**ABSTRACT**

Agriculture is a key source of livelihood. Agriculture provides employment opportunities for village people on large scale in developing country like India. India's agriculture is composed of many crops and according to survey nearly 70% population is depending on agriculture. Most of Indian farmers are adopting manual cultivation due to lagging of technical knowledge. Farmers are unaware of what kind of crops that grows well on their land. When plants are affected by heterogeneous diseases through their leaves that will effects on production of agriculture and profitable loss. Also, reduction in both quality and amount of agricultural production. Leaves are important for fast growing of plant and to increase production of crops. Identifying diseases in plants leave is challenging for farmers also for researchers. Currently farmers are spraying pesticides to the plants but it effects human directly or indirectly by health or also economically. To detect these plant diseases many fast techniques need to be adopt.

Also Plant disease detection is a huge problem and often require professional help to detect the disease. This research focuses on creating a deep learning model that detects the type of disease that affected the plant from the images of the leaves of the plants. The deep learning is done with the help of Convolutional Neural Network by performing transfer learning. The model is created using transfer learning and is experimented with both resnet 34 and resnet 50 to demonstrate that discriminative learning gives better results. This method achieved state of art results for the dataset used. The main goal is to lower the professional help to detect the plant diseases and make this model accessible to as many people as possible. The main goal of this project is to lower the professional help to detect the plant diseases and make this model accessible to as many people as possible.

**INTRODUCTION**

Indian economy is dependent of agricultural productivity. Over 70% of rural homes depend on agriculture. Agriculture pays about 17% to the total GDP and provides employment to over 60% of the population. Therefore, detection of plant diseases plays a vital key role in the arena of agriculture. Indian agriculture is composed of many crops like rice, wheat. Indian farmers also grow sugarcane, oilseeds, potatoes and non-food items like coffee, tea, cotton, rubber. All these crops grow based on strength of leaves and roots. There are things that lead to different disease for the plant leaves, which spoiled crops and finally it will effect on economy of the country. These big losses can be avoided by early identification of plant diseases. Accurate detection of plant disease is needed to strengthen the field of agriculture and economy of our country. Various types of Disease kill leaves in a plant. Farmers get more difficulties in identifying these diseases, they are unable to take precaution on those plants due to lack of knowledge on those diseases. Biomedical is one of the fields to detect plant diseases. In current day among this field, the image processing methods are suitable, efficient and reliable field for disease detection with help of plant leaf images. Farmers need fast and efficient techniques to detect all types of diseases of plants that can save time. These systems that can reduce efforts and use of pesticides. For measurement of yields in agriculture different ideas are proposed by scientists with the help of laboratory and systems for efficient identification of plant leaf diseases.

Image Processing is only an aspect of Computer Vision, and they are not the same. Image Processing systems focus on transforming images from one form to another, and Computer Vision systems help the computer to understand, and get meaning from an image. Many Computer Vision systems employ Image Processing algorithms. For example, a face enhancement app may use computer vision algorithms to detect faces in a photo, and then apply Image Processing techniques like smoothing or grayscale filters to it.

Many advanced Image Processing methods leverage Machine Learning Models like Deep Neural Networks to transform images on a variety of tasks, like applying artistic filters, tuning an image for optimal quality, or enhancing specific image details to maximize quality for computer vision tasks.

Convolutional Neural Networks (CNN) take in an input image and use filters on it, in a way that it learns to do things like object detection, image segmentation and classification. ResNet50 is a residual deep learning neural network model with 50 layers. ResNet was the winning model of the ImageNet (ILSVRC) 2015 competition and is a popular model for image classification, it is also often used as a backbone model for object detection in an image. Which would be used in our project.

**Problem Statement**

“**CROP IDENTIFICATION ALONG WITH HEALTH STATUS USING MOBILE APP**”

The project involved preprocessing of plant images, masking those images, building the CNN model, validating the model, testing the model, and finally deploying the CNN model to an optimised version that can be used in an Android app. The key challenge in the project was preprocessing and masking the images. These images consisted of unwanted information in the background which was being unnecessarily computed while training the model. With the help of OpenCV library, we could cut out the sections of image containing the plant and darken the background of the image. These images were then directly fed to the CNN model, leading to greater accuracy in the results.

**Existing System**

The identification of plants by conventional keys is complex, time consuming, and due to the use of specific botanical terms frustrating for non-experts. This creates a hard to overcome hurdle for novices interested in acquiring species knowledge. Today, there is an increasing interest in automating the process of species identification.

There are many existing plant species and crop identification applications are existing and the technology stack they are using is based on image processing. Image-based methods are considered a promising approach for species identification. A user can take a picture of a plant in the field with the build-in camera of a mobile device and analyze it with an installed recognition application to identify the species or at least to receive a list of possible species if a single match is impossible. But they are many affecting factors that would make the image based calcification weaker and not predictable output.

**Proposed System**

The proposed technique involves three major steps, such as pre-processing, feature importance, and classification. Initially, the data will be pre-processed using the resampling technique. The main objective of this project is to classify a plant seedling and it’s health status from weed using Convolutional Neural Networks. And also develop an Android Application that has ability to classify it effectively can mean better crop and its health status yields and better stewardship of the environment just by capturing the leaf image in our android application.

**LITERATURE SURVEY**

Literature survey is a critical analysis of a portion of the published body of knowledge available through the use of summary, classification, and comparison of previous research studies, reviews of literature, and journal articles. A literature survey examines the current scholarly work available on a particular subject, perhaps within a given time period. It is the summary and synthesis of material gathered from various sources and organized to address an issue, research objective, or problem statement.

1. **Plant identification system using its leaf features**  
   Author: Pradeep Nijalingappa; V. J. Madhumathi  
   Publisher: IEEE  
   Year: 2017  
   In this paper they have done Plant identification based on leaf is becoming one of the most interesting and a popular trend. Each leaf carries unique information that can be used in the identification of plants. In the identification of plants based on leaf, the leaf images needs to be pre-processed accordingly to extract the various critical features. In this paper, we present the identification of plants based on leaf features using Multiclass SVM (MSVM) as a classifier.
2. **An Android Application for Plant Identification**  
   Author: Qian Cheng; Hongyan Zhao;

Publisher: IEEE  
Year: 2018  
This paper presents an Android application to automatically identify plant species using a single leaf image as input. At the pre-processing phase, we proposed an improved segmentation method to eliminate the noise caused by capturing on non-uniform background so we can obtain the binary image which only contains the leaf shape. Then, several morphological features and Hu moment invariants descriptors were extracted as inputs of a joint classifier which combines the back propagation neural network(BPNN) with a weighted k-nearest-neighbor (KNN) to distinguish 220 species of plants. The outputs of the joint classifier are the top ten species that best match the query leaf image. At the end, we implemented these algorithms on Android OS and the application we developed has been downloaded about a million times.

1. Classification of Leaf Images for Species Identification  
   Author: S. Santhosh; A. Fahima Zulfath  
   Publisher: IEEE  
   Year: 2019  
   In this paper they speak about that There are several million species of plants in the world, of which most of the species are not yet identified and recognized. Many species look similar, but they are actually not genetically the same, may belong to different families. It is very important to identify the plant species. Often taxonomist finds it difficult to classify the plants and sometimes may go wrong while grouping them. The key focus of this paper is to identify the variety of species by classifying the plant's leaves. This paper takes leaf image features to classify the species by using Support Vector Machines (SVM). This classification helps to recognize and identify the population of endangered and extint species for preservation.
2. Mobile Leaf Identification System using CNN applied to plants in Hokkaido  
   Author: Tatsuhiro Akiyama; Yosuke Kobayashi;  
   Publisher: IEEE  
   Year: 2019  
   In this paper they explained about The ability to identify plant types is important when conducting vegetation surveys. This ability requires investigators experience. We propose a mobile application using convolutional neural networks (CNNs) that will help beginners identify plant species. We compare three CNN models, VGG19, MobileNet, and MobileNetV2. Our plant identification application using MobileNetV2 shows an average F1 score of 0.992, indicating its high performance and practicality. The implemented system shows a practical performance of 338.1 ms per picture on a tablet-type device.

**SYSTEM REQUIREMENT AND SPECIFICATION**

The study of existing systems helps for a new system to be developed. Analysis starts with requirements and produces a specification of what the specification of what the system does. In order to implement any project, one has to gather requirement specifications. Hence the software and hardware requirements for development of the work along with the functional and non-functional requirements are specified.

**Functional Requirements**

A functional requirement specifies a function that a system or component must be able to perform. These include input, calculator, external interfaces, communications and special management information needs. Functional requirements are also called behavioural requirements because they address what the system does. Functional requirement specification contains the way in which a given task is to be performed, the results to be obtained as well as the elements of functional entities.

**Non-Functional Requirements**

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. Non-functional requirements are often called qualities of a system.

**Usability**: This project will take user input images which are captured using the mobile camera and has a very simple mobile interface to interact with.

**Reusability**: Each module of this project is written and tested independently so that they can be reused. Reusable modules and classes will reduce implementation time, and have eliminated bugs and localized code modification when a change in implementation is required.

**Scalability**: Even if there are additions of any number of nodes into the network the application will function normally. Hence the project provides scalability and also has the ability to work without internet,

**System Requirements**

**Software Requirements**

Operating System - Windows

Programming Language - Python & Kolin

IDE - Jupyter Notebook & Android Studio

Library - Tensorflow

**Hardware Requirements**

Processor - Intel Core i5 and above

Speed - 2 GHz or above

RAM - 4 GB Hard disk - 40GB

Input Device – Mobile Camera

**SYSTEM ANALYSIS**

**Feasibility study**

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

1. Economic Feasibility 2. Technical Feasibility 3. Operational Feasibility

**Economic Feasibility**

Economic Feasibility helps in assessing the viability, cost, and benefits associated with projects before financial resources are allocated. This assessment typically involves a cost/ benefits analysis of the project. The application is so designed that it requires minimal cost and eliminates costs as there would minimal need for manual work. The technologies used, help in understanding the user without any investment. As the machine will be trained it reduces the cost that is required to deploy the man power and also eliminates the problem of time consumption. And also our application would not require the network facility to process the images and give the analysis.

**Technical Feasibility**

Technical feasibility involves evaluation of the hardware and the software requirements of the proposed system. In this project, the technology involved is Machine Learning. The language that is used to implement the concepts of Machine Learning is Python Programming and the tool that is used to execute the Python code is Jupyter Notebook (IPython notebook). And as an interface we have used the kotlin as programming language to develop a simple android Application interface.

**Operational Feasibility**

The application involves design-dependent parameters such as reliability, maintainability, supportability, usability, disposability, sustainability, affordability, and others. It minimizes the drawbacks of the current system by building an application that automatically resolves the user queries and helps to analyse the user data.

**SYSTEM DESIGN**

The purpose of the design phase is to plan a solution of the problem specified by the requirements document. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed; design takes us toward how to satisfy the needs. The design of a system is perhaps the most critical factor affecting the quality of the software; it has a major impact on the later phases particularly testing and maintenance.

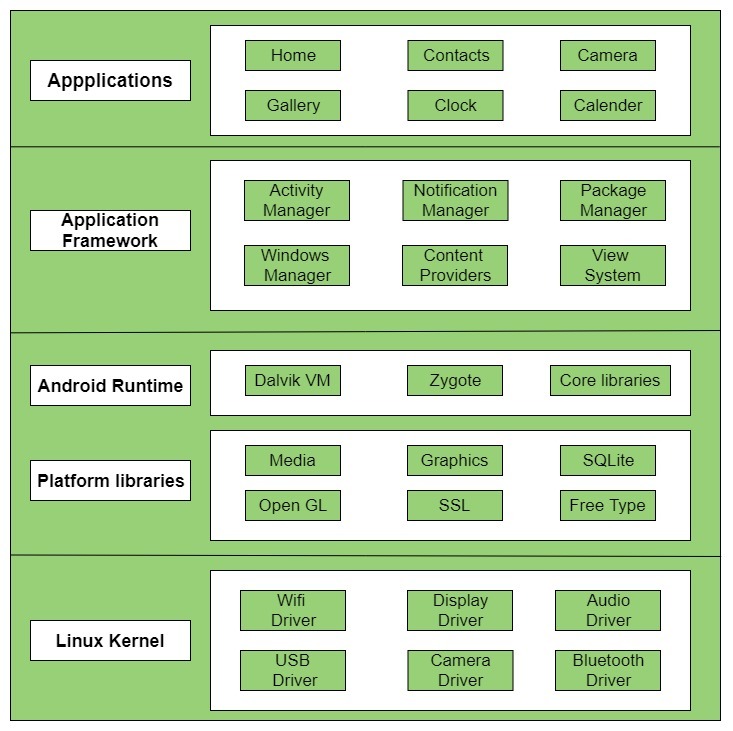
The design activity often results in three separate outputs

– ● Architecture design. ● High level design. ● Detailed design.  
  
**ARCHITECTURAL DESIGN**Architecture focuses on looking at a system as a combination of many different components, and how they interact with each other to produce the desired result. The focus is on identifying components or subsystems and how they connect. In other words, the focus is on what major components are needed.

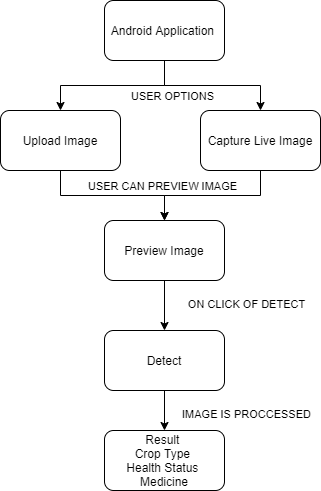
As a developer, the mobile developer Android architecture contains different number of components to support any android device needs. Android software contains an open-source Linux Kernel having collection of number of C/C++ libraries which are exposed through an application framework services.

Among all the components Linux Kernel provides main functionality of operating system functions to smartphones and Dalvik Virtual Machine (DVM) provide platform for running an android application. Linux Kernel is heart of the android architecture. It manages all the available drivers such as display drivers, camera drivers, Bluetooth drivers, audio drivers, memory drivers, etc. which are required during the runtime.

The Linux Kernel will provide an abstraction layer between the device hardware and the other components of android architecture. It is responsible for management of memory, power, devices etc.



**HIGH LEVEL DESIGN**

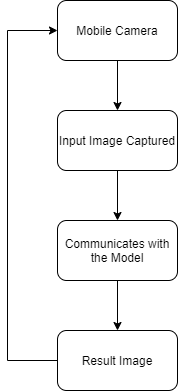
The purpose of this High Level Design (HLD) Document is to add the necessary detail to the current project . description to represent a suitable model for coding.

**DATA FLOW DIAGRAMS**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design). On a DFD, data items flow from an external data source or an internal data store to an internal data store or an external data sink, via an internal process.

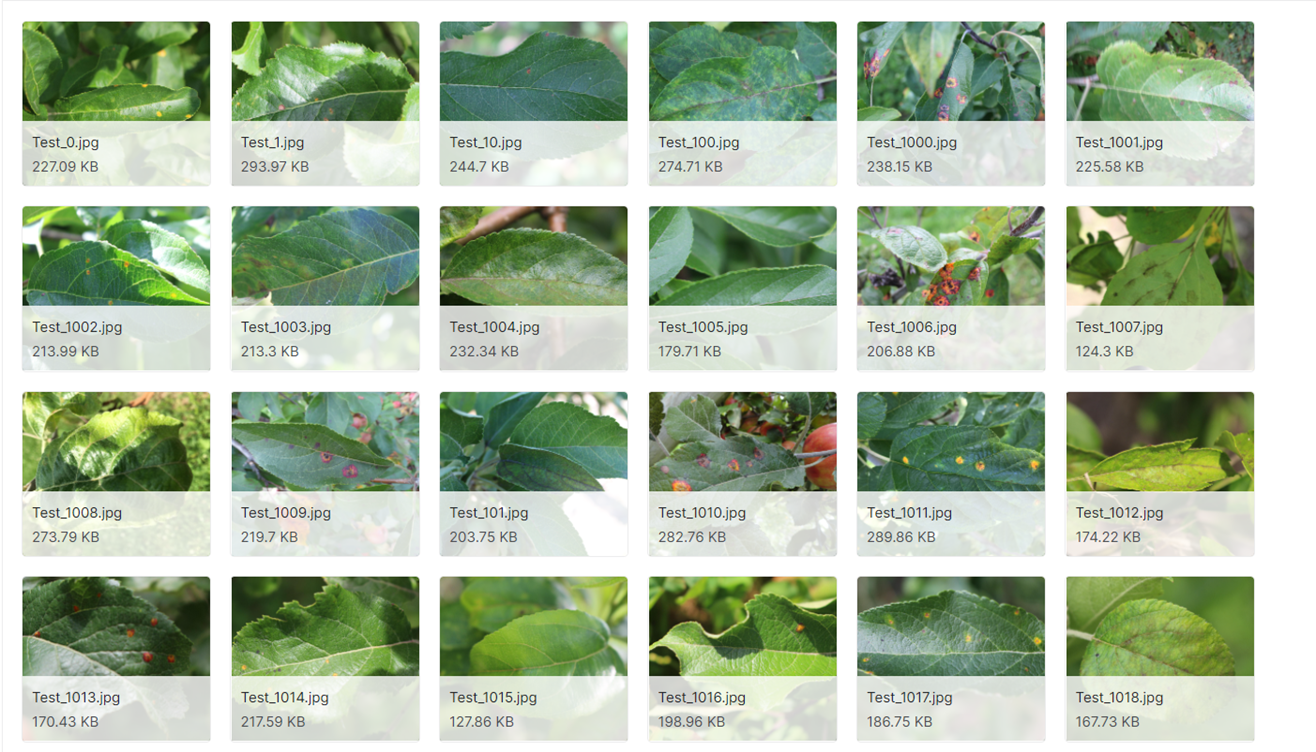
A DFD provides no information about the timing of processes, or about whether processes will operate in sequence or in parallel. It is therefore quite different from a flowchart, which shows the flow of control through an algorithm, allowing a reader to determine what operations will be performed, in what order, and under what circumstances, but not what kinds of data will be input to and output from the system, nor where the data will come from and go to, nor where the data will be stored (all of which are shown on a DFD).

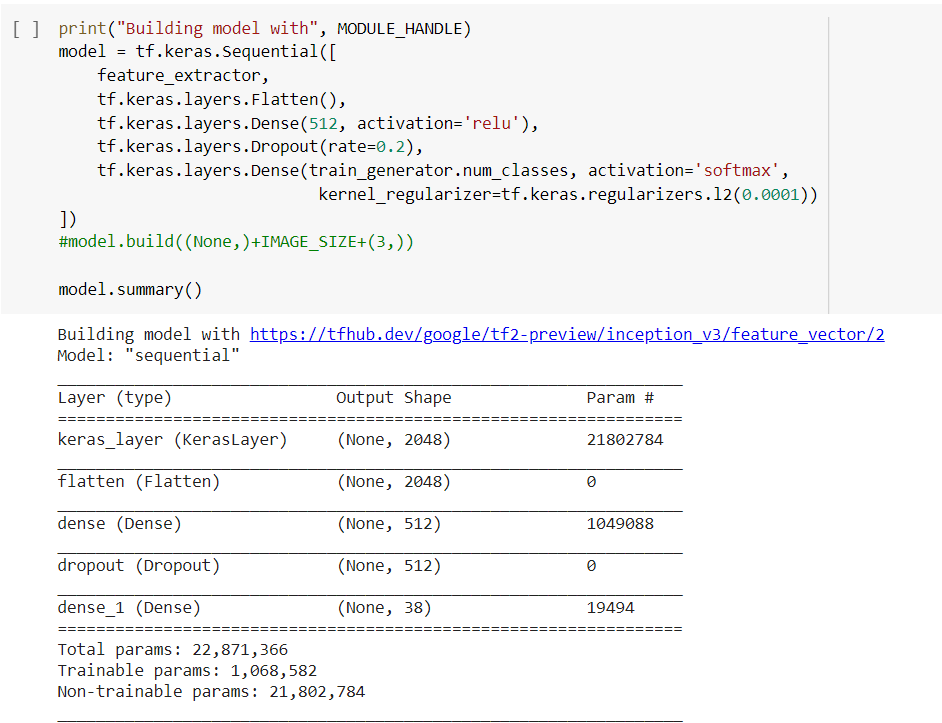
The data flow of our Android application works in the backend with the flow shown below



Module Description

Module 1 – Preparing the Machine Learning Model   
The dataset was taken from the open source <https://www.kaggle.com/emmarex/plantdisease> where we have collected all dataset which was needed in order to develop the calcification model. We have used the CNN machine learning concept in order to develop our model. The Inception network was an important milestone in the development of CNN classifiers. The Inception network on the other hand, was complex (heavily engineered). It used a lot of tricks to push performance; both in terms of speed and accuracy. Its constant evolution lead to the creation of several versions of the network. We have used the inception model 3.





Module 2 – Development of the Android Application for the interface.

Tensorflow Lite Converter converts a Tensorflow model to Tensorflow Lite flat buffer **file**(. **tflite**). Tensorflow Lite flat buffer **file** is deployed to the client, which in our cases can be a mobile device running on iOS or **Android** or an embedded device. There by the above developed model is been converted to a .tflite file which is been used to embed or interact with our Android Application.



**IMPLEMENTATION**

**Technological Survey**

**About Machine Learning**

Machine learning is a buzzword these days, the reason for this is the huge amount of data production by applications and the increase of computation power. The term machine learning was first introduced by Arthur Samuel in 1959. We can define it in a summarized was as: “Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.” Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without beginning 14 explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn from themselves.

**Working of a Machine Learning Model**

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of the predicted output depends up on the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately. Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it we just need to feed the data to generic algorithms, and with the help of these algorithms, the machine builds the logic as per the data and predicts the output. Machine learning has changed our way of thinking about the problem, Block diagram of machine learning algorithm is as follows

**Kaggle**

**Kaggle**, a subsidiary of [Google LLC](https://en.wikipedia.org/wiki/Google_LLC), is an online community of [data scientists](https://en.wikipedia.org/wiki/Data_science) and [machine learning](https://en.wikipedia.org/wiki/Machine_learning) practitioners. Kaggle allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges. One of the biggest hurdles of democratizing Deep Learning has been the need for a GPU accelerated environment. If we want to spend our time efficiently testing and learning across problem statements we need a GPU-based server. But these GPU based environments across platforms like GCP, AWS, Azure, Paper-space can amount to a huge cost. This cost when combined with the cost of storage and if we are using instances that are persistent may really add to the burden on our money bag.

But some amazing news came in when Kaggle and Google Colab introduced GPU-enabled kernels which can be now used to decipher and solve deep learning problems across the problem statements without the need to buy a GPU.

**Python Language**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

**Kotlin Android Development**

Kotlin is a general purpose, free, open source, statically typed “pragmatic” programming language initially designed for the JVM (Java Virtual Machine) and Android that combines object-oriented and functional programming features. It is focused on interoperability, safety, clarity, and tooling support. Versions of Kotlin targeting JavaScript ES5.1 and native code (using LLVM) for a number of processors are in production as well.

Kotlin originated at JetBrains, the company behind IntelliJ IDEA, in 2010, and has been open source since 2012. The Kotlin team currently has more than 90 full-time members from JetBrains, and the [Kotlin project on GitHub](https://github.com/JetBrains/kotlin) has more than 300 contributors. JetBrains uses Kotlin in many of its products including its flagship IntelliJ IDEA. Speaking of avoiding common errors, Kotlin was designed to eliminate the danger of null pointer references and streamline the handling of null values. It does this by making a null illegal for standard types, adding nullable types, and implementing shortcut notations to handle tests for null.

**Jupyter Notebook**

The IPython Notebook is now known as the Jupyter Notebook. It is an interactive computational environment, in which you can combine code execution, rich text, mathematics, plots and rich media.

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results. The IPython notebook combines two components:

A web application: a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output.

**6. SYSTEM DESIGN**

**6.1 UML Diagrams Introduction:**

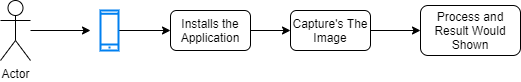
UML is a standard language for specifying, visualizing, constructing, and documenting the artefacts of software systems. UML can be described as a general- purpose visual modelling language to visualize, specify, construct and document software system. Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well like process flow in a manufacturing unit etc. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object oriented analysis and design. The goal of UML can be defined as a simple modelling mechanism to model all possible practical systems in today’s complex environment.

**6.2 Activity Diagram:**

**6.2.1 Definition:**

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams are intended to model both computational and organizational processes. Activity diagrams show the overall flow of control

**6.2.2 Activity Diagram for Plant Doctor**

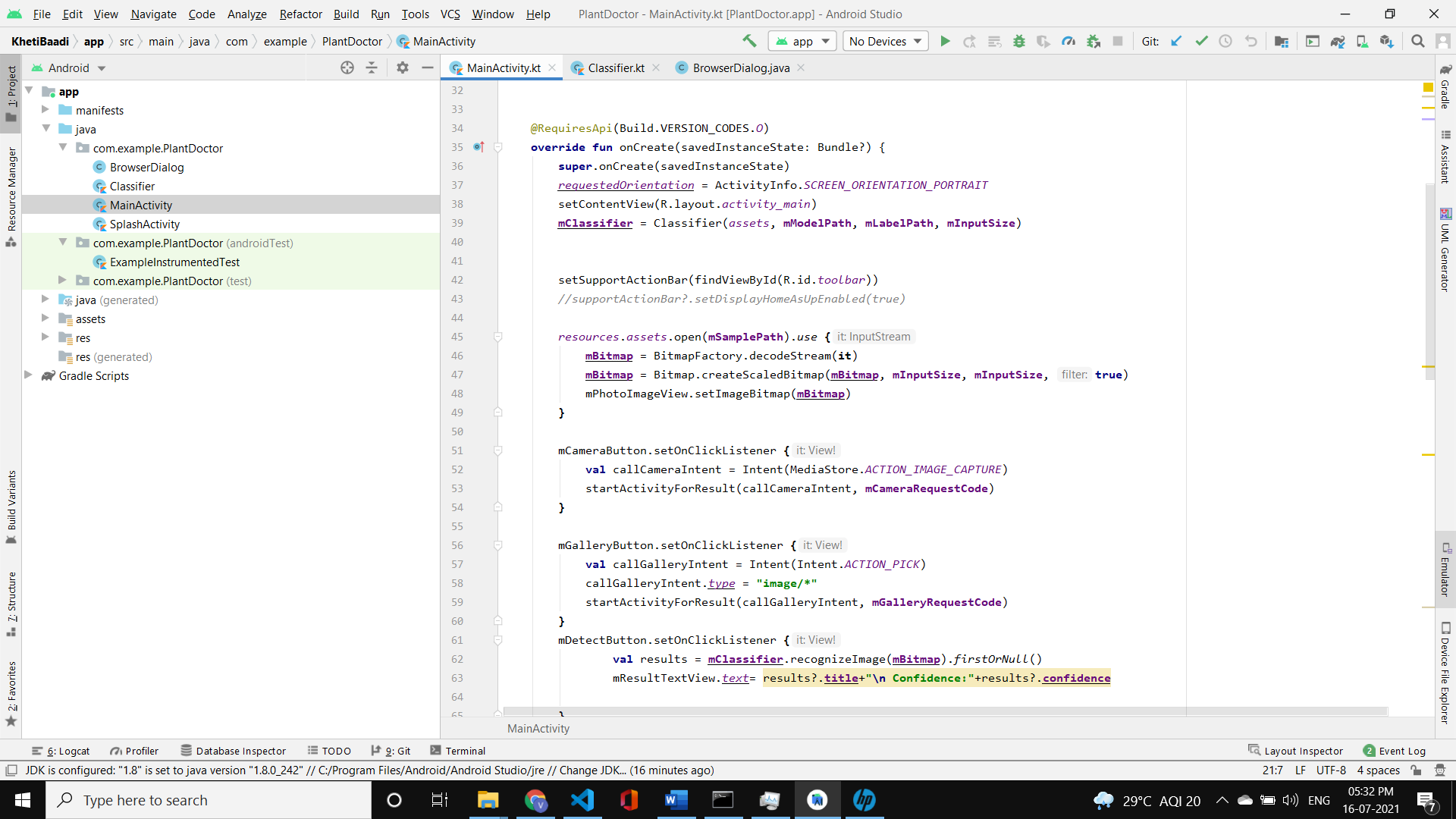
****

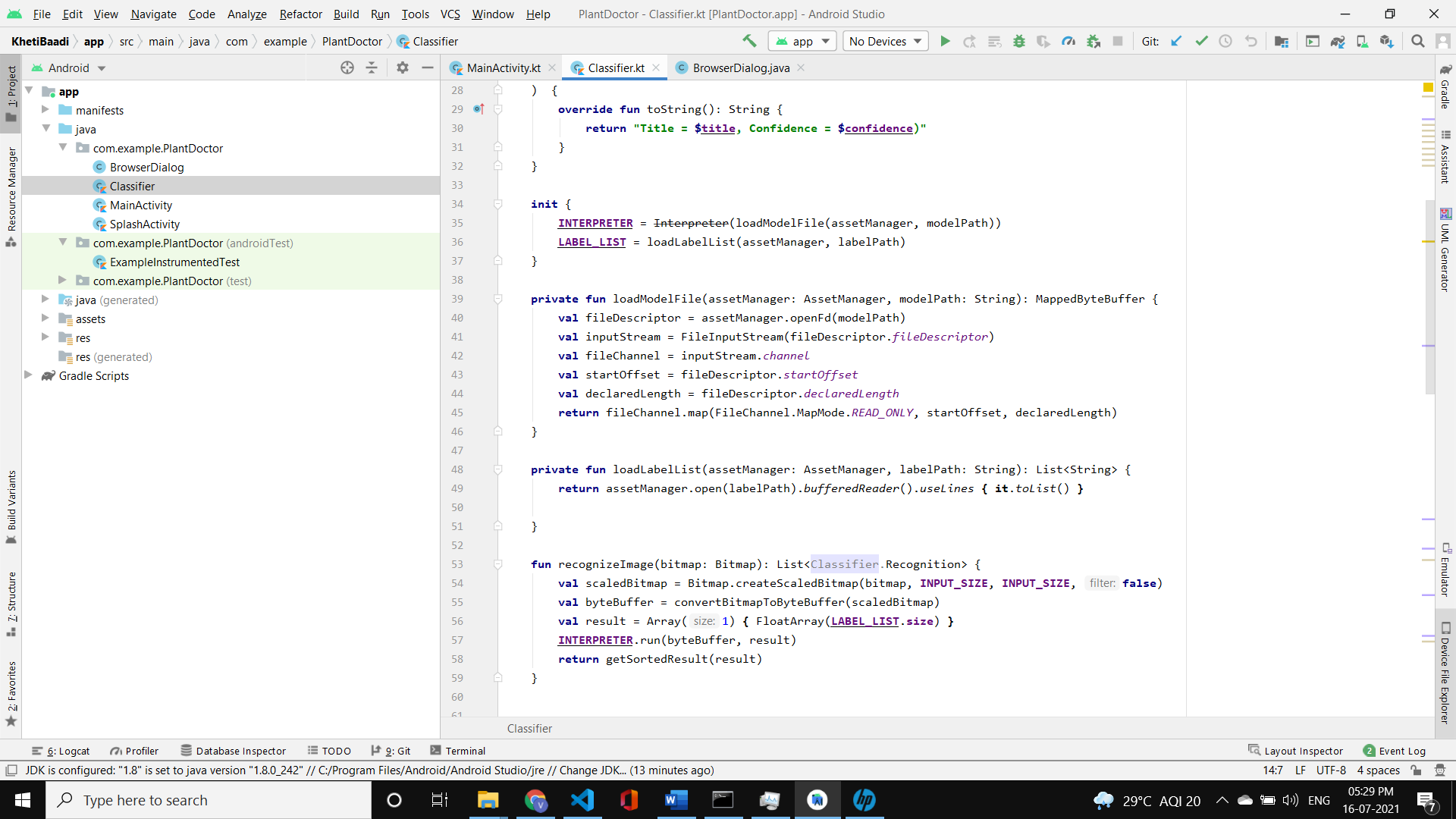
**Figure 6.2.2 Activity diagram for Plant Doctor**

7. IMPLEMENTATION

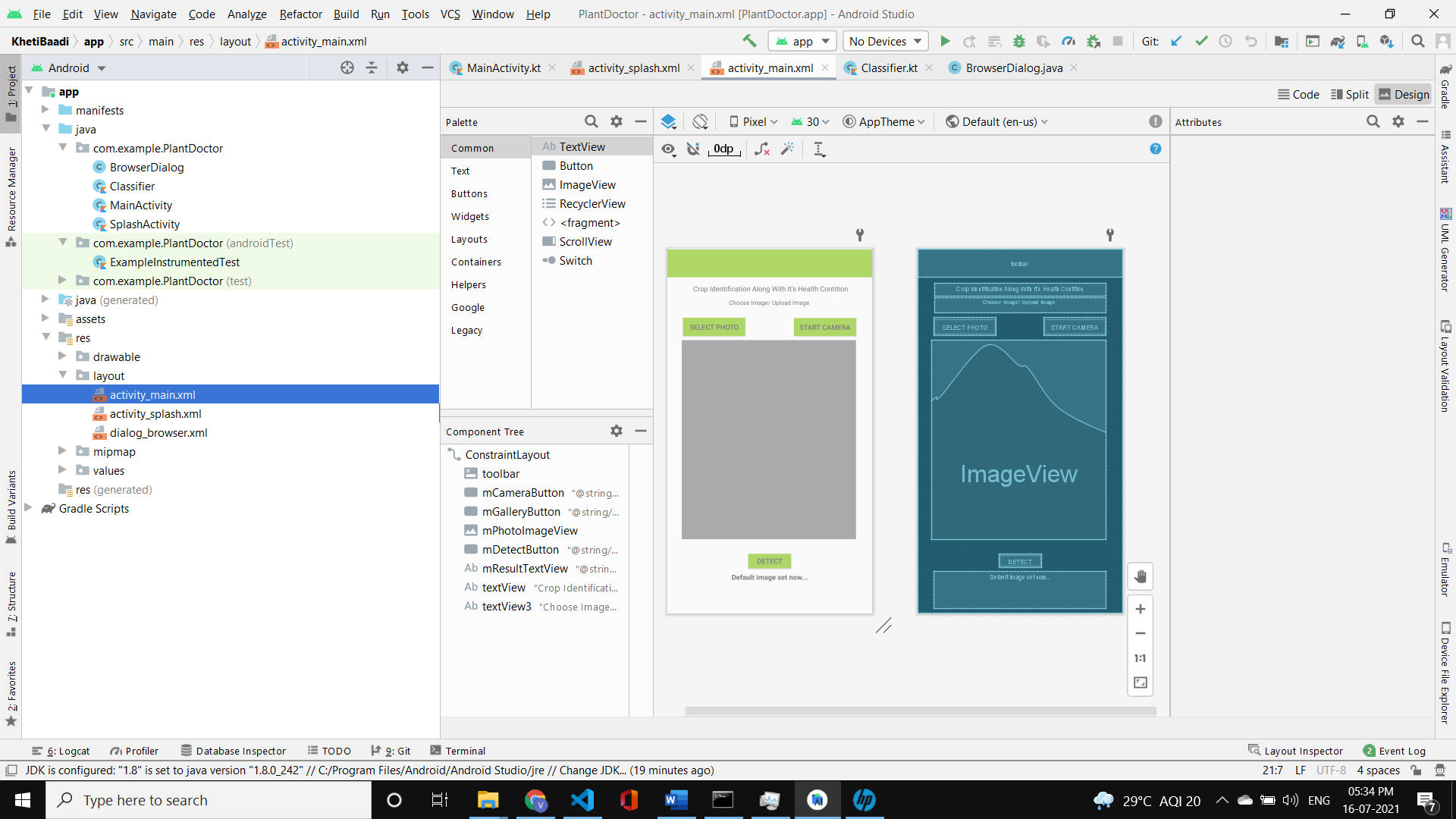
7.1 Coding

ADD ML PART WHICH IS THERE IN THE PPT

  
User interactions function are defined in the above code snippet



Function that Load’s the model in Android Application

  
Designing the user interface

**CHAPTER 7 Add snapshots of the App**

**CONCLUSION AND FUTURE ENHANCEMENT**