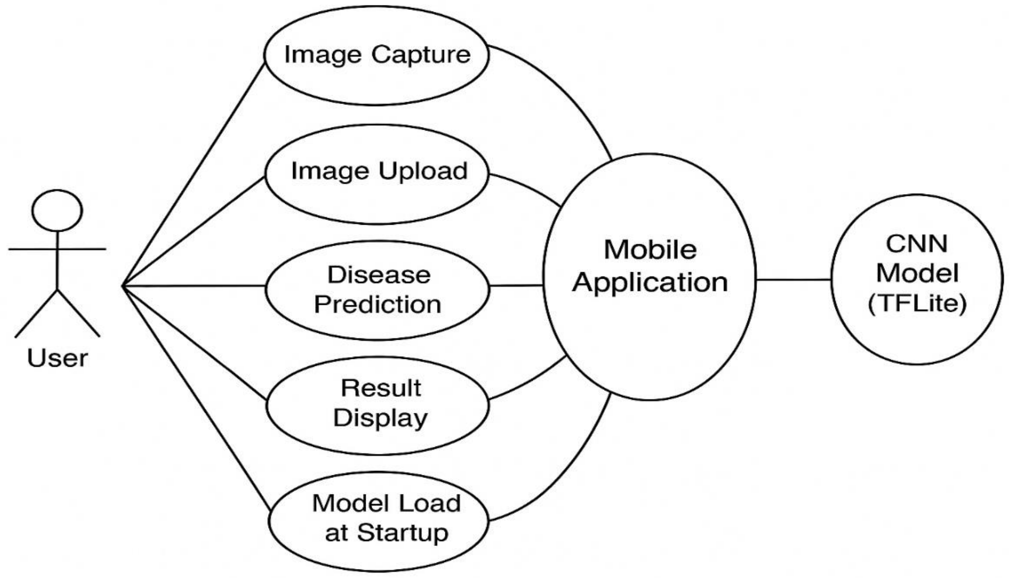
4.2.1 List of Actors

- Farmer
- Mobile Application
- CNN Model

4.2.2 List of Use Case

- Image Capture
- Image Classification
- Result Display
- Model Load

4.2.3 Use Case Diagram



4.2.4 Description of Use Cases

- Image Capture: User opens the app and captures a wheat leaf image using the camera.
- Image Classification: App sends image to TFLite model for prediction.
- Result Display: App shows the predicted class and confidence.
- Model Load: App loads model from local storage during startup.

5. Non - Functional Requirements

5.1 Performance Requirements

Inference time should be less than 1 second on average smartphones.

5.2 Safety Requirements

The app must not collect or store user data or images.

5.3 Security Requirements

Model and app must function without any network communication to ensure data privacy.

5.4 Reliability Requirements

App must be able to reload the model even after device restart or app crash.

5.5 Usability Requirements

User interface should be simple and intuitive, especially for farmers with limited technical knowledge.

5.6 Supportability Requirements

App should support future model updates via in-app assets replacement.

5.7 User Documentation

A simple in-app guide will be included to help users understand how to capture a good image and interpret results.

6. References

- TensorFlow Lite documentation: https://www.tensorflow.org/lite
- Wheat Disease datasets (custom/collected from Kaggle)
- Research papers on CNN for plant disease detection
- Android Developer Documentation: https://developer.android.com
- Agricultural datasets from open repositories (e.g., Kaggle)