A picture containing clipart

Description automatically generated

INFORMATICS INSTITUE OF TECHNOLOGY

DEPARTMENT OF COMPUTING

Module: 5COSC009C.2

Software Development Group Project

Module Leader: Mr. Guhanathan Poravi

**Paddy Weed Detector**

|  |  |  |
| --- | --- | --- |
| Team Hypesters |  |  |
| Name | Iit ID | UoW ID |
| Akila Nanayakkara | 2018400 | w1742308 |
| Kisal Wedage | 2018368 | w1742101 |
| Moveen Wijerathne | 2018541 | w1742328 |
| Amilakelum Kodikara | 2018395 | w1742107 |
| Danushka Samarakoon | 2018382 | w1742106 |
| Malshama Perera | 2018446 | w1742313 |

Table of Contents

[List of Tables 4](#_Toc38885873)

[Table of figures 4](#_Toc38885874)

[1. Implementation 5](#_Toc38885875)

[1.1. Chapter Overview 5](#_Toc38885876)

[1.2. Overview of the Prototype 5](#_Toc38885877)

[1.3. Selection of Technologies 5](#_Toc38885878)

[1.3.1. Programming language for data science part 5](#_Toc38885879)

[1.3.2. Image Processing Library 5](#_Toc38885880)

[1.3.3. Web Frameworks Selection 6](#_Toc38885881)

[1.4. Selection of tools 6](#_Toc38885882)

[1.4.1. Selection of IDE 6](#_Toc38885883)

[1.4.2. Selection of the version control system 6](#_Toc38885884)

[1.5. Technology Stack 7](#_Toc38885885)

[1.6. Data Set 8](#_Toc38885886)

[1.7. Implementation of the features of the prototype in the back end 8](#_Toc38885887)

[1.7.1. Data set Pre-Processing 8](#_Toc38885888)

[1.7.2. Training the model 9](#_Toc38885889)

[1.7.3. Selected Algorithm 11](#_Toc38885890)

[1.7.4. Testing the model 11](#_Toc38885891)

[1.8. Implementation of the GUI 13](#_Toc38885892)

[1.8.1. Implementation of the input page 13](#_Toc38885893)

[1.8.2. Implementation of the output page 15](#_Toc38885894)

[1.9. Combining the back end and the front end 16](#_Toc38885895)

[1.10. Chapter Summary 16](#_Toc38885896)

[2. Testing 17](#_Toc38885897)

[2.1. Chapter Overview 17](#_Toc38885898)

[2.2. Testing Goals and objectives 17](#_Toc38885899)

[2.3. Testing 17](#_Toc38885900)

[2.4. Testing Criteria 17](#_Toc38885901)

[2.5. Testing Functional Requirements 17](#_Toc38885902)

[2.6. Performance Testing 18](#_Toc38885903)

[2.7. Usability Testing 18](#_Toc38885904)

[2.8. Compatibility Testing 18](#_Toc38885905)

[2.9. Chapter Summary 18](#_Toc38885906)

[3. Evaluation 19](#_Toc38885907)

[3.1. Chapter Overview 19](#_Toc38885908)

[3.2. Evaluation Goals 19](#_Toc38885909)

[3.3. Evaluations Benchmark 19](#_Toc38885910)

[3.4. Evaluations Selections 19](#_Toc38885911)

[3.5. Concept and Scope 19](#_Toc38885912)

[3.6. Technical Aspects 19](#_Toc38885913)

[3.7. Impacts 19](#_Toc38885914)

[3.8. Limitations 19](#_Toc38885915)

[3.9. Research question 19](#_Toc38885916)

[3.10. Self-Evaluation 20](#_Toc38885917)

[3.11. Chapter Summary 20](#_Toc38885918)

[4. Conclusion 21](#_Toc38885919)

[4.1. Chapter Overview 21](#_Toc38885920)

[4.2. Achievement of the Project Aim 21](#_Toc38885921)

[4.3. Achieveemnts of Operational Objectives 21](#_Toc38885922)

[4.4. Achievements of Academic Objectives 22](#_Toc38885923)

[4.5. Achievement of Requirements 23](#_Toc38885924)

[4.5.1. Achievement of Functional Requirements 23](#_Toc38885925)

[4.5.2. Achievement of Non-Functional Requirements 24](#_Toc38885926)

[4.6. Milestones Deliverables 24](#_Toc38885927)

[4.7. Problems and Challenges Faced 25](#_Toc38885928)

[4.8. Learning Outcomes 25](#_Toc38885929)

[4.9. Future Enhancements 26](#_Toc38885930)

[4.10. Chapter Summary 26](#_Toc38885931)

[References 27](#_Toc38885932)

# List of Tables

[Table 1 Testing the functional requirements 17](#_Toc38885949)

[Table 2 Performance Testing 18](#_Toc38885950)

[Table 3 Compatibility testing 18](#_Toc38885951)

[Table 4 Achievements of operational objectives 22](#_Toc38885952)

[Table 5 Achievements of academic objectives 23](#_Toc38885953)

[Table 6 Achievement of Functional Requirements 23](#_Toc38885954)

[Table 7 Achievement of Non-Functional Requirements 24](#_Toc38885955)

[Table 8 Milestones Deliverables 25](#_Toc38885956)

# Table of figures

[Figure 1 Technology Stack 7](file:///C:\Users\Akila%20Nanayakkara\Desktop\Paddy%20Weed%20Detector%20-%20Hypesters.docx#_Toc38885933)

[Figure 2 Data Set gray scaling and resizing 8](#_Toc38885934)

[Figure 3 Imports for pre-processing 8](#_Toc38885935)

[Figure 4 Extracting the weights to two pickles 9](#_Toc38885936)

[Figure 5 Imports for training the model 9](#_Toc38885937)

[Figure 6 Calling our pre created pickles to train the model 10](#_Toc38885938)

[Figure 7 Model training and saving the model 10](#_Toc38885939)

[Figure 8 CNC Model Explanation (Saha, 2018) 11](#_Toc38885940)

[Figure 9 Model testing in the console 11](#_Toc38885941)

[Figure 10 Input page 1 Angular 13](#_Toc38885942)

[Figure 11 Input page 2 Angular 14](#_Toc38885943)

[Figure 12 Input page Components 14](#_Toc38885944)

[Figure 13 Code to get the image 15](#_Toc38885945)

[Figure 14 Code to show the added image 15](#_Toc38885946)

[Figure 15 Classifying the input image as weed 15](#_Toc38885947)

[Figure 16 Classifying the input image as paddy 16](#_Toc38885948)

# Implementation

## Chapter Overview

This chapter elaborates about the implementation and elaborates about the selection of technologies and the reason behind it. How the design discussions transformed to code, libraries, datasets used are documented. Languages, IDEs and the version control system are also described. Important code snippets are shown and explained. Then the implementation of components is discussed. Finally, screenshots of the system are given after which, the chapter concludes.

## Overview of the Prototype

Over idea was to classify the weed “echinochloa crus-galli” in Sri Lanka commonly known as “maha maruk” from the basic paddy, the reason for that is that weed is little similar to the normal weed. We created an image classification system to do that. When we insert an image of a paddy or a weed, our classification model will classify whether it is “maha maruk” weed or a normal paddy. Also, when we insert an image which was taken from a camera where the location function is available. We can extract the location from the image and show it from the maps.

## Selection of Technologies

### Programming language for data science part

When considering image processing systems python runs a major task in the current world among the competitors like Java, C, C++ etc. So in this case no point of deciding which language is to be used. Python takes a major part because it is an appropriate and timely choice for image processing. This is because of the availability of many states of Image processing tools in its ecosystem. Tools such as Numpy, OpenCV play a major role.

Machine learning                               : CNN (Convolutional Neural Network), TensorFlow,

Keras Data analysis and visualization   : NumPy, SciPy

* Backed up with extremely good documentation and community support
* Easier to build models (CNN ) compared to other
* Existing literacy with the members, are some reasons to select python as the programming language.

### Image Processing Library

In python there are so many libraries: pillow, SciPy, OpenCV etc. after doing a big study it was decided to use OpenCV. OpenCV-Python is a Python wrapper for theOpenCV python implementation. All the OpenCV array structures are converted to Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib. Which we didn’t use.

TensorFlow is used as the framework to build and train the module. It provides an excellent service rather than other frameworks. TensorFlow has a high-level API name Keras, which is used by the selected module (CNN).

For image processing there are so many libraries. For this project we have use OpenCV, TensorFlow, Keras, NumPy and Matplotlib as the libraries for the data science part.

• OpenCV is a free open source library that is used in image processing. We have used the OpenCV library for image preprocessing. We should preprocess the image before using it for the model training processes. We have used the OpenCV for resize the image and turn the color of the image to gray.

• Keras is an open source neural network library written in python. It is capable of running on top of the TensorFlow. It wraps the efficient numerical computation library and allows you to define and train neural network models in just a few lines of code

• TensorFlow is an open source artificial intelligence library, using data flow graphs to build models. It allows developers to create large-scale neural networks with many layers. TensorFlow is mainly used for: Classification, Perception, Understanding, Discovering, Prediction and Creation.

• NumPy is a library for python programming language. By this library we can convert the digital image to a dimensional array. This dimensional array become the source to the processing part.

• Matplotlib is a plotting library for the python programming language and its numerical mathematics extension NumPy.

### Web Frameworks Selection

Since our proposed solution contains a web application which interact with the user and the backend which deploy the model and handle the application logic, in our case it is pre-processing the input image. Web framework will handle the interaction between frontend and backed other than implement everything from scratch. For python there are two web frameworks to consider, Flask and Django. We used Django for our development and its Rest Api to pass the images from front end to back end.

## Selection of tools

### Selection of IDE

JetBrains PyCharm, sublime text, Programiz most known IDEs for Python development. JetBrains PyCharm was decided to be used as the IDE, because of following reasons

* It has provided us with a powerful solution for a data science program in python language
* Existing experience of the members in the team

Also, for the front-end angular development, we used the Visual Studio Code as code editor.

### Selection of the version control system

Git was decided to be used as the Version Control System,

.

* Changes can be committed offline, pushed when it is convenient, and the history can    be reverted easily.
* Existing experience of the author.

Above reasons was responsible to use Git

## Technology Stack

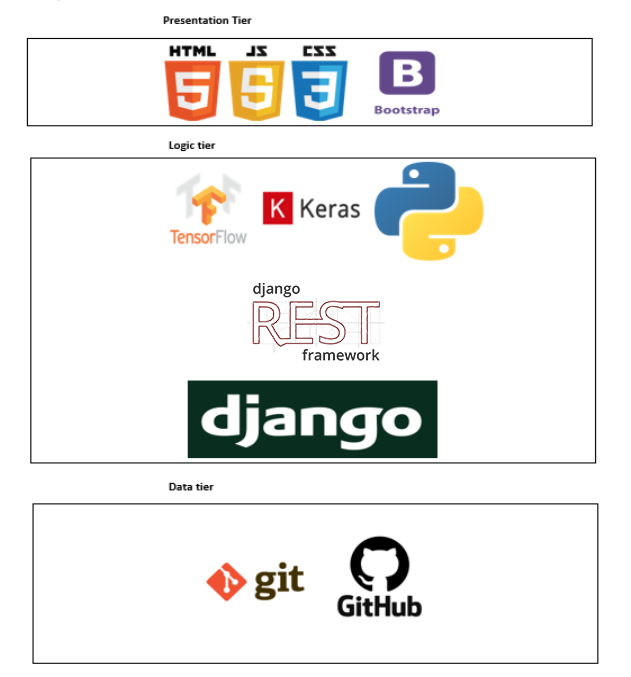


Figure 1 Technology Stack

## Data Set

Any image processing project starts with images. Images with low quality will result in the program to malfunction and not being able to develop. Thus, the very first question was if there are any suitable datasets available for the system to be built.

Since we are specializing in weed, we had a hard time gathering this dataset. Image set with above thousand images and all with quality images wasn't easy. We got an image set from <https://www.kaggle.com/>. The data set was made by combining five to six dataset images. Some camera images were taken to test and train the module.

## Implementation of the features of the prototype in the back end

### Data set Pre-Processing

When we get an image data set, before training the model we should preprocess those images. The reason is there can be various types of images and various size of images. So, before we train them into a model, we should make them all similar to a particular size and color and then we should extract the data from those images.



Figure 2 File path to the data set

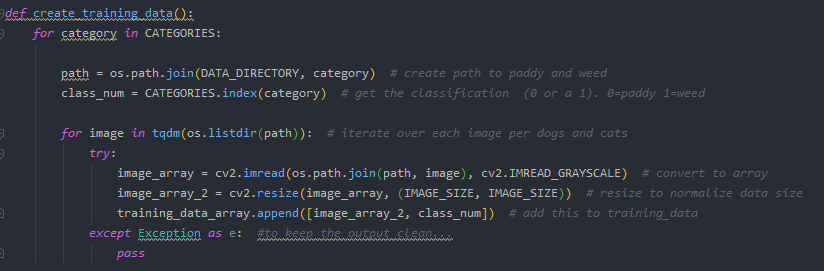


Figure 3 Data Set gray scaling and resizing

In the above code, we take the image data set and then we categorize them as paddy and weed and then we put them into an array. After that we convert those images to gray scale for the easiness of the image data extraction and then we resize all those images to 50\*50.

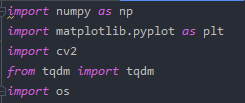


Figure 4 Imports for pre-processing



Figure 5 Extracting the weights to two pickles

In the above code first, we randomize the images and the we reshape our images and then we create two arrays and then we insert our weights of each image to the particular array according to the category. After that we save those two arrays as two pickles to be used in our model training.

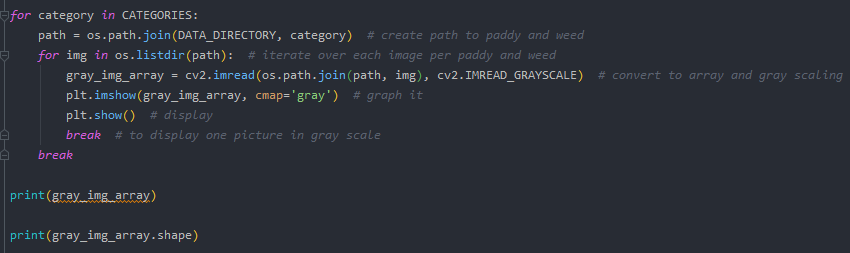


Figure 6 Preview of gray sailing

### Training the model

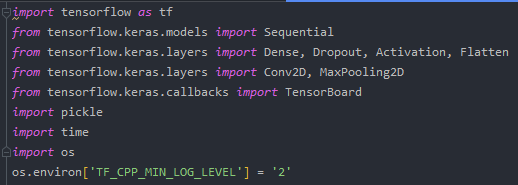


Figure 7 Imports for training the model

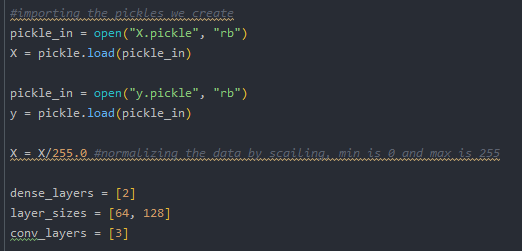


Figure 8 Calling our pre created pickles to train the model

In this code we import the pickles we previously saved and then we normalize the them. After that we create our required layers to train the model.

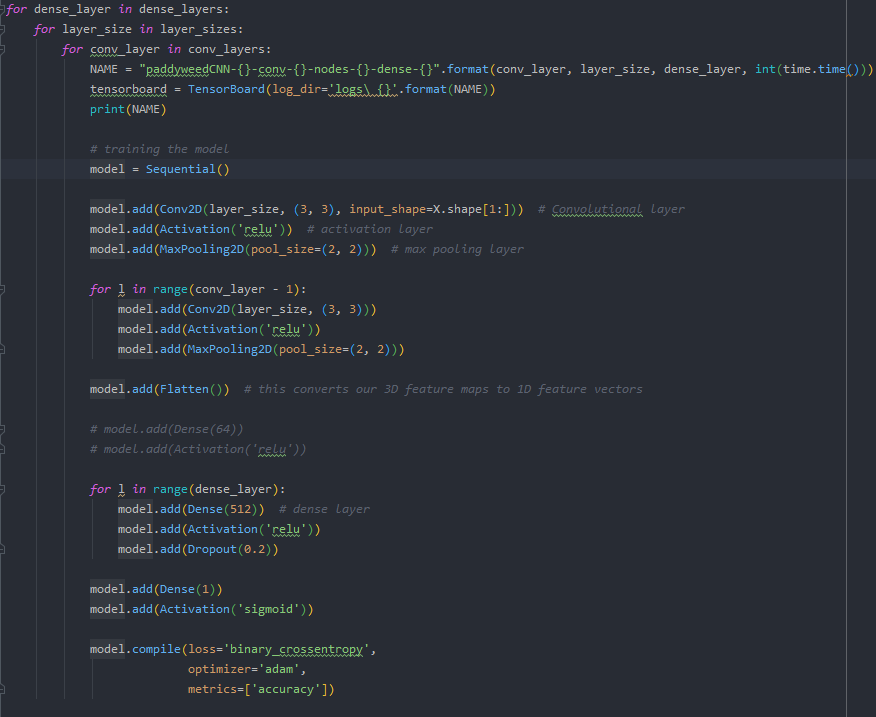


Figure 9 Model training and saving the model

Here according to our layers, we create our model and then save it as in .h5 format.

### Selected Algorithm

We used CNN algorithm to train our model.

A picture containing text, map

Description automatically generated

Figure 10 CNC Model Explanation (Saha, 2018)

A Convolutional Neural Network (**CNN**) is a Deep Learning **algorithm** which can take in an input image, assign learnable weights and biases to various objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics. (Saha, 2018)

### Testing the model

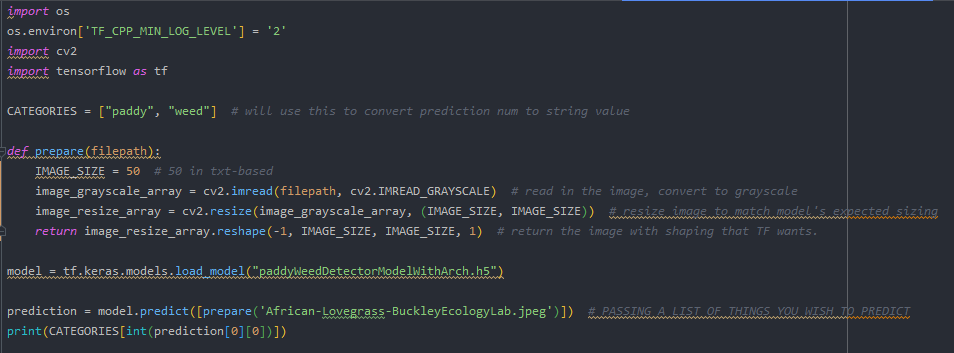


Figure 11 Model testing in the console

In the above snippet we created a console app to test our classification model. In the first place we get the image that we want to classify. Then we pre-processed the image to insert to the model. Then we load the model to our console and then we feed pre-processed image to the model. Finally, the model will give an output saying if the input image is a weed or a paddy.

A close up of a plant

Description automatically generated

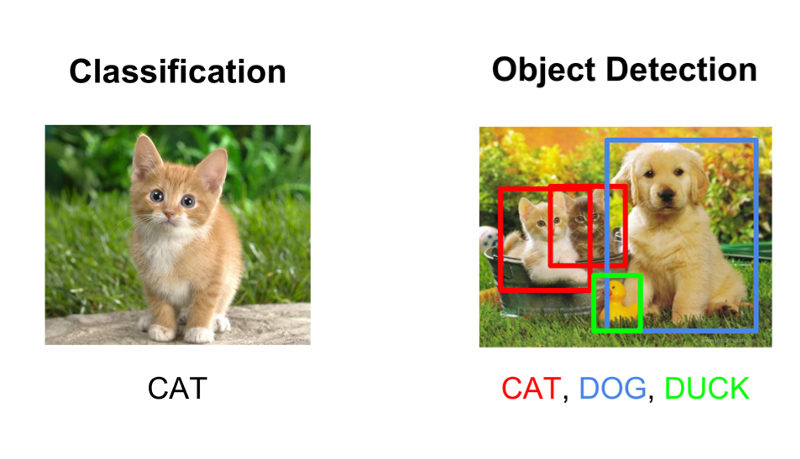
A screenshot of a cell phone

Description automatically generated



But there is exception for our model. In the first place our model is not an image detection model. It’s an image classification model. It means we train our model to specific sets of images. It will get their weights and train. So, if we input a non-related image still our image classification will tell it’s a paddy or a weed. The reason is some of the weight files can be there in the non-related image too. There is a work around for this. First, we should make an image detection model and then pass the detected image to the classifier. We didn’t go through that because our prototype was about the image classifier.





## Implementation of the GUI

The Angular framework was used to build the web application. Angular modules and Bootstrap stylings are included to come up with a high responsive beauteous web application. This web application contains one component called “First-page”. All the relevant sections are included in this component.

### Implementation of the input page

These figures declare the view of the GUI which was implemented for the Paddy Weed Detector.

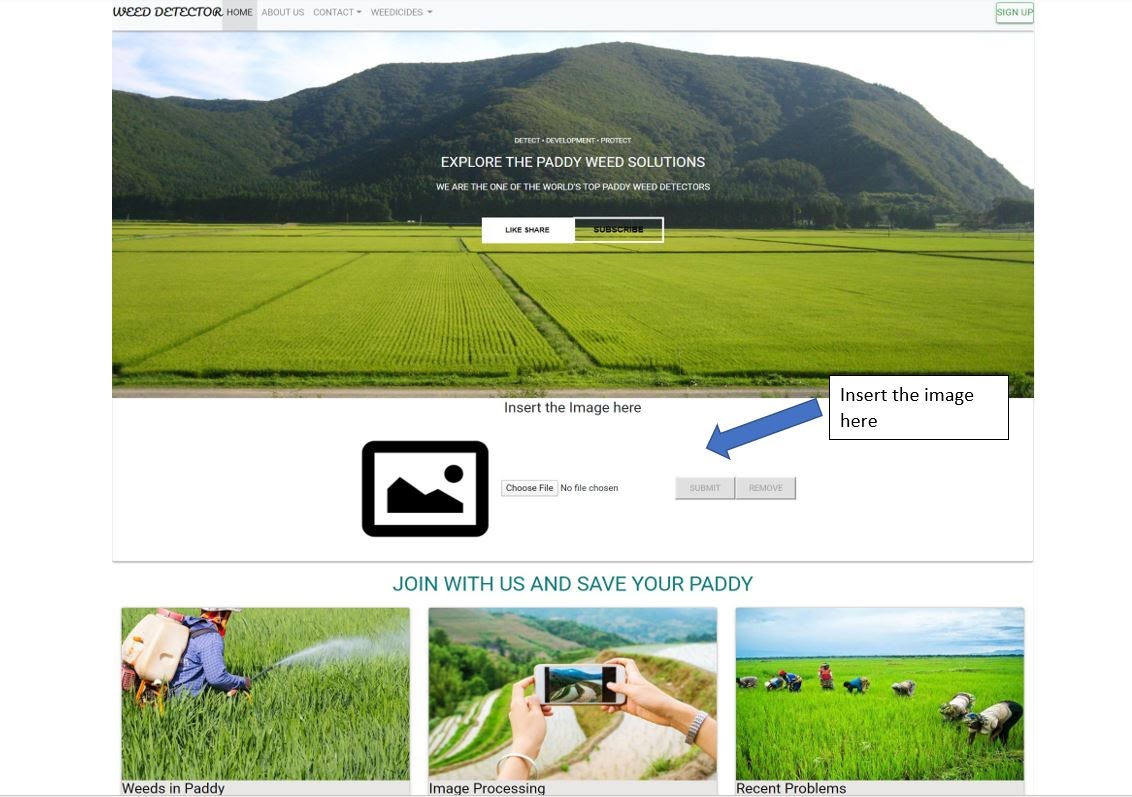


Figure 12 Input page 1 Angular

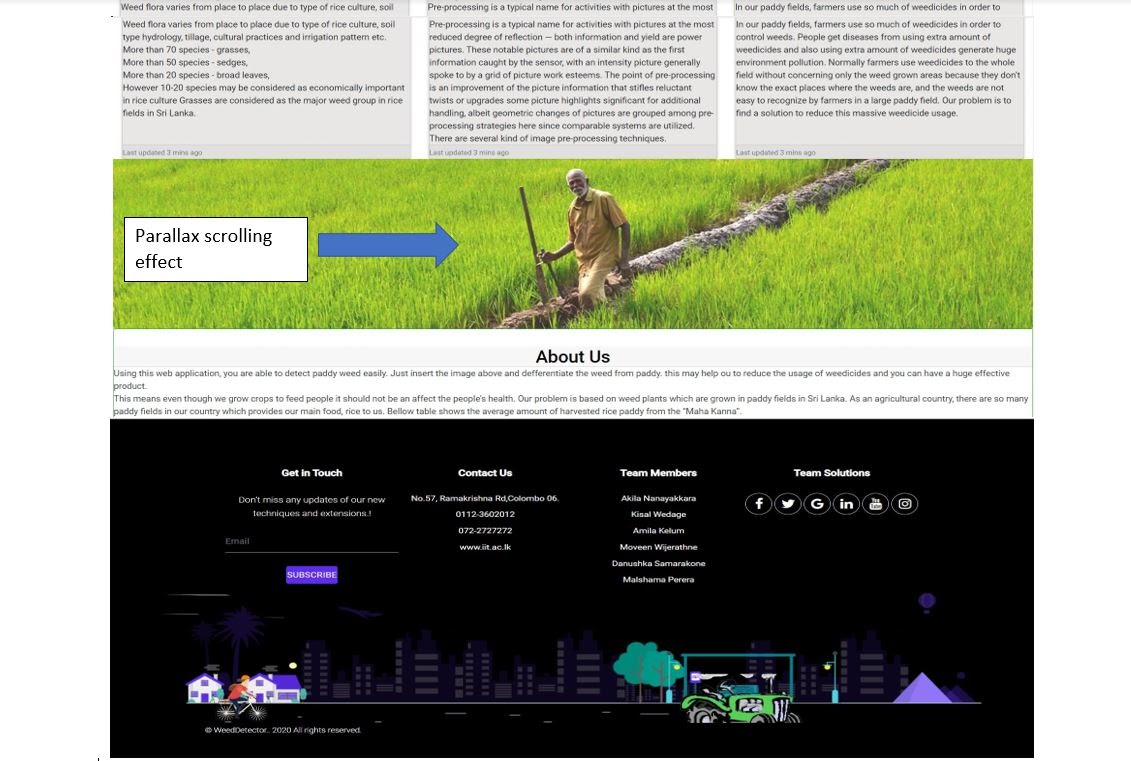


Figure 13 Input page 2 Angular

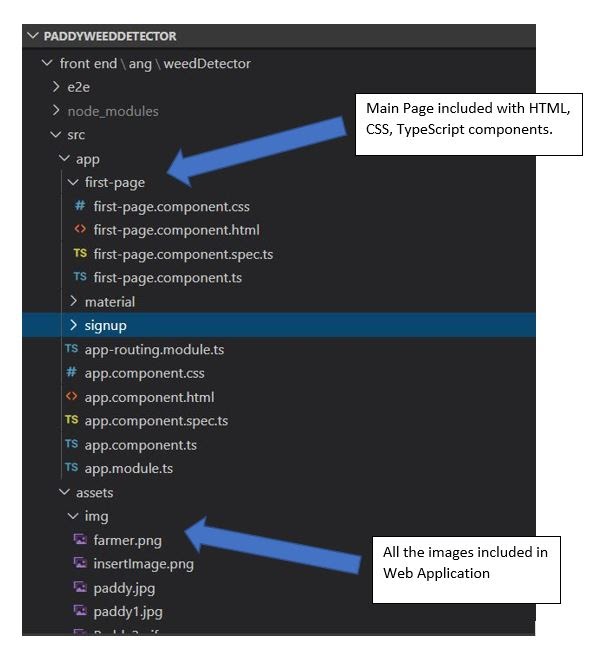


Figure 14 Input page Components

The following figure shows the components that are used for constructing the frontend using angular Js.

The following html and css code snippet declares the image inserting part in the front end.



Figure 15 Code to get the image



Figure 16 Code to show the added image

### Implementation of the output page

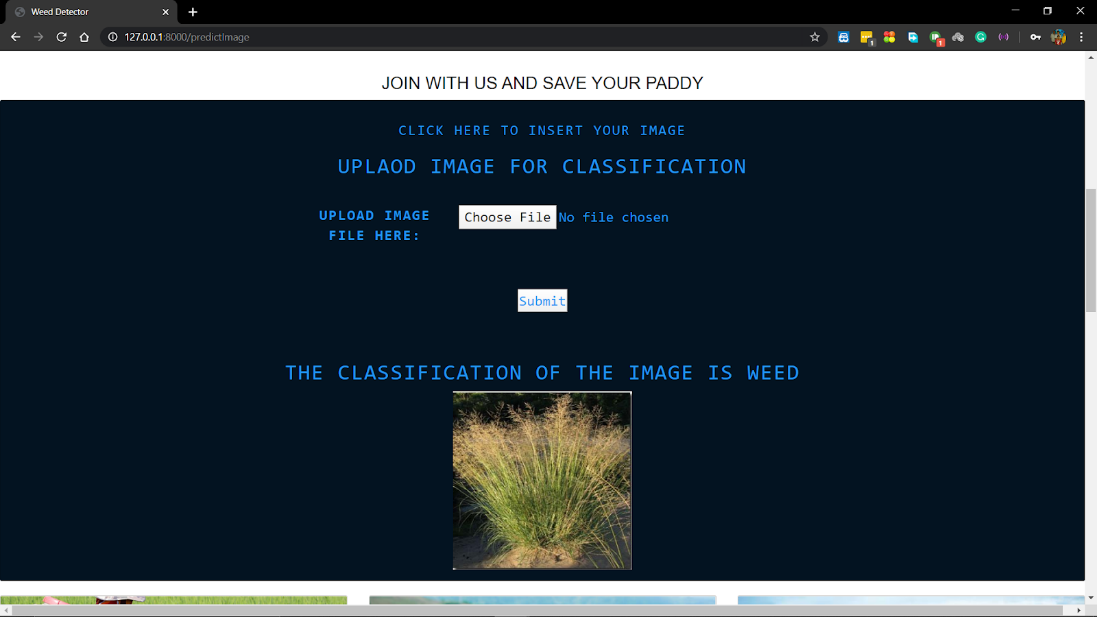


Figure 17 Classifying the input image as weed

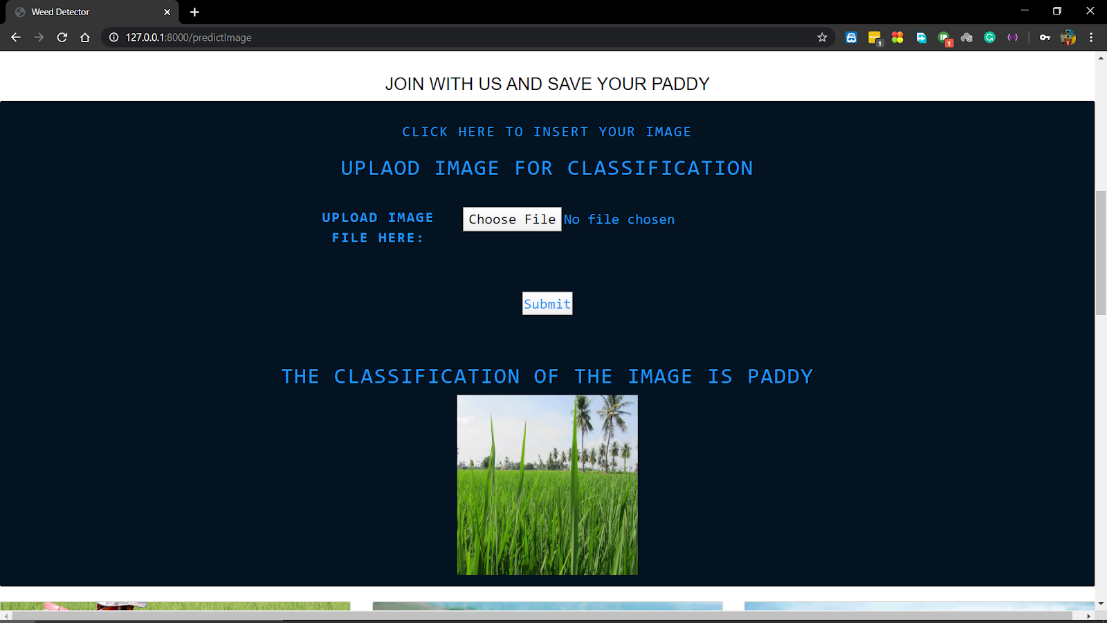


Figure 18 Classifying the input image as paddy

## Combining the back end and the front end

## Chapter Summary

This chapter began with the technologies that are being used, libraries and the tools. The code snippets and the output snippets were provided when it is needed when the functionality is discussed.

# Testing

## Chapter Overview

## Testing Goals and objectives

## Testing

## Testing Criteria

## Testing Functional Requirements

|  |  |  |
| --- | --- | --- |
| ID | Functional Requirements | Status |
| FRT01 | The user is able to put the image into the system. | Success |
| The system must have an option to get the input image from the user. | | |
| FRT02 | The system is able to detect the weed in the paddy field. | Success |
| Main target of the system is to detect the specific weed that is on the image. | | |
| FRT03 | The system is able to give the exact location where the weeds are. | Success |
| FRT04 | The system should give the exact location where the weeds are. | Success |
| One of the features of our program is to show the location where the weeds are. To do that we will extract the location from the image and show it to the user. | | |
| FRT05 | Identify other weed plants other than the “MahaMaruk” weed plant. | Success |

Table 1 Testing the functional requirements

## Performance Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Test Case | Input | Expected Output | Actual Output | Status |
| PT01 | Loading the front end Page | N/A | Less than 25s. | 24.634s. | Pass |
| PT02 | Validate whether input is successful or not. | Paddy Image | Less than 25s. | N/A | Fail |
| PT03 | Generate the Final Result | Paddy Image | 25s. |  | Pass |
| PT04 |  |  |  |  |  |

Table 2 Performance Testing

## Usability Testing

## Compatibility Testing

|  |  |
| --- | --- |
| Browser | Compatible |
| Chrome | Yes |
| FireFox | Yes |
| Edge | Yes |

Table 3 Compatibility testing

## Chapter Summary

# Evaluation

## Chapter Overview

# Evaluation Goals

## Evaluations Benchmark

## Evaluations Selections

## Concept and Scope

## Technical Aspects

## Impacts

## Limitations

## Research question

## Self-Evaluation

## Chapter Summary

# Conclusion

## Chapter Overview

This chapter covers the achievements of the project. The chapter firstly discusses the project aims and objectives. Further, the problems faced, how the existing knowledge of the members utilized with the degree program. Limitations of the projects are outlined followed by the learning outcomes of the project. The chapter proposes the future enhancements.

## Achievement of the Project Aim

“This project aims to build up a system to recognize the weed plants in-between the paddy and to generate the map with location showing where weeds are.”

The aim of the project was successfully achieved during the development time. System that detects the weed and maintaining was successfully designed, developed and evaluated. With the original camera photographs with higher quality images made possible with the system.

## Achieveemnts of Operational Objectives

|  |  |
| --- | --- |
| Description | status |
| **Validation of the idea and the introduction**  To gather proper knowledge about the problem.  1. Research on the problem domain.  2. Get the idea validated according to the problem domain.  3. Conducting interviews with domain experts.  4. Distributing a questionnaire to validate the problem. | **Completed** |
| **Literature Review**  Researching about the problem and its existing solutions.  1. Discovering the existing methods to solve the problem.  2. Comparing the different approaches to solve the problem. | **Completed** |
| **Choosing our approach**  With the help of the literature review, find out the best approach that we can solve the problem. | **Completed** |
| **Requirement gathering**  Gather user requirements mainly to come up with functional and non-functional requirements which are necessary. | **Completed** |
| **Designing the system**  The basic domain models, onion diagram, use case diagrams and its descriptions and class diagrams, use case diagrams. This is the basic overview of the system which anyone can understand easily. | **Completed** |
| **SRS**  Preparing the SRS document with the help of objective three and requirements engineering process. | **Completed** |
| **Proof of Concept**  Initial development stage of the implementation which will include the core functionality and other functionalities depending on what time permits with the help of the best software and hardware which were validated on previous objectives. | **Completed** |
| **Test and evaluate the Proof of Concept**  1. Write test cases.  2. Run the tests on the implementation.  3. Record the outputs.  4. Validate them and check if it meets the requirement. | **Completed** |
| **Build up the Proof of Concept and complete the implementation.** | **Completed** |
| **Test and evaluate the implementation.** | **Completed** |
| **Complete the final documentation.** | **Completed** |
| **Conclude the project and submit.** | **Completed** |

Table 4 Achievements of operational objectives

## Achievements of Academic Objectives

|  |  |
| --- | --- |
| Description | Achievement |
| **Using data science and applying image processing**.  Applying python open cv to do the image processing to identify the weed plants from the paddy field. | **75%** |
| Learning python, angular js, weka and matlab and using them for our project. | **80%** |
| Learning designing, developing, testing and evaluating | **100%** |
| Practicing the software development life cycle  To learn and experience the full SDLC and how to manage a software development project. This will give us exposure in all aspects of software development such as requirement gathering, analysis, design, implementation, testing, deployment and maintenance. | **100%** |

Table 5 Achievements of academic objectives

## Achievement of Requirements

### Achievement of Functional Requirements

|  |  |  |
| --- | --- | --- |
| Requirement | Description | Achievement |
| The connection between the system and the dataset. | As this system is based on image processing and data science, the system must always have the connection with the dataset. | 100% |
| The user must be able to put up the drone and snap the paddy field. | User must capture a picture using a drone. The image quality should be at a certain good level. | 100% |
| The user should be able to put the snap into the software. | The system must have an option to get the input image from the user. | 100% |
| The system should detect the weed in the paddy field. | Main target of the system is to detect the specific weed that is on the image. | 100% |
| User Log-in | - | 0% |
| Identify other weed plants other than the “Maha-Maruk” weed plant. | - | 0% |

Table 6 Achievement of Functional Requirements

### Achievement of Non-Functional Requirements

|  |  |
| --- | --- |
| Requirements | Achievement |
| The software should load up soon and then identify the weed quickly. | 50% |
| Identifying the weed should be accurate. | 80% |
| The system should improve when it gets more and more images | 90% |
| The system is a web application, it can be accessed through a laptop from any location, when the user has a stable internet connection. | 0% |
| The system will keep the photos that farmers insert to the system. In the backend, we can use those images and train our system to be more accurate. | 100% |

Table 7 Achievement of Non-Functional Requirements

## Milestones Deliverables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Milestone/Deliverable | Completed | End Date | Remarks |
| 1 | Project Initiation Phase | 100% | 15/11/19 | Project initiation document (idea pitching) was done on time |
| 2 | Literature Review Phase | 100% | 01/01/20 | Research reports were referred during the phase. Submitted the final report 1st stage report on time |
| 3 | Requirements Gathering  Phase | 100% | 10/02/20 | This phase was started and finished on time |
| 4 | Design Phase | 100% | 09/03/20 | This phase finishes on time |
| 5 | Implementation Phase | 100% | 25/4/20 | Due to various circumstances it was unable to start this phase on time. |
| 6 | Testing and Evaluation  Phase | 100% | 26/4/20 | Due to the delay of the above phase testing  and evaluation phase was in a hurry |
| 7 | Project Closure | 100% | 26/4/20 | Unprepared delys pay the price to rush closure |

Table 8 Milestones Deliverables

## Problems and Challenges Faced

* 1. Limited time
     + The project nearly carried a year. The time frame was highly challenging to get a better project.
  2. Unrest in the country
     + Due to the covid-19 virus curfew was lifted, continuous power cuts on some areas in the country. To gain a better outcome team meetup (physical meetups) is key. It was a challenge to the members to communicate and implement the project through online. This was a major challenge that affected the project.
  3. Lack of expertise
     + It was hard to find the correct and the most accurate solution for an error. This is because of the scarcity of machine learning experts.
  4. Implementation of the data science part
     + Data science part was asked to submit earlier than the project deadline. Rush coding was needed to survive the challenge. This was harder because the library was not familiar. Had to do massive research before handling the libraries.

## Learning Outcomes

The Learning outcomes that were obtained are mentioned below.

* Skills such as time management, problem solving, analytical thinking and team communication were improved.
* An outstanding improvement has been seen in academic writing and formal report writing abilities.
* Skills such as best practices in coding, commenting, design and architectural style have been improved precisely.
* Totally new knowledge was gained through this project, such as new libraries like keras, tensorflow, django.

## Future Enhancements

* The system should be able to detect or locate other weed types or even few more
  + Priority -  High
* The system can be developed for detecting more weeds other than “Mahamuruk” if there were more data sets.
  + Priority -  High
* The system should be run on mobile devices, such as IOS, android with a minimum time taken.
  + Priority - Medium
* The system with user login, with user information database should be considered
  + Priority - low

## Chapter Summary

This chapter concludes the report by discussing the aim and objectives which are in the very first chapter. How the requirements of the system has been satisfied is being discussed. Then the Milestones and Deliverables have been mentioned with an ending date and the problems and the challenges met are being mentioned and discussed. Learning outcomes of the project are noted following, finally with the future enhancements are documented.

# References

Saha, S., 2018. *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way.* [Online]   
Available at: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53  
[Accessed 27 04 2020].