

A PROJECT REPORT ON
CROPIFY – CROP RECOMMENDATION SYSTEM

*Mini Project submitted in partial fulfillment of the requirements for the
award of the degree of*

**BACHELOR OF TECHNOLOGY
IN
INFORMATION TECHNOLOGY
(2018-2022)**

BY

Dhavaleswarapu Anudeep	18241A1215
Potluri Sai Sampath	18241A1249
Tejaswi Krishna Kosaraju	16241A12B1
P Parasuram	18241A1244

Under the esteemed guidance of

V. Padma

Associate Professor, Department of IT



**DEPARTMENT OF INFORMATION TECHNOLOGY
GOKARAJU RANGARAJU INSTITUTE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)
HYDERABAD**



CERTIFICATE

This is to certify that it is a bonafide record of Mini Project work entitled “**Cropify – Crop Recommendation System**” done by **D. Anudeep (18241A1215), P. Sai Sampath (18241A1249), K. Tejaswi Krishna (16241A12B1), P. Parasuram (18241A1244)**, students of **B.Tech (IT)** in the Department of Information Technology, Gokaraju Rangaraju Institute of Engineering and Technology during the period of 2018-2022 in the partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY** from GRIET, Hyderabad.

V. Padma
(Internal Project Guide)

Dr. K. Prasanna Lakshmi
(Head of the Department)

(Project External)

ACKNOWLEDGEMENT

We take immense pleasure in expressing gratitude to our internal guide **V. Padma, Associate Professor**, Information Technology, GRIET. We express our sincere thanks for her encouragement, suggestions, and support, which provided the impetus and paved the way to the successful completion of the project work.

We wish to express our gratitude to **Dr. K. Prasanna Lakshmi, P. Gopala Krishna**, our Project Co-ordinators **K. Archana** and **K. Swanthana**, for their constant support during the project.

We express our sincere thanks to **Dr. Jandhyala N Murthy**, Director, GRIET, and **Dr. J. Praveen**, Principal, GRIET, for providing us the conducive environment for carrying through our academic schedules and project with ease.

We also take this opportunity to convey our sincere thanks to the teaching and non-teaching staff of GRIET College, Hyderabad.



E-mail: anudeepd2@gmail.com
Contact No.: 8074431924



E-mail: saisampathpotturi@gmail.com
Contact No.: 9704431099



E-mail: 16241a12b1@griet.ac.in
Contact No.: 9177317343



E-mail: 18241a1244@griet.ac.in
Contact No.: 6302616792

DECLARATION

This is to certify that the project entitled “**Cropify – Crop Recommendation System**” is a bonafide work done by us in partial fulfilment of the requirements for the award of the degree **BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY** from Gokaraju Rangaraju Institute of Engineering and Technology, Hyderabad.

We also declare that this project is a result of our own effort and has not been copied or imitated from any source. Citations from any websites, books and paper publications are mentioned in the Bibliography.

This work was not submitted earlier at any other University or Institute for the award of any degree.

<i>Dhavaleswarapu Anudeep</i>	<i>18241A1215</i>
<i>Potluri Sai Sampath</i>	<i>18241A1249</i>
<i>Tejaswi Krishna Kosaraju</i>	<i>16241A12B1</i>
<i>P Parasuram</i>	<i>18241A1244</i>

TABLE OF CONTENTS

SERIAL NO.	NAME	PAGE NO.
	Certificate	ii
	Table of Contents	v
	Abstract	vii
1	INTRODUCTION	1
1.1	Introduction to Project	1
1.2	Existing System	2
1.3	Proposed System	2
2	REQUIREMENT ENGINEERING	3
2.1	Hardware Requirements	3
2.2	Software Requirements	3
3	LITERATURE SURVEY	4
4	TECHNOLOGY	5
5	DESIGN REQUIREMENT ENGINEERING	11
5.1	Use case diagram	11
5.2	Class diagram	12
5.3	Sequence diagram	13
5.4	Activity diagram	14
5.5	Architecture	15
6	IMPLEMENTATION	16
6.1	Machine learning module	16

6.1.1	Dataset	16
6.1.2	Data exploration and pre-processing	17
6.1.3	Splitting the data for training and testing	18
6.1.4	Training various models	19
6.1.5	Comparison of accuracy	23
6.1.6	Prediction	23
6.1.7	Dumping the model	24
6.2	IoT module	24
6.3	Web development module	26
7	SOFTWARE TESTING	30
7.1	Unit testing	30
7.2	Integration testing	30
7.3	System testing	30
7.4	Acceptance testing	30
8	RESULTS	32
9	CONCLUSION AND FUTURE ENHANCEMENTS	36
10	BIBLIOGRAPHY	37

ABSTRACT

IoT is the most trending technology used in this decade. Machine learning is powered by data and generates insight from IoT. Machine learning uses this data to identify patterns and builds models that help predict future behaviour and events.

The main idea of this project is to predict the suitable seasonal crop based on N, P, K levels, temperature, humidity, pH, and soil moisture data. We make use of Random Forest ML algorithm to predict the results. The humidity, soil moisture and temperature levels are gathered using the sensors. This is integrated along with an Arduino board. The setup is installed in the farm and the data is gathered in real time which is used for further processing. The remaining parameters are entered by the user. We use a predefined dataset to train the model. The current data from the sensors can now be passed to the model which predicts the crop. A website is designed built using Django framework which displays the result. The users can easily understand what crop to cultivate to produce maximum yield.

1. INTRODUCTION

1.1 Introduction to Project

Agriculture has a major role in the lives of every individual. From the olden times itself agriculture is one of the main practices in India. But as the years progressed, to meet the needs of the growing food requirement, the methods became unnatural. As a result of this the soil's fertility has reduced. Choosing the wrong crop and using fertilisers can cause even more damage.

Precision agriculture is in trend nowadays. Precision agriculture is a modern farming technique that uses the data of soil characteristics, weather conditions and suggests the farmers with the most optimal crop to grow in their farms for maximum yield and profit. This technique can reduce the crop failures and will help the farmers to take informed decision about their farming strategy.

To decrease the burden of losses, there is a need for a better recommendation system. This is where our project comes into picture. The data used in this project is made by augmenting and combining various publicly available datasets. You can access the dataset [here](#). This dataset is simple with only the useful features affecting the yield of the crop. The data has Nitrogen, Phosphorous, Potassium, pH, moisture values of the soil & humidity, temperature required for a particular crop.

Simple soil analysis will get us the values of N, P, K, and pH levels of the soil. We gather temperature, humidity and soil moisture using the respective sensors to suggest the best suitable crop. This method of precision agriculture helps the farmers to make an informed decision.

1.2 Existing System:

The existing systems fail to consider all the parameters at once. Most of these systems just consider the N, P, K levels and do not consider the other important parameters like soil moisture, temperature, and humidity. Because of this, the results might be inaccurate. Some of the existing systems tend to use LORA to transmit the information but it is costly and needs technical skill to setup when an error occurs. Existing systems use SVM or Decision Tree algorithm that has an accuracy comparatively less as it has overfitting and high variance.

1.3 Proposed System:

The proposed system tries to minimise the drawbacks of existing system. It considers N, P, K levels, pH, soil moisture, temperature, and humidity. Since all the vital parameters are being considered, the resultant system will be more accurate. It is cheaper and easy to setup. The replacement is also easy. The parameter transmission delay is reduced. It does not need much technical skills to setup. It uses Random Forest classifier which reduces the problem of high variance and offers high accuracy.

2. REQUIREMENT ENGINEERING

2.1 Hardware Requirements

- Arduino Uno
- Soil moisture sensor
- DHT11 sensor
- Jumper wires
- Processor – Intel i3 (sixth generation or newer)
- Memory – 4GB RAM (recommended)

2.2 Software Requirements

- Arduino IDE
- Django
- HTML + CSS
- Visual Studio Code
- Bootstrap
- Windows 10
- Python

3. LITERATURE SURVEY

Coriander, pulses, cotton, paddy, and other crops were included in the model for prediction. Sorghum, peanuts, sugarcane, bananas, and vegetables are some of the crops grown. In this study, many soil characteristics were considered. pH, depth, erosion, permeability, texture, drainage, and water holding were all factors in predicting the crop and the hue of the soil ensembling was the strategy adopted, which combined the force of using two or more people. For improved prediction, multiple models are used. The method of assembly utilised was known as Majority Voting Technique ^[1]

The crops were examined and graded based on the results of the examination to estimate crop yields. Different data mining techniques have discovered this categorization. This paper gives an overview of several grouping rules, including K-Nearest Neighbor and Naive Bayes. We analysed the categorization rules and determined which ones will match the data set we will be using in our project by using this document ^[2]

4. TECHNOLOGY

4.1 ABOUT PYTHON

Python is a programming language, which means it can be understood by both humans and computers. Python was designed by Guido van Rossum, a Dutch software engineer who wanted to overcome some of the difficulties he encountered in other computer languages at the time.

Python is a high-level computer language for general-purpose programming that is interpreted. Python is a programming language created by Guido van Rossum and initially released in 1991. It features a design philosophy that prioritises code readability and a syntax that allows programmers to express concepts in fewer lines of code, with a lot of whitespaces. It has structures that allow for clear programming at both local and large sizes.

Python has a dynamic system with memory management that is automatic. It features a big and extensive standard library and supports several programming paradigms, including object-oriented, imperative, functional, and procedural.

For a wide range of operating systems, Python interpreters are available. The reference implementation of Python, C Python, is open-source software with a community-based development process, as do almost all its variant implementations. The Python Software Foundation, a non-profit organisation, oversees C Python.

Python is used extensively in both machine learning and web development. For data visualisation, machine learning, natural language processing, complicated data analysis, and more, Python scientific packages abound. Python is a wonderful tool for scientific computing and a viable alternative to commercial products like MATLAB because of all these qualities. Bottle.py, Flask, CherryPy, Pyramid, Django, and web2py are just a few of the frameworks available in Python. Some of the world's most popular websites, including Spotify, Mozilla, Reddit, the Washington Post, and Yelp, employ these frameworks.

4.2 ARDUINO IDE

The Arduino Integrated Development Environment (IDE) is a cross-platform application written in C and C++ functions for Windows, macOS, and Linux. It's used to write and upload programs to Arduino-compatible boards, as well as other vendor development boards with the support of third-party cores.

There are two primary functions in the programme structure. –

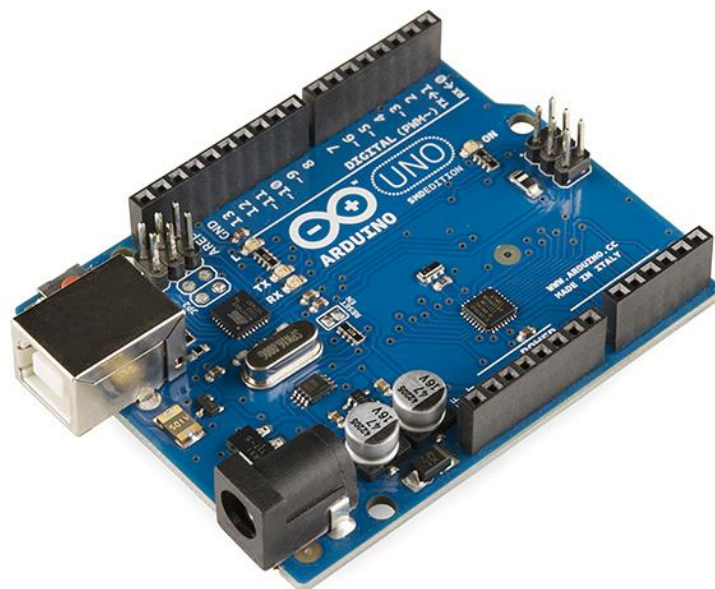
- **setup () function**

- **loop () function**

When a sketch begins, the setup () function is called. Pin modes are used to initialise variables. After each power up or reset of the Arduino board, the setup function will only run once. The loop () function runs indefinitely until it is terminated. Furthermore, the delay () function is used to postpone the Arduino's execution between operations.

4.3 ARDUINO UNO BOARD

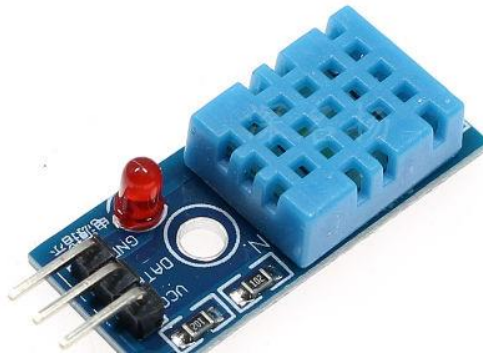
The Arduino Uno is a microcontroller board that uses the ATmega328P. It contains 14 digital input/output pins, 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header, and a reset button. It can be used with a USB cable to connect to a computer or powered by an AC-to-DC adapter or battery.



4.4 DHT11 SENSOR

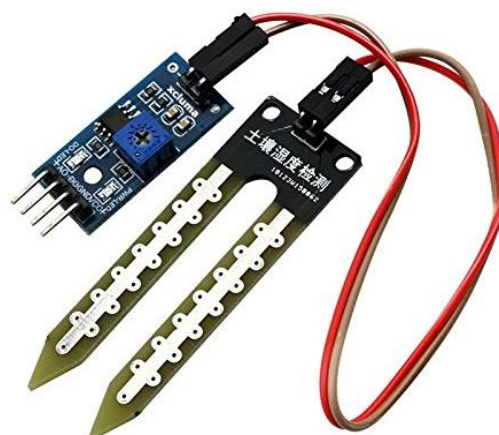
The DHT11 is a temperature and humidity sensor that is widely used. The sensor includes a dedicated NTC for temperature measurement and an 8-bit microcontroller for serial data output of temperature and humidity values. The sensor is factory calibrated, making it simple to connect to other microcontrollers.

The sensor can measure temperature from 0°C to 50°C and humidity from 20% to 90% with an accuracy of $\pm 1^\circ\text{C}$ and $\pm 1\%$.



4.5 SOIL MOISTURE SENSOR

The Soil Moisture Sensor detects variations in earth's electrical conductivity and quantifies soil moisture (soil resistance increases with drought). When a configurable threshold is exceeded, a comparator activates a digital output. Two probes are utilised to measure the volumetric content of water in the soil moisture sensor. The two probes allow current to flow through the soil, and the resistance value is used to calculate the moisture content. When there is more water in the soil, the soil conducts more electricity, resulting in less resistance. As a result, the moisture content will be higher. Because dry soil conducts electricity poorly, when there is less water, the soil conducts less electricity, resulting in increased resistance. As a result, the moisture content will be decreased.



4.6 NUMPY

NumPy is a Python package for array processing. It includes a high-performance multidimensional array object as well as utilities for manipulating them. This is a foundational Python package that adds support for big, multi-dimensional arrays and matrices, as well as a vast library of high-level mathematical methods to operate on these arrays.

4.7 SCIKIT-LEARN

Scikit-learn is a library for machine learning. It includes support vector machines, random forests, gradient boosting, k-means, and DBSCAN, among other classification, regression, and clustering techniques, and is designed to work with the Python numerical and scientific libraries NumPy and SciPy.

4.8 PANDAS

Panda gets its name from the econometric phrase "panel data," which refers to multidimensional structured data sets. It's a data manipulation and analysis library. For managing numerical tables and time series, the library contains data structures and functions. The "Python Data Analysis Library" is another name for it.

4.9 MATPLOTLIB

Matplotlib is a Python 2D plotting package that generates high-quality figures in a range of hardcopy and interactive formats across platforms. Plots, histograms, power spectra, bar charts, error charts, scatterplots, and more are all possible with Matplotlib.

4.10 SEABORN

Seaborn is a Python module for creating statistical visuals. It is based on matplotlib and tightly integrates with pandas data structures. Seaborn assists you in exploring and comprehending your data. Its charting functions work with dataframes and arrays containing entire datasets, performing the necessary semantic mapping and statistical aggregation internally to generate useful graphs. Its dataset-oriented, declarative API allows you to concentrate on the meaning of your charts rather than the mechanics of drawing them.

4.11 JOBLIB

Joblib is a set of Python tools for lightweight pipelining. Transparent disc caching of functions and lazy re-evaluation are two examples (memoize pattern). Parallel computing is easy and straightforward. Joblib contains specific optimizations for NumPy arrays and is built to be fast and robust on huge data. It is released under the BSD licence.

4.12 PYSERIAL

This module isolates the serial port's access. It includes Python backends for Windows, Mac OS X, Linux, BSD (potentially any POSIX compliant system), and IronPython. The "serial" module chooses the suitable backend for you automatically.

4.13 CSV

The CSV (Comma Separated Values) format is the most used spreadsheet and database import and export format. The csv module provides classes for reading and writing CSV tabular data. It enables programmers to say things like "put this data in the format favoured by Excel" or "read data from this file generated by Excel" without having to grasp the specifics of Excel's CSV format. Programmers can also define their own special-purpose CSV formats or describe the CSV formats that other apps understand.

4.14 HTML

HTML, or Hypertext Markup Language, is the industry standard markup language for web-based documents. Additionally, technologies like Cascading Style Sheets (CSS) and computer languages like JavaScript can be beneficial. Web browsers translate HTML documents received from a web server or locally saved files into multimedia web pages. HTML originally included visual clues for the document's appearance and logically represented the structure of a web page. HTML elements are the building blocks of HTML pages. Using HTML techniques, images and other objects, such as interactive forms, can be incorporated in the final page.

4.15 CSS

CSS (Cascading Style Sheets) is a simple system for adding style (such as fonts, colours, and spacing) to web publications. CSS, like HTML and JavaScript, is an important

component of the internet. We may separate presentation from content using CSS, which includes layout, colours, and fonts. Using a separate(.css) file for relevant CSS reduces complexity and redundancy in structure content and allows the.css file to be cached to speed up page load. By delivering the required CSS in a separate file, this separation can improve content accessibility, offer more freedom and control in the specification of presentation characteristics, and allow numerous web pages to share formatting.

4.16 BOOTSTRAP

Bootstrap is a responsive and mobile-first open-source CSS framework for front-end web development. It offers CSS and (optionally) JavaScript design templates for typography, forms, buttons, navigation, and other interface components. After freeCodeCamp (roughly 312,000 stars), Vue.js framework, React library, TensorFlow, and others, Bootstrap is the tenth most starred project on GitHub as of April 2021.

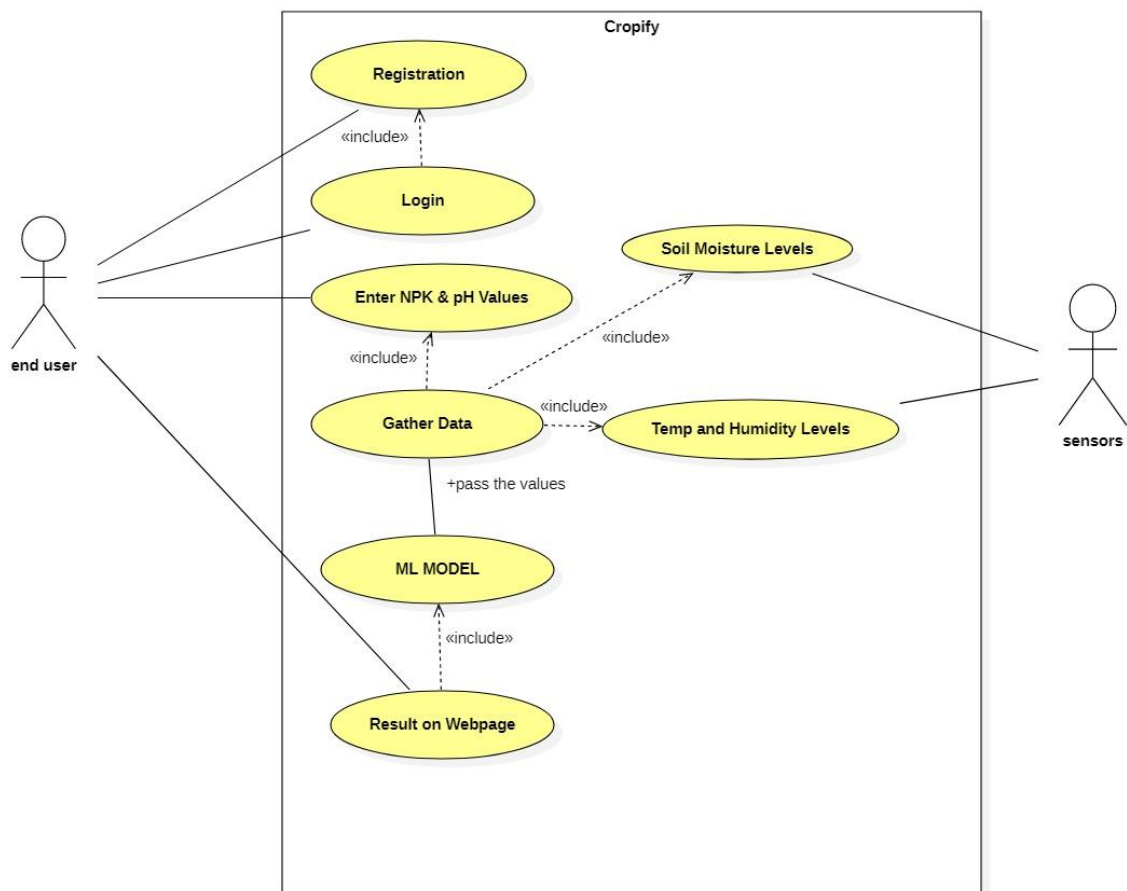
4.17 DJANGO

Django is a Python programming language web application framework. The MVT (Model View Template) design pattern is used. Due to its rapid development characteristic, Django is quite demanding. After gathering customer requirements, it takes less time to construct an application. The tag line for this framework is "The web framework for perfectionists with deadlines." We can construct web applications in a fraction of the time by using Django. Django is structured in such a way that it handles much of the configuration for us, allowing us to focus solely on application development.

5. DESIGN REQUIREMENT ENGINEERING

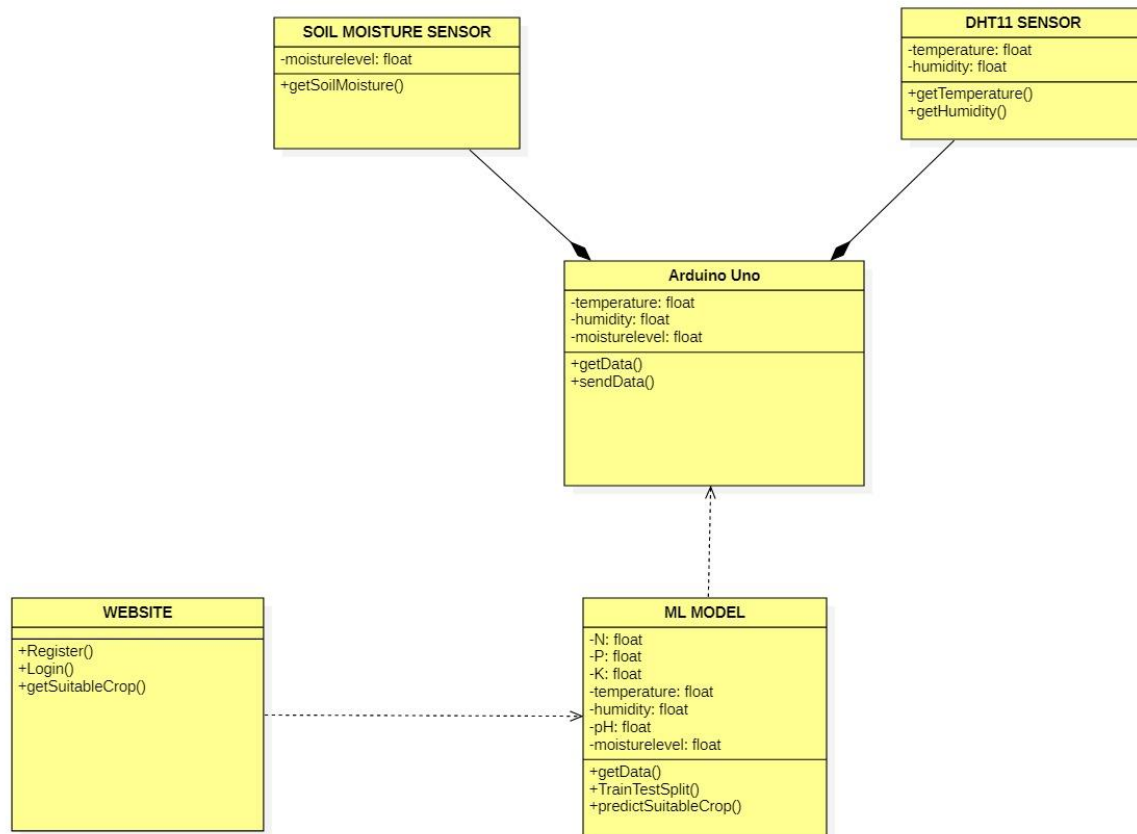
5.1 Use case Diagram

Use case diagram is a behavioural diagram and it is the most widely used UML diagram of all others. It is used for knowing the functional requirements. It consists of actors- primary actor, secondary actor, use case and subject. Primary actor is those who depend on system whereas secondary are those on which system depends. It consists of include and extends features.



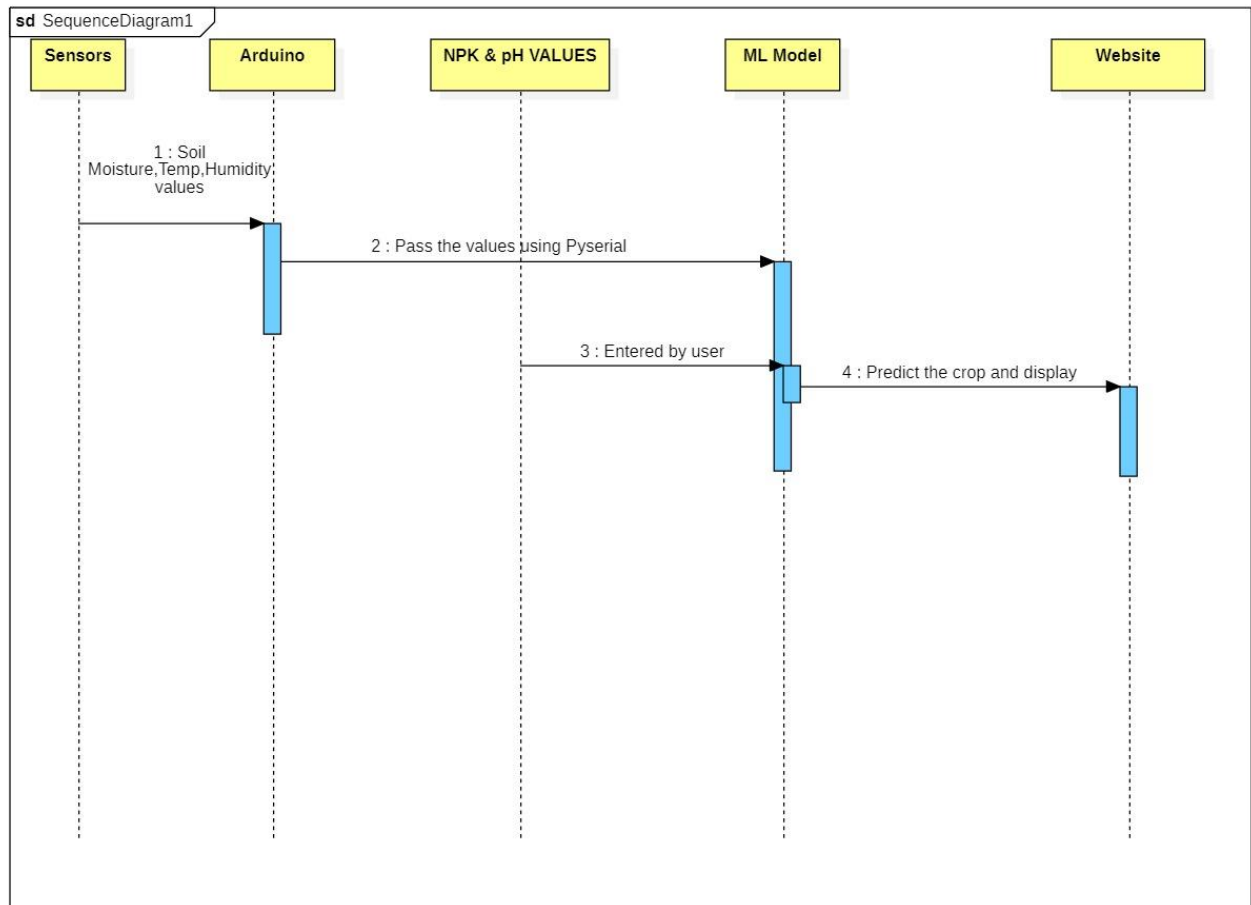
5.2 Class Diagram

Class diagram is an Object Oriented and structural diagram. It consists of classes and their relationships as association or aggregation or composition or realization. Each class consists of three compartments – name of the class, attributes of the class and functionality of the class. We also have private, public options in Class diagram.



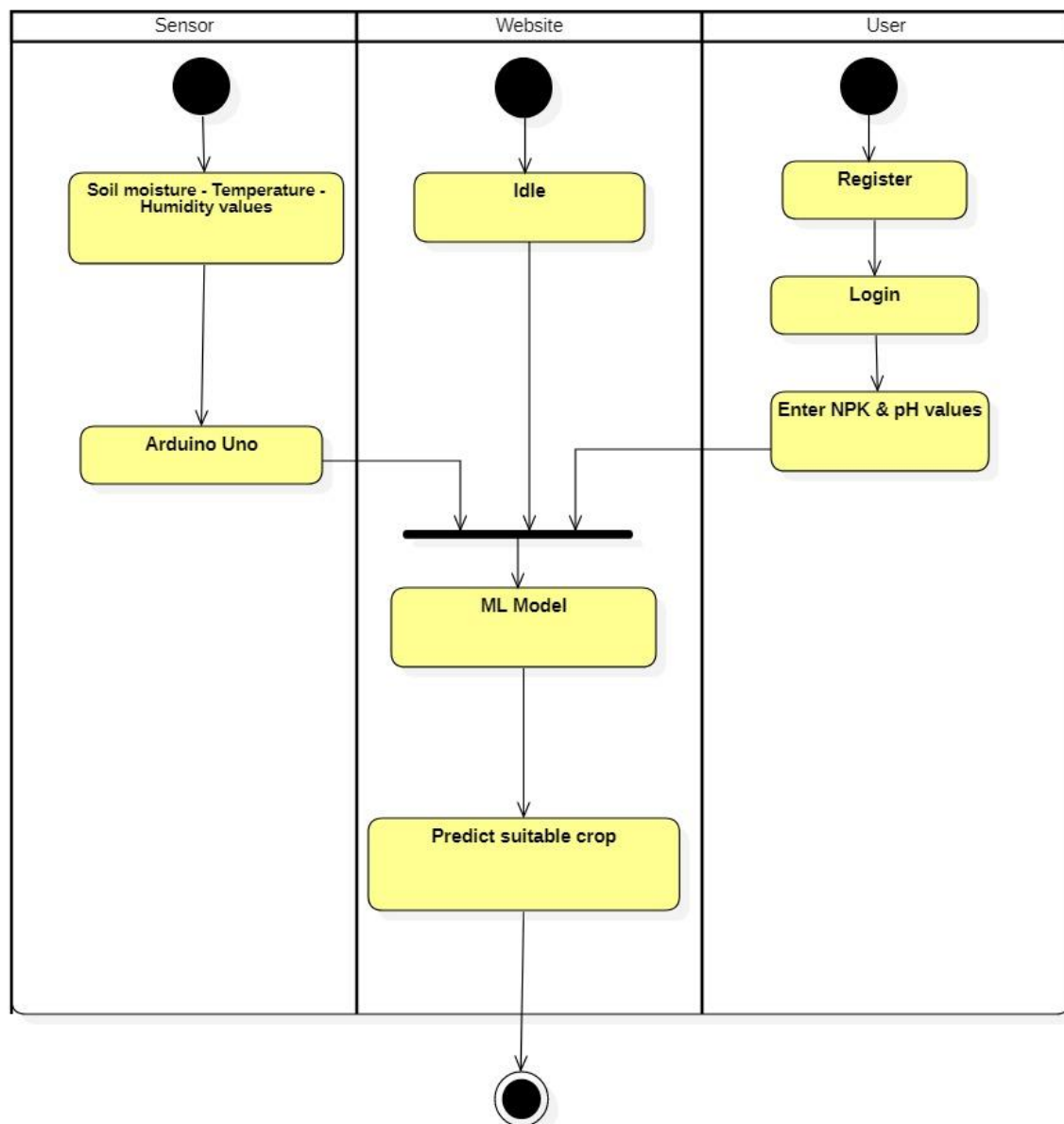
5.3 Sequence diagram

Sequence diagram is a behavioural diagram. It describes the chronological order of messages or events. It consists of package, timelines, activation box, messages, and replies. Inside package, each timeline generally represents a user who connect with other user (i.e., another timeline) through messages. The other use may respond to message using reply. There is also a self-message feature.

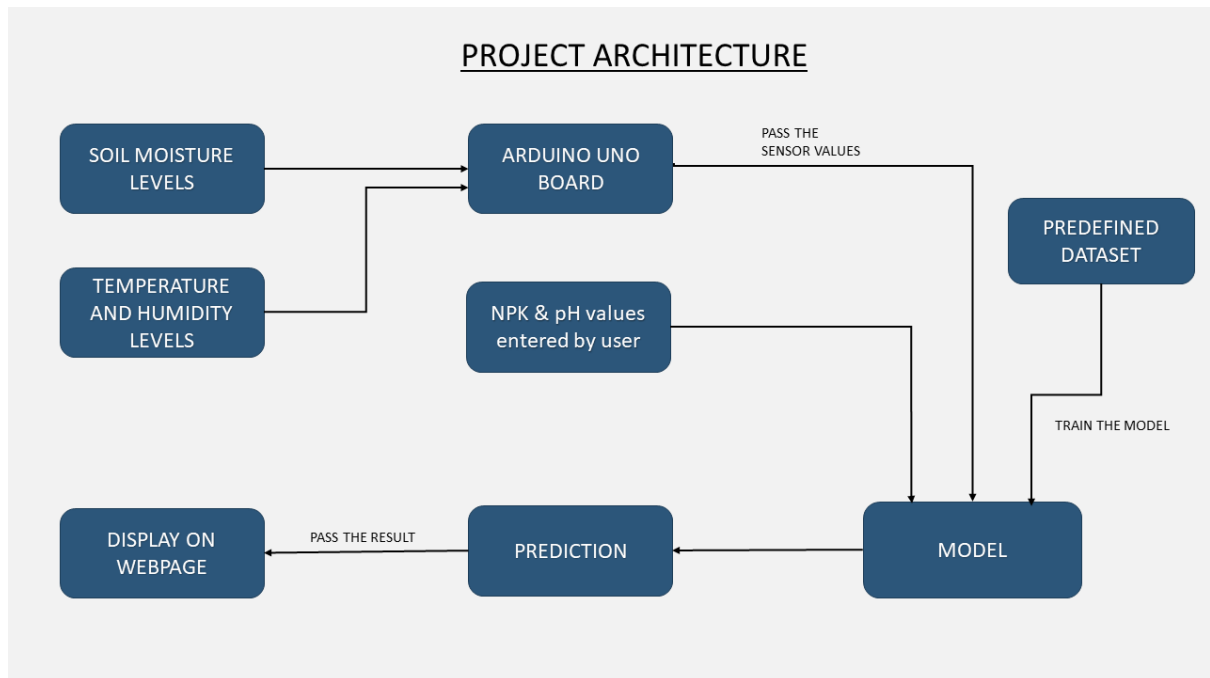


5.4 Activity diagram

Activity diagram is a behavioural diagram. It consists of swimlanes, initial state, action box, fork, join, decision box and final state. Swimlane consists of similar activities grouped together and workflow may change from one swimlane to other. Fork is used for splitting single action into two or more actions whereas join is used for joining two or more actions together.



5.5 Architecture



6. IMPLEMENTATION

6.1 Machine Learning Module

6.1.1 Dataset ^[3]

	A	B	C	D	E	F	G	H
1	N	P	K	temperature	humidity	ph	soil_moisture	label
2	90	42	43	20.87974371	82.00274423	6.502985	202.9355362	rice
3	85	58	41	21.77046169	80.31964408	7.038096	226.6555374	rice
4	60	55	44	23.00445915	82.3207629	7.840207	263.9642476	rice
5	74	35	40	26.49109635	80.15836264	6.980401	242.8640342	rice
6	78	42	42	20.13017482	81.60487287	7.628473	262.7173405	rice
7	69	37	42	23.05804872	83.37011772	7.073454	251.0549998	rice
8	69	55	38	22.70883798	82.63941394	5.700806	271.3248604	rice
9	94	53	40	20.27774362	82.89408619	5.718627	241.9741949	rice
10	89	54	38	24.51588066	83.5352163	6.685346	230.4462359	rice
11	68	58	38	23.22397386	83.03322691	6.336254	221.2091958	rice
12	91	53	40	26.52723513	81.41753846	5.386168	264.6148697	rice
13	90	46	42	23.97898217	81.45061596	7.502834	250.0832336	rice
14	78	58	44	26.80079604	80.88684822	5.108682	284.4364567	rice
15	93	56	36	24.01497622	82.05687182	6.984354	185.2773389	rice
16	94	50	37	25.66585205	80.66385045	6.94802	209.5869708	rice
17	60	48	39	24.28209415	80.30025587	7.042299	231.0863347	rice
18	85	38	41	21.58711777	82.7883708	6.249051	276.6552459	rice
19	91	35	39	23.79391957	80.41817957	6.97086	206.2611855	rice
20	77	38	36	21.8652524	80.1923008	5.953933	224.5550169	rice
21	88	35	40	23.57943626	83.58760316	5.853932	291.2986618	rice
22	89	45	36	21.32504158	80.47476396	6.442475	185.4974732	rice
23	76	40	43	25.15745531	83.11713476	5.070176	231.3843163	rice
24	67	59	41	21.94766735	80.97384195	6.012633	213.3560921	rice
25	83	41	43	21.0525355	82.67839517	6.254028	233.1075816	rice
26	98	47	37	23.48381344	81.33265073	7.375483	224.0581164	rice
27	66	53	41	25.0756354	80.52389148	7.778915	257.0038865	rice
28	97	59	43	26.35927159	84.04403589	6.2865	271.3586137	rice
Crop_recommendation								

6.1.2 Data Exploration and Pre-Processing

```
[1] from __future__ import print_function
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn import tree
import warnings
warnings.filterwarnings('ignore')
import joblib
```

```
[2] df = pd.read_csv('Crop_recommendation.csv')
```

```
[3] df.head()
```

	N	P	K	temperature	humidity	ph	soil_moisture	label
0	90	42	43	20.879744	82.002744	6.502985	202.935536	rice
1	85	58	41	21.770462	80.319644	7.038096	226.655537	rice
2	60	55	44	23.004459	82.320763	7.840207	263.964248	rice
3	74	35	40	26.491096	80.158363	6.980401	242.864034	rice
4	78	42	42	20.130175	81.604873	7.628473	262.717340	rice

```
[4] df.tail()
```

	N	P	K	temperature	humidity	ph	soil_moisture	label
2195	107	34	32	26.774637	66.413269	6.780064	177.774507	coffee
2196	99	15	27	27.417112	56.636362	6.086922	127.924610	coffee
2197	118	33	30	24.131797	67.225123	6.362608	173.322839	coffee
2198	117	32	34	26.272418	52.127394	6.758793	127.175293	coffee
2199	104	18	30	23.603016	60.396475	6.779833	140.937041	coffee

```
[5] df.size
```

```
17600
```

```
[6] df.shape
```

```
(2200, 8)
```

```
[7] df.columns
```

```
Index(['N', 'P', 'K', 'temperature', 'humidity', 'ph', 'soil_moisture',  
      'label'],  
      dtype='object')
```



```
[8] df['label'].unique()
```

```
array(['rice', 'maize', 'chickpea', 'kidneybeans', 'pigeonpeas',  
      'mothbeans', 'mungbean', 'blackgram', 'lentil', 'pomegranate',  
      'banana', 'mango', 'grapes', 'watermelon', 'muskmelon', 'apple',  
      'orange', 'papaya', 'coconut', 'cotton', 'jute', 'coffee'],  
      dtype=object)
```

```
[9] df.dtypes
```

```
N          int64  
P          int64  
K          int64  
temperature  float64  
humidity     float64  
ph           float64  
soil_moisture float64  
label        object  
dtype: object
```

```
[10] df['label'].value_counts()
```

```
pomegranate    100  
orange         100  
grapes         100  
blackgram      100  
maize          100  
papaya         100  
coconut        100  
cotton         100  
mothbeans      100  
watermelon     100  
rice           100  
jute           100  
banana         100  
coffee        100  
mungbean       100  
lentil         100  
mango          100  
pigeonpeas     100  
apple          100  
chickpea       100  
muskmelon      100  
kidneybeans    100  
Name: label, dtype: int64
```

```
[11] features = df[['N', 'P', 'K', 'temperature', 'humidity', 'ph', 'soil_moisture']]  
target = df['label']  
labels = df['label']
```

```
[12] acc = []  
model = []
```

6.1.3 Splitting the data for training and testing

```
[13] # Splitting into train and test data
```

```
from sklearn.model_selection import train_test_split  
Xtrain, Xtest, Ytrain, Ytest = train_test_split(features, target, test_size = 0.2, random_state = 2)
```

6.1.4 Training various models

```
[14] from sklearn.tree import DecisionTreeClassifier

DecisionTree = DecisionTreeClassifier(criterion="entropy",random_state=2,max_depth=5)

DecisionTree.fit(Xtrain,Ytrain)

predicted_values = DecisionTree.predict(Xtest)
x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
model.append('Decision Tree')
print("DecisionTrees's Accuracy is: ", x*100)

print(classification_report(Ytest,predicted_values))
```

```
DecisionTrees's Accuracy is: 90.0
      precision    recall  f1-score   support

   apple         1.00      1.00      1.00        13
  banana         1.00      1.00      1.00        17
blackgram         0.59      1.00      0.74        16
 chickpea         1.00      1.00      1.00        21
   coconut        0.91      1.00      0.95        21
    coffee         1.00      1.00      1.00        22
    cotton         1.00      1.00      1.00        20
   grapes         1.00      1.00      1.00        18
     jute         0.74      0.93      0.83        28
kidneybeans        0.00      0.00      0.00        14
   lentil         0.68      1.00      0.81        23
    maize         1.00      1.00      1.00        21
    mango         1.00      1.00      1.00        26
 mothbeans        0.00      0.00      0.00        19
 mungbean         1.00      1.00      1.00        24
 muskmelon         1.00      1.00      1.00        23
   orange         1.00      1.00      1.00        29
   papaya         1.00      0.84      0.91        19
      .         . . .         . . .         . .
```

```
[14]  pigeonpeas         0.62      1.00      0.77        18
     pomegranate        1.00      1.00      1.00        17
         rice         1.00      0.62      0.77        16
    watermelon         1.00      1.00      1.00        15

      accuracy              0.90       440
    macro avg         0.84      0.88      0.85       440
   weighted avg         0.86      0.90      0.87       440
```

```
[15] from sklearn.model_selection import cross_val_score
```

```
[16] # Cross validation score (Decision Tree)
     score = cross_val_score(DecisionTree, features, target,cv=5)
```

```
[17] score

array([0.93636364, 0.90909091, 0.91818182, 0.87045455, 0.93636364])
```

```
[18] from sklearn.svm import SVC

SVM = SVC(gamma='auto')

SVM.fit(Xtrain,Ytrain)

predicted_values = SVM.predict(Xtest)

x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
model.append('SVM')
print("SVM's Accuracy is: ", x)

print(classification_report(Ytest,predicted_values))
```

```
SVM's Accuracy is: 0.10681818181818181
              precision    recall  f1-score   support

   apple         1.00        0.23        0.38         13
   banana         1.00        0.24        0.38         17
 blackgram         1.00        0.19        0.32         16
  chickpea         1.00        0.05        0.09         21
   coconut         1.00        0.05        0.09         21
    coffee         0.00        0.00        0.00         22
    cotton         1.00        0.05        0.10         20
    grapes         1.00        0.06        0.11         18
     jute         1.00        0.07        0.13         28
 kidneybeans       0.03        1.00        0.07         14
   lentil         0.00        0.00        0.00         23
    maize         0.00        0.00        0.00         21
    mango         0.00        0.00        0.00         26
  mothbeans       0.00        0.00        0.00         19
   mungbean        1.00        0.12        0.22         24
 muskmelon         1.00        0.30        0.47         23
    orange         1.00        0.03        0.07         29
   ...         ...         ...         ...         ...
```

```
[18]
```

papaya	1.00	0.05	0.10	19
pigeonpeas	0.00	0.00	0.00	18
pomegranate	1.00	0.12	0.21	17
rice	0.50	0.06	0.11	16
watermelon	1.00	0.13	0.24	15
accuracy			0.11	440
macro avg	0.66	0.13	0.14	440
weighted avg	0.66	0.11	0.13	440

```
[19] # Cross validation score (SVM)
score = cross_val_score(SVM,features,target,cv=5)
score

array([0.27727273, 0.28863636, 0.29090909, 0.275      , 0.26818182])
```

```
[20] from sklearn.linear_model import LogisticRegression

LogReg = LogisticRegression(random_state=2)

LogReg.fit(Xtrain,Ytrain)

predicted_values = LogReg.predict(Xtest)

x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
model.append('Logistic Regression')
print("Logistic Regression's Accuracy is: ", x)

print(classification_report(Ytest,predicted_values))
```

```
[20] Logistic Regression's Accuracy is: 0.9522727272727273
precision recall f1-score support

apple      1.00      1.00      1.00      13
banana     1.00      1.00      1.00      17
blackgram  0.86      0.75      0.80      16
chickpea   1.00      1.00      1.00      21
coconut    1.00      1.00      1.00      21
coffee     1.00      1.00      1.00      22
cotton     0.86      0.90      0.88      20
grapes     1.00      1.00      1.00      18
jute       0.84      0.93      0.88      28
kidneybeans 1.00      1.00      1.00      14
lentil     0.88      1.00      0.94      23
maize      0.90      0.86      0.88      21
mango      0.96      1.00      0.98      26
mothbeans  0.84      0.84      0.84      19
mungbean   1.00      0.96      0.98      24
muskmelon  1.00      1.00      1.00      23
orange     1.00      1.00      1.00      29
papaya     1.00      0.95      0.97      19
pigeonpeas 1.00      1.00      1.00      18
pomegranate 1.00      1.00      1.00      17
rice       0.85      0.69      0.76      16
watermelon 1.00      1.00      1.00      15

accuracy   0.95      0.95      0.95      440
macro avg  0.95      0.95      0.95      440
weighted avg 0.95      0.95      0.95      440
```

```
[21] # Cross validation score (Logistic Regression)
score = cross_val_score(LogReg, features, target, cv=5)
score
```

```
array([0.95      , 0.96590909, 0.94772727, 0.96818182, 0.94318182])
```

```
[22] from sklearn.ensemble import RandomForestClassifier

RF = RandomForestClassifier(n_estimators=20, random_state=0)
RF.fit(Xtrain, Ytrain)

predicted_values = RF.predict(Xtest)

x = metrics.accuracy_score(Ytest, predicted_values)
acc.append(x)
model.append('RF')
print("RF's Accuracy is: ", x)

print(classification_report(Ytest, predicted_values))
```

```
RF's Accuracy is: 0.990909090909091
```

	precision	recall	f1-score	support
apple	1.00	1.00	1.00	13
banana	1.00	1.00	1.00	17
blackgram	0.94	1.00	0.97	16
chickpea	1.00	1.00	1.00	21
coconut	1.00	1.00	1.00	21
coffee	1.00	1.00	1.00	22
cotton	1.00	1.00	1.00	20
grapes	1.00	1.00	1.00	18
jute	0.90	1.00	0.95	28
kidneybeans	1.00	1.00	1.00	14
lentil	1.00	1.00	1.00	23
maize	1.00	1.00	1.00	21
mango	1.00	1.00	1.00	26
mothbeans	1.00	0.95	0.97	19
mungbean	1.00	1.00	1.00	24
muskmelon	1.00	1.00	1.00	23
orange	1.00	1.00	1.00	29
papaya	1.00	1.00	1.00	19
pigeonpeas	1.00	1.00	1.00	18
pomegranate	1.00	1.00	1.00	17
rice	1.00	0.81	0.90	16
watermelon	1.00	1.00	1.00	15
accuracy			0.99	440
macro avg	0.99	0.99	0.99	440
weighted avg	0.99	0.99	0.99	440

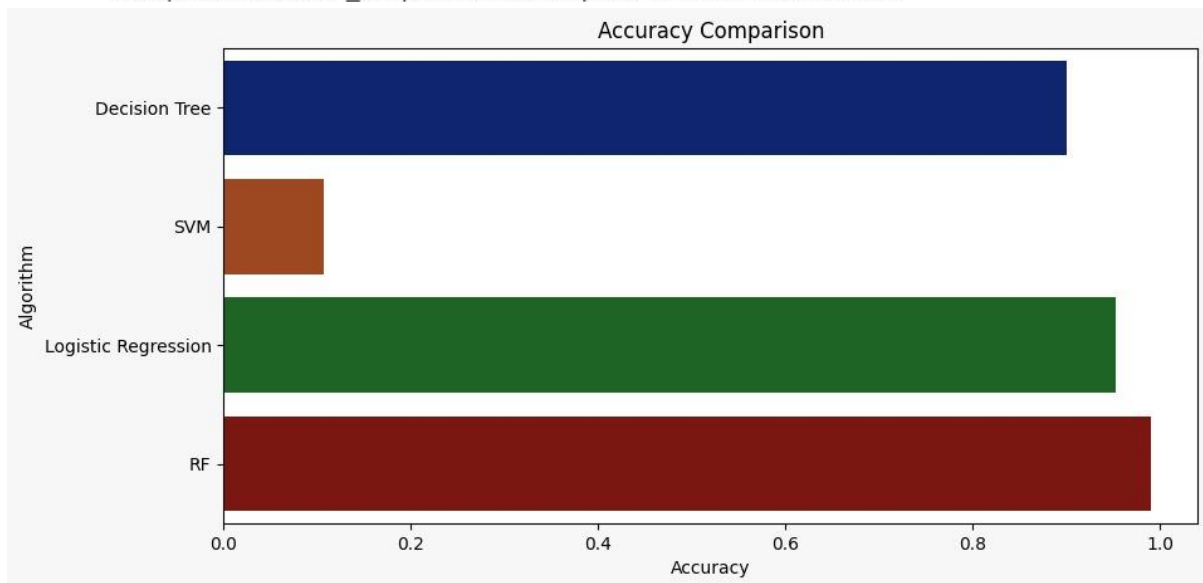
```
[23] # Cross validation score (Random Forest)
score = cross_val_score(RF, features, target, cv=5)
score
```

```
array([0.99772727, 0.99545455, 0.99772727, 0.99318182, 0.98863636])
```

6.1.5 Comparison of Accuracy

```
[24] plt.figure(figsize=[10,5],dpi = 100)
      plt.title('Accuracy Comparison')
      plt.xlabel('Accuracy')
      plt.ylabel('Algorithm')
      sns.barplot(x = acc,y = model,palette='dark')
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f97927d36d0>



```
[25] accuracy_models = dict(zip(model, acc))
      for k, v in accuracy_models.items():
          print (k, '-->', v)

Decision Tree --> 0.9
SVM --> 0.10681818181818181
Logistic Regression --> 0.9522727272727273
RF --> 0.990909090909091
```

6.1.6 Prediction

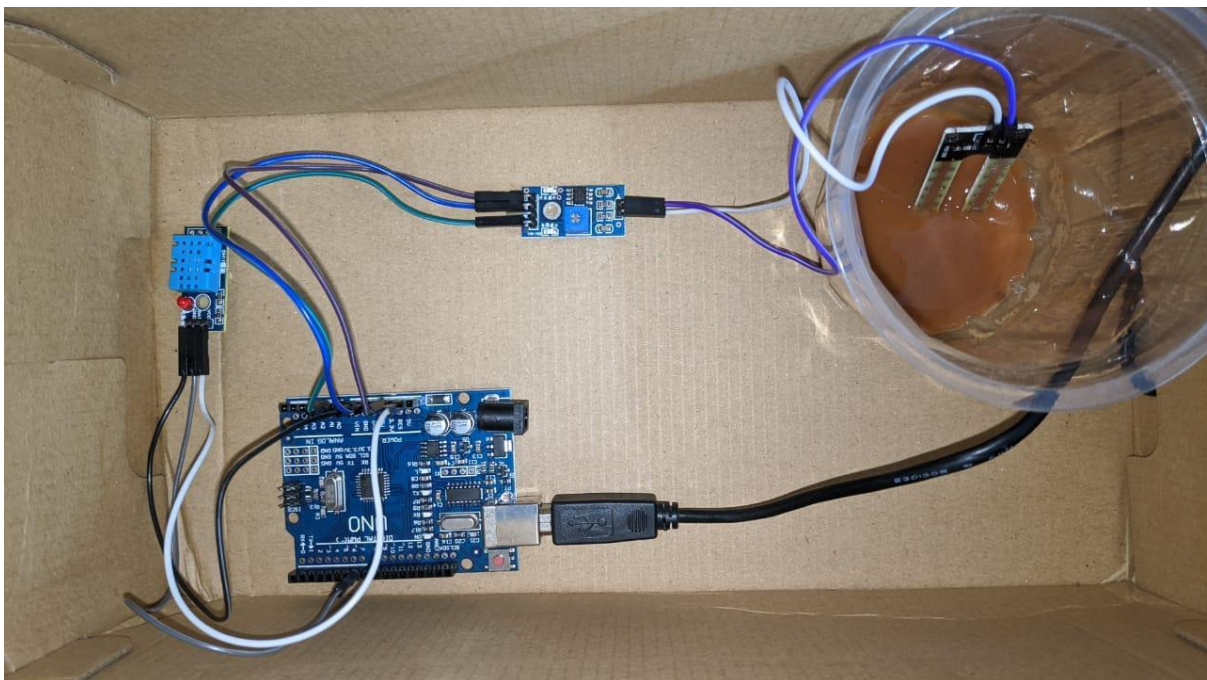
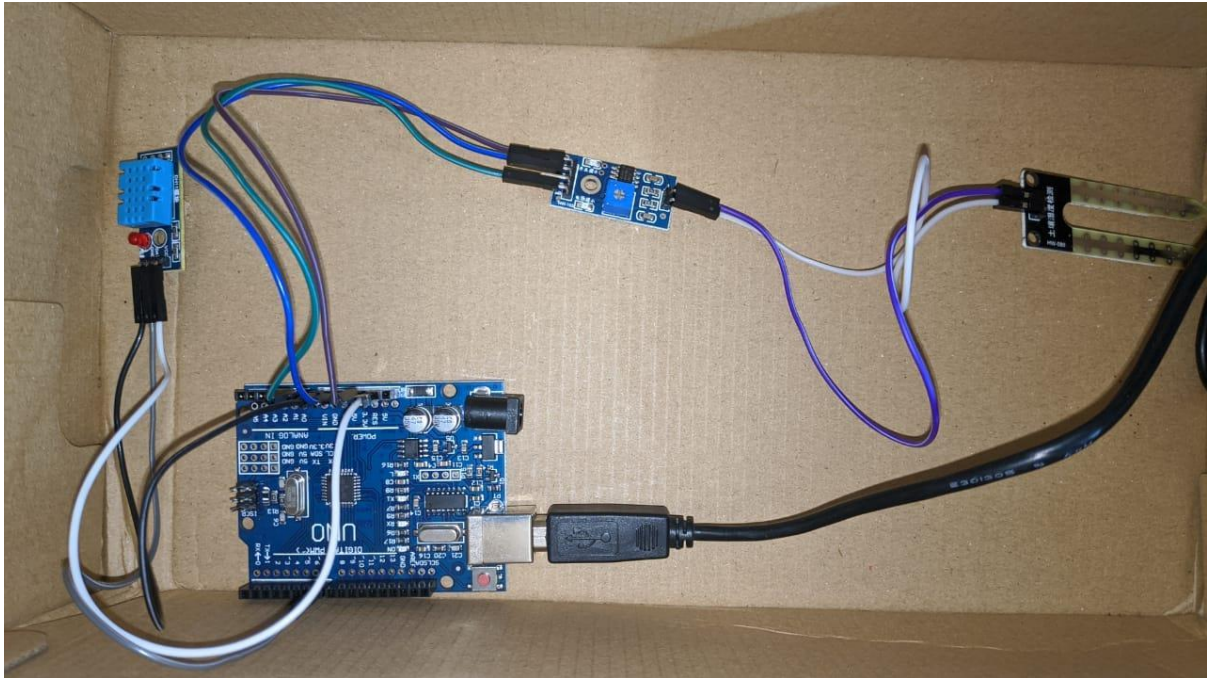
```
[26] data = np.array([[90, 42, 43, 20.8, 82.0, 6.5, 202.9]])
      prediction = RF.predict(data)
      print(prediction)

['rice']
```


6.1.7 Dumping the model.

```
[27] filename = 'finalized_model.sav'  
     joblib.dump(RF, filename)  
  
     ['finalized_model.sav']
```

6.2 IoT Module



mod | Arduino 1.8.15
File Edit Sketch Tools Help

mod

```
#include "DHT.h"

#define DHTPIN 7

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

int sensorPin = A0;
int sensorValue;

void setup() {
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  sensorValue = analogRead(sensorPin);
  float hum = dht.readHumidity();
  float temp = dht.readTemperature();
  Serial.print(temp);
  Serial.print("x");
  Serial.print(hum);
  Serial.print("x");
  Serial.println(sensorValue);
  delay(10000);
}
```

COM3

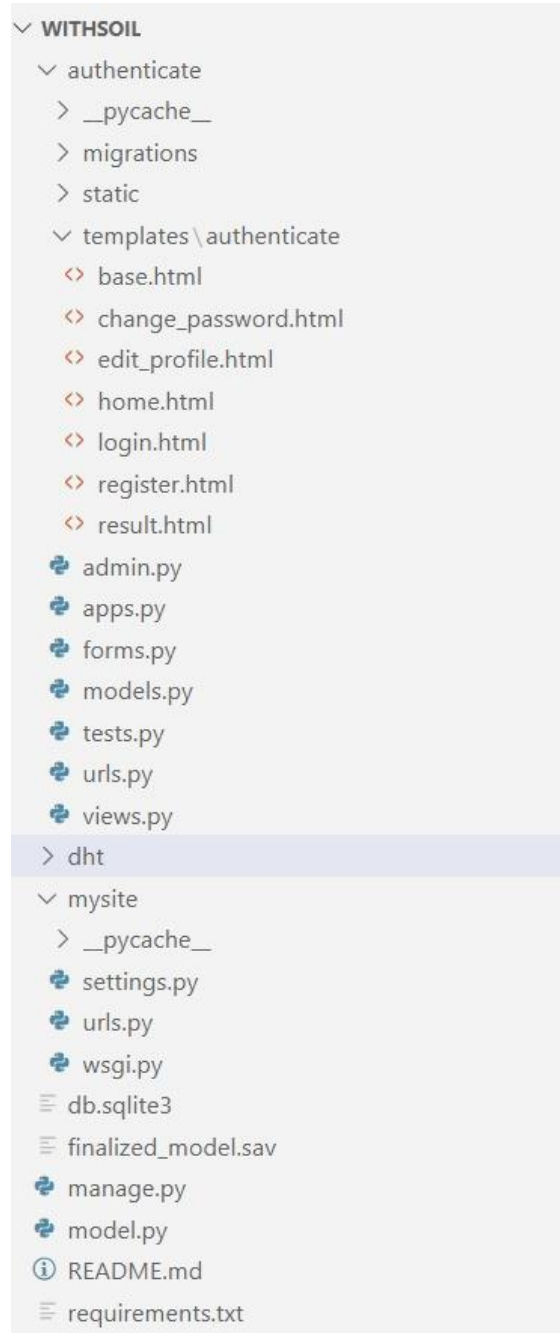
Send

30.80x67.00x281
30.80x67.00x282
30.80x67.00x283

☒ Autoscroll ☐ Show timestamp Newline 9600 baud Clear output

6.3 Web Development Module

Directory Structure



urls.py in mysite app

```
from django.contrib import admin
from django.urls import path, include
from django.contrib.staticfiles.urls import staticfiles_urlpatterns

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('authenticate.urls')),
]

urlpatterns += staticfiles_urlpatterns()
```

urls.py in authenticate app

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.home, name="home"),
    path('login/', views.login_user, name='login'),
    path('logout/', views.logout_user, name='logout'),
    path('register/', views.register_user, name='register'),
    path('edit_profile/', views.edit_profile, name='edit_profile'),
    path('change_password', views.change_password, name='change_password'),
    path('result/', views.result, name='result'),
]
```

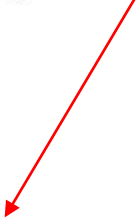
views.py in authenticate app

```
def login_user(request):
    if request.method == 'POST':
        username = request.POST['username']
        password = request.POST['password']
        user = authenticate(request, username=username, password=password)
        if user is not None:
            login(request, user)
            messages.success(request, ('You Have Been Logged In!'))
            return redirect('home')
        else:
            messages.success(request, ('Error Logging In - Please Try Again...'))
            return redirect('login')
    else:
        return render(request, 'authenticate/login.html', {})
```

urls.py in authenticate app

```
from django.urls import path
from . import views

urlpatterns = [
    path('', views.home, name="home"),
    path('login/', views.login_user, name='login'),
    path('logout/', views.logout_user, name='logout'),
    path('register/', views.register_user, name='register'),
    path('edit_profile/', views.edit_profile, name='edit_profile'),
    path('change_password', views.change_password, name='change_password'),
    path('result/', views.result, name='result'),
]
```



views.py in authenticate app

```
def result(request):
    arduino = serial.Serial('com3', 9600)
    print('Established serial connection to Arduino')
    arduino_data = arduino.readline()
    decoded_values = str(arduino_data[0:len(arduino_data)].decode("utf-8"))
    list_values = decoded_values.split('x')
    list_in_floats = []
    for item in list_values:
        list_in_floats.append(float(item))

    cls = joblib.load('finalized_model.sav')
    lis = []
    lis.append(request.GET['N'])
    lis.append(request.GET['P'])
    lis.append(request.GET['K'])
    lis.append(list_in_floats[0])
    lis.append(list_in_floats[1])
    lis.append(request.GET['ph'])
    lis.append(list_in_floats[2])

    ans = cls.predict([lis])
    s = ''.join(map(str, ans))
    lis = [float(x) for x in lis]
    return render(request, "authenticate/result.html", {'ans':s, 'lis':lis})
```



result.html in authenticate/templates

```
{% extends 'authenticate/base.html' %}
{% load static %}
{% block content %}
    <!-- <h1>The results are here</h1><br> -->
    <div class="alert bg-transparent">
        <table class="table" style="text-align: center;">
            <thead>
                <tr>
                    <th scope="col">N</th>
                    <th scope="col">P</th>
                    <th scope="col">K</th>
                    <th scope="col">Temperature</th>
                    <th scope="col">Humidity</th>
                    <th scope="col">pH</th>
                    <th scope="col">Soil Moisture</th>
                </tr>
            </thead>
            <br>
            <tbody>
                <tr>
                    {% for item in lis %}
                        <td>{{item}}</td>
                    {% endfor %}
                </tr>
            </tbody>
        </table>
    </div>
    <!-- <h3>The data you entered is : {{lis}}</h3><br> -->
    <center><h3>The suggested crop is <b style="color: #e62a09">{{ans}}</b></h3></center>
    <br>
    <br>
    
{% endblock %}
```

7. SOFTWARE TESTING

7.1 Unit Testing:

In this type of testing, we split the entire code into small parts and test each part considering each part as one unit. Each part is tested to detect the problems in the early stage because if small bugs are detecting in the unit stages, it would be easy for debugging. There are black box and white box tester in unit testing. Black box testers only have test case which they need to verify against test output. They do not know about the functionality. On the other hand, white box testers know the functionality of the units.

7.2 Integration Testing:

The next stage after unit testing is the integration testing, in this we combine modules like user authentication with the actual logic module and test it against the required functionality. It has four approaches: big bang approach, top-down approach, bottom-up approach, mixed integration. Big bang means as the name suggest, we combine all the modules and test the high-level overview functionality.

7.3 System Testing:

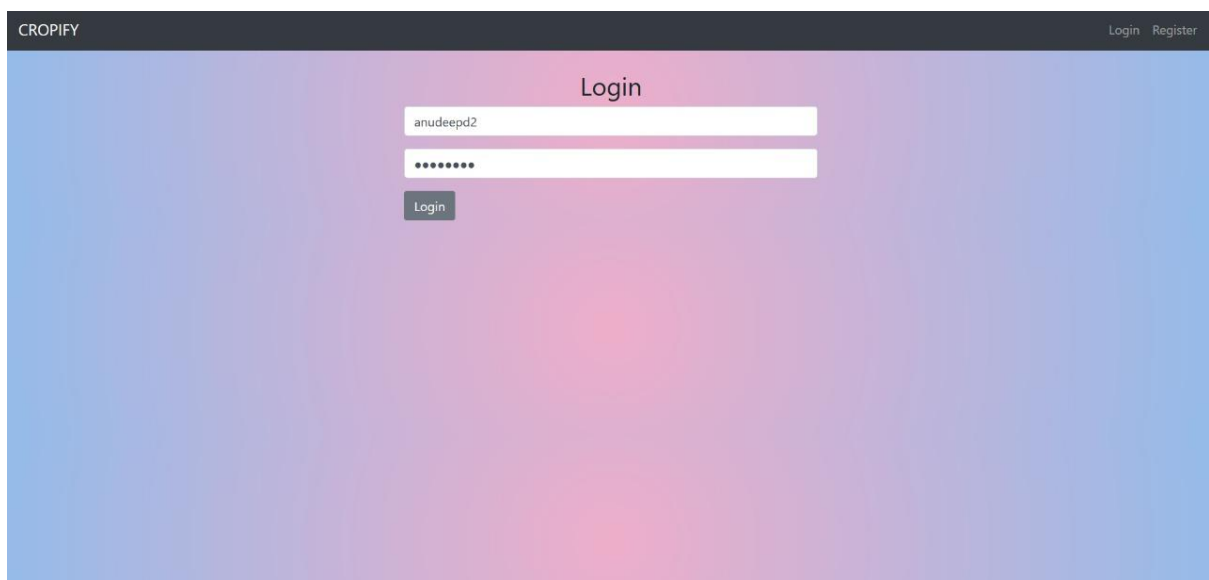
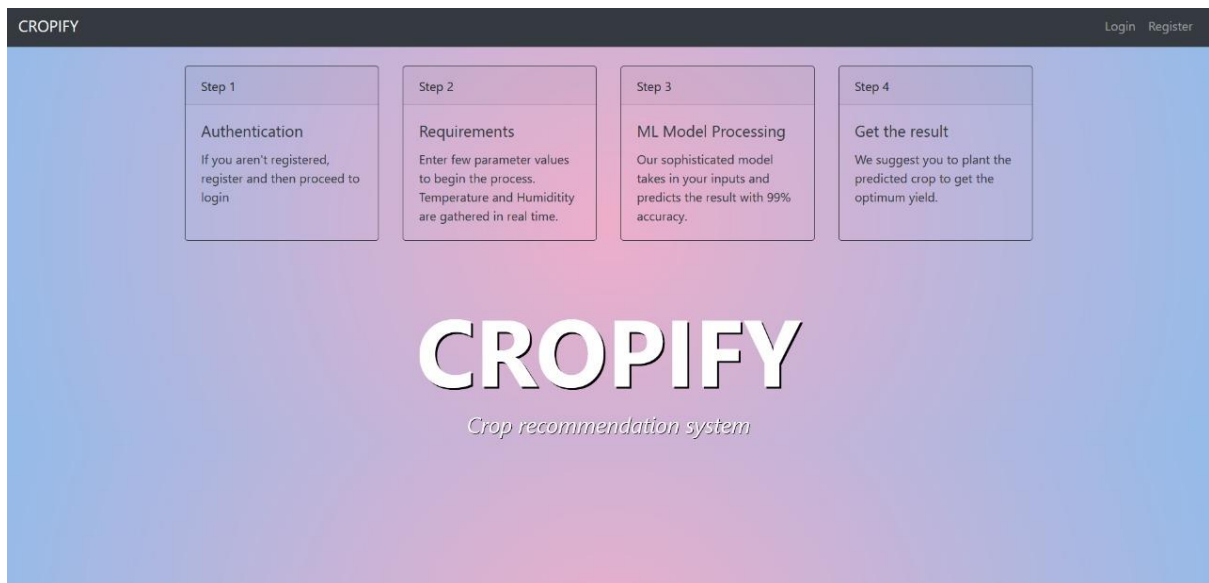
In this testing, the system parameters like scalability, reliability, speed of the application are evaluated. It also checks if the system can handle huge loads and if it can perform well under stress or custom environment. This is accomplished by creating a test environment where the system is evaluated with various test cases and defects are analysed and reported. Once the defects are handled the system is retested again.

7.4 Acceptance Testing:

Acceptance Testing is done to check if the system is meeting the requirements and if it is ready for delivery. This is the last stage of testing a product before delivery. This is done to check how good the product is. Changes are made based on the feedback from the users. Hence this gives scope for improvement and offers better user experience with the product.

There are two popularly known testing techniques under this, namely: Alpha and Beta testing. Alpha testing is done by testing team whereas Beta testing is done by selected end users.

8. RESULTS



CROPIFY

Welcome, Anudeep

LogoutEdit ProfileChange Password

You Have Been Logged In!

Crop Prediction

90

45

30

6.9

Submit


CROPIFY

Welcome, Anudeep

LogoutEdit ProfileChange Password

N	P	K	Temperature	Humidity	pH	Soil Moisture
90.0	45.0	30.0	30.8	67.0	6.9	286.0

The suggested crop is **coffee**



Edit Profile

Username:

First name:

Last name:

Email address:

[Edit Profile](#)

[Click Here To Change Your Password](#)

Change Password

Old password:

New password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

New password confirmation:

[Change Password](#)

You Have Been Logged Out...

x

Step 1

Authentication

If you aren't registered, register and then proceed to login

Step 2

Requirements

Enter few parameter values to begin the process. Temperature and Humidity are gathered in real time.

Step 3

ML Model Processing

Our sophisticated model takes in your inputs and predicts the result with 99% accuracy.

Step 4

Get the result

We suggest you to plant the predicted crop to get the optimum yield.

CROPIFY

Crop recommendation system

9. CONCLUSION AND FUTURE ENHANCEMENTS

9.1 Conclusion

In this project, we prepared a machine learning model based on Random Forest classifier. The model is interpreted in the form of a website. The user needs to be registered to use it. It takes in N, P, K, pH values as input from the user. The temperature, humidity and soil moisture values are gathered using the hardware. When the user enters the required parameters and clicks on submit, a serial communication is established with Arduino Uno and values are collected from the hardware. Together, all these values help in predicting the suitable crop.

9.2 Future enhancements

As of now, we are taking in the values of N, P, K, and pH manually. But in future, we can enhance it by incorporating sensors. Additionally, we can also try to predict the crop yield and give advice on what fertilizers to choose to get the best outcome based on the input parameters.

10. BIBLIOGRAPHY

- [1] [S. Pudumalar, E. Ramanujam, R. H. Rajashree, C. Kavya, T. Kiruthika and J. Nisha, "Crop recommendation system for precision agriculture," 2016 Eighth International Conference on Advanced Computing \(ICoAC\), 2017, pp. 32-36, doi: 10.1109/ICoAC.2017.7951740.](#)
- [2] [R. Kumar, M. P. Singh, P. Kumar and J. P. Singh, "Crop Selection Method to maximize crop yield rate using machine learning technique," 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials \(ICSTM\), 2015, pp. 138-145, doi: 10.1109/ICSTM.2015.7225403.](#)
- [3] Dataset - <https://www.kaggle.com/saisampathpotluri/what-to-cropcsv>
- [4] <https://github.com/Priyabrata017/Crop-prediction-using-Machine-Learning>
- [5] [Rohit Kumar Rajak, Ankit Pawar, Mitalee Pendke , Pooja Shinde, Suresh Rathod, Avinash Devare "Crop Recommendation System to Maximize Crop Yield using Machine Learning Technique" Volume: 04 Issue: 12 | Dec-2017 International Research Journal of Engineering and Technology \(IRJET\)](#)
- [6] <https://www.arduino.cc/en/Guide/ArduinoUno>
- [7] https://create.arduino.cc/projecthub/techno_z/dht11-temperature-humidity-sensor-98b03b
- [8] <https://create.arduino.cc/projecthub/MisterBotBreak/how-to-use-a-soil-moisture-sensor-ce769b>