Senior Project 1

**صورة تحتوي على رسم, التصميم, توضيح

تم إنشاء الوصف تلقائياً**

*Design and Implementation for a Plant Care System (Bloom)*

|  |
| --- |
| Supervised By |

Dr. Fatmah Assiri

|  |
| --- |
| Team Members |

|  |  |
| --- | --- |
| 2110211 | Layali Adel Khayat |
| 2110808 | Laila Najem Alzahrani |
| 2115044 | Khamsa Saleh Alzahrani |
| 2115094 | Souad Mohammad Amal Ridwan |
| 1914977 | Maryam Maki Shami |

**Table of Contents**

[List of Illustrations 4](#_Toc165688515)

[Chapter 1 Introduction: 7](#_Toc165688516)

[1.1 Problem Deﬁnition: 7](#_Toc165688517)

[1.2 Aims and objectives: 7](#_Toc165688518)

[1.2.1 Project Aim: 7](#_Toc165688519)

[1.2.2 The objectives 7](#_Toc165688520)

[1.3 proposed solution: 8](#_Toc165688521)

[1.4 Novelty/Contribution: 8](#_Toc165688522)

[1.5 Report Outline 8](#_Toc165688523)

[1.6 Project Plan (Grant chart): 10](#_Toc165688524)

[1.7 Conclusion: 10](#_Toc165688525)

[Chapter 2 Introduction: 12](#_Toc165688526)

[2.2. Similar Project: 12](#_Toc165688527)

[2.2.1 PictureThis: 12](#_Toc165688528)

[2.2.2 Carl: 13](#_Toc165688529)

[2.2.3 PlantIn: 13](#_Toc165688530)

[2.2.4 Planto: 14](#_Toc165688531)

[2.4.1 Comparison 14](#_Toc165688532)

[2.4.2 Research Gap: 15](#_Toc165688533)

[2.5. Conclusion: 15](#_Toc165688534)

[Chapter 3 Introduction 17](#_Toc165688535)

[3.1. Requirements Gathering 17](#_Toc165688536)

[3.2. Functional Requirements (basic, middle, and high priority requirements) 17](#_Toc165688537)

[3.3. Non-Functional Requirements (basic, middle, and high priority requirements) 18](#_Toc165688538)

[3.4. Use-case Diagram 19](#_Toc165688539)

[3.5. Use-case Specifications 20](#_Toc165688540)

[3.6. ER Diagram 25](#_Toc165688541)

[3.7. Constraints 25](#_Toc165688542)

[3.9 Conclusion 26](#_Toc165688543)

[Chapter 4 Introduction 28](#_Toc165688544)

[4.1. Product Backlog 28](#_Toc165688545)

[4.2. Sprint Backlog 28](#_Toc165688546)

[4.3. Task and Their Allocation 29](#_Toc165688547)

[4.4. Burndown Chart 29](#_Toc165688548)

[4.5. Conclusion 30](#_Toc165688549)

[Chapter 5 Introduction 32](#_Toc165688550)

[5.1. Class Diagram 32](#_Toc165688551)

[5.2. Activity Diagram 33](#_Toc165688552)

[5.3 Conclusion 33](#_Toc165688553)

[Chapter 6 Introduction 35](#_Toc165688554)

[6.1. Engineering Standards 35](#_Toc165688555)

[6.1.1. Interaction Capability 35](#_Toc165688556)

[6.1.2. Functional Suitability 35](#_Toc165688557)

[6.1.3. Maintainability 36](#_Toc165688558)

[6.2. Programming Language and Tools 36](#_Toc165688559)

[6.3. Code Snippets of main Function 38](#_Toc165688560)

[6.4. Unit Testing 39](#_Toc165688561)

[6.5. Conclusion 39](#_Toc165688562)

[Reference 40](#_Toc165688563)

# List of Illustrations

#### Figures

[***Figure 1*** *Gantt Chart for milestone and deadline 10*](#_Toc165688733)

[***Figure 2*** *PictureThis Application 12*](#_Toc165688734)

[***Figure 3*** *Carl Application 13*](#_Toc165688735)

[***Figure 4*** *PlantIn Application 13*](#_Toc165688736)

[***Figure 5*** *Planto Application 14*](#_Toc165688737)

[***Figure 6*** *Use-case Diagram 19*](#_Toc165688738)

[***Figure 7*** *ER Diagram 25*](#_Toc165688739)

[***Figure 8*** *Sprint 1 Backlog 28*](#_Toc165688740)

[***Figure 9*** *Task and their allocation 29*](#_Toc165688741)

[***Figure 10*** *Burndown Chart 29*](#_Toc165688742)

[***Figure 11*** *Class Diagram 32*](#_Toc165688743)

[***Figure 12*** *Activity Diagram 33*](#_Toc165688744)

#### Tables

[***Table 1*** *Comparison Between Current Solution Apps Features 14*](#_Toc165681640)

[***Table 2*** *Functional Requirements 17*](#_Toc165681641)

[***Table 3*** *Non-Functional Requirements 18*](#_Toc165681642)

[***Table 4*** *Use-Case Specification – Take Picture 20*](#_Toc165681643)

[***Table 5*** *Use-Case Specification – Identify plant 20*](#_Toc165681644)

[***Table 6*** *Use-Case Specification – View information 21*](#_Toc165681645)

[***Table 7*** *Use-Case Specification – Create account 21*](#_Toc165681646)

[***Table 8*** *Use-Case Specification – login 22*](#_Toc165681647)

[***Table 9*** *Use-Case Specification – Save plant 22*](#_Toc165681648)

[***Table 10*** *Use-Case Specification – Set Reminder 23*](#_Toc165681649)

[***Table 11*** *Use-Case Specification – Sell product 23*](#_Toc165681650)

[***Table 12*** *Use-Case Specification – Buy product 24*](#_Toc165681651)

[***Table 13*** *Product Backlog 28*](#_Toc165681652)

[***Table 14*** *Engineering Standards 35*](#_Toc165681653)

[***Table 15*** *Test cases 39*](#_Toc165681654)

**Chapter 1: Introduction**

# 

# Chapter 1 Introduction:

The Kingdom of Saudi Arabia faces numerous challenges in agriculture, despite its essential role as the primary food source. Some people face challenges when they take an interest in agriculture due to ignorance or lack of necessary components like fertile land, water, seeds, and suitable weather conditions, and the overwhelming amount of information they might need. Agriculture is a complex puzzle, and it takes more than just planting seeds and hoping for rain. It needs awareness of different aspects of a plant, weather, environment, watering frequency, adaptation abilities, and more. Therefore, people usually depend on sources that may be incomplete to obtain information about plants or methods of growing them. These sources are usually not rich in sufficient information or are written in another language that the person does not speak. We created a solution that is an efficient, organized, and complete source for all information needed to grow a plant, take care of it, and maintain it in the long run in a simple to understand way. As well as an E-store to trade necessary resources.

# 1.1 Problem Deﬁnition:

# Numerous individuals face difficulty in recognizing their plants, which can result in uncertainty regarding their care needs. This lack of knowledge often leads to challenges in providing adequate attention.

# 1.2 Aims and objectives:

# 1.2.1 Project Aim:

The goal is to leverage machine learning technology in creating a mobile application where users can identify the plant type using image processing and then can access comprehensive plant information and detailed care methods for both agriculture and gardening. Additionally, the application will include a convenient store section offering a variety of agricultural resources among registered users.

# 1.2.2 The objectives

1-To create a centralized source of plant information.

2-To allow users to identify plants using image processing.

3- To facilitate resource exchange among users to enhance community collaboration.

# 1.3 proposed solution:

The proposed solution is to develop an application that simplifies the process of identifying plant type using image processing and then obtaining plant information. The application will also allow the user to create a watering schedule with a notification system for watering the plant it will also create a community for interested users to exchange resource by creating an e-store for planting products.

# 1.4 Novelty/Contribution:

* Providing centralized plant information in Arabic.
* Using machine learning technology to simplify searching via image processing and plant identification.
* Hosting an E-store platform for users to exchange resources.

# 1.5 Report Outline

**Chapter 1. Introduction**

Introduction

1.1. Problem Definition

1.2. Aims & Objectives

1.3. Proposed Solution

1.4. Novelty/Contribution

1.5. Report Outline

1.6. Project Plan (Grant chart)

1.7. Conclusion

**Chapter 2. Related Works**

Introduction

2.1. Scientific Papers

2.2. Existing Systems/4 similar projects

2.4. Comparison Results

2.5. Conclusion

**Chapter 3. Requirement Gathering and Analysis**

Introduction

3.1. Requirements Gathering

3.2. Functional Requirements (basic, middle, and high priority requirements)

3.3. Non-Functional Requirements

3.4. Use-case Diagram

3.5. Use-case Specifications

3.6. Design Constraints

3.7. Conclusion

**Chapter 4. Methodology and Tools**

Introduction

4.1. Product Backlog

4.2. Sprint Backlog

4.3. Tasks and their allocation.

4.4. Burn down chart.

4.5. Conclusion

College of Computer Science and

Engineering

University of Jeddah

Chapter 5. Analysis and Design

Introduction

5.1. Class diagram

5.2 ER/DFD/sequence/activity/state diagrams (any other relevant diagrams)

5.3 Conclusion

**Chapter 6. Implementation and Testing (Sprint 1)**

Introduction

6.1. Engineering Standards

6.2. Programming Language and Tools

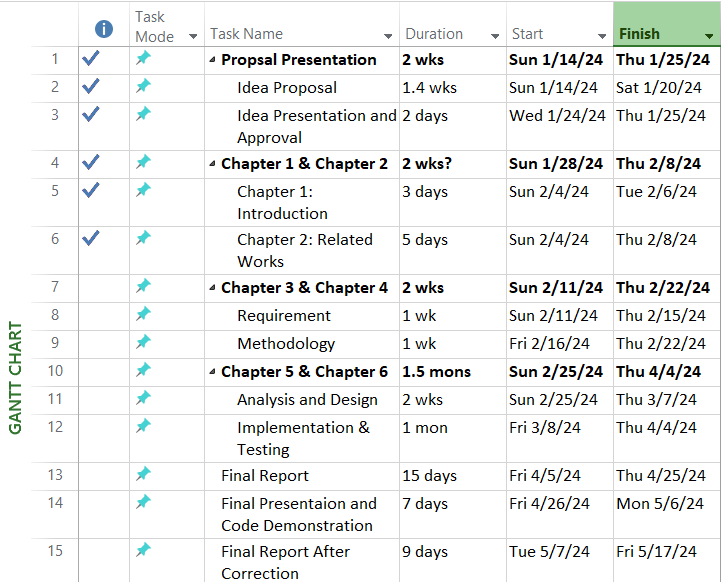
6.3. Code Snippets of Main Functions

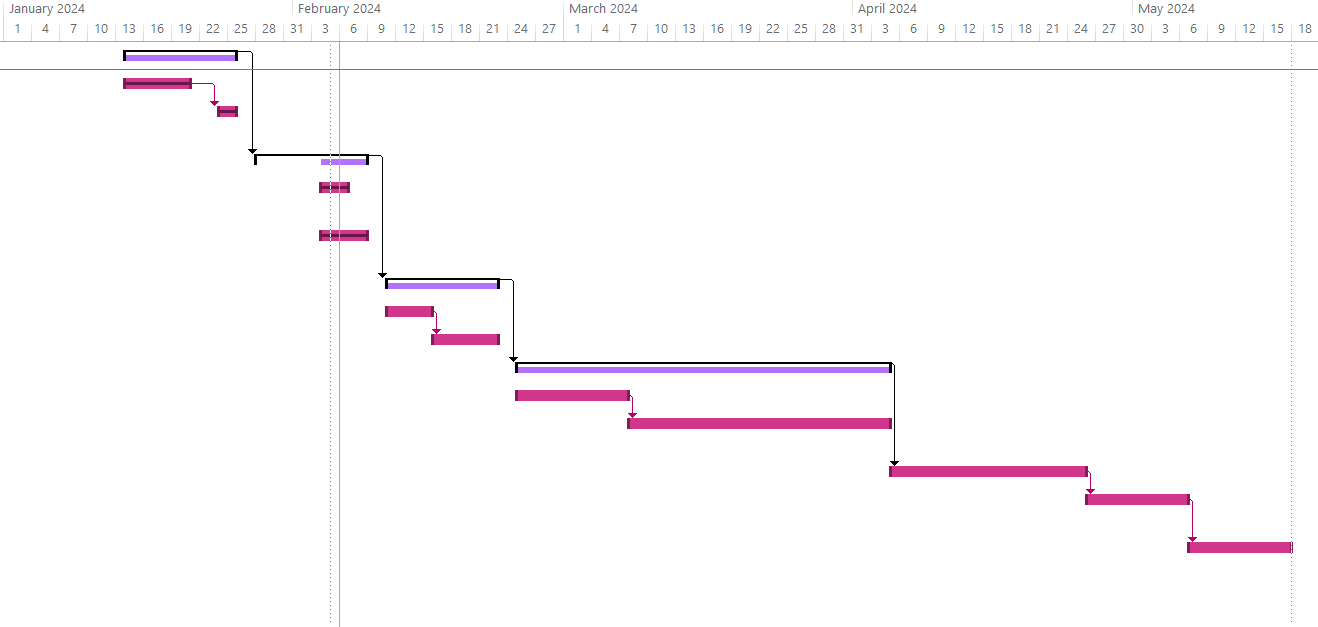
6.4. Sprint 1 Interfaces

6.5. Unit Testing

6.6. Conclusion

# 1.6 Project Plan (Grant chart):

****

****

**Figure 1** Gantt Chart for milestone and deadline.

# 

# 1.7 Conclusion:

This chapter is an introduction to our project. Through it, we identified the problem that we have and reviewed it to propose the best and most appropriate solution to it, and that is what we arrived at in the proposed solution. Moreover, the aims and objectives of the projects were mentioned, in addition to an attached Gantt chart outlining the progress of the project's tasks.

**Chapter 2: Related Works**

## Chapter 2 Introduction:

There are numerous types of plants due to biodiversity, and the efforts to understand them created the need for classifications, which are only truly understood and known by taxonomy experts. However, in recent years there have been efforts to create automated solutions for this [1], which include the utilization of machine learning. This can also help to save biodiversity and greenify places [1], and aid in the Saudi greenifying 2030 vision. The flat nature of plant morphology helps in its identification. Thankfully, computer vision and machine learning methods, which have proven highly successful in various fields, are now being utilized to accurately identify organisms, particularly plants, with precision levels exceeding (90% or more) when identifying a plant by its leaf [1]. Based on this study, we can see that the average person can’t recognize a plant species without it being a majorly common plant, such as apples, cherry blossoms, basil, etc. but when it comes to less commonly known plants, especially ones associated with gardening and agriculture for multiple purposes, things can get complicated. This creates the need for an automated solution which has been developed through machine learning and computer vision, and that’s where our proposed solution, the Bloom application, comes into hand.

# 2.2. Similar Project:

## 2.2.1 PictureThis:

A logo of a flower

Description automatically generatedPictureThis is an application that uses image recognition technology to identify plants and flowers from photos taken by users. It provides detailed information in English about the identified plant species, including their common and scientific names, care instructions, growth habits, and even tips for plant maintenance. The application also enables users to connect with fellow gardening enthusiasts, fostering a community where they can share their plant photos and exchange tips and experiences. Additionally, PictureThis offers a premium version with more features such as unlimited identification requests and access to a larger plant database.

**Figure 2** PictureThis Application.

## 2.2.2 Carl:

Carl is an app for plant identification that helps users identify plants, weeds, and trees with a snapshot using artificial intelligence in English. Additionally, the application offers details about the identified plant, including its common name, scientific name, and description. Carl gives users access to plant specialists for expert assistance and guidance. Other features include a plant disease identification tool, a plant care guide, a watering reminder, and a plant light meter tool.

**Figure 3** Carl Application.

## 2.2.3 PlantIn:

PlantIn is an application employing image recognition technology to aid users in identifying plants and mushrooms effortlessly through photos. It offers details on identified plants in English, encompassing common and scientific names, traits, general info, care tips, and maintenance needs. Users can interact with others, ask questions, and exchange plant images. Additionally, PlantIn offers a premium tier featuring perks such as season pass collection, light meter, and water calculator functionalities.



**Figure 4** PlantIn Application.

## 2.2.4 Planto:

An AI plant identifier application typically utilizes artificial intelligence and image recognition technology to identify plants based on photos uploaded by users in English. These applications analyze visual features such as leaf shape, color, and texture to match them against a database of known plant species. They can be useful for plant lovers, botany students, or anyone interested in identifying plants in their environment, especially for users comfortable with English.

**Figure 5** Planto Application

A black and white logo

Description automatically generated

# 2.4.1 Comparison

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Number** | **App Name** | **Features** | | | | | |  |
| **Plant Care Information** | **Plant Identification Throw Image** | | **Watering Reminder** | **Community Tab** | **E-Store for Selling and Buying Resources** | **Language** |
| **1** | PictureThis | Yes | Yes | Yes | | No | No | English |
| **2** | Carl | Yes | Yes | Yes | | No | No | English |
| **3** | PlantIn | Yes | Yes | Yes | | Yes | No | English |
| **4** | Planto | Yes | Yes | Yes | | No | No | English |
| **5** | Bloom (our application) | Yes | Yes | Yes | | Yes | Yes | English/Arabic |

**Table 1** Comparison Between Current Solution Apps Features.

## 2.4.2 Research Gap:

Based on the comparison table above, we concluded that all applications have the feature of identifying plants and caring for them, in addition to the watering reminders. Some of the applications aren’t concerned about creating communities, and none of the applications facilitate selling and buying resources between users. Additionally, it's worth noting that our application will be available in Arabic, aiming to target non-English speakers and enhance accessibility.

## 2.5. Conclusion:

We examined some scientific papers to look further into the problem we’re handling and have concluded that plant identification can be a challenge to the public. AI has been and is being used for the purpose of plant identification, which resulted in the emergence of different applications based on image recognition technology. However, not all applications have the same features, and none of them have the option of resource selling and buying.

**Chapter 3: Requirement Gathering**

**and Analysis**

# Chapter 3 Introduction

In this chapter, we have defined requirements gathering and the functional, and non-functional requirements, as well as a use case diagram with its specifications and an ER diagram.

# 3.1. Requirements Gathering

We looked at similar apps and extracted the most important requirements that they either have or lack.

# 3.2. Functional Requirements (basic, middle, and high priority requirements)

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Requirement statement** | **Priority** |
| FR1 | The system must allow the user to take a picture of a plant. | High |
| FR2 | The system must identify the plant. | High |
| FR3 | The system must show the plant information. | High |
| FR4 | The system shall allow the user to create an account with a username and password. | Medium |
| FR5 | The system shall allow the user to login by entering a username and a password. | Medium |
| FR6 | The system shall save the plant to their account. | Medium |
| FR7 | The system shall allow the user to set a reminder. | Low |
| FR8 | The system shall allow the user to add a product for sale in the store. | Medium |
| FR9 | The system shall allow the user to buy a product. | Medium |

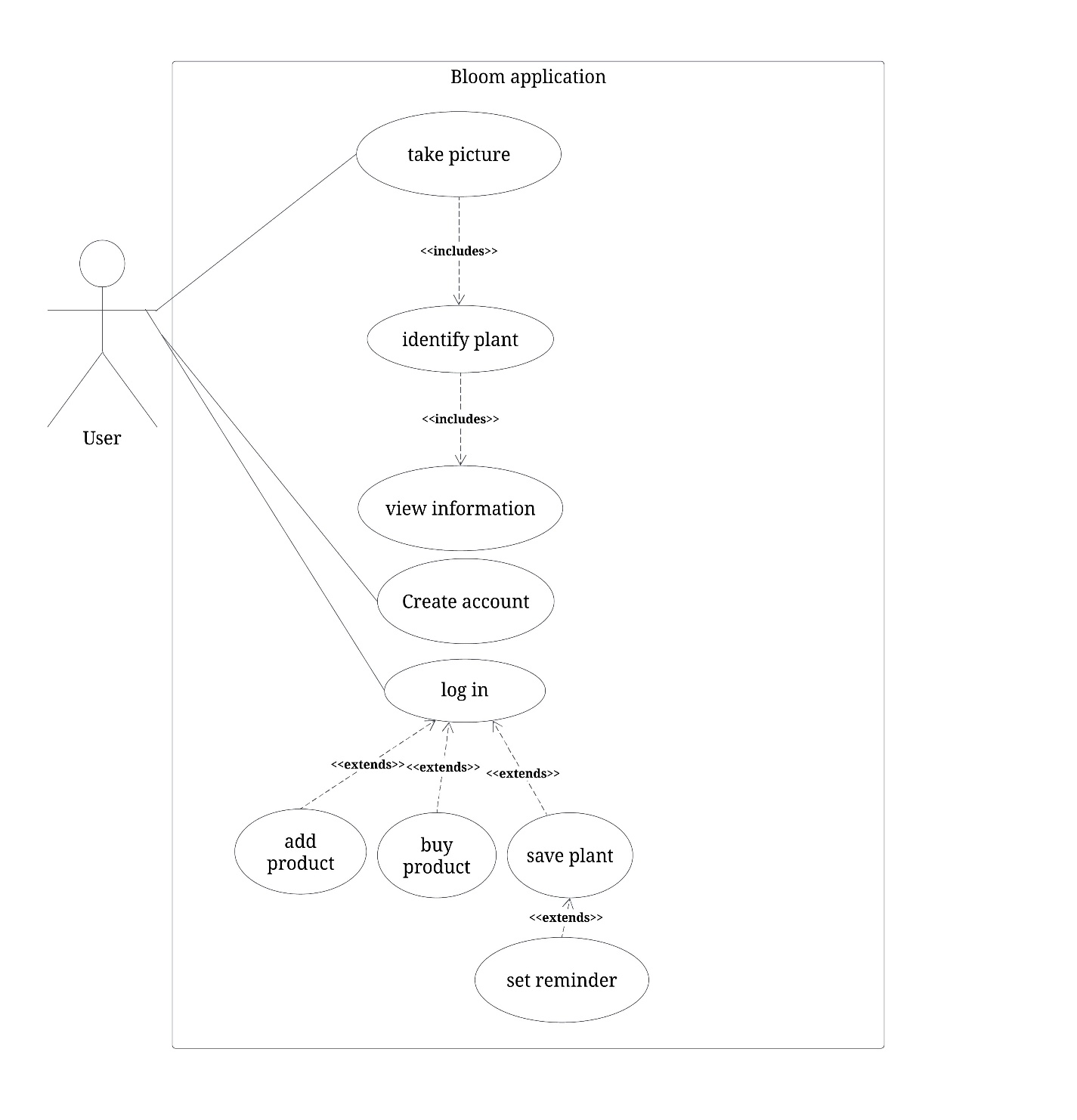
**Table 2** Functional Requirements.

# 3.3. Non-Functional Requirements (basic, middle, and high priority requirements)

|  |  |  |
| --- | --- | --- |
| **Requirement ID** | **Requirement statement** | **Priority** |
| NF1 | Confidentiality:   * The system should allow the user to create a username and password for their account. | High |
| NF2 | Reliability:   * We will apply different type of testing to make sure that our system is reliable. | High |
| NF3 | Useability:   * The system will be designed according to UI standers to make it easy for the user to use. * The system must allow easy navigation within the application. | Medium |

**Table 3** Non-Functional Requirements.

# 3.4. Use-case Diagram



**Figure 6** Use-case Diagram.

# 3.5. Use-case Specifications

|  |  |
| --- | --- |
| Use Case name | Take Picture |
| Use Case- Description | The system must allow the user to take pictures of plants for identification. |
| Actors | User |
| Precondition | The user must have the application installed. |
| Post Condition | The user successfully captures a picture of a plant. |

***Table 4*** *Use-Case Specification – Take Picture.*

|  |  |
| --- | --- |
| Use Case name | Identify plant |
| Use Case- Description | The system must allow the user to identify plants by capturing and submitting photos. |
| Actors | User |
| Precondition | The user must take or submit a photo of a plant to be identified. |
| Post Condition | The system successfully identifies the plant from the submitted photo. |

***Table 5*** *Use-Case Specification – Identify plant.*

|  |  |
| --- | --- |
| Use Case name | View information |
| Use Case- Description | The system must allow the user to view detailed information about a plant. |
| Actors | User |
| Precondition | The system should have identified a plant from the user’s photo. |
| Post Condition | The user successfully views detailed information about the selected plant. |

***Table 6*** *Use-Case Specification – View information.*

|  |  |
| --- | --- |
| Use Case name | Create account |
| Use Case- Description | The system must allow the user to create a new account to access application features. |
| Actors | User |
| Precondition | The user must have the application installed and have attempted to save the information of a plant identified or tried to access the E-store. |
| Post Condition | The user successfully creates a new account. |

**Table 7** Use-Case Specification – Create account.

|  |  |
| --- | --- |
| Use Case name | login |
| Use Case- Description | A user provides a username and password to gain access to the system |
| Actors | User |
| Precondition | The user must have a registered account with the system |
| Post Condition | Upon successful authentication, the user gains access to the system |

**Table 8** Use-Case Specification – login.

|  |  |
| --- | --- |
| Use Case name | Save plant |
| Use Case- Description | The system must allow the user to save plant information for future reference. This functionality extends from the “View Plant Information” use case. |
| Actors | User |
| Precondition | The user must have the app installed, be logged in, and have viewed plant information. |
| Post Condition | The user successfully saves the plant information for future reference. |

**Table 9** Use-Case Specification – Save plant.

|  |  |
| --- | --- |
| Use Case name | Set Reminder |
| Use Case- Description | The system must allow the user to set a reminder for plant care tasks. |
| Actors | User |
| Precondition | The user must have an account and be logged in, and saved plant information. |
| Post Condition | The user successfully sets a reminder for watering of the saved plant. |

**Table 10** Use-Case Specification – Set reminder.

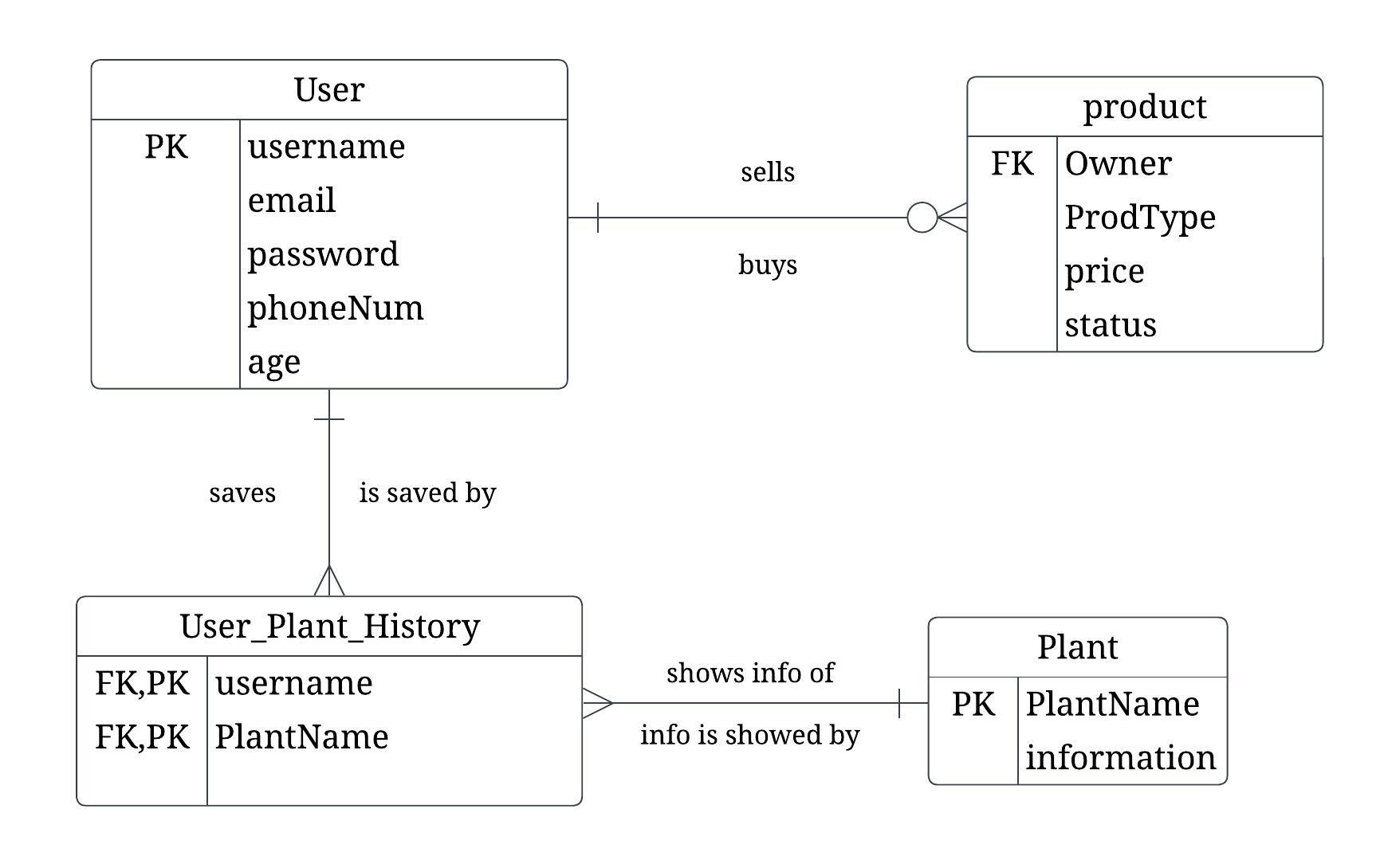
|  |  |
| --- | --- |
| Use Case name | Sell product |
| Use Case- Description | The system must allow the user to add a product to the E-store to sell. |
| Actors | User |
| Precondition | The user must have viewed the store and be logged in. |
| Post Condition | The user successfully adds a product to sell |

**Table 11** Use-Case Specification – Sell product.

|  |  |
| --- | --- |
| Use Case name | Buy product. |
| Use Case- Description | The system must allow the user to pay for a product from the store after viewing its offerings. |
| Actors | User |
| Precondition | The user must have logged in, viewed the store, and added a product to their cart. |
| Post Condition | The user successfully completes the payment for the selected product. |

**Table 12** Use-Case Specification – Buy product.

# 3.6. ER Diagram

****

**Figure 7** ER Diagram.

# 3.7. Constraints

Technical Constraints:

* Our system will be operating on Android. because of the technical challenges in ensuring consistent performance and functionality across all of the different platforms, and the limitations of the IOS system due to its closed source status.
* The accuracy of plant identification is dependent on the application dataset. If the dataset is not inclusive enough, the application may struggle to identify lesser-known or newly discovered plants. As well as specific parameters such as color, depth, height, width, and lighting of the image affecting the identification process.
* Ensuring compatibility with android

Time and Schedule Constraints:

* There may be a limited timeframe to develop and launch the application, especially because of deadlines. This constraint can affect the application.
* The project's financial resources may limit acquiring necessary tools and technologies, potentially affecting the development timeline.

# 3.9 Conclusion

This chapter covered the functional, the non-functional requirements, the use case diagram with its specifications, the ER diagram, and the constraints of our application development.

**Chapter 4: Methodology and Tools**

## Chapter 4 Introduction

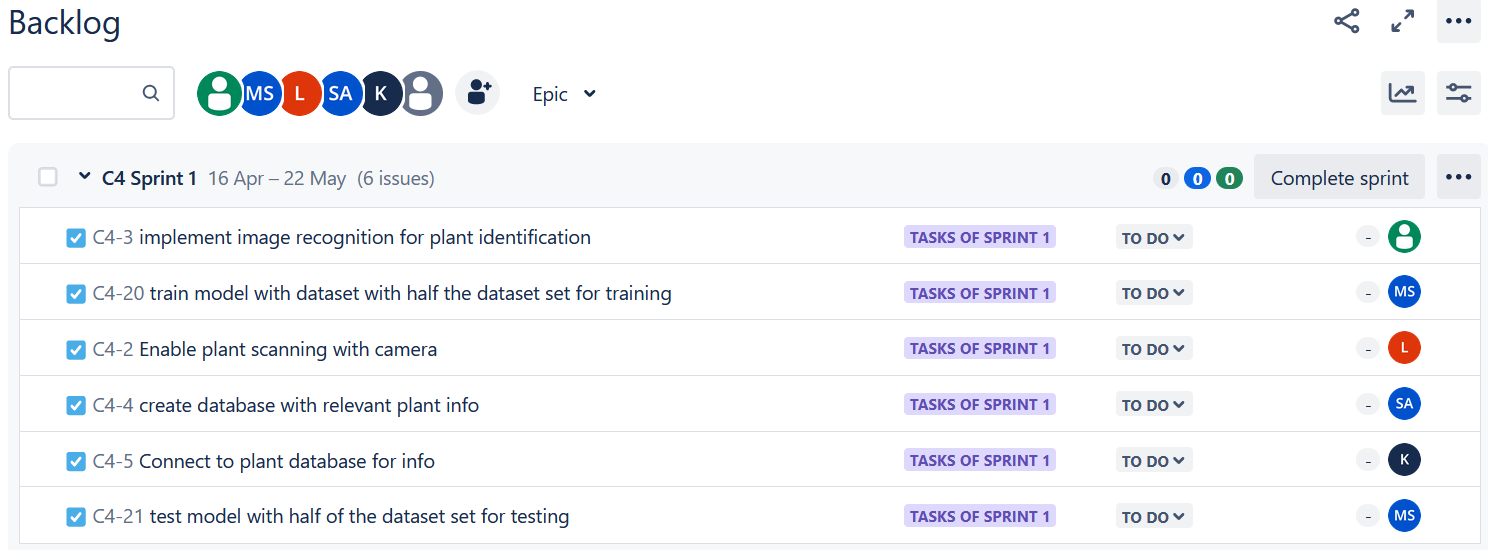
In this chapter, we're diving into Agile project management basics. We'll cover Product Backlog and Sprint Backlog, which will help us organize what needs to be done and when. Then, we'll talk about Tasks and how we assign them to team members. Finally, we'll look at the Burndown Chart, a handy tool for keeping track of progress.

## 4.1. Product Backlog

|  |  |
| --- | --- |
| US1 | The system should use Machen learning to identify the plants. |
| US2 | As a user, I want to scan a plan through a camera. |
| US3 | As a user, I can get information about the plant. |
| US4 | As a user, I can create an account. |
| US5 | As a user, I can log in to the system with my username and password. |
| US6 | As a user, I can set a water reminder for the plant. |
| US7 | As a user, I can add products to the e-store. |
| US8 | As a user, I can buy products from the e-store. |

**Table 13** Product Backlog.

## 4.2. Sprint Backlog



**Figure 8** Sprint 1 Backlog.

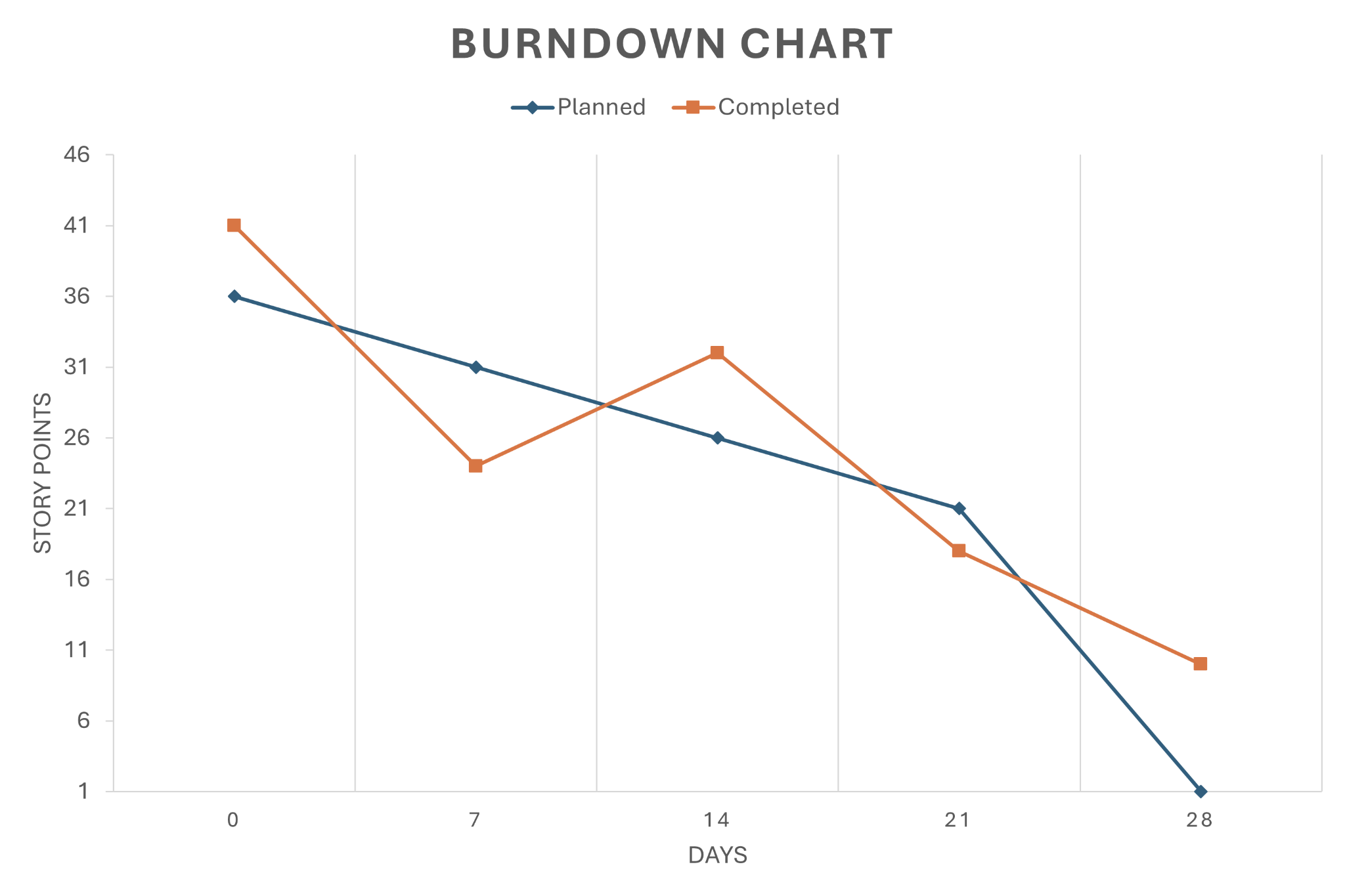
## 4.3. Task and Their Allocation

صورة تحتوي على نص, برمجيات, أيقونة الحاسوب, صفحة ويب

تم إنشاء الوصف تلقائياً

**Figure 9** Task and their allocation.

## 4.4. Burndown Chart



**Figure 10** Burndown Chart.

## 4.5. Conclusion

We established the foundation for our project's development by delineating the product backlog. This comprehensive inventory encapsulates the essential user stories. Through this meticulous planning, we've outlined the core features necessary for our system's functionality.

Furthermore, we introduced the sprint backlog, breaking down the user stories into specific tasks allocated to team members. This strategic approach ensures an organized development process, fostering collaboration and accountability within the team.

Additionally, we emphasized the importance of tracking progress through the burndown chart, enabling us to monitor our development velocity and make necessary adjustments to ensure project success.

**Chapter 5: Analysis and Design**

## Chapter 5 Introduction

In this chapter, we will draw the class diagram to design our application, as well as additional diagrams we chose that are relevant and appropriate for our application.

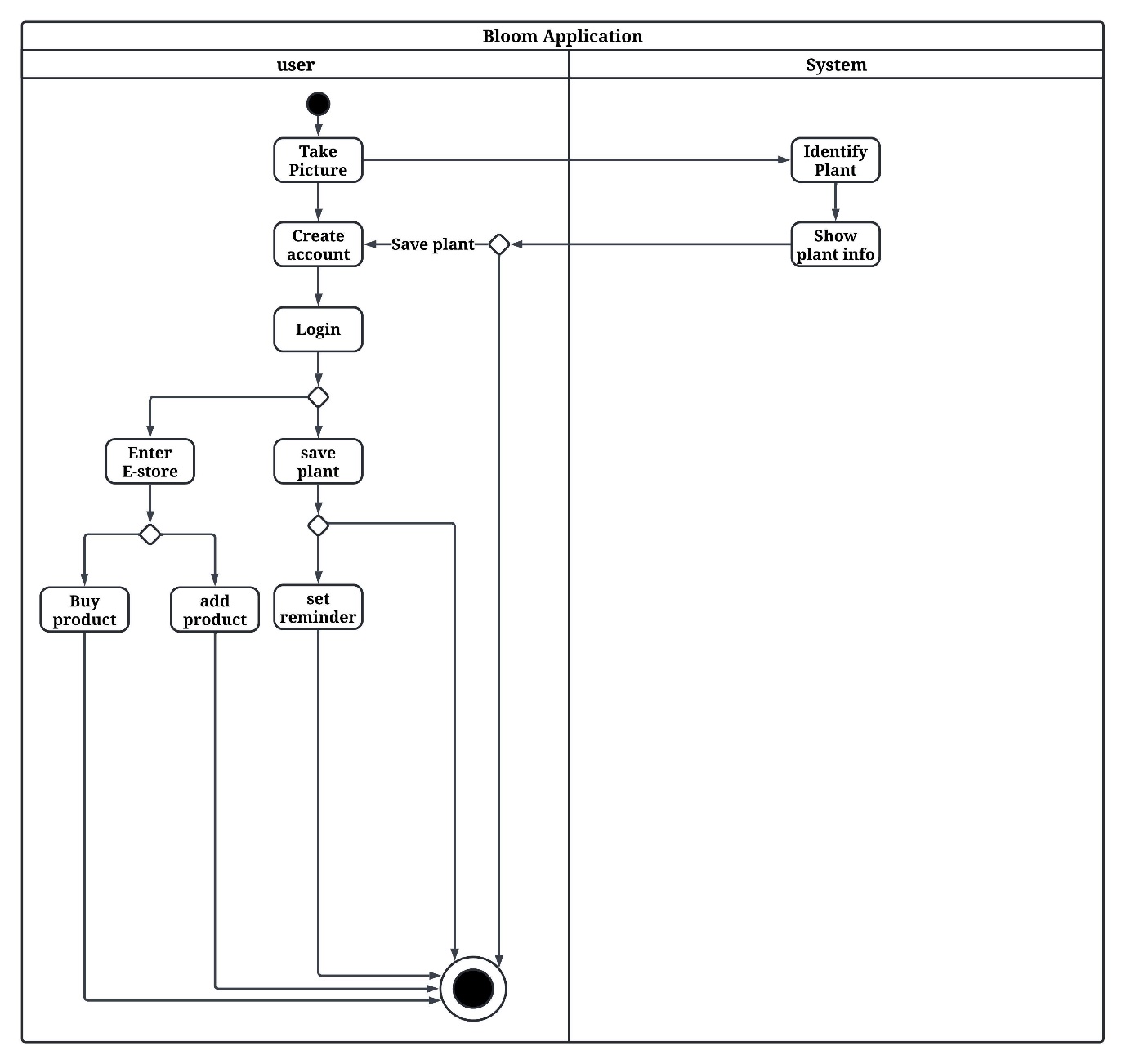
## 5.1. Class Diagram

A diagram of a company

Description automatically generated

**Figure 11** Class Diagram.

## 5.2. Activity Diagram



**Figure 12** Activity Diagram.

## 5.3 Conclusion

In this chapter, we modeled a class diagram and an activity diagram that were suitable for our project.

**Chapter 6: Implementation**

# Chapter 6 Introduction

In this chapter, we will give a thorough rundown of our project's implementation details. We will describe Engineering Standards, Programming Language and Tools used in the development process then, Code Snippets of main Function and write test cases. Due to gap between the current completed tasks of Sprint 1 and the remaining requirements mentioned in chapter 6 outline, we have encountered difficulty in completing critical activities such as sprint 1 interface, acceptance testing, integration testing, system testing, and usability testing. We opted to only doing select chapter 6 tasks that suit the work we have achieved thus far the best.

# 6.1. Engineering Standards

# 6.1.1. Interaction Capability

* **Learnability**

The degree to which the functions of a product or system can be learned to be used by specified users within a specified amount of time. We will make our app easy to use and learn.

# 6.1.2. Functional Suitability

* **Functional Completeness**

Degree to which the set of functions covers all the specified tasks and intended users' objectives. Our app will behave as we have specified in the requirements document.

* **Functional Appropriateness**

Degree to which the functions facilitate the accomplishment of specified tasks and objectives. Our app will finish the specified tasks when the user initiates them.

# 6.1.3. Maintainability

* **Reusability**

Degree to which a product can be used as an asset in more than one system, or in building other assets. Our app will be built from reused ML models that we will improve on, use the TensorFlow lite application to connect to the camera and use XML codes to build our interface, which can all be reused or expanded on if needed [3].

# 

# 

**Table 14** Engineering Standards factors and subfactors according to the iso25000 model.

# 6.2. Programming Language and Tools

#### 6.2.1. Python

The programming language we found to be most suitable is Python. It is an object-oriented, high-level programming language with dynamic semantics. Its simple syntax promotes readability, reduces maintenance costs, and supports modules and packages for modularity and code reuse.

Python offers powerful machine learning capabilities, simplifying data validation, processing, and analysis processes. Its simple syntax and extensive library ecosystem reduce routine tasks, allowing developers to focus on code.

Python offers flexibility, can run on various operating systems, is easy to read, has a large developer community, and is growing in popularity, making it easy to find solutions and resources for machine learning models [1].

#### 6.2.2. PIL (Python Imaging Library)

PIL is a powerful software tool crafted for image processing tasks within Python. It equips developers with a wide range of tools to handle different image file formats, manipulate images, and perform essential image processing operations. Developed by Fredrik Lundh and Contributors, PIL is indispensable for tasks such as image resizing, cropping, filtering, and basic enhancement. Despite its name, the Pillow library has superseded PIL, offering enhanced features and resolving core functionality issues**.**

#### 6.2.3. TensorFlow, Keres API and TensorFlow Lite

TensorFlow provides developers with multiple levels of abstraction, allowing you to select the one that best meets your needs. Create and train models using the high-level Keras API, which simplifies getting started with machine learning.

If you need more flexibility, eager execution enables quick iteration and intuitive debugging. For large ML training tasks, use the Distribution Strategy API to distribute training across different hardware configurations without changing the model definition.

TensorFlow Lite is a mobile library for deploying models on mobile, microcontrollers, and other edge devices [9].

#### 6.2.4. CNN (Convolutional Neural Networks)

A Convolutional Neural Network (CNN), also known as ConvNet, is a type of deep learning algorithm that is specifically designed for tasks that require object recognition, such as image classification, detection, and segmentation. CNNs are used in a variety of practical applications, including autonomous vehicles, security camera systems, and more [3].

#### 6.2.5. Google Collab

Collab is a hosted Jupyter Notebook service that requires no setup and gives you free access to computing resources like GPUs and TPUs. Collab is especially suitable for machine learning, data science, and education [7].

#### 6.2.6. Kaggle

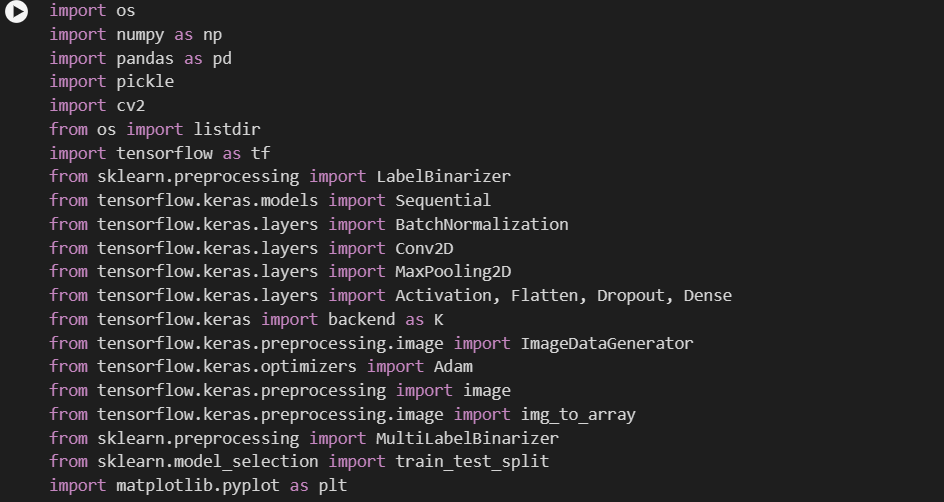
Kaggle is an online community for data scientists and machine learning enthusiasts. Kaggle enables users to collaborate with others, find and publish datasets, use GPU-integrated notebooks, and compete with other data scientists to solve data science problems [8].

# 6.3. Code Snippets of The Core Functionality

**6.3.1. Model**

**Before:**

Relevant libraries imports for model architecture and image processing.

****

**After:**

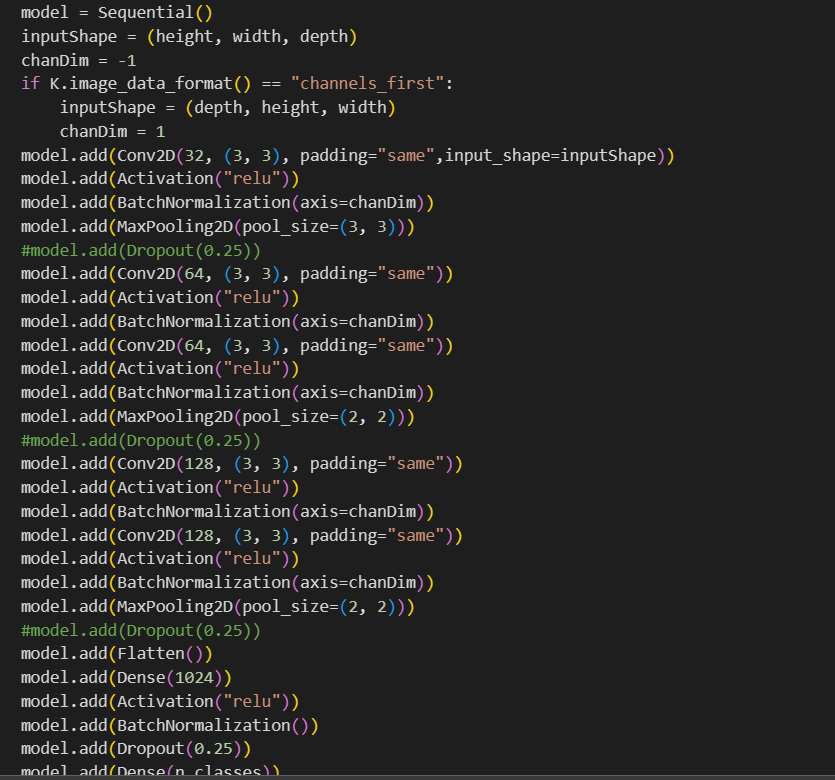
**A screenshot of a computer program

Description automatically generated**

**6.3.1.1. Model Architecture**

**Before:**

**A screen shot of a computer program

Description automatically generated**Used a sequential model and architecture.

**After:**

**A screen shot of a computer program

Description automatically generated**Used the VGG16 model architecture.

**6.3.1.2. Optimizer**

Changed the model name to indicate the difference.

**Before:**

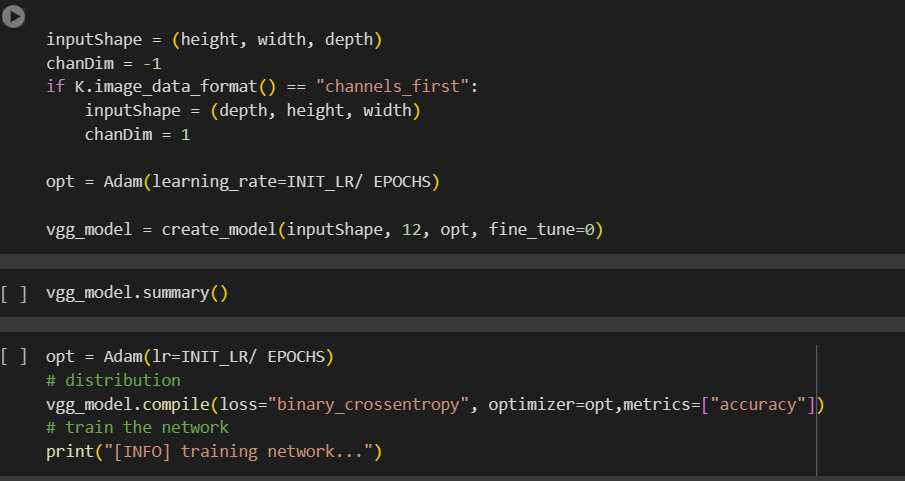
Used an old, outdated syntax that results in errors.

**A screen shot of a computer

Description automatically generated**

**After:**

Used a more recent syntax, and created and compiled the model using it.

****

**6.3.1.3. Training the model**

Changed the model name to indicate the difference.

**Before:**

**A computer code on a black background

Description automatically generated**

**After:**

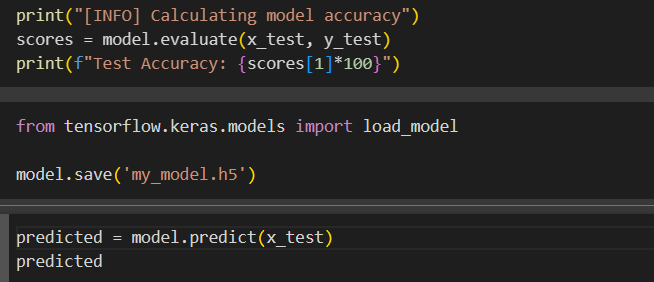
**A computer code on a black background

Description automatically generated**

**6.3.1.4. Predictions, Truths and Testing**

Changed the model name to indicate the difference.

**Before:**

****

**After:**

**A screen shot of a computer program

Description automatically generated**

**6.3.2. Image processing**

**Before:**

Used open CV image processing functions.

**A computer screen shot of a program code

Description automatically generated**

**After:**

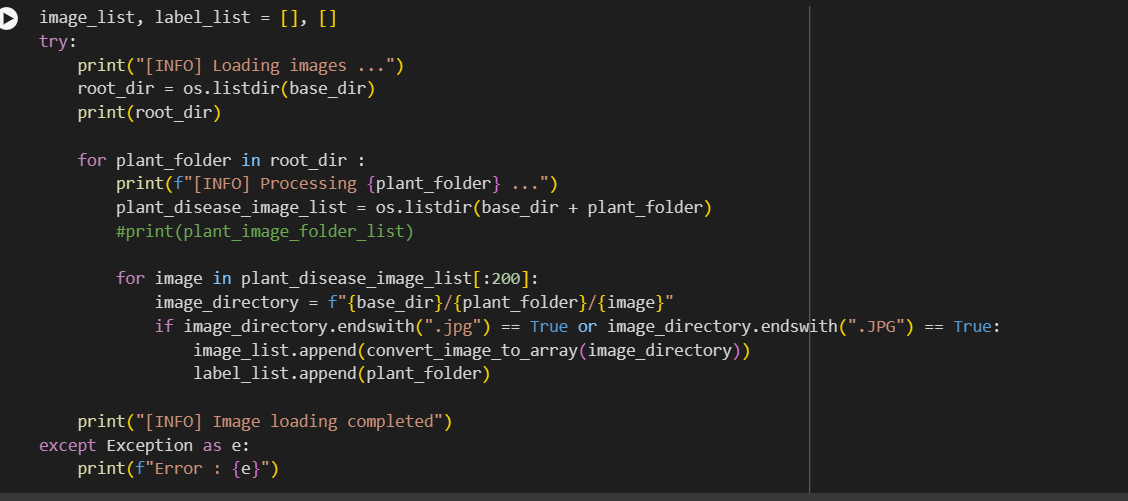
Used PIL functions to improve image processing.

**A screen shot of a computer program

Description automatically generated**

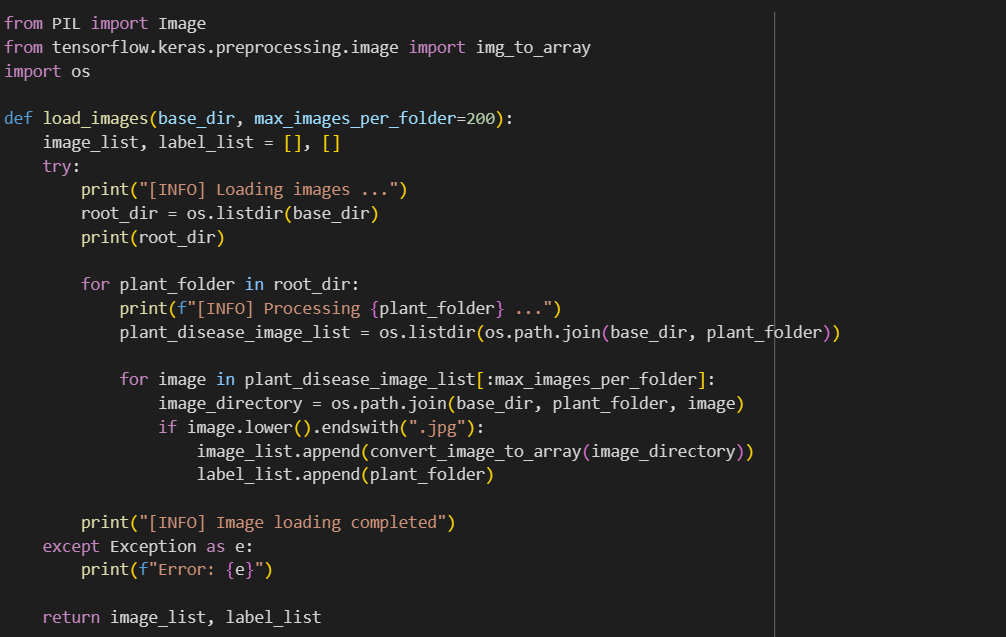
**Before:**

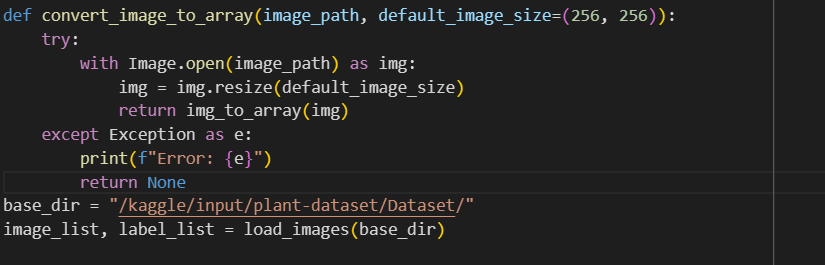
Used OpenCV image processing functions.

****

**After:**

Used PIL functions to improve image processing.

****

****

**6.3.3. Improving Accuracy**

**6.3.3.1. Model and Layers**

**Before:**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A white rectangular object with black text

Description automatically generated**

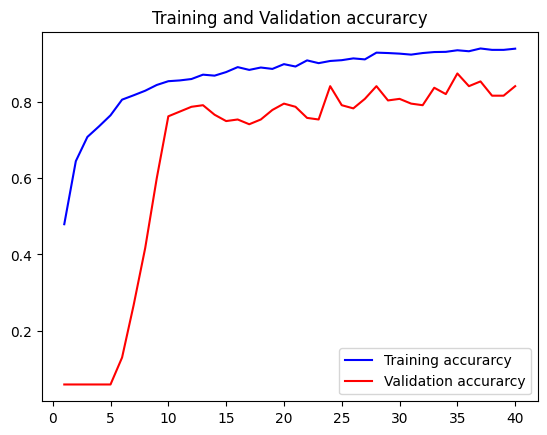
**A screenshot of a computer

Description automatically generatedAfter:**

**6.3.3.2. Training and Validation Accuracy and Loss**

**A graph of a graph showing a line and a line

Description automatically generated with medium confidenceBefore:**

****

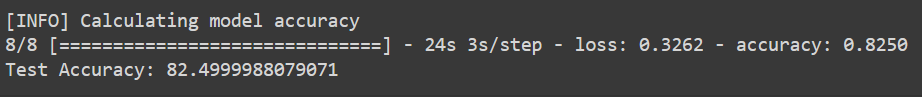
**A graph of a training and validation accuracy

Description automatically generatedA graph of training and validation

Description automatically generatedAfter:**

**Testing Accuracy and Heatmap of Predictions and Truths**

**Before:**

****

**A screenshot of a computer screen

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**After:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a graph

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

# 6.4. Unit Testing

# 6.4.1. Test Cases

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Case Description** | **Test Steps** | **Test Data** | **Expected Results** | **Actual Results** | **Pass/Fail** |
| Model testing | | | | | | |
| TC01 | Check if the model identifies the class of a plant in the dataset. | 1-Split the dataset into training and validation sets.  2-Create plant classes to label the images.  3- Label the images.  4-Train the model with the training set.  5-Test and validate the model with the testing dataset that was hidden from the model while training. | The testing dataset that is isolated from the model during training. | The predictions of the plants in the testing dataset matches the truths of the plant classes. | Majority of the predictions match the truths, with negligible mistakes. | Pass |
| TC02 | Check if the model identifies the classes of plants that are not in the training dataset. | 1-Change the testing dataset into an external dataset with images of plants the model didn’t train on.  2-Test the model with predictions and truths. | Plant image from a class not of the 12 classes the model is trained on. | The model doesn’t identify the unfamiliar plant class. | The model doesn’t identify the unfamiliar plant class and mistakes it for another plant. | Pass |
| TC03 | Check if the model identifies an outsourced test dataset with plant images from the 12 familiar classes. | 1-Change the testing dataset into an external dataset with images of plants the model is trained on.  2-Test the model with predictions and truths. | Plant image from a class from one of the 12 classes the model is trained on. | The model identifies the plants from the 12 familiar classes. | The model identifies 11 of the familiar plant class. The exception being soybean leaves. It needs certain angles and lighting and the absence of fruits, texts, or any objects other than the leaf from the image. | Pass mostly |

**Table 15** Test cases.

# 6.5. Conclusion

In this chapter, we described the engineering standards, programing language and the tools we used throughout sprint 1. For Sprint 1, we completed a core functionality, and focused solely on it. This includes thorough test cases and providing code snippets.

# Reference

1. Mata-Montero, E. and Carranza-Rojas, J. (2016) ‘Automated Plant Species Identification: Challenges and opportunities’, *ICT for Promoting Human Development and Protecting the Environment*, pp. 26–36. doi:10.1007/978-3-319-44447-5\_3.
2. App ref.
3. Corbo, A. (2023, January 3). How is Python used in machine learning? Built In. [https://builtin.com/machine-learning/python-machine-learni](https://builtin.com/machine-learning/python-machine-learning)
4. *Image classification*. (n.d.). TensorFlow. <https://www.tensorflow.org/lite/examples/image_classification/overview>
5. ISO 25010. (n.d.). https://iso25000.com/index.php/en/iso-25000-standards/iso-25010
6. Keita, Z. (2023, November 14). *An Introduction to Convolutional Neural Networks (CNNs)*. <https://www.datacamp.com/tutorial/introduction-to-convolutional-neural-networks-cn>
7. PIL, [Pillow (PIL Fork) 10.3.0 documentation](https://pillow.readthedocs.io/en/stable/)
8. Team, K. (n.d.). *Keras documentation: About Keras 3*. https://keras.io/about/
9. Google Colab. (n.d.). *Colab.google*. colab.google. https://colab.google/
10. Uslu, Ç. (2022, March 16). *What is Kaggle?* <https://www.datacamp.com/blog/what-is-kaggle>[ng](https://builtin.com/machine-learning/python-machine-learning)
11. *Why TensorFlow*. (n.d.). TensorFlow. <https://www.tensorflow.org/about>