AI-BASED MEDICINAL PLANT DETECTION

Minor project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

In

Computer Science and Engineering

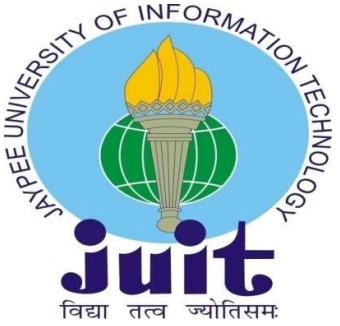
By

Chandan Kumar (211301)

Awanish Thakur (211114)

UNDER THE SUPERVISION OF

Mr. Faisal Firdous



Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat, 173234, Himachal Pradesh, INDIA

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 17- April - 2024

Type of Document (Tick): | B.Tech./B.Sc./BBA/Other

Name: Chandan kumar, Awanish Thakur **Department:** CSE **Enrolment No** 211301, 211114

Contact No. 7982268819, 8450823139 **E-mail.** 211301@juitsolan.in, 211114@juitsolan.in

Name of the Supervisor: Mr. faisal firdous

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters):

AI-BASED MEDICINAL PLANT DETECTION

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

- Total No. of Pages = 44
- Total No. of Preliminary pages = 4
- Total No. of pages accommodate bibliography/references = 2

Chandan kumar

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at 8 (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)

Faisal 18-5-24

TABLE OF CONTENT

CERTIFICATE

We hereby certify that the work being presented in the project report titled "AI-Based Medicinal Plant Recognition" in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering and submitted to the Department of Computer Science And Engineering, Jaypee University of Information Technology, Waknaghat is an authentic record of my work carried out during the period from January 2024 to May 2024 under the supervision of Mr. Faisal Firdous, Department of Computer Science and Engineering, Jaypee University of Information Technology, Waknaghat.

The matter presented in this project report has not been submitted for the award of any other degree at this or any other university.

Chandan Kumar, Awanish Thakur
211301, 211114

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Faisal Firdous
17-5-24
Assistant Professor
Computer Science & Engineering and Information Technology
Jaypee University of Information Technology, Waknaghat,

AKCNOWLEDGEMENT

Firstly, we express our heartiest thanks and gratefulness to almighty God for His divine blessing making it possible to complete the project work successfully.

We are grateful and wish my profound indebtedness to Supervisor **Mr. Faisal Firdous, Designation**, Department of CSE Jaypee University of Information Technology, Waknaghat. Deep Knowledge & keen interest of my supervisor in the field of "**Research Area**" to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

We would like to express my heartiest gratitude to **Mr. Faisal Firdous,** Department of CSE, for his kind help in finishing my project.

We would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, we must acknowledge with due respect the constant support and patience of my parents.

Chandan Kumar, Awanish Thakur

211301, 211114

ABSTRACT

The comprehensive project titled "AI-Based Medicinal Plant Recognition" is a significant endeavor that harnesses the power of machine learning and image processing to recognize and classify different medicinal plant species. The project, built using a combination of C# and Python, two powerful and versatile programming languages, demonstrates a successful integration of various technologies and techniques to solve a complex problem. The project utilizes several libraries to perform various tasks. Emgu CV, a cross-platform .Net wrapper to the OpenCV image processing library, is used for image processing tasks. The tensorFlow library is used for building and training the machine learning models. The project's workflow begins with the collection of plant images, which are then pre-processed using various techniques such as data augmentation and background removal. These preprocessing steps are crucial for improving the performance of the machine learning models by increasing the diversity of the training data and reducing unnecessary information in the images. The pre-processed images are then fed into a machine-learning model for training. Pre-trained models such as InceptionV3, Inception ResNet V2, and VGG16 are being used in this project. The trained model can then be used to classify new plant images, providing the plant's species.

The project also includes a database component, where the plant images and possibly the model's predictions are stored and managed. This PostgreSQL database serves as a model for efficient storage and retrieval of plant images and model predictions in similar projects, facilitating data management in large-scale plant recognition studies. The project has achieved promising results, with the machine learning models demonstrating high accuracy in plant recognition tasks. The results indicate the potential of the system in various fields such as botany, agriculture, and environmental science. However, the project acknowledges several limitations. These include hardware constraints, dependency on the quality of input images, potential overfitting due to limited dataset size, and the inherent lack of explainability in deep learning models. The report also highlights the time-consuming nature of the model training process and the project's reliance on external libraries. We also outline several avenues for future work. These include the development of a custom machine learning model, improving the quality and diversity of the dataset, and the creation of a user-friendly web application. The report also suggests exploring real-time recognition, integration with Geographic Information System (GIS) platforms, improving model explainability, and ensuring cross-platform compatibility.

Chapter 01: INTRODUCTION

1.1 Introduction

The rapid evolution of technology has ushered in new possibilities across various sectors, including botany. Traditional plant identification methods, which predominantly rely on physical characteristics and expert knowledge, can be labour-intensive and prone to errors. This has stimulated a growing interest in utilizing technology to automate the process of plant recognition. Responding to this burgeoning need, the "AI-Based Medicinal Plant Recognition" project aspires to develop an automated system that can accurately recognize and classify different medicinal plant species from images.

The application of image processing technology has the potential to revolutionize the field of botany. It could assist researchers in cataloguing plant species, a task that traditionally requires extensive knowledge and experience. According to the Royal Botanic Gardens, Kew, there are an estimated 390,000 species of vascular plants worldwide. An automated system could significantly expedite the cataloguing process and make it more accessible.

In the realm of medicinal plants, particularly focusing on Indian Medicinal Plants, this technology could be transformative. India, often referred to as the "Botanical Garden of the World", is home to an estimated 45,000 plant species, including about 7,000 medicinal plants, as per the Botanical Survey of India. These medicinal plants have immense potential in the healthcare sector. However, their identification and preservation remain a challenge due to the vast diversity and lack of accessible knowledge. Our project aims to aid in the recognition of these plants, thereby facilitating their use and preservation.

The World Health Organization estimates that 80% of the population in some Asian and African countries depends on traditional medicine for primary health care. Many of these treatments rely on medicinal plants, highlighting the importance of accurate plant identification. By automating this process, our project could make these treatments more accessible and safer, reducing the risk of using the wrong plant species.

Despite the promising potential of the project, it is not without limitations. The quality of input images is a significant factor, as poor-quality images can lead to inaccurate predictions. The project also faces the risk of overfitting due to the limited size of the dataset. Overfitting is a very common issue in machine learning where a model performs good on the training data and fails to generalize on unseen data.

Hardware constraints also pose a challenge, as the training of machine learning models requires substantial computational resources. Without high-performance hardware, the training process can be time-consuming and inefficient.

Another limitation is the inherent lack of explainability in deep learning models. While these models can make accurate predictions, understanding the reasoning behind these predictions is complex. This lack of transparency can be a barrier to the adoption of the project in fields where interpretability is crucial.

1.2 Objective

The objective of the "AI-Based Medicinal Plant Recognition" project is to develop an automated, accurate, and efficient system that leverages advanced image processing and machine learning techniques to recognize and classify Indian medicinal plant species from images. This system aims to facilitate the identification and preservation of these valuable plant species, thereby contributing to the healthcare sector and biodiversity conservation efforts. Furthermore, the project seeks to make the process of medicinal plant identification more accessible, reducing the reliance on extensive botanical knowledge and experience.

1.3 Motivation

The motivation behind the " AI-based medicinal plant recognition " project stems from the increasing need for efficient and accurate recognition systems specifically for medicinal plants. Medicinal plants play a crucial role in healthcare, with an estimated 70-80% of the world's population relying on traditional medicine, primarily plant-based, for primary healthcare. However, the identification and classification of medicinal plants can be a challenging task due to the vast diversity of plant species and their morphological similarities.

Traditional methods of medicinal plant identification often rely on expert knowledge and can be time-consuming, making them impractical for large-scale applications. Furthermore, these methods can be prone to errors due to the subjective nature of visual identification and the variability in plant appearance due to factors such as growth stage and environmental conditions.

The advent of machine learning and image processing technologies presents an opportunity to address these challenges. These technologies can automate the process of medicinal plant recognition, making it faster and more accurate. They can also handle

large amounts of data, making them suitable for large-scale applications.

In the field of healthcare and botany, an automated medicinal plant recognition system could have significant benefits. For instance, it could enable early and accurate identification of medicinal plants, assist in cataloging medicinal plant species, and aid in biodiversity monitoring. However, developing such a system presents several challenges, including the need for large amounts of annotated medicinal plant images for training the machine learning models and the need to handle variations in plant appearance.

The " AI-based medicinal plant recognition project was motivated by the potential of these technologies to revolutionize medicinal plant recognition and the desire to overcome the associated challenges. The project aims to develop an automated system that can recognize and classify different medicinal plant species based on images, providing a valuable tool for various fields**.**

1.4 Language Used

The project is built using a combination of C# and Python programming languages, leveraging their respective strengths. It utilizes several libraries to perform various tasks, including Emgu CV for image processing, Tensorflow for machine learning, and Npgsql for database connectivity. The project employs several images processing techniques, including data augmentation and background removal, to preprocess the plant images. The images are then used to train a machine-learning model.

1.5 Technical Requirements

1. **Programming Languages**: The project is developed using C# and Python. C# is used for its robustness and extensive support for object-oriented programming, making it ideal for building the project's structure and handling database operations. Python, renowned for its simplicity and extensive support for scientific computing and machine learning, is used for implementing machine learning models and image processing tasks.
2. **Machine Learning Library**: Tensorflow and Timm, the open-source machine learning library, is used for building and training machine learning models. It provides a comprehensive ecosystem of tools, libraries, and community resources that allows researchers to push the state-of-the-art in machine learning.
3. **Image Processing Library**: Emgu CV, a cross-platform .Net wrapper to the

OpenCV image processing library, is used for image processing tasks. It provides extensive functionalities for processing images and is widely used in real-time image processing.

4. **Database Connectivity**: Npgsql, a .NET data provider for PostgreSQL, is used for database connectivity. It allows the project to interact with the PostgreSQL database, enabling efficient storage and retrieval of plant images and potentially the model's predictions.
5. **Python Runtime for .NET**: Python.Runtime, a part of the Python for .NET package, is used to run Python scripts from the C# code. It provides a powerful application scripting tool and allows C# to use Python libraries.
6. **Operating System**: The project is developed and tested on a Windows operating system. However, with minor modifications, it can be made compatible with other operating systems like Linux or MacOS.
7. **Hardware Requirements**: The project requires a computer with a minimum of 8GB RAM for smooth operation. For training the machine learning models, a GPU is recommended for faster computation, although it's not mandatory as the models can also be trained on a CPU.
8. **Software Requirements**: The project requires a Python environment with Tensorflow installed for running the machine learning scripts. It also requires a PostgreSQL database for storing and managing the plant images and potentially the model's predictions.

1.6 Deliverables/Outcomes

1. **Automated Medicinal Plant Recognition System**: The primary deliverable of this project is a system that can automatically recognize and classify different medicinal plant species based on images. This system will leverage machine learning and image processing techniques to provide accurate and efficient plant recognition.
2. **Preprocessed Image Dataset**: The project will produce a dataset of preprocessed medicinal plant images. These images will undergo various preprocessing steps such as data augmentation and background removal to improve the performance of the machine learning models.
3. **Trained Machine Learning Model**: The project will deliver a machine learning

model trained on the preprocessed image dataset. This model will be capable of classifying different medicinal plant species based on images.

4. **Database of Plant Images and Model Predictions**: The project will set up a PostgreSQL database for efficient storage and retrieval of plant images and potentially the model's predictions. This database will be managed using the Npgsql library in C#.
5. **Project Report**: A comprehensive report detailing the project's methodology, results, and implications will be produced. This report will serve as a record of the project's findings and a guide for future research in this area.
6. **Source Code**: All source code developed for this project will be provided. This includes C# code for the project structure and database operations and Python code for the machine learning models and image processing tasks.

Chapter 02: Feasibility Study, Requirements Analysis and Design

In this chapter, we first conducted a feasibility study for our AI-based medicinal plant recognition system. We defined the problem as the need for accurate identification of medicinal plants from images, a task that is complex due to the high variability in plant appearance and the similarity between many medicinal and non-medicinal plants.

Our proposed solution is an AI system that uses image preprocessing techniques and machine learning models to identify medicinal plants. We found this solution to be technically feasible given the availability of necessary technologies like C#, Python, Tensorflow, and Emgu CV. The economic benefits of automating medicinal plant identification support its economic feasibility. The system can be integrated into existing workflows, making it operationally feasible. The project does not violate any data protection or social media laws, confirming its legal feasibility. Finally, the project is scheduled to be completed within a reasonable timeframe, supporting its schedule feasibility.

Next, we performed a requirements analysis to clearly define what our system should do. This involved identifying the functional and non-functional requirements of the system.

Finally, we designed the system based on the requirements. This involved deciding on the system architecture, the technologies to be used, and the design of the user interface. The design phase ensured that the system would meet the identified requirements and be user-friendly.

This chapter laid the groundwork for the development of our AI-based medicinal plant recognition system by ensuring its feasibility, clearly defining its requirements and designing the system to meet these requirements.

2.1 Feasibility Study

2.1.1 Problem Definition

The problem our project addresses is the identification of medicinal plants based on images. This is a significant issue in the field of herbal medicine and botany, where accurate identification of medicinal plants is crucial. Due to weather changes, the leaves change

throughout the year, which makes it complicated. Medicinal plant identification cannot work without professionals with rich knowledge and experience. Misidentification can lead to the use of incorrect or potentially harmful plants when we have thousands of herbs with many different properties. Manual identification can be time-consuming and requires expert knowledge, which may not always be readily available.

2.1.2 Problem Analysis

The problem of medicinal plant identification involves classifying images of plants into their respective species. This is a complex task due to the high variability in plant appearance due to factors such as age, lighting conditions, and viewpoint. Additionally, many medicinal plants can look similar to non-medicinal plants, adding to the complexity of the task. Traditional image processing techniques may not be sufficient to handle this complexity.

2.1.3 Solution

Our solution is to develop an AI-based system that can automatically identify medicinal plant species from images. The system uses a combination of image preprocessing techniques and machine learning models to achieve this. The image preprocessing involves data augmentation and background removal to prepare the images for the machine learning model. The machine learning model is trained using Python scripts, and the trained model is then used to make predictions on new images. The feasibility of this solution is supported by the availability of the necessary technology (C#, Python, Tensorflow, Emgu CV), the economic benefits of automating medicinal plant identification, and the operational feasibility of integrating this system into existing workflows. The project is also legally feasible as it does not violate any data protection or social media laws, and it is scheduled to be completed within a reasonable timeframe.

Table 2.1: Literature Review Table

S.No.	Year and Author Name	Dataset	Method	Accuracy	Limitation
1	Biplob Dey Jannatul	5878 images of 30	VGG16, VGG19, DenseNet201, InceptionResNetV	P1=99.64% In PF1=97%	Environmental and natural impact is not recorded

	Ferdous [2024]	medicinal species	2,and InceptionV3		
2	*Parismita Sharma, ∗Parvez Aziz Boruaha [2023]	77500 images of 117 species	Unet(gray scale)	Train Accuracy 96.33% Value accuracy 95.49%	Cover only a small area or local plant
3	*Pushpa B R ∗N. Shobha Rani [2023]	The datasets consist of 5900 images of 40 plant species and 6900 leaf images of 80 species	Data augmentation is applied to medicinal plant datasets.		Consists data of from a botanical garden in Kerala.
4	*S. Kavitha *T. Satish Kumar	500 images of each 6 plant species.	DL model	98.3%	This sample is just for six medicinal plants.
5	*Xin Sun * Huinan Qian [2022]	It contains 95 herb categories with 5640 images	The DNN method is used	Top 1=68.89% Top 5=88.03%	Need Professionals to identify very high resources
6	Nilesh S. Bhelkar Dr. Avinash	In this work, 45 distinct	Deep learning (DL) model Deep	97.65%	The data is only limited to Indian herbal plants

	Sharma	medicinal plant leaves were used	convolutional neural networks		
7	Rahim Azadnia Mohammed Maitham Al-Amidi	5 Different plant species	Convoluti onal Neural Network (CNN)	99.3%	The leaves change throughout the year due to changes in weather conditions and in different places same species look different
8	Samreen Naeem , Aqib Ali Christophe Chesneau	6 plant species taken from the agriculture department.	multi-layer perceptron (MLP), LogitBoost (LB), Bagging (B) with REPTree, Random Forest (RF), and Simple Logistic (SL)	99.01%	This study is limited to six medicinal plant leaves while there are millions of types of medicinal plants/herbs in the world
9	Rahim Azadnia Kamran Kheiralipour	4 plants from 40 locations.	Artificial neural network	100%	in different places plant of the same species look different Here it takes the sample of IRAN
10	Jianqing Wang , Weitao Mo ,	14196 images of 182 plant category	CA and SA module network (CCSMNet) for the CHS image recognition	99.27%	The data is only limited for Chinese herbal plants

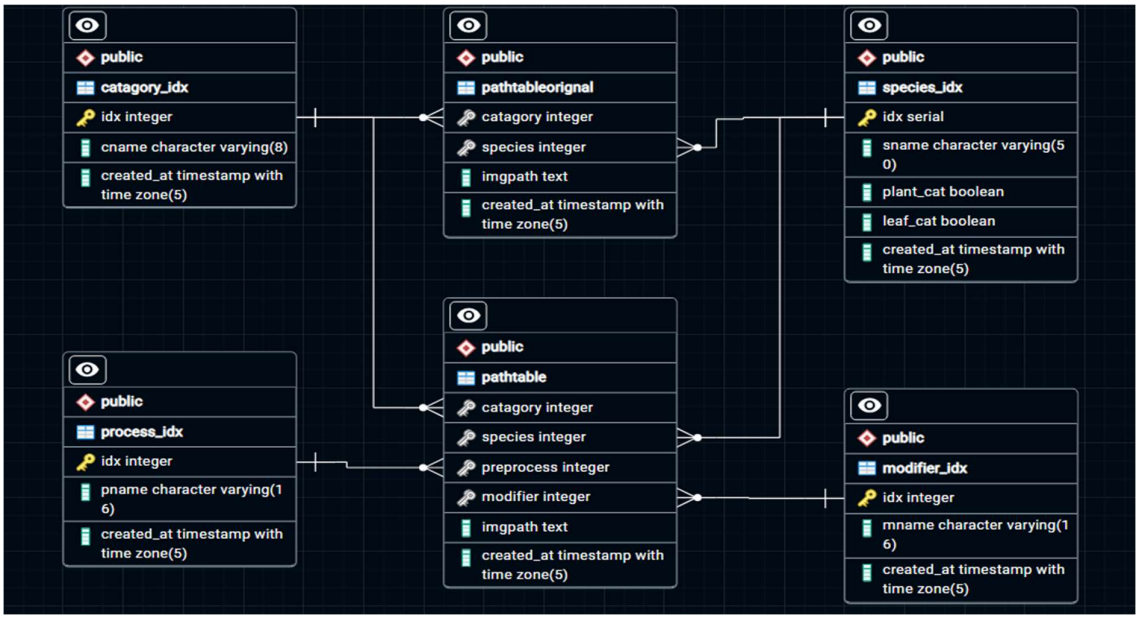
2.2 E-R Diagram / Data-Flow Diagram (DFD)

2.2.1 E-R Diagram

In the context of our project, the main entities are the Original Image Path table, Pre-processed Image Path table, Category_idx, process_idx, species_idx, modifier_idx

- Original Image path table (pathtableorignal) holds the location of the original image. This entity has the following attributes.
    - Category: This attribute references to the category_idx and holds the information about whether the image is of a plant or leaf.
    - Species: This attribute reference to the species_idx and hold the information of the of plant image.
    - Path: This attribute holds the actual path of the location where the image is being saved.
- Preprocessed Image Path Table (pathtable) holds the location of the preprocessed image.
    - Category: This attribute references to the category_idx and holds the information about whether the image is of a plant or leaf.
    - Species: This attribute reference to the species_idx and hold the information of the of plant image.
    - Preprocessed: This attribute references to the process_idx and holds information on the pre-process action performed on the image.
    - Modifier: This attribute reference to the modifier_idx and holds information of the modifier that is being performed during preprocess.
    - Path: This attribute holds the actual path of the location where the image is being saved.
- Category_idx holds the index of the category.
    - Idx: This attribute holds the unique index for all the categories of image.
    - Cname : This attribute holds the category name that is part of the image information.
- Process_idx holds the index of the process.
    - Idx: This attribute holds the unique index for all the processes performed on the image.
    - Pname : This attribute holds the process name that is part of the image information.

- Species_idx holds the index of the species.
    - Idx: This attribute holds the unique index for all the species of image.



    - Sname : This attribute holds the species name that is part of the image information.
- Modifier_idx holds the index of the modifier.
    - Idx: This attribute holds the unique index for all the modifiers implemented on an image during preprocessing.
    - Mname : This attribute holds the modifier name that is part of the image information.

Figure 2.1: ER Diagram of the database used in the project

2.2.1 Data-Flow Diagram

In the context of our project, the Data-Flow Diagram looks like:

- User uploads Image to the system.
- System sends Image to image preprocessing module and stores image information in the database.
- In the preprocessing module, the Image gets preprocessed and passed to the prediction module for prediction.
- Prediction uses a trained model to predict plant names from preprocessed images.
- After prediction data is stored in the database.

- System sends prediction results back to the user.

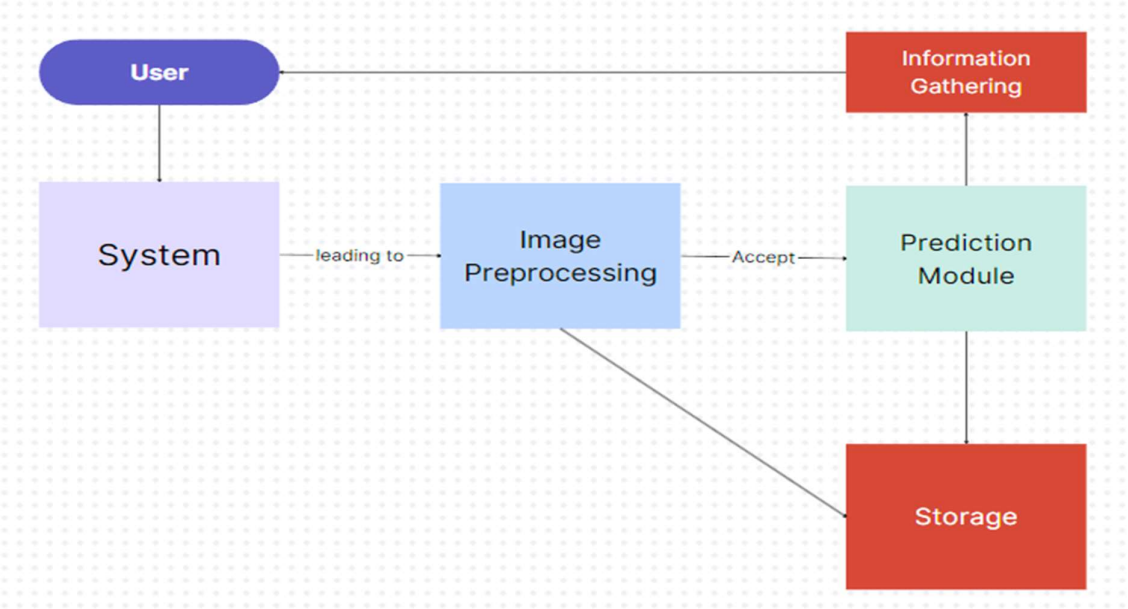


Figure 2.2: Data-Flow Diagram

Chapter 03: IMPLEMENTATION

3.1 Date Set Used in the Project

The data set used for this project was sourced from Kaggle, a popular online platform for data science and machine learning. The specific data set is the "Indian Medicinal Leaves Dataset" (source: **Kaggle**).

This data set was chosen because it provides a comprehensive collection of images of various Indian medicinal plants. It includes images of leaves from different angles and in different lighting conditions, which adds to the variability and complexity of the data, making it ideal for training a robust machine-learning model.

The data set is well-organized and labeled, which facilitated the data preprocessing and model training stages of the project. The labels provided with the data set were used as the ground truth during model training and evaluation.

In summary, the "Indian Medicinal Leaves Dataset" provided a rich, varied, and well-structured data source for our AI-based medicinal plant recognition system.

3.2 Data Set Features

3.2.1 Types of Data Set

The data set used in this project is an image data set. It consists of digital images of various Indian medicinal plants. Each image represents a different instance of a plant leaf, and each instance belongs to a specific class, which is the species of the medicinal plant.

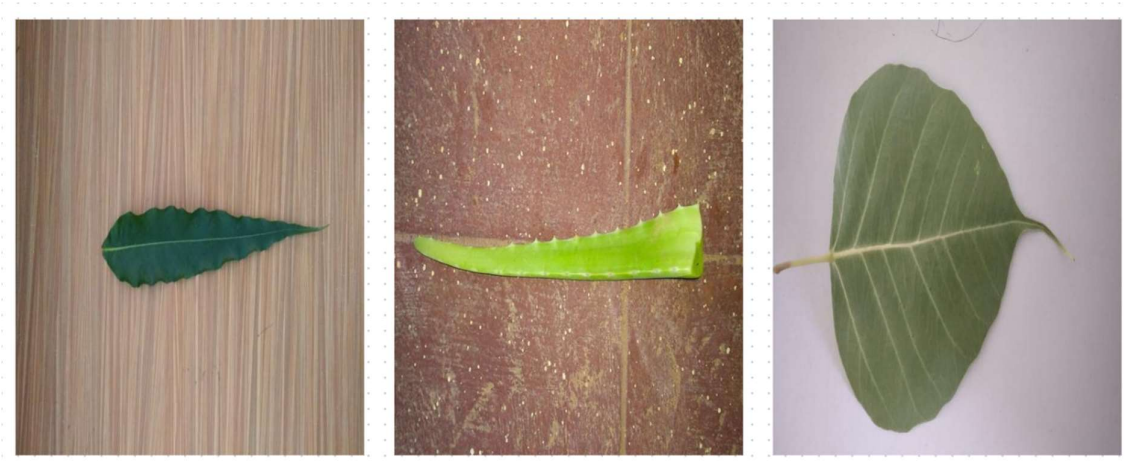


Figure 3.1: Original Image from the dataset

For this project, we select five species total of 445 images divided in an 8:2 ratio to train and test the model.

```
Train data count: 353
Test data count: 92
Species dict: {1: 'Aloevera', 2: 'Amla', 3: 'Amruthaballi', 4: 'Arali', 5: 'Ashoka'}
```

Figure 3.2: Data Count using the program with species name

Table 3.1: Data distribution table in test, train and total

Data distribution table

	Train	Test	Total
Aloevera	93	24	117
Amruthaballi	72	19	91
Arali	71	18	89
Ashoka	64	17	81
Amla	53	14	67

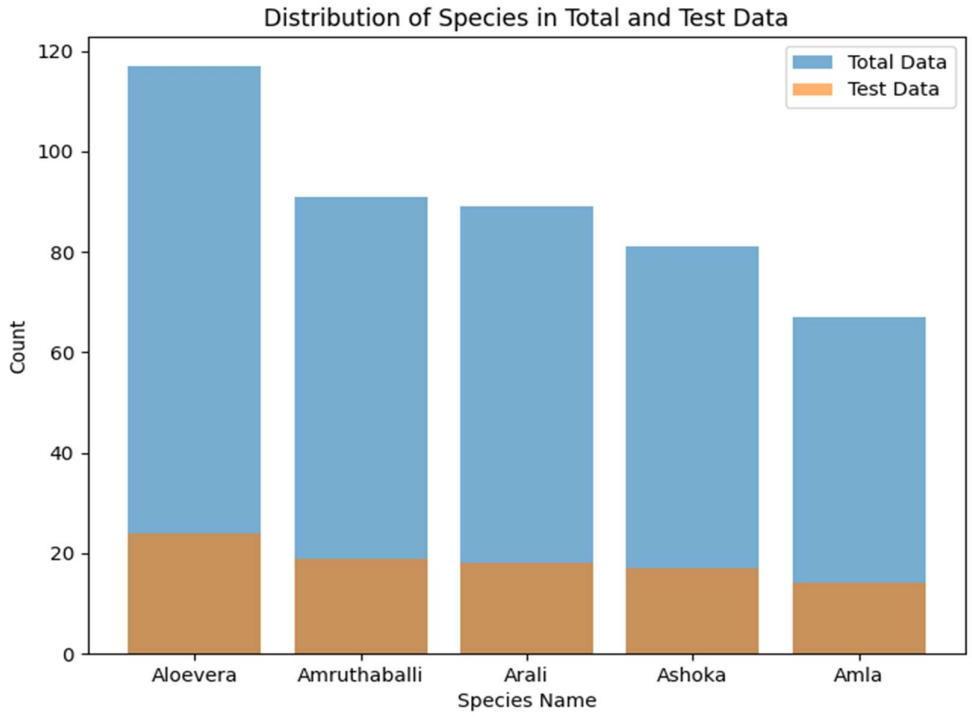


Figure 3.3: Visualization of data in total and test.

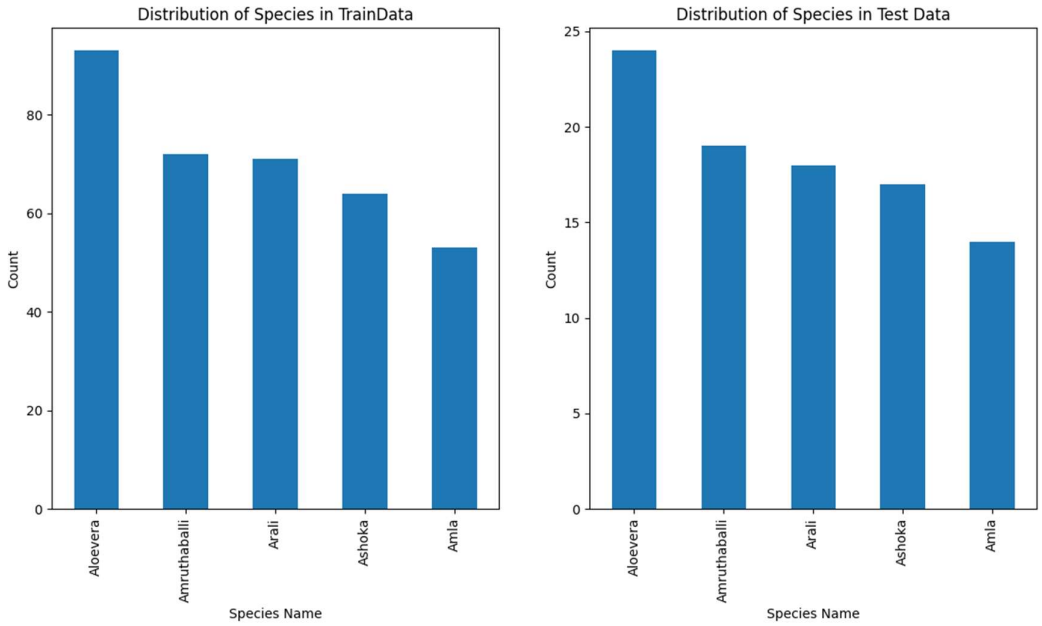


Figure 3.4: Independent Visualization of Data in test and train

3.2.2 Number of Attributes, Fields, and Description of the Data Set

The primary attribute of this data set is the image itself. Each image is a multi-dimensional array of pixel intensities in the red, green, and blue (RGB) color channels. The data set is labeled, meaning that each image is associated with a class label that identifies the species of the medicinal plant in the image. This label is used as the ground truth when training the machine learning model.

The data set contains images of several different species of medicinal plants. Each species is represented by multiple images, providing a variety of examples for the model to learn from.

In terms of fields, the data set primarily consists of two fields: the image data and the corresponding class label. The image data is the primary feature used for model training, while the class label is the target output.

The images in the data set vary in size and are in color (RGB), providing rich information for the model to learn the distinguishing features of different medicinal plant species. The variability in the images, in terms of lighting conditions, angles, and plant conditions, adds to the complexity and robustness of the data set.

In addition to the raw image data and class labels, the data set was further processed and stored in a database for efficient access and management. The DBMain class in the code handles the parsing of the data from the original directory and feeds it into the database. This allows for efficient retrieval and manipulation of the data during the preprocessing, model training, and prediction stages of the project.

3.3 Design of Problem Statement

The problem statement for this project is the identification of medicinal plants from images. The design of the solution to this problem involves several stages, each handled by different components of the system.

1.Data Collection and Database Feeding: The first stage involves collecting images of medicinal plants and feeding them into a database for efficient access and management. This is handled by the DBMain class in the code.

2.Image Preprocessing: The next stage is preprocessing the images to prepare them for the machine learning model. This involves data augmentation and background

removal, which are handled by the DataAugmentation, BckRemove, and Augmentation_BckRem classes.

3.Model Training: Once the images are preprocessed, they are used to train a machine-learning model. This involves feeding the preprocessed images and their corresponding labels into a machine-learning algorithm, which learns to associate the images with the labels. This is handled by the PythonMain class and its PyMain method.

4.Prediction: After the model is trained, it can be used to identify the medicinal plant in a new image. This involves feeding the new image into the trained model, which outputs a prediction of the plant species. This is handled by the Prediction class and its PredictMain method.

5.Result Display: Finally, the prediction results are displayed to the user. This involves retrieving the prediction from the model and presenting it in a user-friendly format.

3.4 Algorithm / Pseudo code of the Project Problem

High-level pseudocode representation of our project:

Algorithm for Medicinal Plant Recognition System:

1. Start
2. Initialize the database using DBMain class
3. For each image in the dataset:
   3.1. Load the image
   3.2. Preprocess the image using DataAugmentation and BckRemove classes
   3.3. Store the preprocessed image in the database
4. Train the machine learning model using the preprocessed images and their labels:
   4.1. Load the preprocessed images and their labels from the database
   4.2. Feed the images and labels into the machine-learning algorithm
   4.3. Train the model using PythonMain.PyMain method

4.4. Store the trained model

5. For each new image to be identified:

5.1. Load the new image

5.2. Preprocess the new image in the same way as the training images

5.3. Feed the preprocessed image into the trained model

5.4. Get the prediction from the model using Prediction.PredictMain method

5.5. Display the prediction to the user

6. End

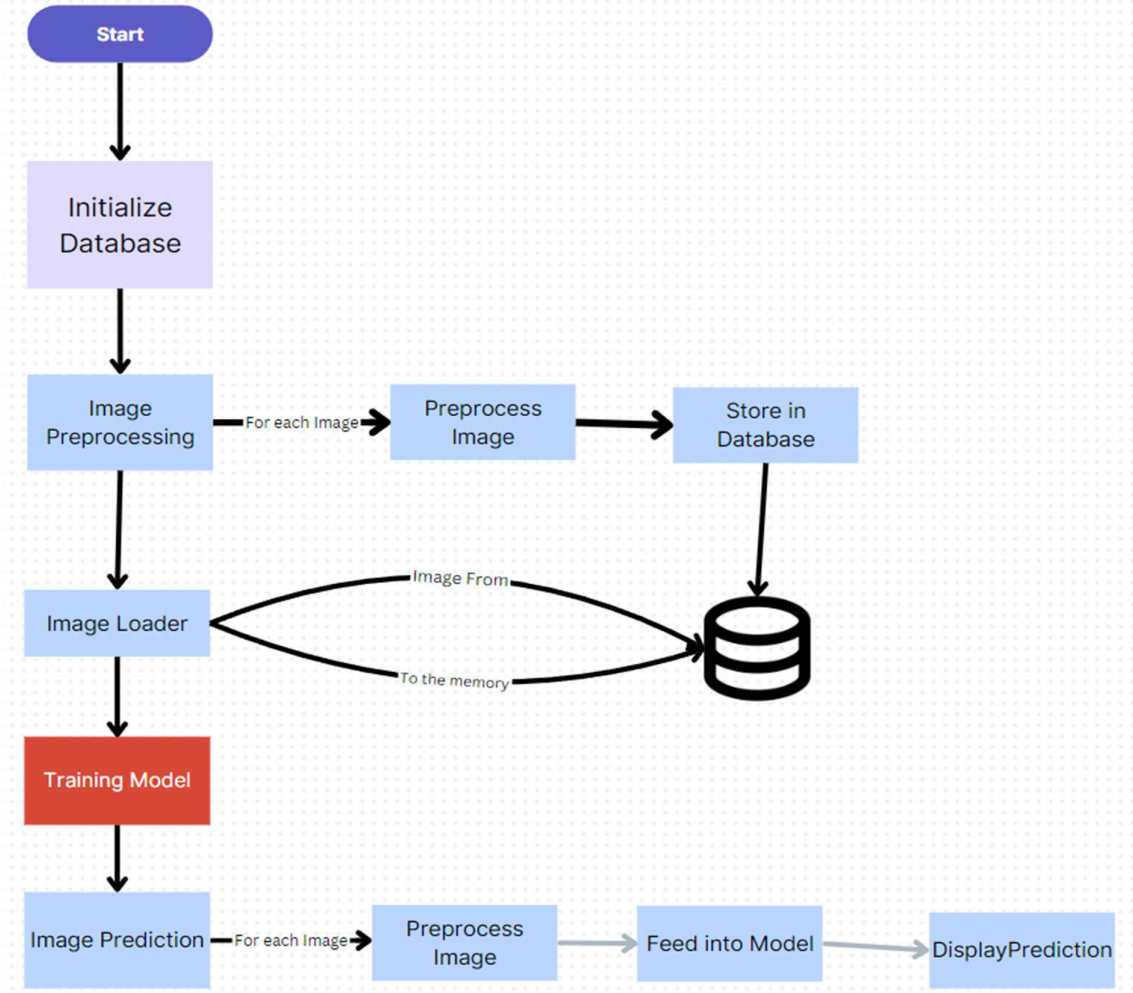


Figure 3.5: Program flow chart

3.5 Stages of the Project

Our project consists of various stages including initialization, data processing, and model training to predict results and display.

```
namespace Minor_Project_Ai_Plant_Recognition
{
    /// <summary>
    /// The Program class is the entry point of the application.
    /// </summary>
    0 references
    internal class Program
    {
        Tabnine | 0 references
        public static void Main(string[] args)
        {
            WriteLine("Starting Project");

            _ = new Final();
            Final.FinalMain();

            WriteLine("thanks for you corporation...");
        }
    }
}
```

Figure 3.6: Starting point of the program

1. **Initialization**: The initialization of the database is a crucial step. This is typically done using a method in our DBMain class.

   We are connecting to our database using credentials in our GetConnection() method. This method ensures a secure connection and gives output accordingly.

```
private NpgsqlConnection GetConnection()
{
    string connString = $"Host={Host};Username={User};Password=
      {Password};Database={DBname};Port={Port};Include Error Detail = True";
    var connection = new NpgsqlConnection(connString);

    try
    {
        connection.Open();
        WriteLine("Successfully connected to the database.");
    }
    catch (Exception ex)
    {
        WriteLine("Failed to connect to the database.");
        WriteLine("Error: " + ex.Message);
    }

    return connection;
}
```

Figure 3.7: Code to connect to the Database

We are performing CRUD operations in our database to create, read, update, and delete data from our database to parse data for further processing.

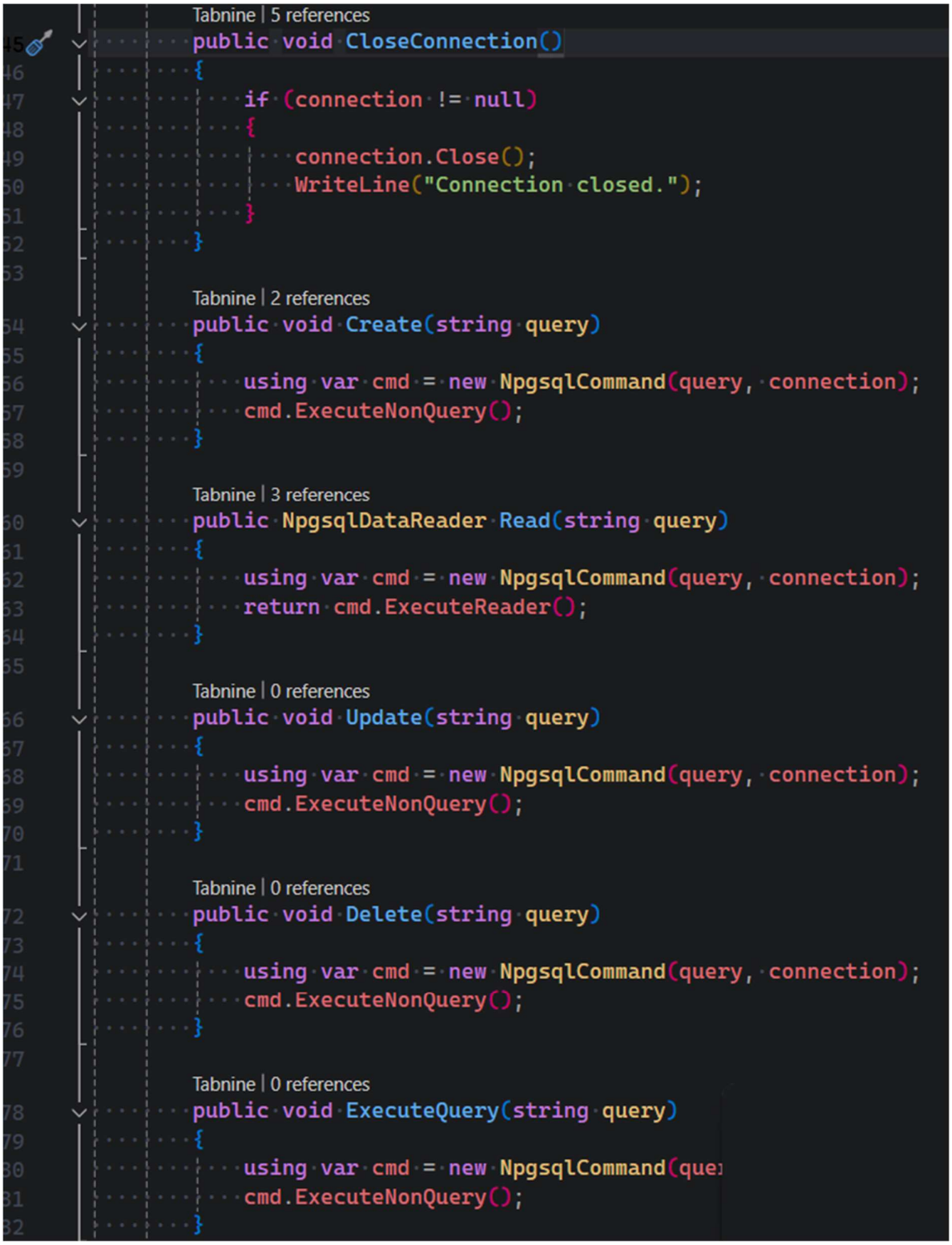
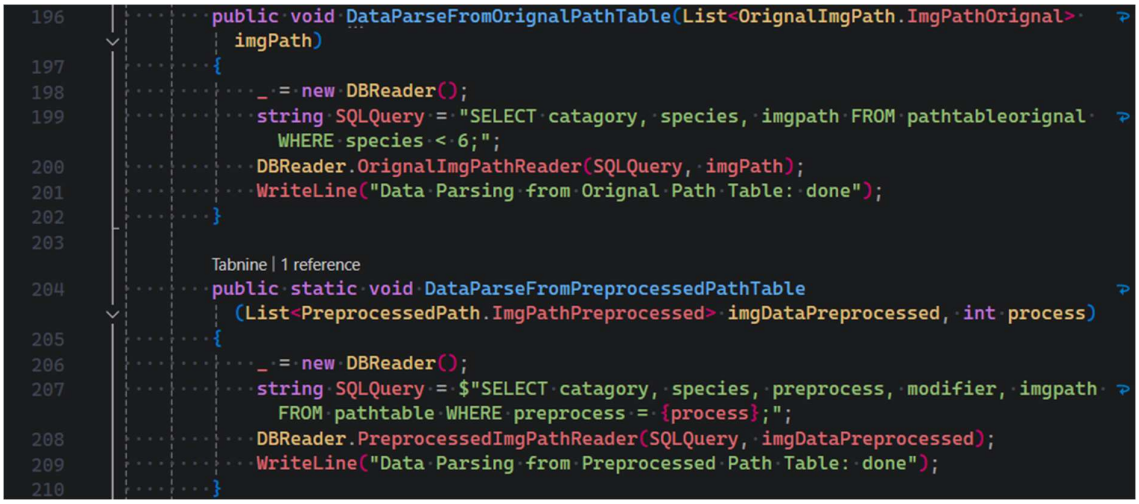


Figure 3.7: Code to perform CRUD operation in database

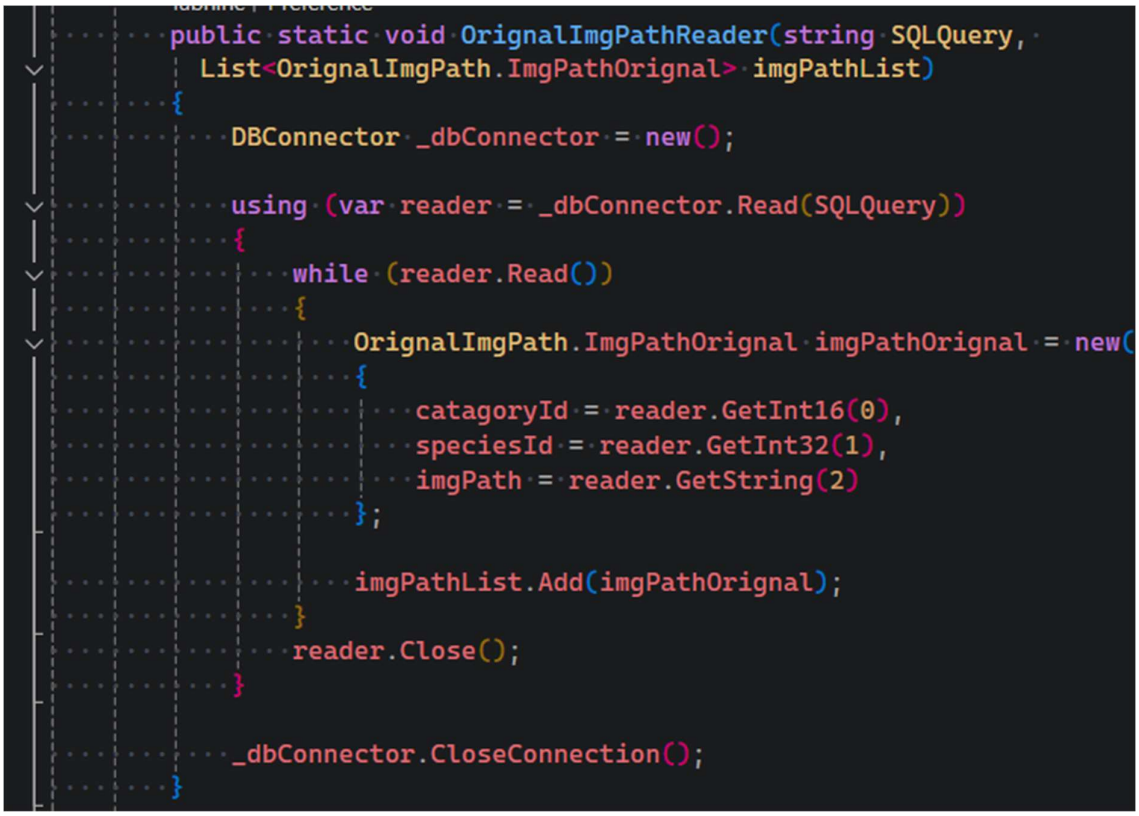
We are parsing data from the database table by executing SQL Query from specified methods and storing it in the data structure.



```
196     public void DataParseFromOrignalPathTable(List<OrignalImgPath.ImgPathOrignal>
            imgPath)
197     {
198         _ = new DBReader();
199         string SQLQuery = "SELECT catagory, species, imgpath FROM pathtableorignal
                WHERE species < 6;";
200         DBReader.OrignalImgPathReader(SQLQuery, imgPath);
201         WriteLine("Data Parsing from Orignal Path Table: done");
202     }
203
        Tabnine | 1 reference
204     public static void DataParseFromPreprocessedPathTable
            (List<PreprocessedPath.ImgPathPreprocessed> imgDataPreprocessed, int process)
205     {
206         _ = new DBReader();
207         string SQLQuery = $"SELECT catagory, species, preprocess, modifier, imgpath
                FROM pathtable WHERE preprocess = {process};";
208         DBReader.PreprocessedImgPathReader(SQLQuery, imgDataPreprocessed);
209         WriteLine("Data Parsing from Preprocessed Path Table: done");
210     }
```

Figure 3.8: Code to execute SQL Query

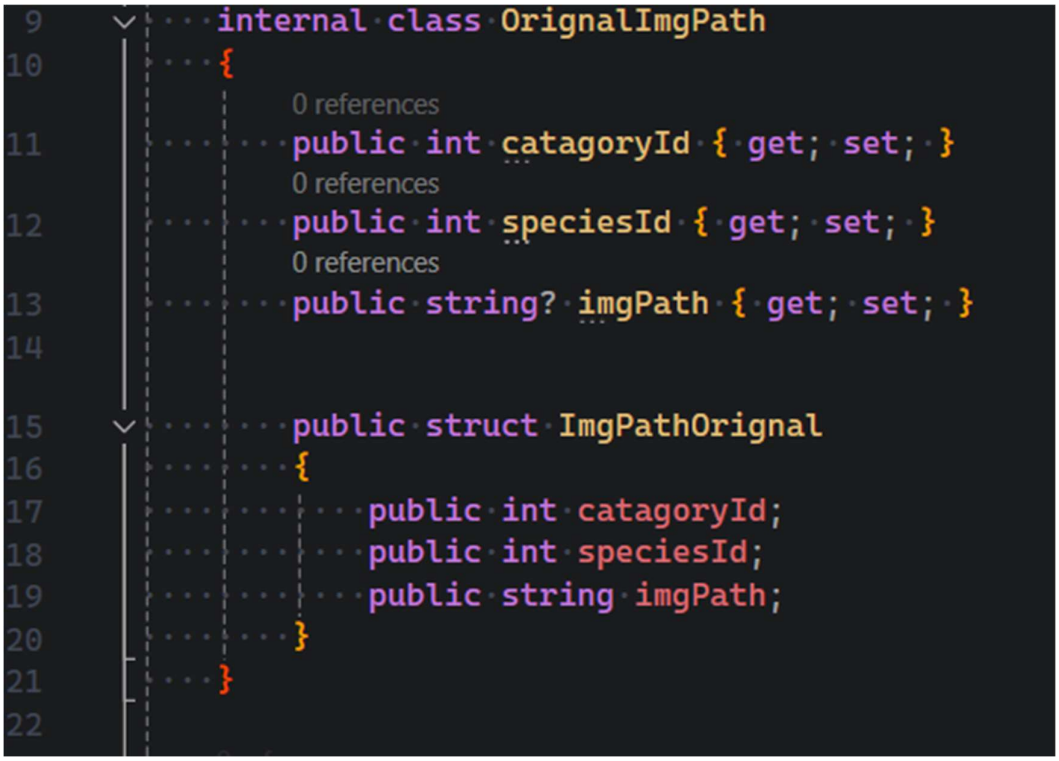
2. **Data Collection and Preprocessing**: The DataAugmentation and BckRemove classes handle the preprocessing of images. After preprocessing, the photos are stored in the database.



```
public static void OrignalImgPathReader(string SQLQuery,
    List<OrignalImgPath.ImgPathOrignal> imgPathList)
{
    DBConnector _dbConnector = new();

    using (var reader = _dbConnector.Read(SQLQuery))
    {
        while (reader.Read())
        {
            OrignalImgPath.ImgPathOrignal imgPathOrignal = new(
            {
                catagoryId = reader.GetInt16(0),
                speciesId = reader.GetInt32(1),
                imgPath = reader.GetString(2)
            };

            imgPathList.Add(imgPathOrignal);
        }
        reader.Close();
    }

    _dbConnector.CloseConnection();
}
```

Figure 3.9: Code to parse and store data

Data structure to store data parsed from the table.



```
internal class OrignalImgPath
{
    0 references
    public int catagoryId { get; set; }
    0 references
    public int speciesId { get; set; }
    0 references
    public string? imgPath { get; set; }


    public struct ImgPathOrignal
    {
        public int catagoryId;
        public int speciesId;
        public string imgPath;
    }
}
```

Figure 3.10: Code to create data structure

Although we performed training on the original species due to computation constraints. We performed augmentation to increase our data.

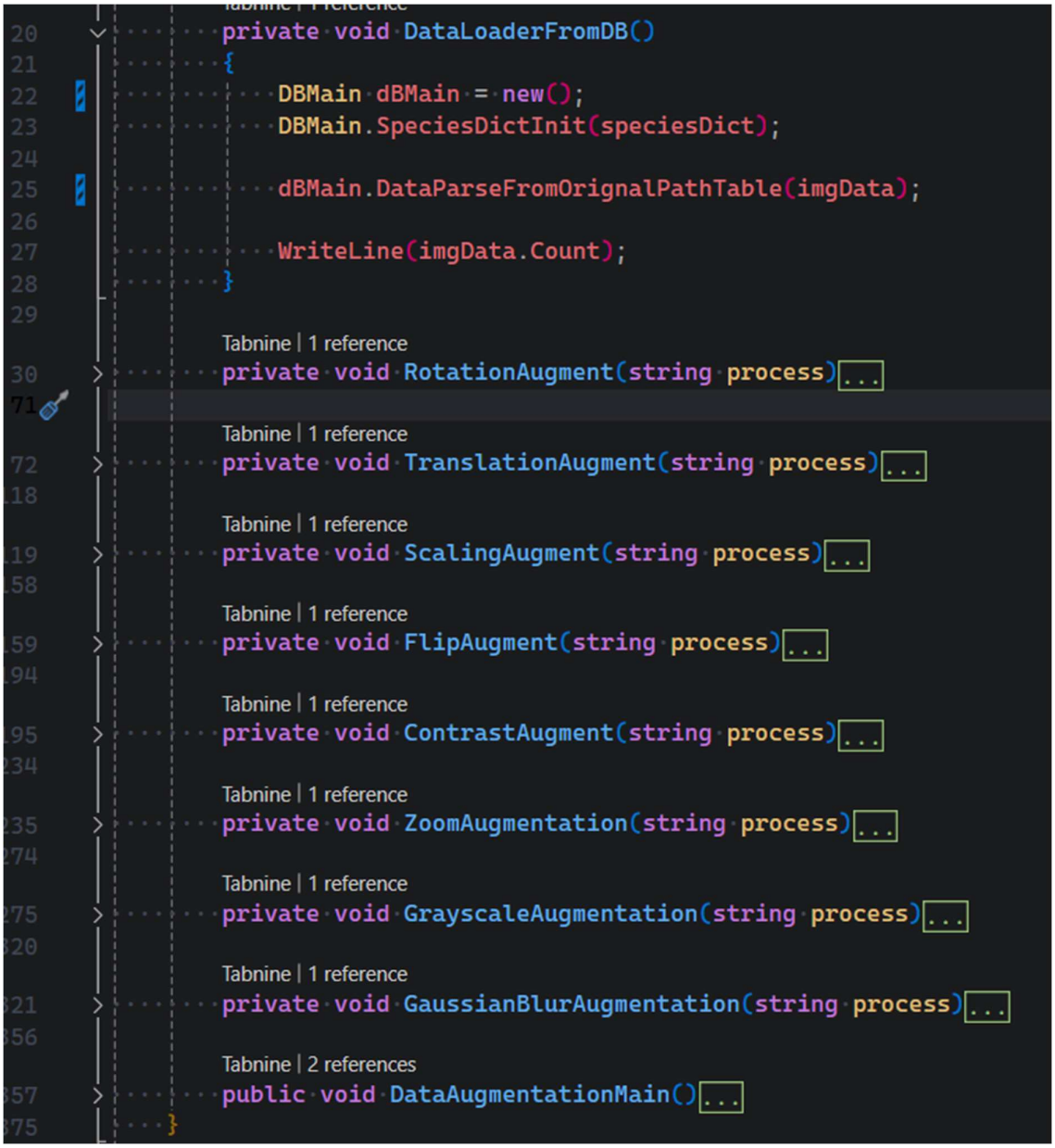
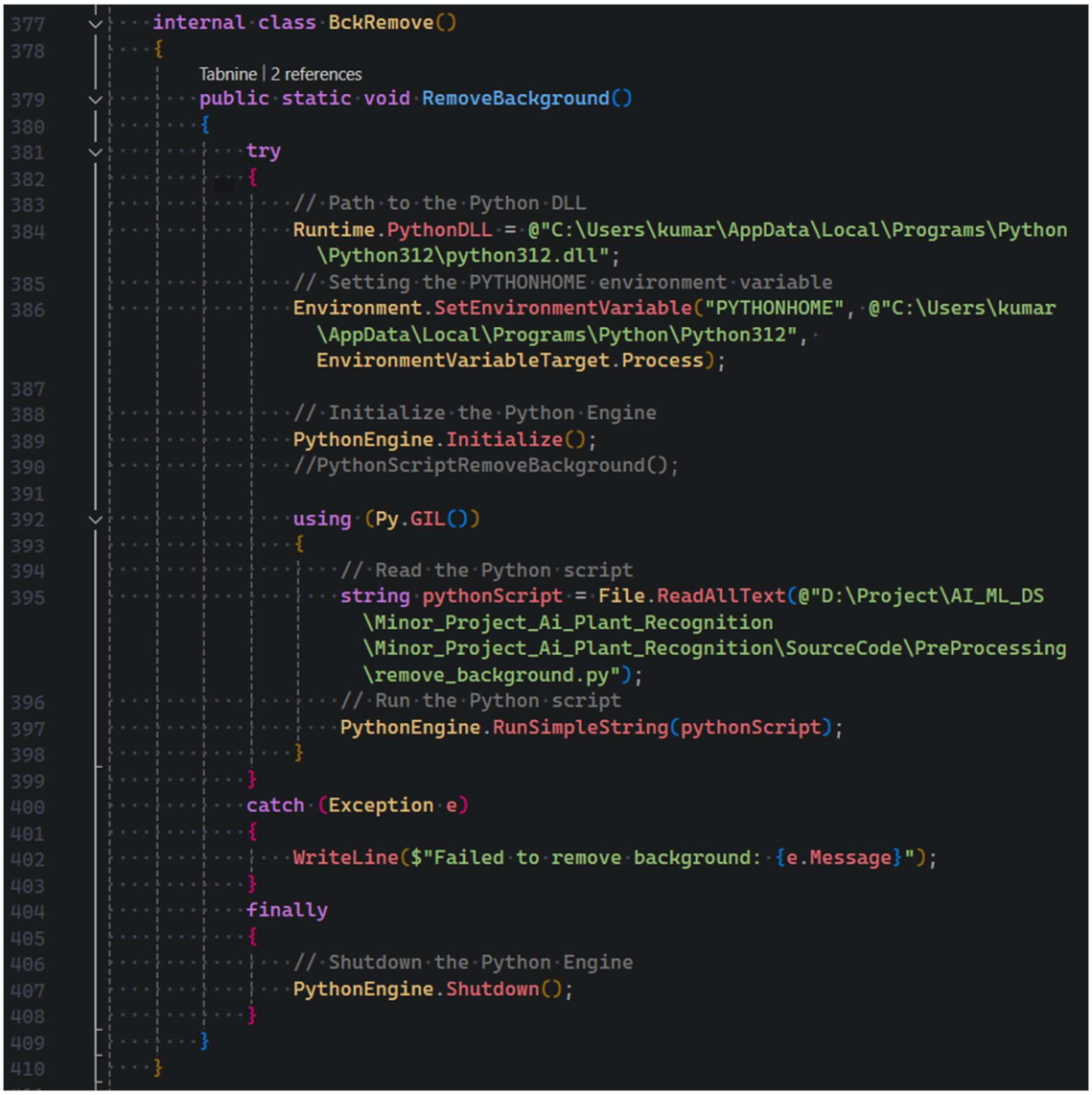


Figure 3.11: Methods of the augmentation.

We are also using Python to harness its power of extensive libraries and modules to remove background.



```
internal class BckRemove()
{
    Tabnine | 2 references
    public static void RemoveBackground()
    {
        try
        {
            // Path to the Python DLL
            Runtime.PythonDLL = @"C:\Users\kumar\AppData\Local\Programs\Python
              \Python312\python312.dll";
            // Setting the PYTHONHOME environment variable
            Environment.SetEnvironmentVariable("PYTHONHOME", @"C:\Users\kumar
              \AppData\Local\Programs\Python\Python312",
              EnvironmentVariableTarget.Process);

            // Initialize the Python Engine
            PythonEngine.Initialize();
            //PythonScriptRemoveBackground();

            using (Py.GIL())
            {
                // Read the Python script
                string pythonScript = File.ReadAllText(@"D:\Project\AI_ML_DS
                  \Minor_Project_Ai_Plant_Recognition
                  \Minor_Project_Ai_Plant_Recognition\SourceCode\PreProcessing
                  \remove_background.py");
                // Run the Python script
                PythonEngine.RunSimpleString(pythonScript);
            }
        }
        catch (Exception e)
        {
            WriteLine($"Failed to remove background: {e.Message}");
        }
        finally
        {
            // Shutdown the Python Engine
            PythonEngine.Shutdown();
        }
    }
}
```

Figure 3.12: Code to execute Python script from C# code.

3. **Model Training**: The training of the machine learning model is handled by the PythonMain class and its PyMain method.

   As we discussed earlier we are using Python to use its modules such as TensorFlow and renbg.

```python
from Models.inception_model import InceptionModel
from Models.inception_resnetV2_model import InceptionResNetModel
from Models.densenet_model import DensenetModel
from Models.vgg16_model import VGGModel
from Models.vit_model import VITMain
import model_data_prepare as mdp

if __name__ == "__main__":
    data_parser = mdp.Main()
    traindata, testdata, pname = data_parser.data_prepare_starter(0)

    print("staring training InceptionV3")
    inception_model = InceptionModel(traindata, testdata, pname)
    x_train, y_train, x_test, y_test = inception_model.data_prepare()
    inception_model.train_model(x_train, y_train, x_test, y_test)

    print("staring training InceptionResNetV2")
    inception_resnetV2_model = InceptionResNetModel(traindata, testdata, pname)
    x_train, y_train, x_test, y_test = inception_resnetV2_model.data_prepare()
    inception_resnetV2_model.train_model(x_train, y_train, x_test, y_test)

    print("staring training DenseNet50")
    Densenet121_model = DensenetModel(traindata, testdata, pname)
    x_train, y_train, x_test, y_test = Densenet121_model.data_prepare()
    Densenet121_model.train_model(x_train, y_train, x_test, y_test)

    print("staring training VGG16")
    vgg16_model = VGGModel(traindata, testdata, pname)
    x_train, y_train, x_test, y_test = vgg16_model.data_prepare()
    vgg16_model.train_model(x_train, y_train, x_test, y_test)
```

Figure 3.13: Code to call class for training model.

```python
    def model(self):
        base_model = InceptionV3(weights='imagenet', include_top=False)

        x = base_model.output
        x = GlobalAveragePooling2D()(x)

        x = Dense(2048, activation='relu')(x)
        x = BatchNormalization()(x)  # Add batch normalization layer
        x = Dropout(0.5)(x)  # Add dropout layer to prevent overfitting

        predictions = Dense(5, activation='softmax')(x)
        model = Model(inputs=base_model.input, outputs=predictions)

        for layer in base_model.layers:
            layer.trainable = False

        optimizer = Adam(learning_rate=0.01)

        model.compile(optimizer=optimizer,
                      loss='categorical_crossentropy', metrics=['accuracy'])

        return model
```

Figure 3.14: Code of the model.

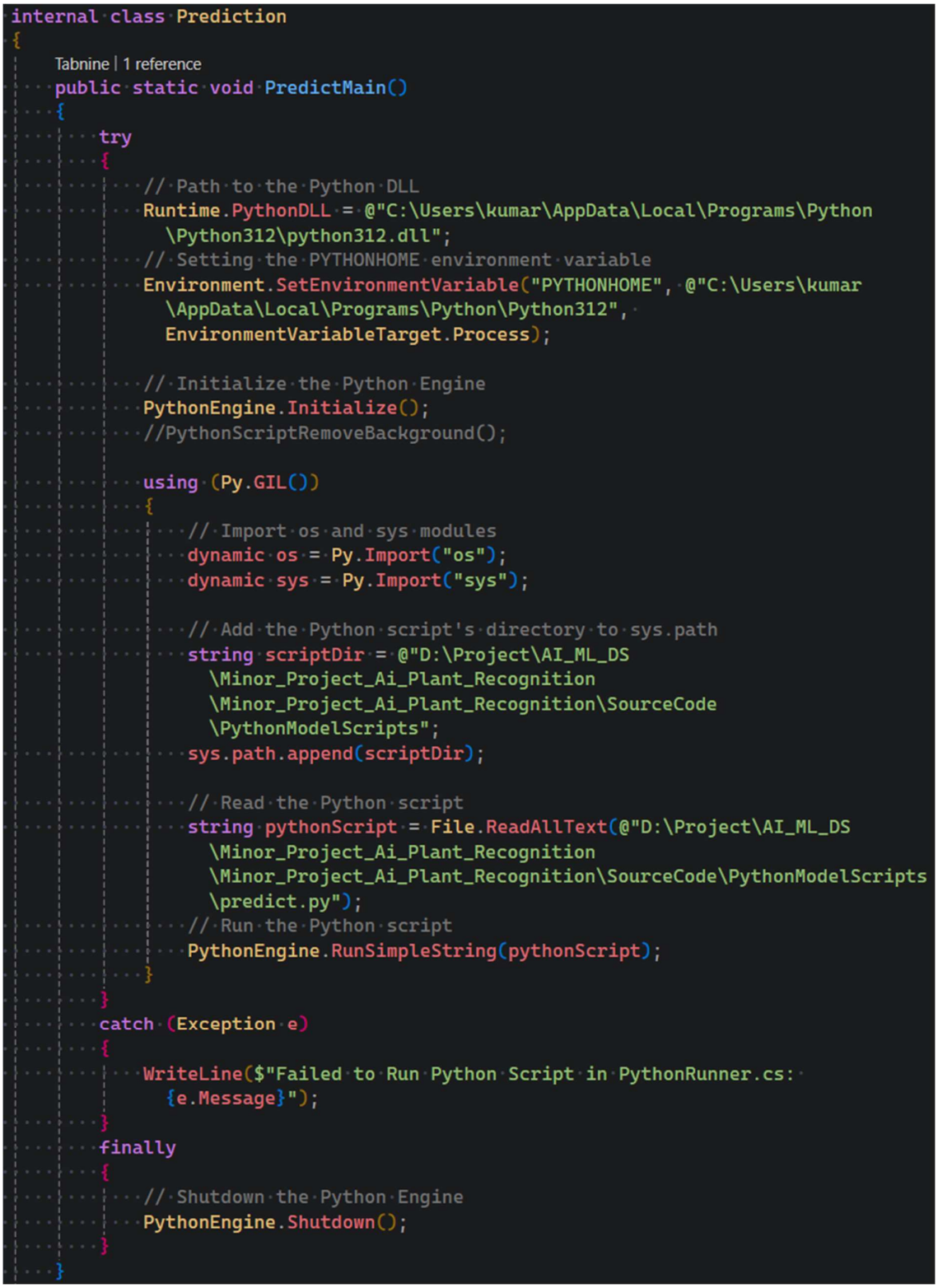
```
def train_model(self, x_train, y_train, x_test, y_test):
    model = self.model()

    # Early stopping
    early_stop = EarlyStopping(monitor='val_loss', patience=5)

    # Reduce learning rate when a metric has stopped improving
    lr_reduce = ReduceLROnPlateau(
        monitor='val_loss', factor=0.1, patience=2, verbose=1, min_delta=1e-4,
        min_lr=0.0001)

    # Save the best model after every epoch
    checkpoint = ModelCheckpoint(
        f'InceptionV3_{self.processed_data_type}.keras', monitor='val_loss',
        verbose=1, save_best_only=True)

    # Train the model
    history = model.fit(x_train, y_train, validation_data=(
        x_test, y_test), epochs=10, batch_size=32, callbacks=[early_stop, lr_reduce,
        checkpoint])
```

Figure 3.15: Code to train model

4. **Prediction**: The prediction of the medicinal plant in a new image is handled by the Prediction class and its PredictMain method.

```
def predict():
    img_folder_path = r"D:\Dataset\medai\test"
    species_dict = {1: 'Aloevera', 2: 'Amla',
                    3: 'Amruthaballi', 4: 'Arali', 5: 'Ashoka'}
    models_dir = r'D:\Project\AI_ML_DS\Minor_Project_Ai_Plant_Recognition
      \Minor_Project_Ai_Plant_Recognition\models_files'
    model_files = os.listdir(models_dir)
    img_files = os.listdir(img_folder_path)

    # Load all models at the start
    models = {model_file: load_model(os.path.join(
        models_dir, model_file)) for model_file in model_files}

    # Initialize DataFrame with image names as index and model names as columns
    results_df = pd.DataFrame(index=img_files, columns=model_files)

    for i, model_file in enumerate(model_files):
        model = models[model_file]
        for img in img_files:
            img_path = os.path.join(img_folder_path, img)
            predictions = predict_plant(img_path, model)
            prediction, probability = predictions[0]
            species = species_dict[int(prediction)]
            # Store the prediction in the DataFrame
            results_df.loc[img, model_file] = species

    mode_result = results_df.mode(axis=1)

    results_df['Majority'] = mode_result.apply(
        lambda row: np.random.choice(row[~pd.isnull(row)]), axis=1)
    # Print the final DataFrame
    print(results_df)
```

Figure 3.16: Code to predict using the trained model.



```
internal class Prediction
{
    Tabnine | 1 reference
    public static void PredictMain()
    {
        try
        {
            // Path to the Python DLL
            Runtime.PythonDLL = @"C:\Users\kumar\AppData\Local\Programs\Python
              \Python312\python312.dll";
            // Setting the PYTHONHOME environment variable
            Environment.SetEnvironmentVariable("PYTHONHOME", @"C:\Users\kumar
              \AppData\Local\Programs\Python\Python312",
              EnvironmentVariableTarget.Process);

            // Initialize the Python Engine
            PythonEngine.Initialize();
            //PythonScriptRemoveBackground();

            using (Py.GIL())
            {
                // Import os and sys modules
                dynamic os = Py.Import("os");
                dynamic sys = Py.Import("sys");

                // Add the Python script's directory to sys.path
                string scriptDir = @"D:\Project\AI_ML_DS
                  \Minor_Project_Ai_Plant_Recognition
                  \Minor_Project_Ai_Plant_Recognition\SourceCode
                  \PythonModelScripts";
                sys.path.append(scriptDir);

                // Read the Python script
                string pythonScript = File.ReadAllText(@"D:\Project\AI_ML_DS
                  \Minor_Project_Ai_Plant_Recognition
                  \Minor_Project_Ai_Plant_Recognition\SourceCode\PythonModelScripts
                  \predict.py");
                // Run the Python script
                PythonEngine.RunSimpleString(pythonScript);
            }
        }
        catch (Exception e)
        {
            WriteLine($"Failed to Run Python Script in PythonRunner.cs:
              {e.Message}");
        }
        finally
        {
            // Shutdown the Python Engine
            PythonEngine.Shutdown();
        }
    }
```

Figure 3.17: Code to call predict python script from C#

5. **Result Display**: The display of the prediction results to the user is typically done using a simple print statement.

```python
def predict():
    img_folder_path = r"D:\Dataset\medai\test"
    species_dict = {1: 'Aloevera', 2: 'Amla',
                    3: 'Amruthaballi', 4: 'Arali', 5: 'Ashoka'}
    models_dir = r'D:\Project\AI_ML_DS\Minor_Project_Ai_Plant_Recognition
      \Minor_Project_Ai_Plant_Recognition\models_files'
    model_files = os.listdir(models_dir)
    img_files = os.listdir(img_folder_path)

    # Load all models at the start
    models = {model_file: load_model(os.path.join(
        models_dir, model_file)) for model_file in model_files}

    # Initialize DataFrame with image names as index and model names as columns
    results_df = pd.DataFrame(index=img_files, columns=model_files)

    for i, model_file in enumerate(model_files):
        model = models[model_file]
        for img in img_files:
            img_path = os.path.join(img_folder_path, img)
            predictions = predict_plant(img_path, model)
            prediction, probability = predictions[0]
            species = species_dict[int(prediction)]
            # Store the prediction in the DataFrame
            results_df.loc[img, model_file] = species

    mode_result = results_df.mode(axis=1)

    results_df['Majority'] = mode_result.apply(
        lambda row: np.random.choice(row[~pd.isnull(row)]), axis=1)
    # Print the final DataFrame
    print(results_df)
```

Figure 3.18: Code to output predicted result.

```
                  DenseNet_Orignal_Data.keras InceptionV3_Orignal_Data.keras Inception_ResNet_Orignal_Data.keras VGG16_Orignal_Data.keras      Majority
aloevera 1.jpeg                     Aloevera                       Aloevera                          Aloevera                  Aloevera      Aloevera
aloevera 2.jpeg                     Aloevera                       Aloevera                          Aloevera                  Aloevera      Aloevera
aloevera 3.jpg                      Aloevera                       Aloevera                          Aloevera                  Aloevera      Aloevera
aloevera 4.jpeg                     Aloevera                       Aloevera                          Aloevera                  Aloevera      Aloevera
amla (2).jpg                            Amla                           Amla                              Amla                  Aloevera          Amla
amla (3).jpg                            Amla                           Amla                              Amla                      Amla          Amla
amla (4).jpg                            Amla                           Amla                              Amla                    Ashoka          Amla
amla (5).jpg                            Amla                          Ashoka                             Amla                      Amla          Amla
Amla.jpg                                Amla                           Amla                              Amla                      Amla          Amla
Amruthaballi 1.jpeg                    Arali                  Amruthaballi                             Arali             Amruthaballi  Amruthaballi
Amruthaballi 2.jpeg          Amruthaballi                  Amruthaballi                             Arali             Amruthaballi  Amruthaballi
Amruthaballi 3.jpeg          Amruthaballi                         Arali                     Amruthaballi             Amruthaballi  Amruthaballi
Amruthaballi 4.jpeg          Amruthaballi                  Amruthaballi                             Arali             Amruthaballi  Amruthaballi
Amruthaballi 5.jpg           Amruthaballi                  Amruthaballi                     Amruthaballi                  Aloevera  Amruthaballi
arali 1.jpg                             Arali                          Arali                     Amruthaballi                  Aloevera         Arali
arali 2.jpeg                            Arali                          Arali                             Arali             Amruthaballi         Arali
arali 3.jpeg                            Arali                          Arali                             Arali             Amruthaballi         Arali
arali 4.jpeg                            Arali                          Arali                             Arali             Amruthaballi         Arali
arali 5.jpeg                 Amruthaballi                  Amruthaballi                             Arali                      Arali  Amruthaballi
ashoka (2).jpg                          Ashoka                         Ashoka                            Ashoka                    Ashoka        Ashoka
ashoka (3).jpg                          Ashoka                         Ashoka                            Ashoka                    Ashoka        Ashoka
ashoka 3.jpeg                           Ashoka                         Ashoka                             Arali                    Ashoka        Ashoka
ashoka.jpeg                             Ashoka                           Amla                            Ashoka                    Ashoka        Ashoka
ashoka.jpg                              Ashoka                         Ashoka                            Ashoka                    Ashoka        Ashoka
thanks for you corporation...
thanks for you corporation...
```

Figure 3.19: Prediction results of the models

Chapter 04: RESULTS

4.1 Discussion on the Results Achieved

In this project, we utilized several pre-trained models for the task of medicinal plant recognition from images. Specifically, we used InceptionV3, InceptionResNetV2, DenseNet121, and VGG16. These models are all variants of Convolutional Neural Networks (CNNs), which are particularly effective for image classification tasks.

1.**InceptionV3**: This model performed well on our dataset, demonstrating the effectiveness of its 'network within a network' structure and 'Inception modules' for efficient computation and deep learning.

Table 4.1: Performance Matrix of Inception V3

Performance Matrix	Train	Test
Accuracy	0.99716	0.97826
Loss	0.01045	0.16642
R Squre	0.99	0.94
Mean R Square	0.001	0.008

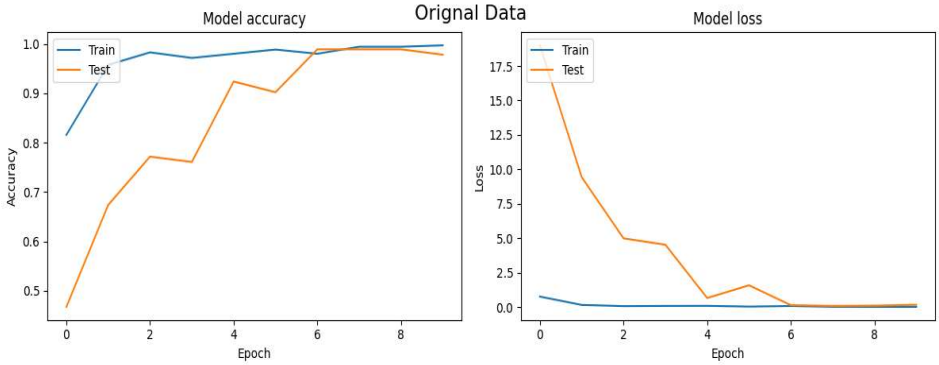


Figure 4.1: Accuracy and Loss of the Inception V3

We can perceive from this graph that the model starts performing well after the 7th or 8th epochs and further iterating is not feasible.

2.**InceptionResNetV2**: This model combines the advantages of Inception networks and ResNets. It showed good performance on our dataset, benefiting from its

'Inception modules' and 'skip connections' that allow for efficient computation and improved gradient flow.

Table 4.1: Performance Matrix of Inception ResNet V2

Performance Matrix	Train	Test
Accuracy	0.99433	0.98913
Loss	0.00909	0.09937
R Squre	0.99	0.97
Mean R Square	0.001	0.003

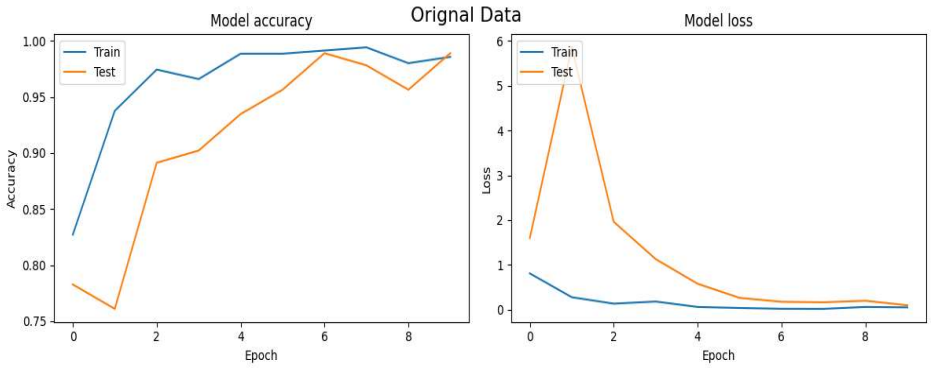


Figure 4.2: Accuracy and Loss of the Inception Resnet V2

We can see here that our model sharply loses accuracy around 5th epochs from 97-98 to 94 to 95 but it regains its accuracy around epoch 9 or 10 and start performing well.

3.**DenseNet121**: DenseNet121, with its unique structure where each layer is connected to every other layer in a feed-forward fashion, also showed promising results. The improved gradient flow and low redundancy of DenseNets were beneficial for our task.

Table 4.1: Performance Matrix of DenseNet121

Performance Matrix	Train	Test
Accuracy	0.9745	0.95652
Loss	0.27276	0.58964
R Squre	0.93	0.88
Mean R Square	0.009	0.017

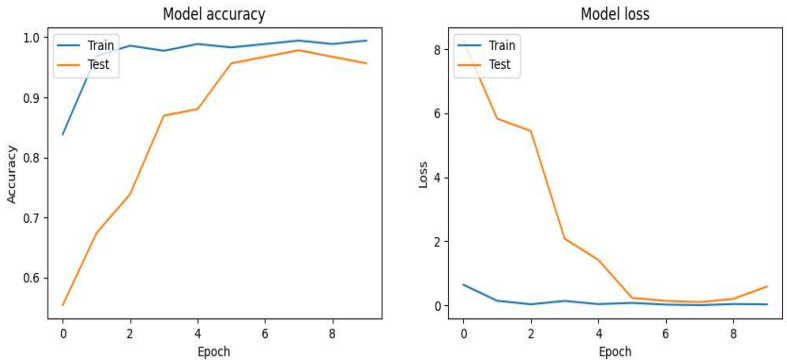


Figure 4.3: Accuracy and Loss of the DenseNet 121

This dense model shows good accuracy but during validation, loss is very high which is not ideal.

4.**VGG16**: VGG16, with its simplicity and depth, was effective for our image classification task. Its structure of stacked 3x3 convolutional layers demonstrated the power of depth in neural networks.

Table 4.1: Performance Matrix of VGG16

Performance Matrix	Train	Test
Accuracy	0.898	0.8695
Loss	0.5133	0.5489
R Squre	0.7552	0.6921
Mean R Square	0.038	0.047

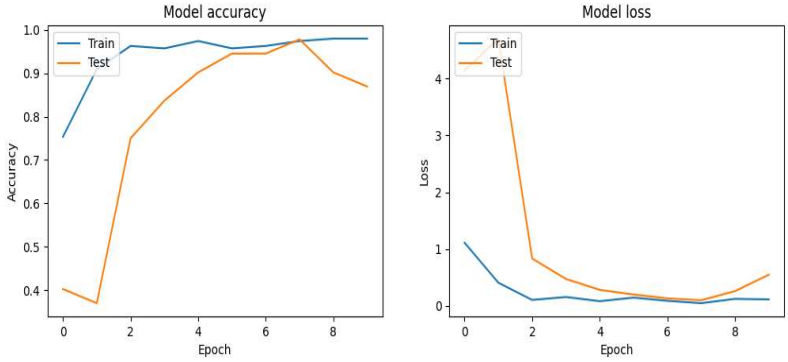


Figure 4.4: Accuracy and Loss of the VGG16

We can see here that our model sharply loses accuracy around 7th epochs from 97-98 to 89 but it never regains its accuracy but we can improve it after some optimization.

Each of these models contributed to the overall performance of our medicinal plant recognition system. By comparing their performance, we were able to gain insights into the strengths and weaknesses of each model, and how they can be effectively utilized for image classification tasks.

All these models are running on their base configuration and pre-trained weight at the drop out 0.5, learning rate 0.01, and neuron 2048. As I selected only five species and only one leaf, we need only one layer but we can increase the layer as well as neurons at each layer to capture more features and information. It will significantly increase accuracy at the cost of resources and time. As we increase epochs and reduce the learning rate the accuracy of the model starts approaching 99.9. When we are finetuning the model by an unfreezing layer of the pre-trained model and hyperparameter tuning it starts approaching 100 percent accuracy with shallow loss but the time to train the model also increases significantly.

Right now, the best-performing models are inceptionv3, inception resnet v3, and denseness. All these models perform well in their base parameters and are good to use in prediction. To test these train models, we collect some images from Google that are neither part of training nor testing and run prediction. The result of the prediction is stored in the table for each model in a row and their prediction in columns.

Figure 4.5: Images to run prediction taken from Google.

	DenseNet_Orignal_Data.keras	InceptionV3_Orignal_Data.keras	Inception_ResNet_Orignal_Data.keras	VGG16_Orignal_Data.keras	Majority
aloevera 1.jpeg	Aloevera	Aloevera	Aloevera	Aloevera	Aloevera
aloevera 2.jpeg	Aloevera	Aloevera	Aloevera	Aloevera	Aloevera
aloevera 3.jpg	Aloevera	Aloevera	Aloevera	Aloevera	Aloevera
aloevera 4.jpeg	Aloevera	Aloevera	Aloevera	Aloevera	Aloevera
amla (2).jpg	Amla	Amla	Amla	Aloevera	Amla
amla (3).jpg	Amla	Amla	Amla	Aloevera	Amla
amla (4).jpg	Amla	Amla	Amla	Aloevera	Amla
amla (5).jpg	Amla	Amla	Amla	Aloevera	Amla
Amla.jpg	Amla	Amla	Amla	Amla	Amla
Amruthaballi 1.jpeg	Arali	Arali	Arali	Amruthaballi	Arali
Amruthaballi 2.jpeg	Amruthaballi	Amruthaballi	Amruthaballi	Amruthaballi	Amruthaballi
Amruthaballi 3.jpeg	Amruthaballi	Arali	Amruthaballi	Amruthaballi	Amruthaballi
Amruthaballi 4.jpeg	Amruthaballi	Arali	Amruthaballi	Amruthaballi	Amruthaballi
Amruthaballi 5.jpg	Amruthaballi	Amruthaballi	Amruthaballi	Amruthaballi	Amruthaballi
arali 1.jpg	Arali	Arali	Arali	Aloevera	Arali
arali 2.jpeg	Arali	Arali	Arali	Amruthaballi	Arali
arali 3.jpeg	Arali	Arali	Arali	Amruthaballi	Arali
arali 4.jpeg	Arali	Arali	Arali	Amruthaballi	Arali
arali 5.jpeg	Amruthaballi	Amruthaballi	Arali	Amruthaballi	Amruthaballi
ashoka (2).jpg	Ashoka	Ashoka	Ashoka	Ashoka	Ashoka
ashoka (3).jpg	Ashoka	Ashoka	Ashoka	Aloevera	Ashoka
ashoka 3.jpeg	Ashoka	Ashoka	Ashoka	Ashoka	Ashoka
ashoka 4.jpeg	Arali	Aloevera	Arali	Aloevera	Arali
ashoka.jpeg	Ashoka	Arali	Ashoka	Aloevera	Ashoka
ashoka.jpg	Ashoka	Ashoka	Ashoka	Aloevera	Ashoka

Figure 4.6: Prediction Result

4.2 Application of the Minor Project

The " AI-Based Medicinal Plant Recognition " project has several practical applications that extend beyond the scope of academic research.

1. **Medicinal Plant Identification**: The primary application of this project is in the field of medicinal plant identification. The automated system developed in this project can be used by botanists, herbalists, and researchers to accurately identify and classify different medicinal plant species based on images. This can significantly speed up the process of plant identification and reduce the risk of misidentification.

2. **Biodiversity Studies**: The machine learning model trained in this project can be used in biodiversity studies to monitor and track the presence of different medicinal plant species in a given area. This can provide valuable data for conservation efforts and help in understanding the impact of environmental changes on plant diversity.

3. **Pharmaceutical Industry**: The pharmaceutical industry can leverage the outcomes of this project for the identification and classification of medicinal plants used in drug production. This can streamline the process of raw material acquisition and quality control, ensuring that the correct plant species are used in pharmaceutical products.

4. **Education**: The project can also serve as an educational tool for students studying botany or related fields. It can provide a practical demonstration of how machine

learning and image processing techniques can be applied in the field of plant recognition.

5. **Database Management**: The PostgreSQL database set up for this project can be used as a model for efficient storage and retrieval of plant images and model predictions in similar projects. This can facilitate data management in large-scale plant recognition studies.

The "AI-Based Medicinal Plant Recognition" project has wide-ranging applications that can benefit various fields including botany, conservation biology, the pharmaceutical industry, and education. Future research can explore these applications in more detail and develop ways to optimize the system for specific use cases.

4.3 Limitation of the Minor Project

While the " AI-Based Medicinal Plant Recognition " project has demonstrated promising results, there are several limitations that need to be addressed:

1. **Hardware Constraints**: One of the major limitations of this project is the hardware constraint. The machine learning model requires a significant number of computational resources for training and inference. This can be a challenge for users with limited hardware capabilities.

2. **Quality of Images**: The performance of the model heavily depends on the quality of the input images. Poor-quality images, such as those with low resolution, poor lighting, or occlusions, can lead to inaccurate predictions.

3. **Limited Dataset**: The model's performance is also limited by the size and diversity of the training dataset. If the dataset does not cover all possible variations of the plant species (e.g., different growth stages, lighting conditions, angles), the model may not generalize well to new images.

4. **Overfitting**: There is a risk of overfitting, especially if the model is trained on a small or unbalanced dataset. Overfitting can lead to poor performance when the model is applied to new, unseen data.

5. **Lack of Explainability**: Machine learning models, especially deep learning models, are often criticized for their lack of explainability. It can be difficult to understand

why the model made a certain prediction, which can be a problem in applications where interpretability is important.

6. **Time-Consuming Training Process**: Training a machine learning model can be a time-consuming process, especially for large datasets or complex models. This can be a limitation in scenarios where quick results are needed.

7. **Dependency on External Libraries**: The project relies on several external libraries for machine learning and image processing. Any changes or updates in these libraries can potentially affect the project's performance.

These limitations provide opportunities for future work to improve the system's performance and usability.

4.4 Future Work

The " AI-Based Medicinal Plant Recognition project has laid a solid foundation for further research and development in the field of automated plant recognition. Here are some potential directions for future work:

1. **Developing a Custom Model**: One of the main areas of future work is to develop a custom machine learning model tailored to the specific needs of this project. This could involve exploring different architectures, training strategies, and feature extraction methods to improve the model's performance.

2. **Improving Dataset Quality**: Another important area of future work is to improve the quality of the dataset. This could involve collecting more high-quality images, augmenting the existing dataset with different transformations, and ensuring a balanced representation of different plant species.

3. **Web Application Development**: Developing a web application for the project will make it more accessible to users. The application could provide a user-friendly interface for uploading images, running the model, and displaying the results. It could also include features for managing and browsing the dataset.

4. **Real-Time Recognition**: In the future, the project could be extended to support real-time plant recognition. This would involve optimizing the model and the image processing pipeline to work with live video feeds.

5. **Integration with GIS Systems**: The project could be integrated with Geographic Information System (GIS) platforms to provide location-based plant recognition services. This could be useful for biodiversity studies and conservation efforts.

6. **Model Explainability**: Future work could also focus on improving the explainability of the model. This could involve implementing techniques like SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-Agnostic Explanations) to provide insights into the model's decision-making process.

7. **Cross-Platform Compatibility**: Ensuring that the project works seamlessly across different platforms (Windows, Linux, macOS) and devices (desktop, mobile) can increase its usability and reach.

Numerous opportunities for future work can build upon the achievements of the " AI-Based Medicinal Plant Recognition project and push the boundaries of what is currently possible in the field of automated plant recognition.

References

1) **Dataset:** B R, Pushpa; Rani, Shobha (2023), "Indian Medicinal Leaves Image Datasets", Mendeley Data, V3, doi: 10.17632/748f8jkphb.3
2) "State of the World's Plants and Fungi | Kew," *Kew.org*, 2016. https://www.kew.org/science/state-of-the-worlds-plants-and-fungi (accessed May 16, 2024).
3) "The impact of disasters and crises on agriculture and food security," 2017. Available: https://openknowledge.fao.org/server/api/core/bitstreams/0f03a24f-8d37-4700-aef4-fc436ff32021/content
4) *Bsienvis.nic.in*, 2015. http://www.bsienvis.nic.in/Database/Plant-Discoveries-2020_20977.aspx (accessed May 16, 2024).
5) WWF, "BENDING THE CURVE OF BIODIVERSITY LOSS LIVING PLANET REPORT 2020," 2020. Available: https://www.wwf.org.uk/sites/default/files/2020-09/LPR20_Full_report.pdf
6) X. Qin, Zichen Vincent Zhang, C. Huang, M. Dehghan, O. R. Zaïane, and M. Jägersand, "U2-Net: Going deeper with nested U-structure for salient object detection," *Pattern recognition*, vol. 106, pp. 107404–107404, Oct. 2020, doi: https://doi.org/10.1016/j.patcog.2020.107404.
7) B. Dey, J. Ferdous, R. Ahmed, and J. Hossain, "Assessing deep convolutional neural network models and their comparative performance for automated medicinal plant identification from leaf images," *Heliyon*, vol. 10, no. 1, pp. e23655–e23655, Jan. 2024, doi: https://doi.org/10.1016/j.heliyon.2023.e23655.
8) P. Sarma, Parvez Aziz Boruah, and Rubul Buragohain, "MED 117: A dataset of medicinal plants mostly found in Assam with their leaf images, segmented leaf frames and name table," *Data in brief*, vol. 47, pp. 108983–108983, Apr. 2023, doi: https://doi.org/10.1016/j.dib.2023.108983.
9) S. Kavitha, T. Satish Kumar, E. Naresh, V. H. Kalmani, Kalyan Devappa Bamane, and Piyush Kumar Pareek, "Medicinal Plant Identification in Real-Time Using Deep Learning Model," SN Computer Science/SN computer science, vol. 5, no. 1, Dec. 2023, doi: https://doi.org/10.1007/s42979-023-02398-5.
10) P. Sarma, Parvez Aziz Boruah, and Rubul Buragohain, "MED 117: A dataset of medicinal plants mostly found in Assam with their leaf images, segmented leaf frames and name table," *Data in brief*, vol. 47, pp. 108983–108983, Apr. 2023, doi:

https://doi.org/10.1016/j.dib.2023.108983.

11) X. Sun, H. Qian, Y. Xiong, Y. Zhu, Z. Huang, and F. Yang, "Deep learning-enabled mobile application for efficient and robust herb image recognition," *Scientific reports*, vol. 12, no. 1, Apr. 2022, doi: https://doi.org/10.1038/s41598-022-10449-9.

12) S. Naeem *et al.*, "The Classification of Medicinal Plant Leaves Based on Multispectral and Texture Feature Using Machine Learning Approach," *Agronomy*, vol. 11, no. 2, pp. 263–263, Jan. 2021, doi: https://doi.org/10.3390/agronomy11020263.

13) J. Wang *et al.*, "Combined Channel Attention and Spatial Attention Module Network for Chinese Herbal Slices Automated Recognition," *Frontiers in neuroscience*, vol. 16, Jun. 2022, doi: https://doi.org/10.3389/fnins.2022.920820.

14) N. S. Bhelkar and A. Sharma, "Identification and classification of medicinal plants using leaf with deep convolutional neural networks," *ResearchGate*, Oct. 05, 2022. https://www.researchgate.net/publication/364182850_Identification_and_classification_of_medicinal_plants_using_leaf_with_deep_convolutional_neural_networks (accessed May 17, 2024).

15) Rahim Azadnia, Mohammed Maitham Al-Amidi, H. Mohammadi, Mehmet Akif Cifci, Avat Daryab, and E. Cavallo, "An AI Based Approach for Medicinal Plant Identification Using Deep CNN Based on Global Average Pooling," *Agronomy*, vol. 12, no. 11, pp. 2723–2723, Nov. 2022, doi: https://doi.org/10.3390/agronomy12112723.

16) Rahim Azadnia and Kamran Kheiralipour, "Recognition of leaves of different medicinal plant species using a robust image processing algorithm and artificial neural networks classifier," *Journal of applied research on medicinal and aromatic plants*, vol. 25, pp. 100327–100327, Dec. 2021, doi: https://doi.org/10.1016/j.jarmap.2021.100327.

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 17/5/2024

Type of Document (Tick): | **B.Tech./B.Sc./BBA/Other** ✓

Name: Awanish Thakur, Chandan Kumar **Department:** CSE **Enrolment No** 211114, 211301

Contact No. 8415023139 **E-mail.** 211114@juitsolan.in, 211301@juitsolan.in

Name of the Supervisor: Mr FAISAL FIRDOUS

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters):

AI-BASED MEDICINAL PLANT DETECTION

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at 8(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)

Faisal
17-5-24

report

ORIGINALITY REPORT

8%	4%	3%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	fastercapital.com Internet Source	1%
2	Submitted to Jaypee University of Information Technology Student Paper	1%
3	www.mdpi.com Internet Source	1%
4	Pushpa B R, N. Shobha Rani. "DIMPSAR: Dataset for Indian medicinal plant species analysis and recognition", Data in Brief, 2023 Publication	1%
5	open-innovation-projects.org Internet Source	1%
6	let.de Internet Source	<1%
7	Submitted to Liverpool John Moores University Student Paper	<1%
8	Submitted to University of Derby Student Paper	<1%

9	Submitted to University of Sheffield Student Paper	<1 %
10	docplayer.net Internet Source	<1 %
11	Jinjie Fang, Linshan Yang, Xiaohu Wen, Haijiao Yu, Weide Li, Jan F. Adamowski, Rahim Barzegar. "Ensemble learning using multivariate variational mode decomposition based on the transformer for multistep-ahead streamflow forecasting", Journal of Hydrology, 2024 Publication	<1 %
12	sciencescholar.us Internet Source	<1 %
13	Submitted to SASTRA University Student Paper	<1 %
14	Submitted to Rajasthan Technical University, Kota Student Paper	<1 %
15	Binbin Yong, Zijian Xu, Xin Wang, Libin Cheng, Xue Li, Xiang Wu, Qingguo Zhou. "IoT-based intelligent fitness system", Journal of Parallel and Distributed Computing, 2017 Publication	<1 %
16	Submitted to Northwest Missouri State University Student Paper	<1 %

17	Submitted to University of Westminster Student Paper	<1 %
18	www.irjmets.com Internet Source	<1 %
19	Biplob Dey, Jannatul Ferdous, Romel Ahmed, Juel Hossain. "Assessing deep convolutional neural network models and their comparative performance for automated medicinal plant identification from leaf images", Heliyon, 2023 Publication	<1 %
20	kipdf.com Internet Source	<1 %
21	Thottempudi Naga Sri Yaswanth, K.S Vijaya Lakshmi, Kallepalli Guru Vamsi, Devarapalli Supraja, K Suvarna Vani. "Classification of Medicinal Leaves using SVM", 2023 World Conference on Communication & Computing (WCONF), 2023 Publication	<1 %
22	Submitted to University of Wales Institute, Cardiff Student Paper	<1 %
23	Samreen Naeem, Aqib Ali, Christophe Chesneau, Muhammad H. Tahir, Farrukh Jamal, Rehan Ahmad Khan Sherwani, Mahmood Ul Hassan. "The Classification of	<1 %

Medicinal Plant Leaves Based on Multispectral and Texture Feature Using Machine Learning Approach", Agronomy, 2021

Publication

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		