

**AGRICULTURAL CROP AND FERTILIZER RECOMMENDATIONS BASED ON VARIOUS PARAMETERS**

**PROJECT REPORT**

*Submitted by*

**Under the Guidance of Mrs.NITHYA**

*DEPARTMENT OF COMPUTER SCIENCE*

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**TEAM MEMBERS** GOKULNATH S(714521104016) MARISH P (714521104028)

HARINI R (714521104017)

LAKSHANA K (714521104025)

TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER NUMBER** | **TITLE** | **PAGE NUMBER** |
|  | Abstract | I |
| 1 | Introduction | 1 |
| 2 | System Analysis | 6 |
| 3 | Software Requirements Specifications | 9 |
| 4 | Technology | 10 |
| 5 | Design | 12 |
| 6 | Result | 24 |
| 7 | Maintenance | 63 |
| 8 | Conclusion | 68 |
|  | Appendices | 69 |
|  | References | 71 |

# ABSTRACT

Agriculture, as a cornerstone of global food security, faces the constant challenge of optimizing crop yield while mitigating the impact of plant diseases. The emergence of advanced technologies, particularly in the fields of computer vision and machine learning, provides an opportunity to revolutionize agricultural practices. This project presents an integrated agricultural decision support system for disease prediction and crop yield estimation, leveraging image analysis of fruits in conjunction with weather datasets. The system aims to enhance precision agriculture practices by providing early detection of plant diseases and accurate predictions of crop yields. The research involves the collection of diverse datasets containing images of plants, labelled with disease information, and integrates meteorological data to capture environmental variables. The methodology encompasses advanced image processing techniques for the extraction of relevant features from plant images. A deep learning model, specifically a Convolutional Neural Network (CNN), is developed to analyse and classify images, predicting the likelihood of diseases affecting crops from fruits. Based on affected disease, recommend the natural fertilizers. The model is trained on a comprehensive dataset that includes historical weather data, enabling it to discern patterns and correlations between environmental conditions using Multilayer perceptron algorithm. Finally, prediction details send to farmers in terms of SMS alert.

I

# INTRODUCTION

Crop recommendation systems that use machine learning (MLP) may build complicated models that can understand complex correlations between different input factors and the best crop selections. Through the use of MLP algorithms and taking into account the farm's location data, these systems are able to offer customised suggestions that improve both agricultural sustainability and production.This technique is unique in that it may adjust and change over time as a result of its ongoing learning from fresh data, which keeps the suggestions current and useful. Essentially, the incorporation of MLP into crop recommendation systems represents a major advancement in precision agriculture by providing farmers with insightful information to help them make decisions that maximise output while minimising environmental effect. The MLP learning process is dynamic and iterative, enabling the system to constantly improve its predictions in response to fresh input. This flexibility guarantees that the suggestions made stay current and correct, taking into account modifications to farming methods and environmental circumstances over time. Multilayer Perceptron (MLP) neural networks are one sophisticated method for creating such systems. These systems use machine learning and artificial intelligence to analyse a variety of datasets covering a broad range of topics, such as soil quality, climate, past crop yields, and regional agricultural practises. Furthermore, MLP is resilient in real-world agricultural contexts where variables do not always follow exact patterns because of its capacity to manage uncertainties and noise in the data. A new age in agriculture is being ushered in by the combination of location-specific data, large datasets, and the adaptive capabilities of machine learning in crop recommendation systems. These systems open the door to a more resilient and fruitful agricultural future by providing farmers with tools for intelligent decision-making and supporting the larger goals of precision and sustainable farming.

# MODULE SPECIFICATIONS

* DATASETS ACQUISITION
* PREPROCESSING
* FEATURES EXTRACTION
* MODEL TRAINING
* CROP DETAILS

# DATASETS ACQUISITION

In this module, we can upload the crop yield datasets in the form of CSV file format. And also store the data in database for future purpose. The dataset includes the temperature, rain fall, pH value, nitrogen, phosphorus, potassium values. These values are extracted from Kaggle website and values are stored in the form of integer values. And also upload the soil datasets that are collected from KAGGLE sources in terms of image format.

# PREPROCESSING

Data pre-processing is an important step in the data mining process. The phrase ["garbage in, garbage out"](https://en.wikipedia.org/wiki/GIGO) is particularly applicable to data mining and [machine learning](https://en.wikipedia.org/wiki/Machine_learning) projects. Data preparation and filtering steps can take considerable amount of processing time. In this module, we can eliminate the irrelevant values and also estimate the missing values of data. Finally provide structured datasets. Then in soil image, we can analyze the noises and remove the noises using Median filtering algorithm.

Median filtering is a widely used technique for removing noise from images, including soil images. It works by replacing each pixel in an image with the median value of its neighbouring pixels. In the case of salt and pepper noise, which is

common in soil images, median filtering can be particularly effective at removing the noise while preserving the edges and details of the image.

# FEATURES EXTRACTION

Feature selection refers to the process of reducing the inputs for processing and analysis, or of finding the most meaningful inputs. In this module, select the multiple features from uploaded datasets. And train the datasets with various crop labels with multiple attribute values. We can train the datasets to multiple crops such as rice, maize and so on. Features are collected from soil image datasets such as color, shape and texture datasets. Build a CNN architecture that is appropriate for the task of soil feature extraction. This can involve choosing the number and size of convolutional and pooling layers, selecting an activation function, and deciding on the loss function and optimizer.

# MODEL TRAINING

In this module, we can implement multi-layer perceptron algorithm to classify the uploaded datasets. MLP algorithm is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. Crop prediction using MLP (Multilayer Perceptron) algorithm is a machine learning technique that involves the use of artificial neural networks to predict crop yields based on environmental data. The MLP algorithm is a type of feedforward neural network that is capable of learning complex relationships between input variables and output predictions. The process of crop prediction using MLP algorithm typically involves several steps. First, historical data on crop yields, weather conditions, soil quality, and other environmental factors are collected and pre-processed.

The data is then divided into training and validation sets, with the training set used to train the MLP model and the validation set used to evaluate its performance. The MLP model consists of an input layer, one or more hidden layers, and an output layer. Each layer contains a set of neurons, which are connected to the neurons in adjacent layers by weighted connections. During training, the weights of these connections are adjusted in order to minimize the error between the predicted crop yields and the actual crop yields. Once the MLP model has been trained and validated, it can be used to make predictions about future crop yields based on new input data. The input data typically includes information on weather conditions, soil quality, crop type, and other relevant factors. The MLP model processes this inputs data and produces an output prediction of the expected crop yield.

Then image is training using CNN algorithm. Split the dataset into training, validation, and testing sets. The training set is used to train the CNN, while the validation set is used to tune the model's hyperparameters and prevent overfitting. The testing set is used to evaluate the final performance of the model. Build a CNN architecture that is appropriate for the task of soil image classification. This can involve choosing the number and size of convolutional and pooling layers, selecting an activation function, and deciding on the loss function and optimizer. Train the CNN on the training set using backpropagation and gradient descent. Monitor the model's performance on the validation set and adjust the hyperparameters as necessary to prevent overfitting. Evaluate the final performance of the CNN on the testing set. This can involve calculating metrics such as accuracy, precision, recall, and F1 score.

# CROP PREDICTION

Appropriate prediction of crop productivity is required for efficient planning of land usage and economic policy. In recent times, forecasting of crop productivity at the within-field level has increased. The most influencing factor for crop productivity is weather conditions. In this module, we can test the datasets by using deep learning algorithm. Finally provide the details about crops and yield information with improved accuracy rate. The accuracy of the project is calculated

in terms of Precision, Recall and F-measure values. After the model has been validated, it can be deployed for use in the real world. In this module, we predict the crops based on user input and classify with trained model file This may involve integrating the model into an existing system for crop management or using it to make predictions for a specific region or field.

# SYSTEM ANALYSIS

**EXISTING SYSTEM**

Agriculture is one of the important industrial sectors in India and the country’s economy is highly dependent on it for rural sustainability. Due to some factors like climate changes, unpredicted rainfall, decrease of water level, use of pesticides excessively etc. The level of agriculture in India is decreased. To know the level of production we performed descriptive analytics on the agriculture data. The main objective of this research work is to provide a methodology so that it can perform descriptive analytics on crop yield production in an effective manner. Although, some studies revealed statistical information about the agriculture in India, few studies have investigated crop prediction based on the historic climatic and production data. Classification algorithms accept been acclimated for assorted purposes including classification, clustering, agent quantization, arrangement association, action approximation, forecasting, ascendancy applications and optimization. In existing system implement Principal component regression is a two steps method. Firstly, in PCR method stores the variables and reduces the dimensionality of the data and again stores in the structure of table. It extracts the most variation of data and performs feature selection so that dimensionality could be reduced. And find the first factor in the direction maximizes the dispersion of the observation, and other factor should be also maximizing in the dispersion of diagonal of the first factor. And then we can rotate the factor perpendicular and for more dimensional data we can continue in a combined approach. The result of PCR is the representation of the variation of the sample of less dimensional data. Secondly, we try to fit a linear regression between the samples on the factors which are mostly correlated with factors. PCR solves the multidimensional space problem and co-linearity problem in an efficient manner.

# DISADVANTAGES

* Labeleddata-based classification
* Provide high number of false positive
* Binary classification can be occurred
* Computational complexity

# PROPOSED SYSTEM

Predicting crop yield based on the environmental, soil, water and crop parameters has been a potential research topic. Deep-learning-based models are broadly used to extract significant crop features for prediction. Though these methods could resolve the yield prediction problem there exist the following inadequacies: Unable to create a direct non-linear or linear mapping between the raw data and crop yield values; and the performance of those models highly relies on the quality of the extracted features. Deep reinforcement learning provides direction and motivation for the aforementioned shortcomings. Combining the intelligence of reinforcement learning and deep learning, deep reinforcement learning builds a complete crop yield prediction framework that can map the raw data to the crop prediction values. In this project, we use deep neural networks to predict yield, check yield, and yield difference of corn hybrids from genotype and environment data. Deep neural networks belong to the class of representation learning models that can find the underlying representation of data without handcrafted input of features. Deep neural networks have multiple stacked non- linear layers which transform the raw input data into higher and more abstract representation at each stacked layer. As such, as the network grows deeper, more complex features are extracted which contribute to the higher accuracy of results. Given the right parameters, deep neural networks are known to be universal approximate functions, which means that they can approximate almost any function, although it may be very challenging to find the right parameters

# ADVANTAGES

* Accuracy is high
* Parallel processing
* Reduce number of false positive rate
* Time and computational complexity can be reduced

# SOFTWARE REQUIREMENTS SPECIFICATIONS

**HARDWARE REQUIREMENTS**

* Processor : Intel core processor 2.6.0 GHZ
* RAM : 1GB
* Hard disk : 160 GB
* Compact Disk : 650 Mb
* Keyboard : Standard keyboard
* Monitor : 15-inchcolor monitor

# SOFTWARE REQUIREMENTS

* Operating system : Windows OS
* Front End : PYTHON
* Back End : MYSQL
* IDE : PYCHARM

# TECHNOLOGY

**MULTI-LAYER PERCEPTRON**

A Multi-Layer Perceptron (MLP) is a type of artificial neural network that is commonly used in machine learning. It is composed of multiple layers of interconnected nodes or neurons, where each neuron is a mathematical function that receives input signals, performs a computation on them, and produces an output. The basic architecture of an MLP includes an input layer, one or more hidden layers, and an output layer. The input layer receives the input data, and each subsequent hidden layer applies a non-linear transformation to the output of the previous layer. The output layer produces the final output of the network. During training, an MLP adjusts the weights and biases of the neurons in the network to minimize a cost or loss function, which measures the difference between the predicted output and the actual output. This process, called backpropagation, uses the chain rule of calculus to compute the gradient of the loss function with respect to the weights and biases. Once the MLP is trained, it can be used to make predictions on new data by feeding the data through the input layer and propagating it through the network to produce an output. MLPs are widely used in applications such as image and speech recognition, natural language processing, and predictive modeling.

# PSEUDOCODE

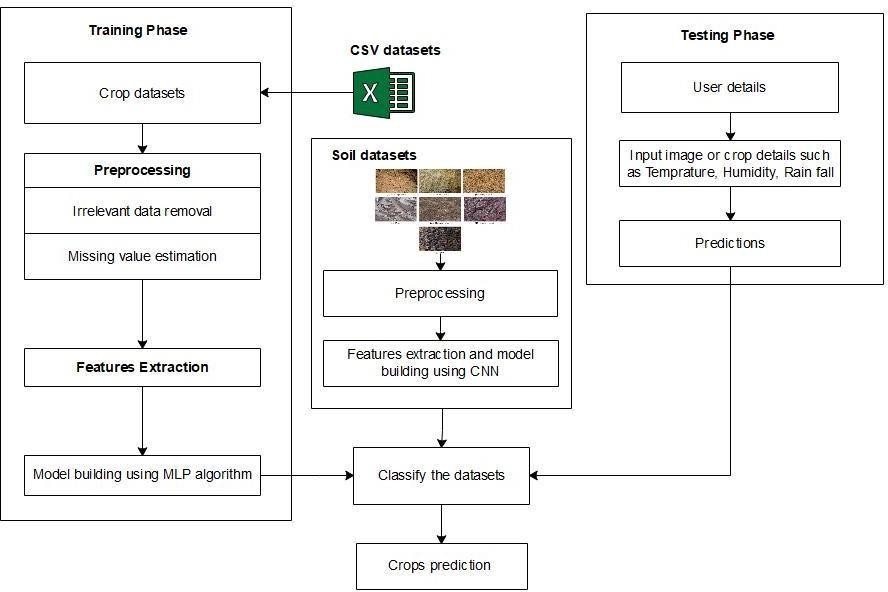
* Initialize the weights and biases of the network randomly.
* Set the learning rate and the number of epochs.
* For each epoch do:
* Shuffle the training data.
* For each training example do:
* Feedforward the input through the network and compute the output.
* ii. Calculate the error between the predicted output and the actual output.
* iii. Backpropagate the error through the network to update the weights and biases.
* Calculate the average error for the epoch.
* Test the network on the validation set to evaluate its performance.
* Repeat steps 3 and 4 until the network achieves satisfactory performance on the validation set.
* Test the final network on the test set to evaluate its generalization performance.

# DESIGN

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture; collectively these are called architecture description languages (ADLs).

## Various organizations define systems architecture in different ways, including:

* + An allocated arrangement of physical elements which provides the design solution for a consumer product or life-cycle process intended to satisfy the requirements of the functional architecture and the requirements baseline.
  + Architecture comprises the most important, pervasive, top-level, strategic inventions, decisions, and their associated rationales about the overall structure (i.e., essential elements and their relationships) and associated characteristics and behavior.
  + If documented, it may include information such as a detailed inventory of current hardware, software and networking capabilities; a description of long-range plans and priorities for future purchases, and a plan for upgrading and/or replacing dated equipment and software



* + The composite of the design architectures for products and their life-cycle processes.

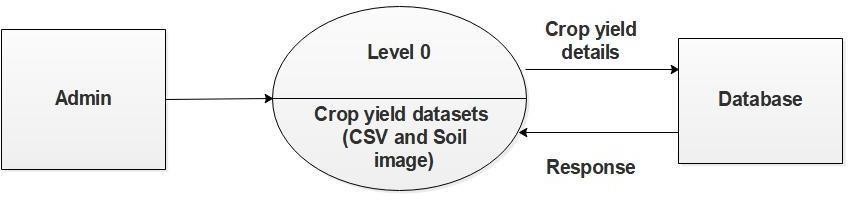
# DATAFLOW DIAGRAM

A two-dimensional diagram explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must identify external inputs and outputs, determine how the inputs and outputs relate to each other, and explain with graphics how these

connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

**Data flow Symbols:**

|  |  |
| --- | --- |
| **Symbol** | **Description** |
|  | An **entity**. A source of data or a destination for data. |
|  | A **process** or task that is performed by the system. |
|  | A **data store**, a place where data is held between processes. |
|  | A **data flow**. |

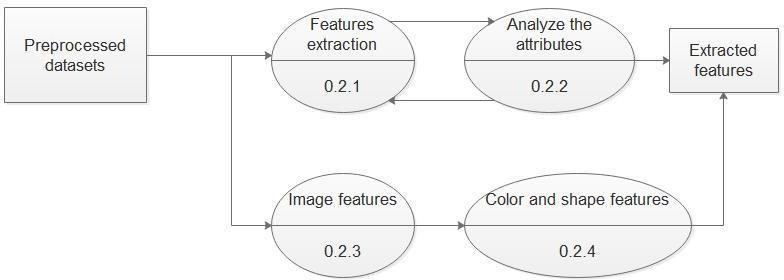
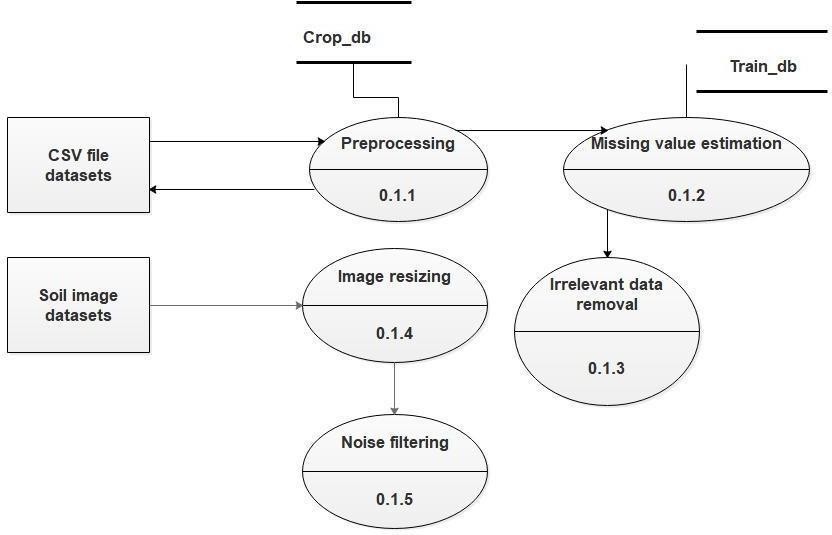


# LEVEL 0

The Level 0 DFD shows how the system is divided into 'sub-systems' (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

# LEVEL-1

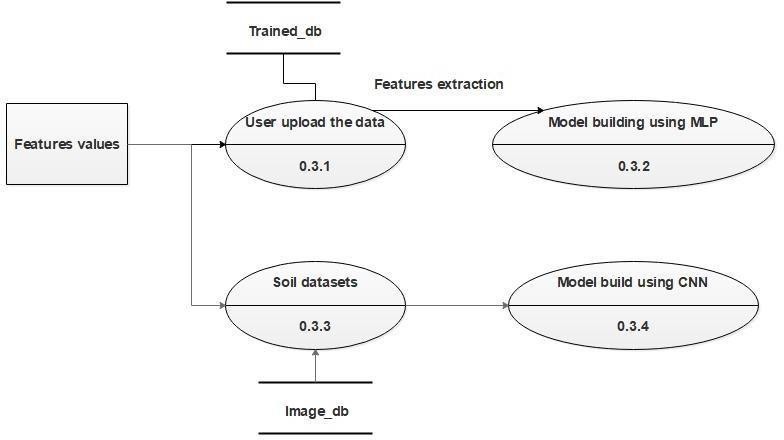
The next stage is to create the Level 1 Data Flow Diagram. This highlights the main functions carried out by the system. As a rule, to describe the system was using between two and seven functions - two being a simple system and seven being a complicated system. This enables us to keep the model manageable on screen or paper.



# LEVEL-2

A Data Flow Diagram (DFD) tracks processes and their data paths within the

business or system boundary under investigation. A DFD defines each domain boundary and illustrates the logical movement and transformation of data within the defined boundary. The diagram shows 'what' input data enters the domain, 'what' logical processes the domain applies to that data, and 'what' output data leaves the domain. Essentially, a DFD is a tool for process modeling and one of the oldest.

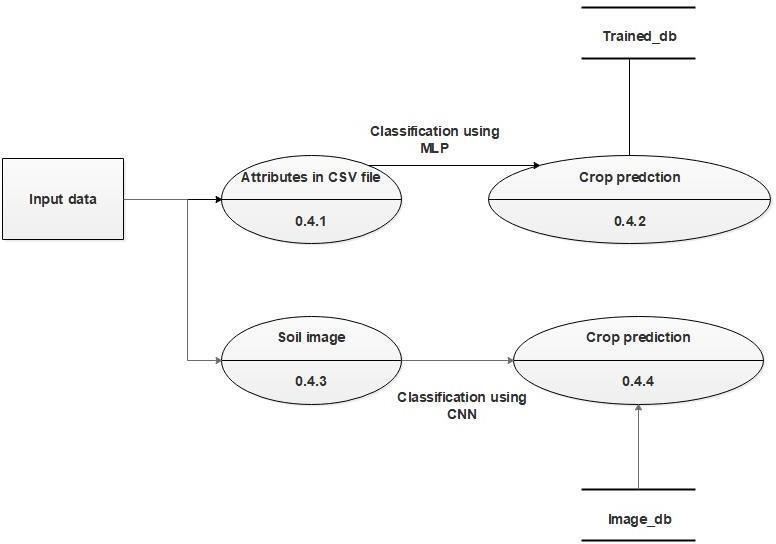


# LEVEL-3

A data flow diagram (DFD) is a graphical representation of the flow of data through an information system. A DFD shows the flow of data from data sources and data stores to processes, and from processes to data stores and data sinks. DFDs are used for modelling and analyzing the flow of data in data processing systems, and are usually accompanied by a data dictionary, an entity-relationship model, and a number of process descriptions.

# LEVEL 4

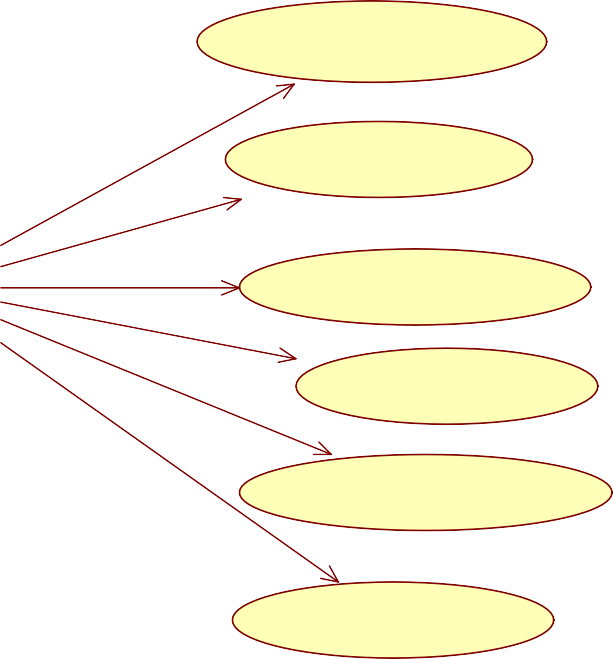
A DFD may look similar to a flow chart. However, there is a significant difference with the data flow diagram. The arrows in DFDs show that there is a flow of data between the two components and not that the component is sending the data that must be executed in the following component. A component in DFD may not



continue execution when sending data and during execution of the component receiving the data. The component sending data can send multiple sets of data along several connections. In fact, a DFD node can be a component that never ends.

# USECASE DIAGRAM

In its most basic form, a use case diagram is a depiction of a user's interaction with the system that illustrates the connection between the user and the many use cases that the user is involved in. A "system" in this sense refers to something that is being created or run, like a website. The "actors" are individuals or groups functioning inside the system in designated roles.



**User**

## Class Diagram

A class diagram, as defined by the Unified Modeling Language (UML), is a kind of static structural diagram that illustrates a system's classes, properties, functions, and interactions between objects. The fundamental component of object-oriented modeling is the class diagram. It is utilized for both technical modeling—which converts the models into computer code—and general conceptual modeling of the applications systematic.

**Features selection**

+Upload the datasets()

+Features Extraction()

+MLP algorithm()

+CNN algorithm()

+CSV datasets

+Soil image datasets

+User details

+MLP classification()

+CNN classification()

+Crop details()

## Sequence Diagram

An object's interactions are arranged chronologically in a sequence diagram. It shows the classes and objects that are a part of the scenario as well as the messages that are passed between the objects in order for the scenario to work. Sequence diagrams are commonly linked to the realizations of use cases in the Logical View of the system that is being developed. Event diagrams or event scenarios are other names for sequence diagrams.

Datasets acquisition

Preprocessing

Rules construction

Classification

1. : Upload crop datasets()



1. : Upload soil image datasets()
2. : Missing value estimation()
3. : Irrelevant data removal()
4. : Noise removal()
5. : Features selection()
6. : MLP algorithm()
7. : CNN algorithm()
8. : Crop prediction()



## Activity Diagram

The activity diagram shows a unique kind of state diagram in which the majority of states are action states and the majority of transitions are brought about by the

fulfillment of actions in the source states. One may refer to the action as a system operation. As a result, the control flow is transferred across operations. This flow may occur concurrently, forked, or sequentially. Activity diagrams use a variety of features to address various forms of flow control.

# DATABASE DESIGN

Database Design is a collection of processes that facilitate the designing, development, implementation and maintenance of enterprise data management systems

It helps produce database systems

1. That meet the requirements of the users
2. Have high performance.

The main objectives of database designing are to produce logical and physical designs models of the proposed database system.

The logical model concentrates on the data requirements and the data to be stored independent of physical considerations. It does not concern itself with how the data will be stored or where it will be stored physically.

The physical data design model involves translating the logical design of the database onto physical media using hardware resources and software systems such as database management systems (DBMS).

Table structure for table admintb

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Null | Default |
| UserName | varchar(250) | Yes | NULL |
| Password | varchar(250) | Yes | NULL |

Table structure for table querytb

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Null | Default |
| *id* | bigint(250) | Yes | NULL |
| UserName | varchar(250) | Yes | NULL |
| Nitrogen | varchar(250) | Yes | NULL |
| Phosphorus | varchar(250) | Yes | NULL |
| Potassium | varchar(250) | Yes | NULL |
| Temperature | varchar(250) | Yes | NULL |
| Humidity | varchar(250) | Yes | NULL |
| PH | varchar(250) | Yes | NULL |
| Rainfall | varchar(250) | Yes | NULL |
| DResult | varchar(250) | Yes | NULL |
| CropInfo | varchar(250) | Yes | NULL |
| Fertilizer | varchar(1000) | Yes | NULL |
| Location | varchar(250) | Yes | NULL |

Table structure for table regtb

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Null | Default |
| Name | varchar(250) | Yes | NULL |
| Gender | varchar(250) | Yes | NULL |
| Age | varchar(250) | Yes | NULL |
| Email | varchar(250) | Yes | NULL |
| Mobile | varchar(250) | Yes | NULL |
| Address | varchar(250) | Yes | NULL |
| UserName | varchar(250) | Yes | NULL |
| Password | varchar(250) | Yes | NULL |

# RESULT

**SYSTEM IMPLEMENTATION**

Testing software in a "black box" means that the tester is not aware of the inner workings, architecture, or language of the module under test. Black box tests, like the majority of other test types, need to be created from a definite source document, such a requirements or specification document. In this type of testing, the software being tested is handled like a black box. One is unable to "see" into it. Without taking the software's operation into account, the test generates inputs and reacts to outputs, realization. Languages used for programming allow humans and machines to communicate. Features of programming languages and coding styles have a significant impact on the maintainability and quality of software. The following features are taken into consideration during coding.

* Translation from design to code is simple
* Memory efficiency
* Code efficiency
* Upgradability

The project's implementation phase is when the theoretical design is transformed into a functional system. As a result, it might be said to be the most important phase in creating a new system that is successful and instilling trust in the user regarding its functionality.

# SOURCE CODE

*# Importing essential libraries*

**import** numpy **as** np **import** pandas **as** pd **import** pickle **import** warnings

*# Loading the dataset*

df = pd.read\_csv(**'./Crop\_recommendation.csv'**)

**'''def clean\_dataset(df):**

**assert isinstance(df, pd.DataFrame), "df needs to be a pd.DataFrame" df.dropna(inplace=True)**

**indices\_to\_keep = ~df.isin([np.nan, np.inf, -np.inf]).any(1) return df[indices\_to\_keep].astype(np.float64)**

**df = clean\_dataset(df)'''**

*#import pandas as pd*

**import** matplotlib.pyplot **as** plt

*# read-in data*

*#data = pd.read\_csv('./test.csv', sep='\t') #adjust sep to your needs*

**import** seaborn **as** sns sns.countplot(df[**'label'**],label=**"Count"**) plt.show()

df.label=df.label.map({**'rice'**:0,

**'maize'**:1,

**'chickpea'**:2, **'kidneybeans'**:3, **'pigeonpeas'**:4, **'mothbeans'**:5, **'mungbean'**:6, **'blackgram'**:7, **'lentil'**:8, **'pomegranate'**:9,

**'banana'**:10,

**'mango'**:11,

**'grapes'**:12,

**'watermelon'**:13, **'muskmelon'**:14, **'apple'**: 15,

**'orange'**: 16,

**'papaya'**: 17,

**'coconut'**: 18,

**'cotton'**: 19,

**'jute'**: 20,

**'coffee'**: 21})

*# Replacing the 0 values from ['Glucose','BloodPressure','SkinThickness','Insulin','BMI'] by NaN* df\_copy = df.copy(deep=**True**) df\_copy[[**'N'**,**'P'**,**'K'**,**'temperature'**,**'humidity'**,**'ph'**,**'rainfall'**]] =

df\_copy[[**'N'**,**'P'**,**'K'**,**'temperature'**,**'humidity'**,**'ph'**,**'rainfall'**]].replace(0,np.NaN)

*# Model Building*

**from** sklearn.model\_selection **import** train\_test\_split df.drop(df.columns[np.isnan(df).any()], axis=1)

X = df.drop(columns=**'label'**) y = df[**'label'**]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.20, random\_state=0)

**from** sklearn.neural\_network **import** MLPClassifier

**from** sklearn.metrics **import** classification\_report

classifier = MLPClassifier(random\_state=0) classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test) print(classification\_report(y\_test, y\_pred))

clreport = classification\_report(y\_test, y\_pred) print(**"Accuracy on RandomForestClassifier training set:**

**{:.2f}"**.format(classifier.score(X\_train, y\_train))) print(**"Accuracy on RandomForestClassifier test set:**

**{:.3f}"**.format(classifier.score(X\_test, y\_test)))

filename = **'crop-prediction-rfc-model.pkl'**

pickle.dump(classifier, open(filename, **'wb'**))

**from** sklearn.neighbors **import** KNeighborsClassifier

**from** sklearn.metrics **import** classification\_report

classifier = KNeighborsClassifier() classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test) print(classification\_report(y\_test, y\_pred))

clreport = classification\_report(y\_test, y\_pred) print(**"Accuracy on KNeighborsClassifier training set:**

**{:.2f}"**.format(classifier.score(X\_train, y\_train))) print(**"Accuracy on KNeighborsClassifier test set:**

**{:.3f}"**.format(classifier.score(X\_test, y\_test)))

**from** sklearn.svm **import** SVC

**from** sklearn.metrics **import** classification\_report

classifier = SVC() classifier.fit(X\_train, y\_train)

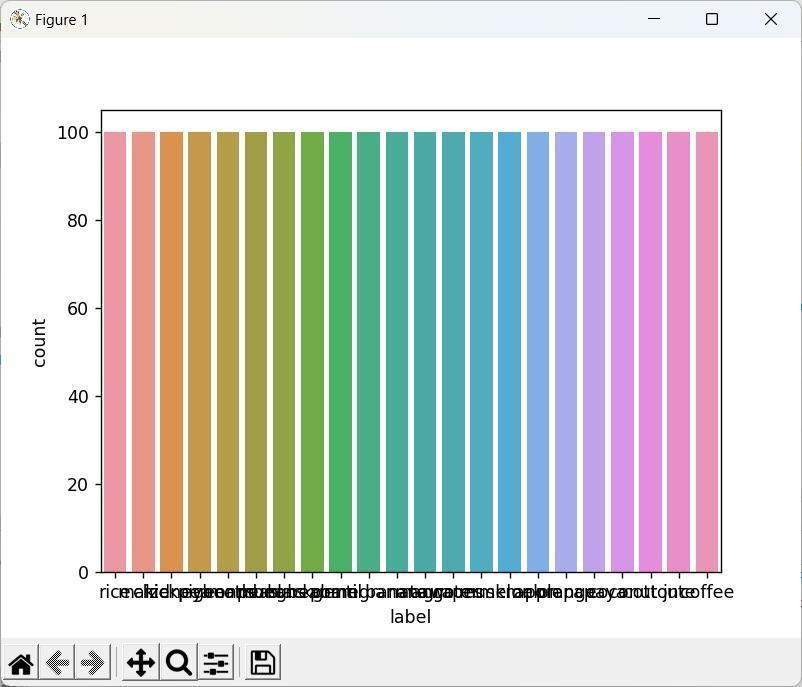
y\_pred = classifier.predict(X\_test) print(classification\_report(y\_test, y\_pred))

clreport = classification\_report(y\_test, y\_pred)

print(**"Accuracy on SVC training set: {:.2f}"**.format(classifier.score(X\_train, y\_train)))

print(**"Accuracy on SVC test set: {:.3f}"**.format(classifier.score(X\_test, y\_test)))

*# Creating a pickle file for the classifier #filename = 'crop-prediction-rfc-model.pkl'*



*#pickle.dump(classifier, open(filename, 'wb'))*

"F:\New Python Projects 2024- 2025\CropRecommendationPyNew\venv\Scripts\python.exe""F:/New Python Projects 2024-2025/CropRecommendationPyNew/Crop/CropDeployment.py" F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\seaborn\\_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be

`data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

FutureWarning

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\neural\_network\multilayer\_perceptron.py:566: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and the optimization hasn't converged yet.

% self.max\_iter, ConvergenceWarning) precision recall f1-score support

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | 0.96 | 0.92 | 0.94 | 25 |
| 1 | 0.94 | 0.89 | 0.91 | 18 |
| 2 | 1.00 | 1.00 | 1.00 | 23 |
| 3 | 1.00 | 1.00 | 1.00 | 20 |
| 4 | 1.00 | 1.00 | 1.00 | 22 |
| 5 | 0.95 | 0.84 | 0.89 | 25 |
| 6 | 1.00 | 1.00 | 1.00 | 17 |
| 7 | 0.87 | 0.91 | 0.89 | 22 |
| 8 | 0.84 | 0.94 | 0.89 | 17 |
| 9 | 1.00 | 1.00 | 1.00 | 23 |
| 10 | 1.00 | 1.00 | 1.00 | 18 |
| 11 | 0.95 | 1.00 | 0.98 | 21 |
| 12 | 1.00 | 1.00 | 1.00 | 18 |
| 13 | 1.00 | 1.00 | 1.00 | 17 |
| 14 | 1.00 | 1.00 | 1.00 | 23 |
| 15 | 1.00 | 1.00 | 1.00 | 18 |
| 16 | 1.00 | 1.00 | 1.00 | 23 |
| 17 | 1.00 | 0.95 | 0.98 | 21 |
| 18 | 1.00 | 1.00 | 1.00 | 15 |
| 19 | 0.88 | 0.94 | 0.91 | 16 |
| 20 | 0.91 | 0.95 | 0.93 | 21 |
| 21 | 1.00 | 1.00 | 1.00 | 17 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| accuracy |  | 0.97 | 440 |  |
| macro avg | 0.97 | 0.97 | 0.97 | 440 |
| weighted avg | 0.97 | 0.97 | 0.97 | 440 |

Accuracy on RandomForestClassifier training set: 0.98 Accuracy on RandomForestClassifier test set: 0.968

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:30: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations method='lar', copy\_X=True, eps=np.finfo(np.float).eps,

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:167: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations

method='lar', copy\_X=True, eps=np.finfo(np.float).eps,

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:284: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations eps=np.finfo(np.float).eps, copy\_Gram=True, verbose=0,

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:862: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations eps=np.finfo(np.float).eps, copy\_X=True, fit\_path=True,

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:1101: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations eps=np.finfo(np.float).eps, copy\_X=True, fit\_path=True,

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:1127: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations eps=np.finfo(np.float).eps, positive=False):

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:1362: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations max\_n\_alphas=1000, n\_jobs=None, eps=np.finfo(np.float).eps,

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:1602: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations max\_n\_alphas=1000, n\_jobs=None, eps=np.finfo(np.float).eps,

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\linear\_model\least\_angle.py:1738: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations eps=np.finfo(np.float).eps, copy\_X=True, positive=False):

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\decomposition\online\_lda.py:29: DeprecationWarning: `np.float` is a deprecated alias for the builtin `float`. To silence this warning, use `float` by itself. Doing this will not modify any behavior and is safe. If you specifically wanted the numpy scalar type, use `np.float64` here.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations EPS = np.finfo(np.float).eps

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\neighbors\base.py:908: DeprecationWarning: `np.int` is a deprecated alias for the builtin `int`. To silence this warning, use `int` by itself. Doing this will not modify any behavior and is safe. When replacing `np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the precision. If you wish to review your current use, check the release note link for additional information.

Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations self.\_y = np.empty(y.shape, dtype=np.int)

precision recall f1-score support

0 0.91 0.80 0.85 25

1 0.95 1.00 0.97 18

2 1.00 1.00 1.00 23

3 0.91 1.00 0.95 20

4 1.00 0.91 0.95 22

5 1.00 1.00 1.00 25

6 1.00 1.00 1.00 17

7 1.00 1.00 1.00 22

8 1.00 1.00 1.00 17

9 1.00 1.00 1.00 23

10 1.00 1.00 1.00 18

11 1.00 1.00 1.00 21

12 1.00 1.00 1.00 18

13 1.00 1.00 1.00 17

14 1.00 1.00 1.00 23

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 15 | 1.00 | 1.00 | 1.00 | | 18 |
| 16 | 1.00 | 1.00 | 1.00 | | 23 |
| 17 | 1.00 | 1.00 | 1.00 | | 21 |
| 18 | 1.00 | 1.00 | 1.00 | | 15 |
| 19 | 1.00 | 0.94 | 0.97 | | 16 |
| 20 | 0.79 | 0.90 | 0.84 | | 21 |
| 21 | 1.00 | 1.00 | 1.00 | | 17 |
| accuracy |  |  | 0.98 | 440 | |
| macro avg | 0.98 | 0.98 | | 0.98 | 440 |
| weighted avg | 0.98 | | 0.98 | 0.98 | 440 |

Accuracy on KNeighborsClassifier training set: 0.99 Accuracy on KNeighborsClassifier test set: 0.977

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.

"avoid this warning.", FutureWarning)

F:\New Python Projects 2024-2025\CropRecommendationPyNew\venv\lib\site- packages\sklearn\metrics\classification.py:1437: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples.

'precision', 'predicted', average, warn\_for) precision recall f1-score support

0 1.00 0.04 0.08 25

1 0.00 0.00 0.00 18

2 1.00 0.09 0.16 23

3 0.00 0.00 0.00 20

4 0.00 0.00 0.00 22

5 0.00 0.00 0.00 25

6 1.00 0.24 0.38 17

7 0.00 0.00 0.00 22

8 0.00 0.00 0.00 17

9 1.00 0.09 0.16 23

10 1.00 0.11 0.20 18

11 1.00 0.05 0.09 21

12 1.00 0.06 0.11 18

13 1.00 0.18 0.30 17

14 1.00 0.30 0.47 23

15 1.00 0.50 0.67 18

16 0.00 0.00 0.00 23

17 1.00 0.05 0.09 21

18 0.04 1.00 0.07 15

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 19 | 1.00 | 0.44 | 0.61 | | 16 |
| 20 | 0.00 | 0.00 | 0.00 | | 21 |
| 21 | 1.00 | 0.06 | 0.11 | | 17 |
| accuracy |  |  | 0.13 | 440 | |
| macro avg | 0.59 | 0.14 | | 0.16 | 440 |
| weighted avg | 0.59 | | 0.13 | 0.15 | 440 |

Accuracy on SVC training set: 1.00 Accuracy on SVC test set: 0.127

Process finished with exit code 0

Main:

**from** flask **import** Flask, render\_template, flash, request, session,send\_file

**from** flask **import** render\_template, redirect, url\_for, request

*#from wtforms import Form, TextField, TextAreaField, validators, StringField,*

*SubmitField*

**from** werkzeug.utils **import** secure\_filename

**import** datetime **import** mysql.connector **import** sys

**import** pickle

**import** numpy **as** np

app = Flask( name ) app.config[**'DEBUG'**]

app.config[**'SECRET\_KEY'**] = **'7d441f27d441f27567d441f2b6176a'**

@app.route(**"/"**)

**def** homepage():

**return** render\_template(**'index.html'**) @app.route(**"/AdminLogin"**)

**def** AdminLogin():

**return** render\_template(**'AdminLogin.html'**)

@app.route(**"/UserLogin"**)

**def** UserLogin():

**return** render\_template(**'UserLogin.html'**)

@app.route(**"/NewUser"**)

**def** NewUser():

**return** render\_template(**'NewUser.html'**)

@app.route(**"/NewQuery1"**)

**def** NewQuery1():

**return** render\_template(**'NewQueryReg.html'**)

@app.route(**"/UploadDataset"**)

**def** UploadDataset():

**return** render\_template(**'ViewExcel.html'**)

@app.route(**"/AdminHome"**)

**def** AdminHome():

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

cur = conn.cursor() cur.execute(**"SELECT \* FROM regtb "**) data = cur.fetchall()

**return** render\_template(**'AdminHome.html'**,data=data)

@app.route(**"/UserHome"**)

**def** UserHome():

user = session[**'uname'**]

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM regtb where username='"**+ user + **"'"**)

data = cur.fetchall()

**return** render\_template(**'UserHome.html'**,data=data)

@app.route(**"/UQueryandAns"**)

**def** UQueryandAns():

uname = session[**'uname'**]

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM Querytb where UserName='"**+ uname + **"' and DResult='waiting'"**)

data = cur.fetchall()

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM Querytb where UserName='"**+ uname + **"' and DResult !='waiting'"**)

data1 = cur.fetchall()

**return** render\_template(**'UserQueryAnswerinfo.html'**, wait=data, answ=data1 )

@app.route(**"/adminlogin"**, methods=[**'GET'**, **'POST'**]) **def** adminlogin():

error = **None**

**if** request.method == **'POST'**:

**if** request.form[**'uname'**] == **'admin' and** request.form[**'password'**] == **'admin'**:

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM regtb "**) data = cur.fetchall()

flash(**'You are Logged In...!'**)

**return** render\_template(**'AdminHome.html'** , data=data)

## else:

flash(**'Username Or Password is Wrong...!'**)

**return** render\_template(**'AdminLogin.html'**, error=error)

@app.route(**"/userlogin"**, methods=[**'GET'**, **'POST'**]) **def** userlogin():

**if** request.method == **'POST'**: username = request.form[**'uname'**]

password = request.form[**'password'**] session[**'uname'**] = request.form[**'uname'**]

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

cursor = conn.cursor()

cursor.execute(**"SELECT \* from regtb where username='"**+ username + **"' and Password='"**+ password + **"'"**)

data = cursor.fetchone()

**if** data **is None**:

print(**"hai"**)

flash(**'Username Or Password is Wrong...!'**) **return** render\_template(**'UserLogin.html'**)

## else:

print(data[0])

session[**'uid'**] = data[0]

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM regtb where username='"**+ username + **"'**

**and Password='"**+ password + **"'"**) data = cur.fetchall() flash(**'You are Logged In...!'**)

**return** render\_template(**'UserHome.html'**, data=data)

@app.route(**"/newuser"**, methods=[**'GET'**, **'POST'**]) **def** newuser():

**if** request.method == **'POST'**:

name1 = request.form[**'name'**] gender1 = request.form[**'gender'**] Age = request.form[**'age'**]

email = request.form[**'email'**] pnumber = request.form[**'phone'**] address = request.form[**'address'**]

uname = request.form[**'uname'**] password = request.form[**'psw'**]

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

cursor = conn.cursor() cursor.execute(

**"INSERT INTO regtb VALUES ('"**+ name1 + **"','"**+ gender1 + **"','"**+ Age +

**"','"**+ email + **"','"**+ pnumber + **"','"**+ address + **"','"**+ uname + **"','"**+ password

+ **"')"**)

conn.commit() conn.close()

*# return 'file register successfully'*

**return** render\_template(**'UserLogin.html'**)

@app.route(**"/newquery"**, methods=[**'GET'**, **'POST'**]) **def** newquery():

**if** request.method == **'POST'**: uname = session[**'uname'**]

nitrogen = request.form[**'nitrogen'**] phosphorus = request.form[**'phosphorus'**] potassium = request.form[**'potassium'**] temperature = request.form[**'temperature'**]

humidity = request.form[**'humidity'**] ph = request.form[**'ph'**]

rainfall = request.form[**'rainfall'**] location = request.form[**'select'**]

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

cursor = conn.cursor()

cursor.execute(

**"INSERT INTO Querytb VALUES ('','"**+ uname + **"','"**+ nitrogen + **"','"**+ phosphorus + **"','"**+ potassium + **"','"**+temperature+**"','"**+humidity +**"','"**+ ph

+**"','"**+ rainfall +**"','waiting','','','"**+location+**"')"**) conn.commit()

conn.close()

*# return 'file register successfully'*

uname = session[**'uname'**]

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM Querytb where UserName='"**+ uname + **"' and DResult='waiting'"**)

data = cur.fetchall()

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM Querytb where UserName='"**+ uname + **"' and DResult !='waiting'"**)

data1 = cur.fetchall()

**return** render\_template(**'UserQueryAnswerinfo.html'**, wait=data, answ=data1)

@app.route(**"/excelpost"**, methods=[**'GET'**, **'POST'**]) **def** excelpost():

**if** request.method == **'POST'**:

file = request.files[**'fileupload'**] file\_extension = file.filename.split(**'.'**)[1]

print(file\_extension)

*#file.save("static/upload/" + secure\_filename(file.filename))*

**import** pandas **as** pd

**import** matplotlib.pyplot **as** plt df = **''**

**if** file\_extension == **'xlsx'**:

df = pd.read\_excel(file.read(), engine=**'openpyxl'**)

**elif** file\_extension == **'xls'**:

df = pd.read\_excel(file.read())

**elif** file\_extension == **'csv'**:

df = pd.read\_csv(file)

print(df)

**import** seaborn **as** sns sns.countplot(df[**'label'**], label=**"Count"**) plt.savefig(**'static/images/out.jpg'**)

iimg = **'static/images/out.jpg'**

*#plt.show()*

*#df = pd.read\_csv("./Heart/Heartnew.csv") #def clean\_dataset(df):*

*#assert isinstance(df, pd.DataFrame), "df needs to be a pd.DataFrame" #df.dropna(inplace=True)*

*#indices\_to\_keep = ~df.isin([np.nan, np.inf, -np.inf]).any(1) #return df[indices\_to\_keep].astype(np.float64)*

*#df = clean\_dataset(df) #print("Preprocessing Completed")*

print(df)

*# import pandas as pd*

**import** matplotlib.pyplot **as** plt

*# read-in data*

*# data = pd.read\_csv('./test.csv', sep='\t') #adjust sep to your needs*

**import** seaborn **as** sns sns.countplot(df[**'label'**], label=**"Count"**) plt.show()

df.label = df.label.map({**'rice'**: 0,

**'maize'**: 1,

**'chickpea'**: 2,

**'kidneybeans'**: 3,

**'pigeonpeas'**: 4,

**'mothbeans'**: 5,

**'mungbean'**: 6,

**'blackgram'**: 7,

**'lentil'**: 8,

**'pomegranate'**: 9,

**'banana'**: 10,

**'mango'**: 11,

**'grapes'**: 12,

**'watermelon'**: 13,

**'muskmelon'**: 14,

**'apple'**: 15,

**'orange'**: 16,

**'papaya'**: 17,

**'coconut'**: 18,

**'cotton'**: 19,

**'jute'**: 20,

**'coffee'**: 21})

*# Replacing the 0 values from ['Glucose','BloodPressure','SkinThickness','Insulin','BMI'] by NaN* df\_copy = df.copy(deep=**True**)

df\_copy[[**'N'**, **'P'**, **'K'**, **'temperature'**, **'humidity'**, **'ph'**, **'rainfall'**]] = df\_copy[

[**'N'**, **'P'**, **'K'**, **'temperature'**, **'humidity'**, **'ph'**, **'rainfall'**]].replace(0, np.NaN)

*# Model Building*

**from** sklearn.model\_selection **import** train\_test\_split df.drop(df.columns[np.isnan(df).any()], axis=1) X = df.drop(columns=**'label'**)

y = df[**'label'**]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.20, random\_state=0)

**from** sklearn.neural\_network **import** MLPClassifier

**from** sklearn.metrics **import** classification\_report classifier = MLPClassifier(random\_state=0) classifier.fit(X\_train, y\_train)

y\_pred = classifier.predict(X\_test) print(classification\_report(y\_test, y\_pred))

clreport = classification\_report(y\_test, y\_pred)

print(**"Accuracy on training set: {:.2f}"**.format(classifier.score(X\_train, y\_train))) print(**"Accuracy on test set: {:.3f}"**.format(classifier.score(X\_test, y\_test)))

Tacc = **"Accuracy on training set: {:.2f}"**.format(classifier.score(X\_train, y\_train))

Testacc = **"Accuracy on test set: {:.3f}"**.format(classifier.score(X\_test, y\_test))

*# Creating a pickle file for the classifier* filename = **'crop-prediction-rfc-model.pkl'** pickle.dump(classifier, open(filename, **'wb'**))

print(**"Training process is complete Model File Saved!"**) df= df.head(200)

*#read\_csv(..., skiprows=1000000, nrows=999999)*

**return** render\_template(**'ViewExcel.html'**, data=df.to\_html(), dataimg=iimg ,tacc=Tacc,testacc=Testacc,report=clreport)

@app.route(**"/AdminQinfo"**)

**def** AdminQinfo():

*#uname = session['uname']*

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM Querytb where DResult='waiting'"**) data = cur.fetchall()

**return** render\_template(**'AdminQueryInfo.html'**, data=data ) @app.route(**"/answer"**)

**def** answer():

Answer = **''**

Prescription=**''**

id = request.args.get(**'lid'**)

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

cursor = conn.cursor()

cursor.execute(**"SELECT \* FROM Querytb where id='"**+ id + **"'"**) data = cursor.fetchone()

**if** data:

UserName = data[1] nitrogen = data[2] phosphorus = data[3] potassium = data[4] temperature = data[5] humidity = data[6] ph = data[7]

rainfall = data[8]

## else:

**return 'Incorrect username / password !'**

nit = float(nitrogen)

pho = float(phosphorus) po = float(potassium) te = float(temperature) hu = float(humidity) phh = float(ph)

ra = float(rainfall)

*#age = int(age)*

filename = **'crop-prediction-rfc-model.pkl'**

classifier = pickle.load(open(filename, **'rb'**))

data = np.array([[nit, pho, po, te, hu, phh, ra ]]) my\_prediction = classifier.predict(data)

print(my\_prediction)

crop = **''**

fertilizer = **''**

**if** my\_prediction == 0: Answer = **'Predict'**

crop = **'rice'**

## fertilizer ='4 kg of gypsum and 1 kg of DAP/cent can be applied at 10 days after sowing'

**elif** my\_prediction == 1: Answer = **'Predict'**

crop = **'maize'**

## fertilizer = 'The standard fertilizer recommendation for maize consists of 150 kg ha−1 NPK 14–23–14 and 50 kg ha−1 urea'

**elif** my\_prediction == 2: Answer = **'Predict'**

crop = **'chickpea'**

## fertilizer = 'The generally recommended doses for chickpea include 20–30 kg nitrogen (N) and 40–60 kg phosphorus (P) ha-1. If soils are low in potassium (K), an application of 17 to 25 kg K ha-1 is recommended'

**elif** my\_prediction == 3: Answer = **'Predict'**

crop = **'kidneybeans'**

## fertilizer = 'It needs good amount of Nitrogen about 100 to 125 kg/ha'

**elif** my\_prediction == 4: Answer = **'Predict'**

crop = **'pigeonpeas'**

## fertilizer = 'Apply 25 - 30 kg N, 40 - 50 k g P 2 O 5 , 30 kg K 2 O per ha area as Basal dose at the time of sowing.'

**elif** my\_prediction == 5: Answer = **'Predict'**

crop = **'mothbeans'**

## fertilizer = 'The applications of 10 kg N+40 kg P2O5 per hectare have proved the effective starter dose'

**elif** my\_prediction == 6: Answer = **'Predict'**

crop = **'mungbean'**

## fertilizer = 'Phosphorus and potassium fertilizers should be applied at 50-50 kg ha-1'

**elif** my\_prediction == 7: Answer = **'Predict'**

crop = **'blackgram'**

## fertilizer = 'The recommended fertilizer dose for black gram is 20:40:40 kg NPK/ha.'

**elif** my\_prediction == 8:

Answer = **'Predict'**

crop = **'lentil'**

## fertilizer = 'The recommended dose of fertilizers is 20kg N, 40kg P, 20 kg K and

**20kg S/ha.'**

**elif** my\_prediction == 9: Answer = **'Predict'**

crop = **'pomegranate'**

## fertilizer = 'The recommended fertiliser dose is 600–700 gm of N, 200–250 gm of P2O5 and 200–250 gm of K2O per tree per year'

**elif** my\_prediction == 10: Answer = **'Predict'**

crop = **'banana'**

## fertilizer = 'Feed regularly using either 8-10-8 (NPK) chemical fertilizer or organic composted manure'

**elif** my\_prediction == 11: Answer = **'Predict'**

crop = **'mango'**

## fertilizer = '50 gm zinc sulphate, 50 gm copper sulphate and 20 gm borax per tree/annum are recommended'

**elif** my\_prediction == 12: Answer = **'Predict'**

crop = **'grapes'**

## fertilizer = 'Use 3 pounds (1.5 kg.) of potassium sulfate per vine for mild deficiencies or up to 6 pounds (3 kg.)'

**elif** my\_prediction == 13: Answer = **'Predict'**

crop = **'watermelon'**

## fertilizer = 'Apply a fertilizer high in phosphorous, such as 10-10-10, at a rate of 4 pounds per 1,000 square feet (60 to 90 feet of row)'

**elif** my\_prediction == 14: Answer = **'Predict'**

crop = **'muskmelon'**

## fertilizer = 'Apply FYM 20 t/ha, NPK 40:60:30 kg/ha as basal and N @ 40 kg/ha 30 days after sowing.'

**elif** my\_prediction == 15: Answer = **'Predict'**

crop = **'apple'**

## fertilizer = 'Apple trees require nitrogen, phosphorus and potassium,Common granular 20-10-10 fertilizer is suitable for apples'

**elif** my\_prediction == 16: Answer = **'Predict'**

crop = **'orange'**

## fertilizer = 'Orange farmers often provide 5,5 – 7,7 lbs (2,5-3,5 kg) P2O5 in every adult tree for 4-5 consecutive years'

**elif** my\_prediction == 17: Answer = **'Predict'**

crop = **'papaya'**

## fertilizer = 'Generally 90 g of Urea, 250 g of Super phosphate and 140 g of Muriate of Potash per plant are recommended for each application'

**elif** my\_prediction == 18: Answer = **'Predict'**

crop = **'coconut'**

## fertilizer = 'Organic Manure @50kg/palm or 30 kg green manure, 500 g N, 320 g P2O5 and 1200 g K2O/palm/year in two split doses during September and May'

**elif** my\_prediction == 19: Answer = **'Predict'**

crop = **'cotton'**

## fertilizer = 'N-P-K 20-10-10 per hectare during sowing (through the sowing machine)'

**elif** my\_prediction == 20: Answer = **'Predict'**

crop = **'jute'**

## fertilizer = 'Apply 10 kg of N at 20 - 25 days after first weeding and then again on 35 - 40 days after second weeding as top dressing'

**elif** my\_prediction == 21: Answer = **'Predict'**

crop = **'coffee'**

## fertilizer = 'Coffee trees need a lot of potash, nitrogen, and a little phosphoric acid. Spread the fertilizer in a ring around each Coffee plant'

**else**:

Answer = **'Predict'**

## crop='Crop info not Found!'

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

cursor = conn.cursor() cursor.execute(

**"update Querytb set DResult='"**+Answer+**"', CropInfo='"**+ crop

+**"',Fertilizer='"**+fertilizer+**"' where id='"**+ str(id) + **"' "**) conn.commit()

conn.close()

conn3 = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

cur3 = conn3.cursor()

cur3.execute(**"SELECT \* FROM regtb where UserName='"**+ str(UserName)

+ **"'"**)

data3 = cur3.fetchone()

**if** data3:

phnumber = data3[4] print(phnumber)

sendmsg(phnumber, **"Predict Crop Name : "**+ crop +**" For More info Visit in Site"**)

*# return 'file register successfully'*

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM Querytb where DResult !='waiting '"**) data = cur.fetchall()

**return** render\_template(**'AdminAnswer.html'**, data=data)

@app.route(**"/AdminAinfo"**)

**def** AdminAinfo():

conn = mysql.connector.connect(user=**'root'**, password=**''**, host=**'localhost'**, database=**'2croprecomdb'**)

*# cursor = conn.cursor()*

cur = conn.cursor()

cur.execute(**"SELECT \* FROM Querytb where DResult !='waiting'"**) data = cur.fetchall()

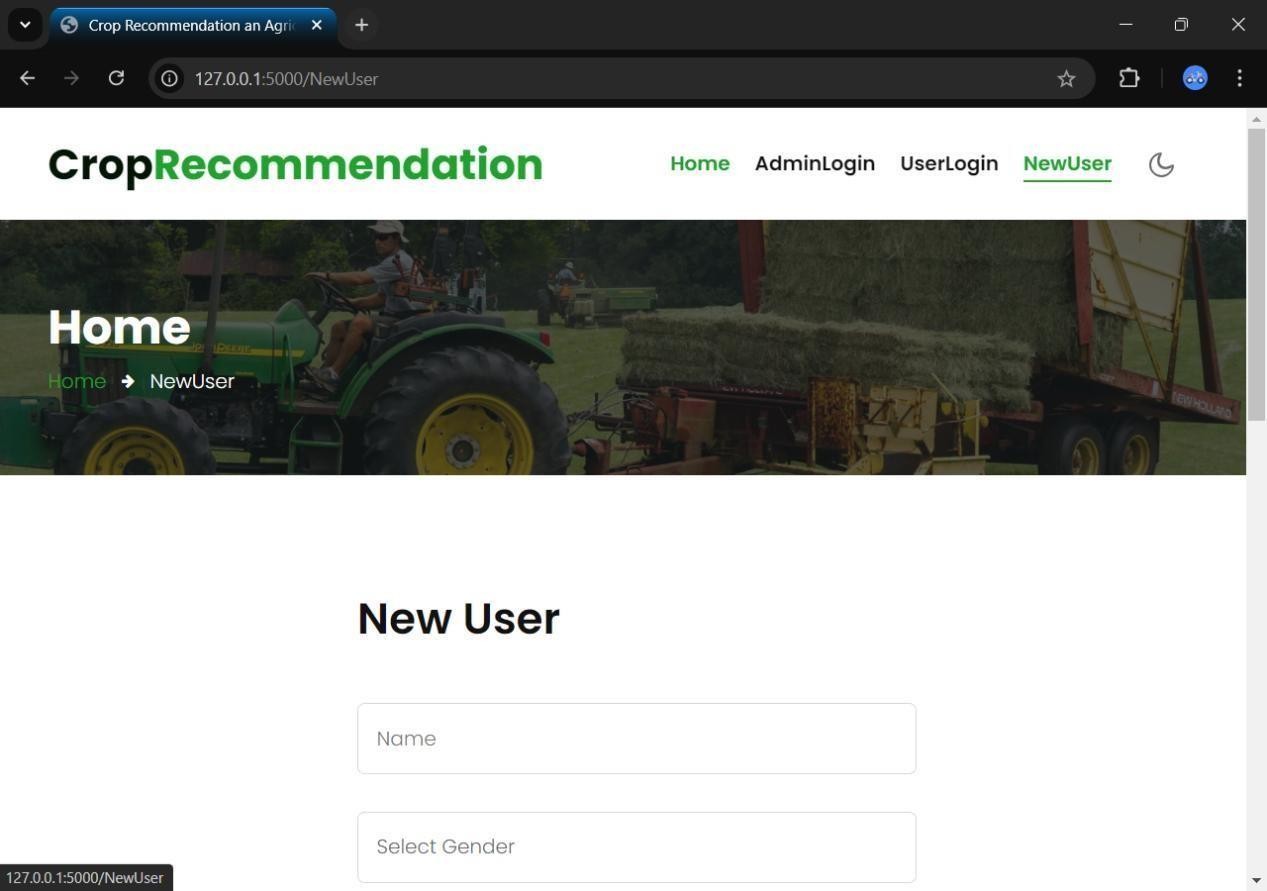
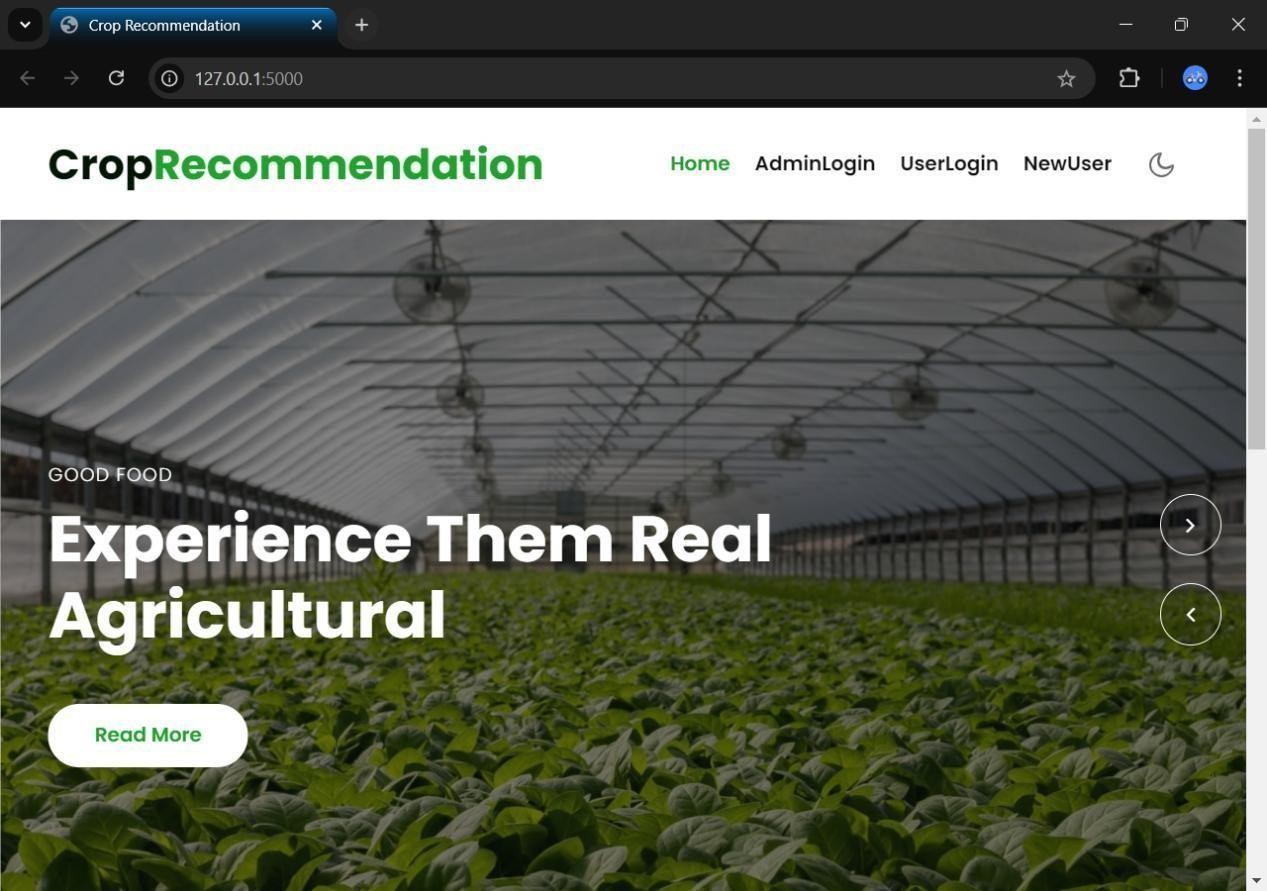
**return** render\_template(**'AdminAnswer.html'**, data=data )

**def** sendmsg(targetno,message):

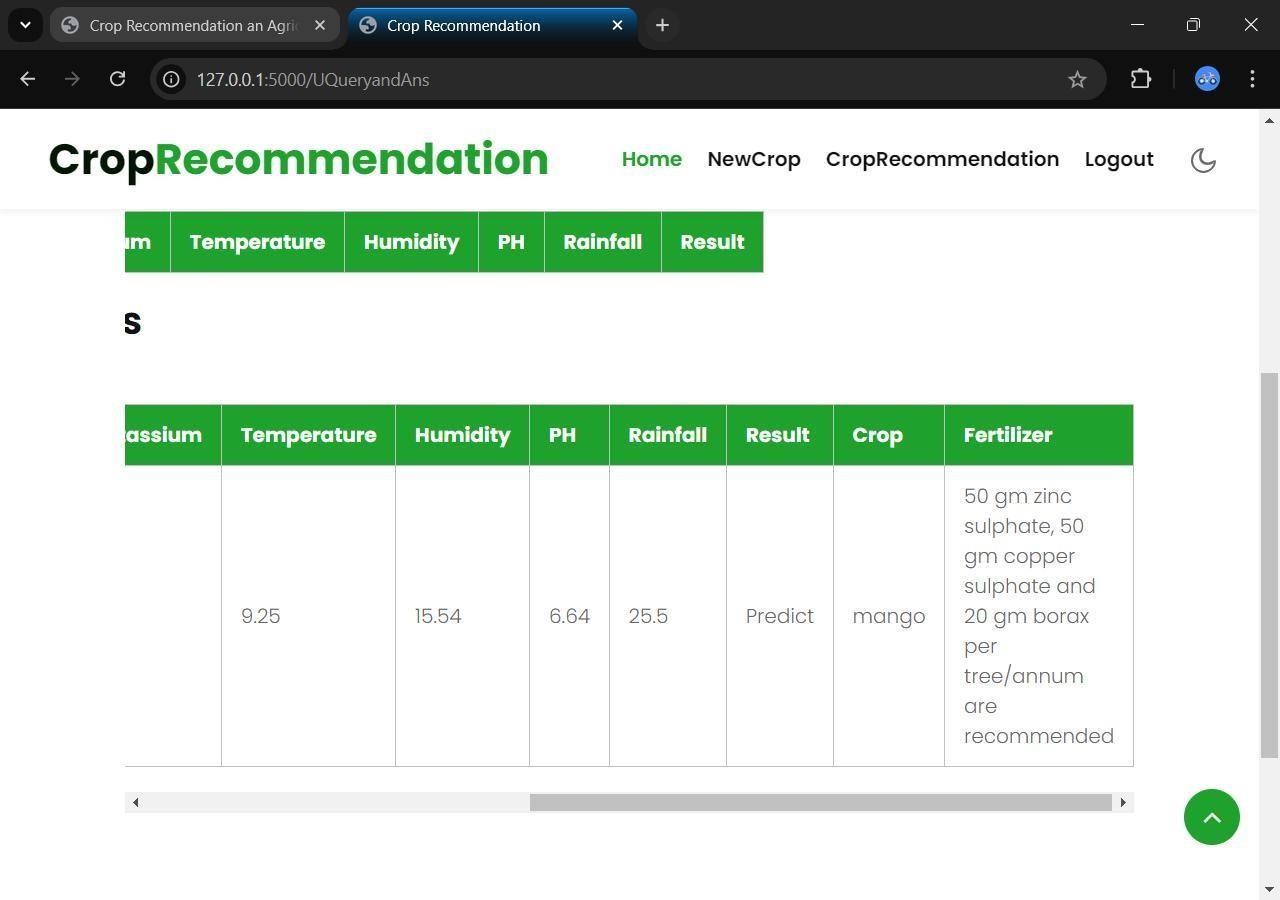
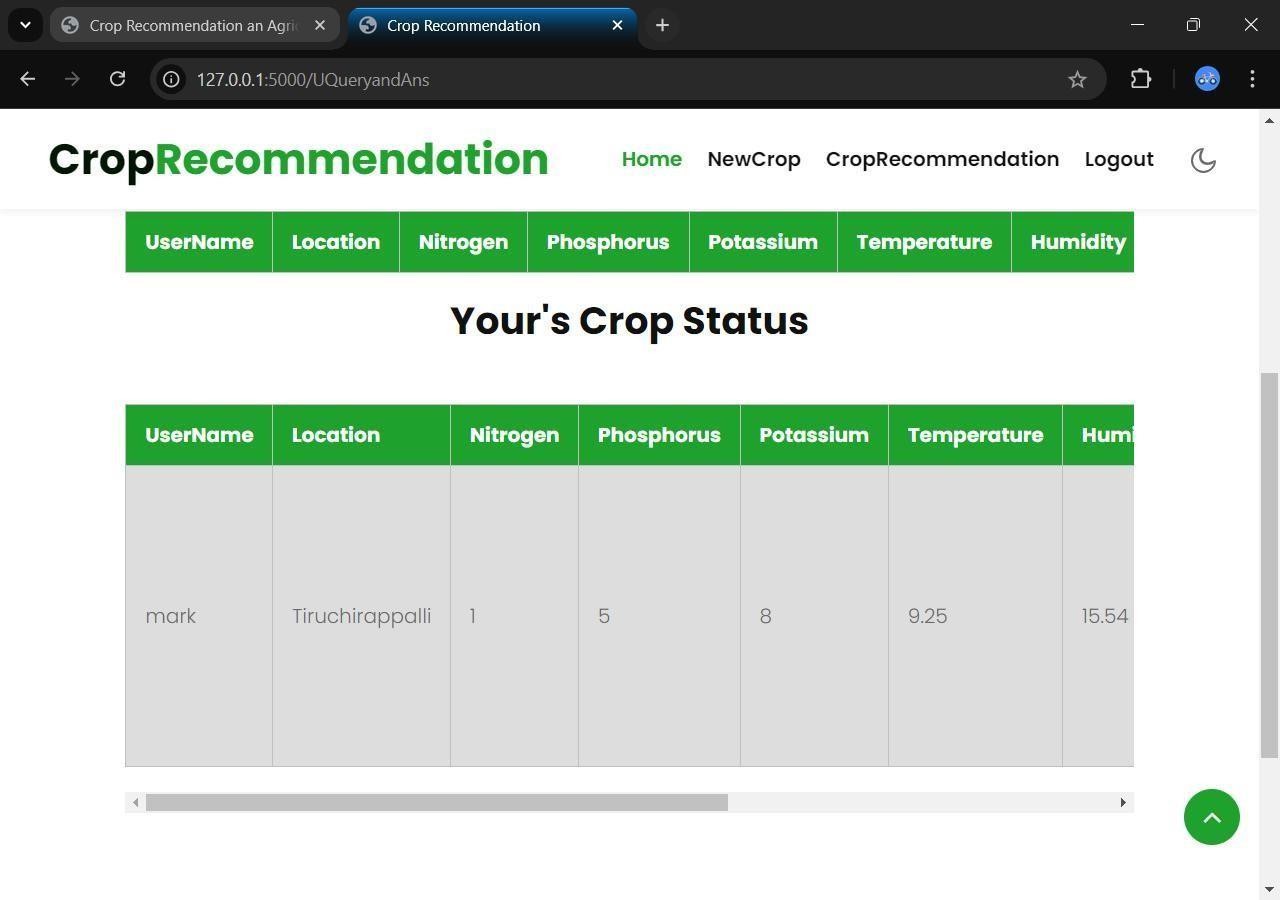
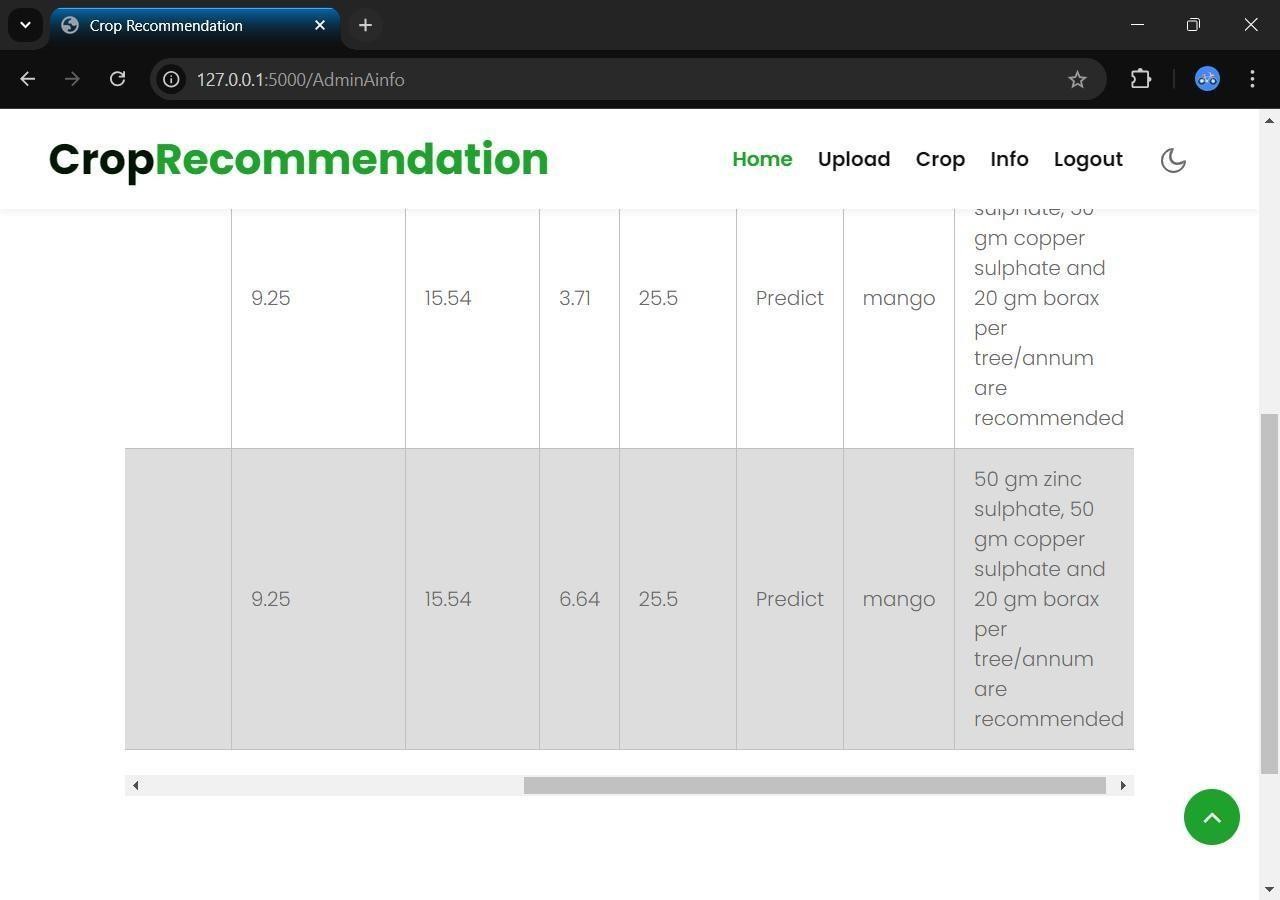
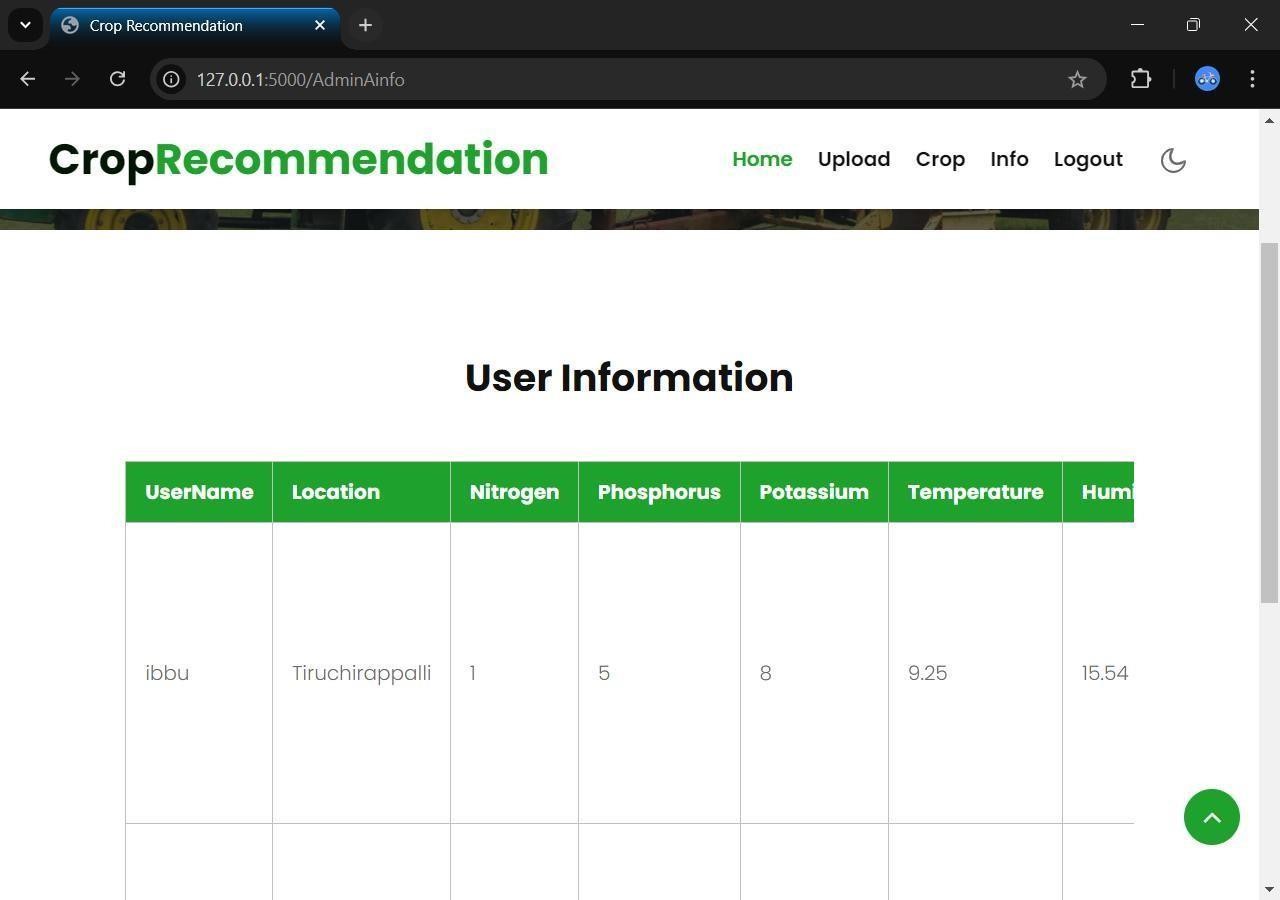
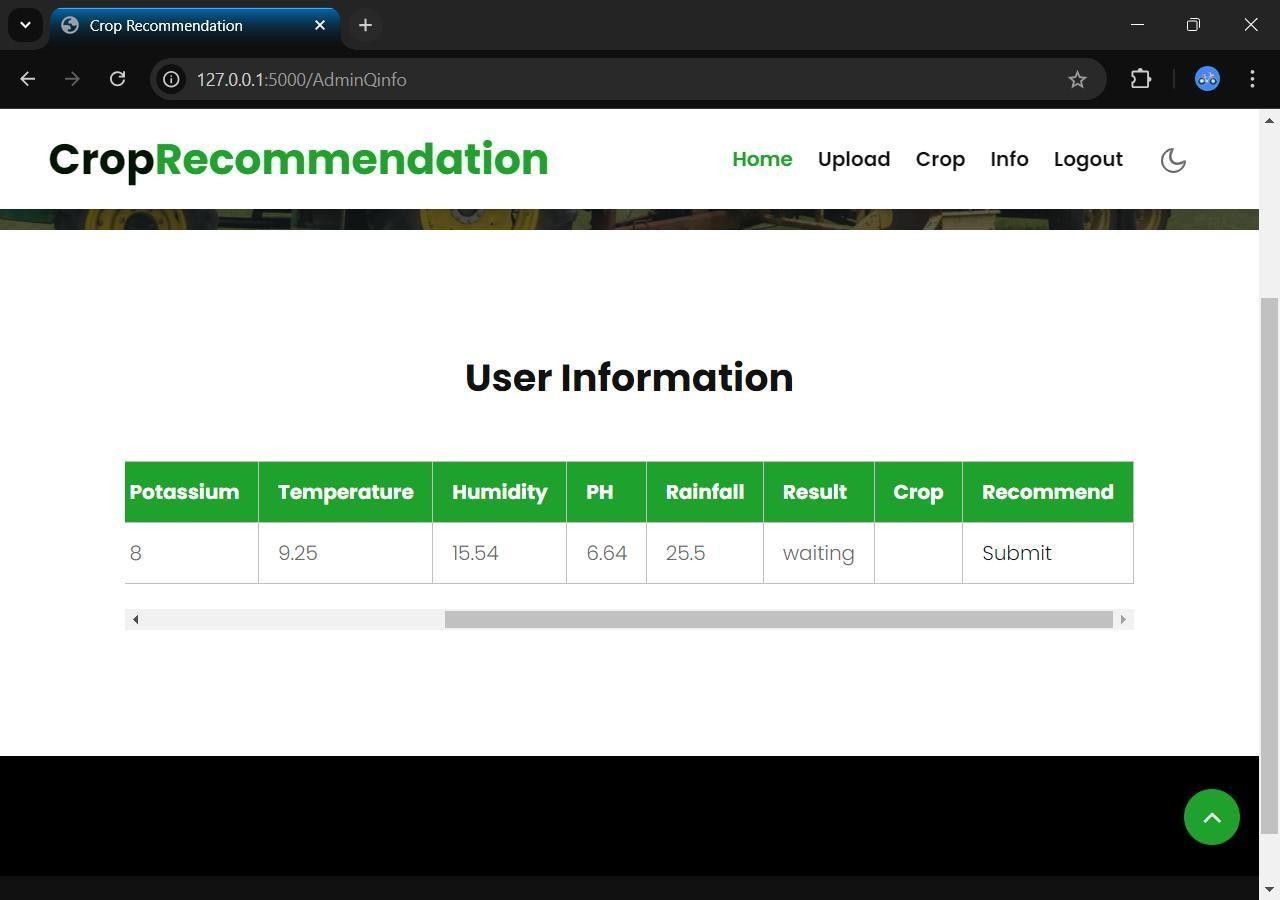
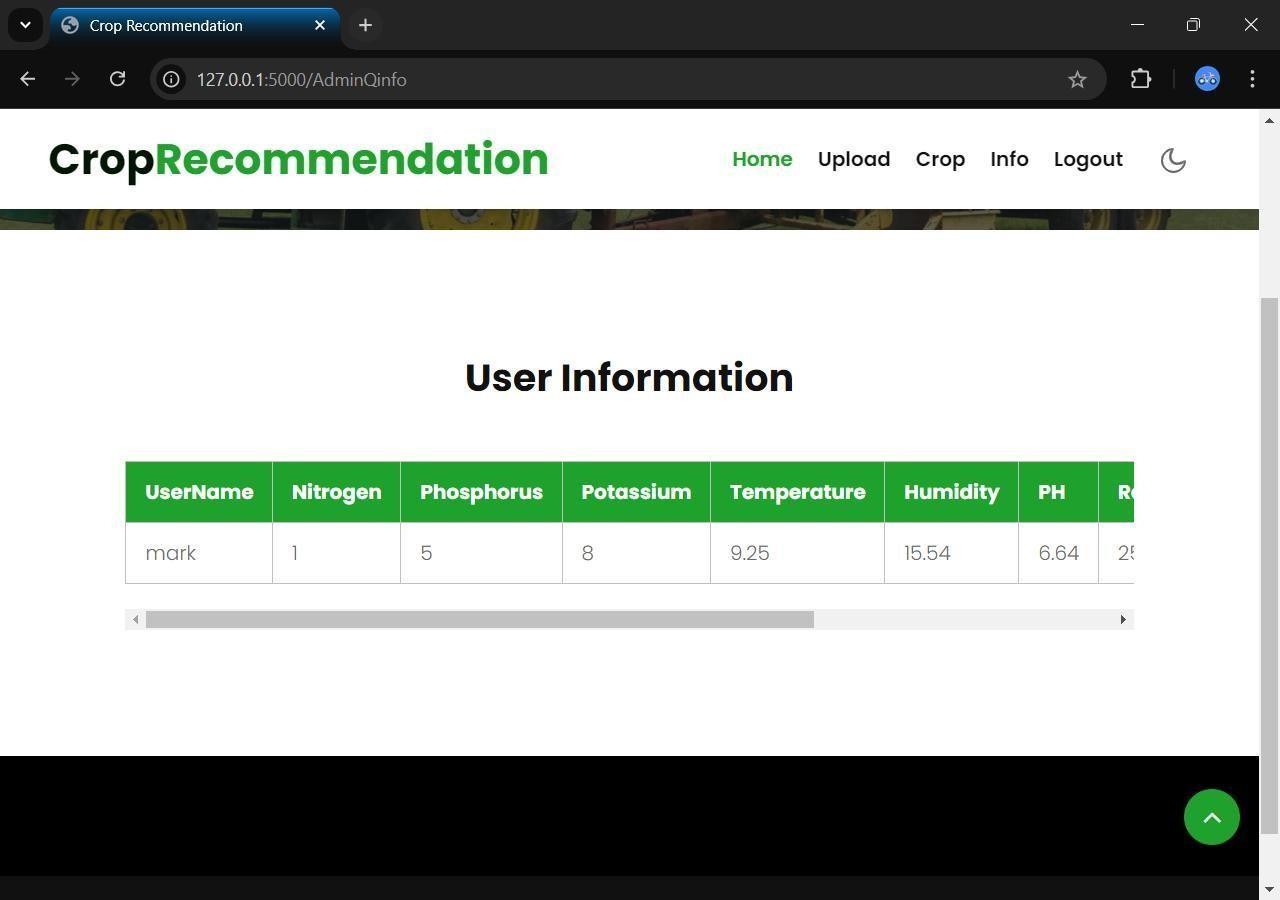
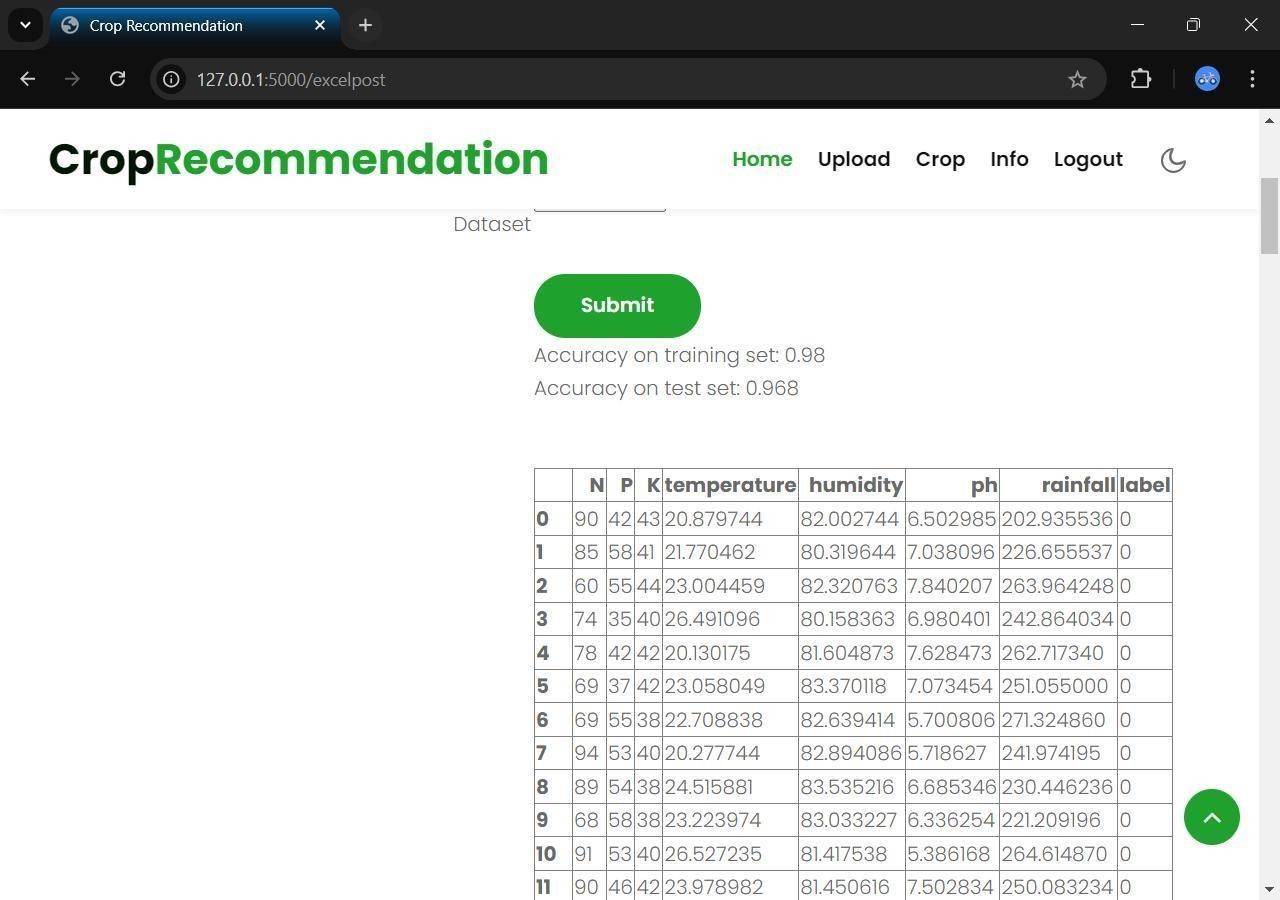
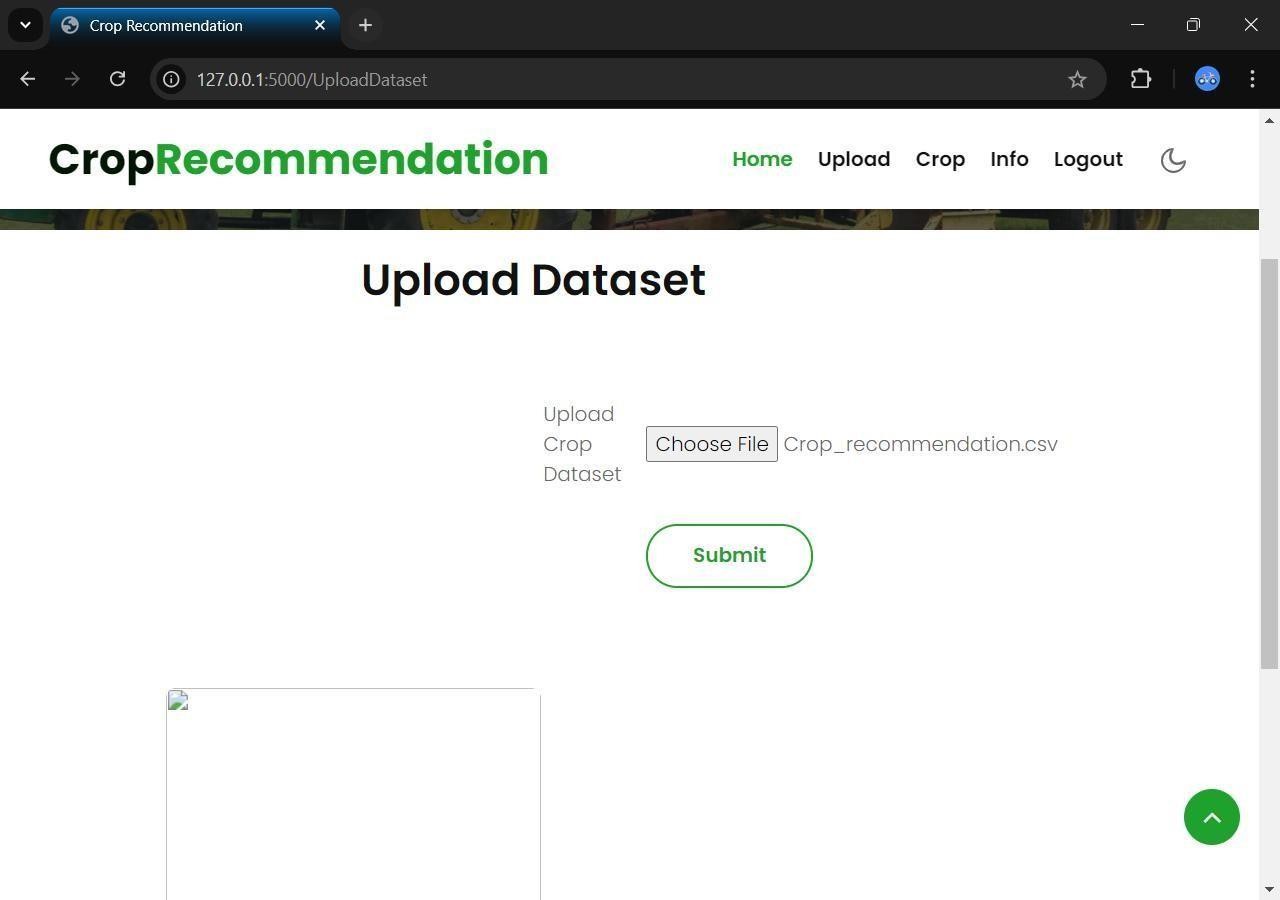
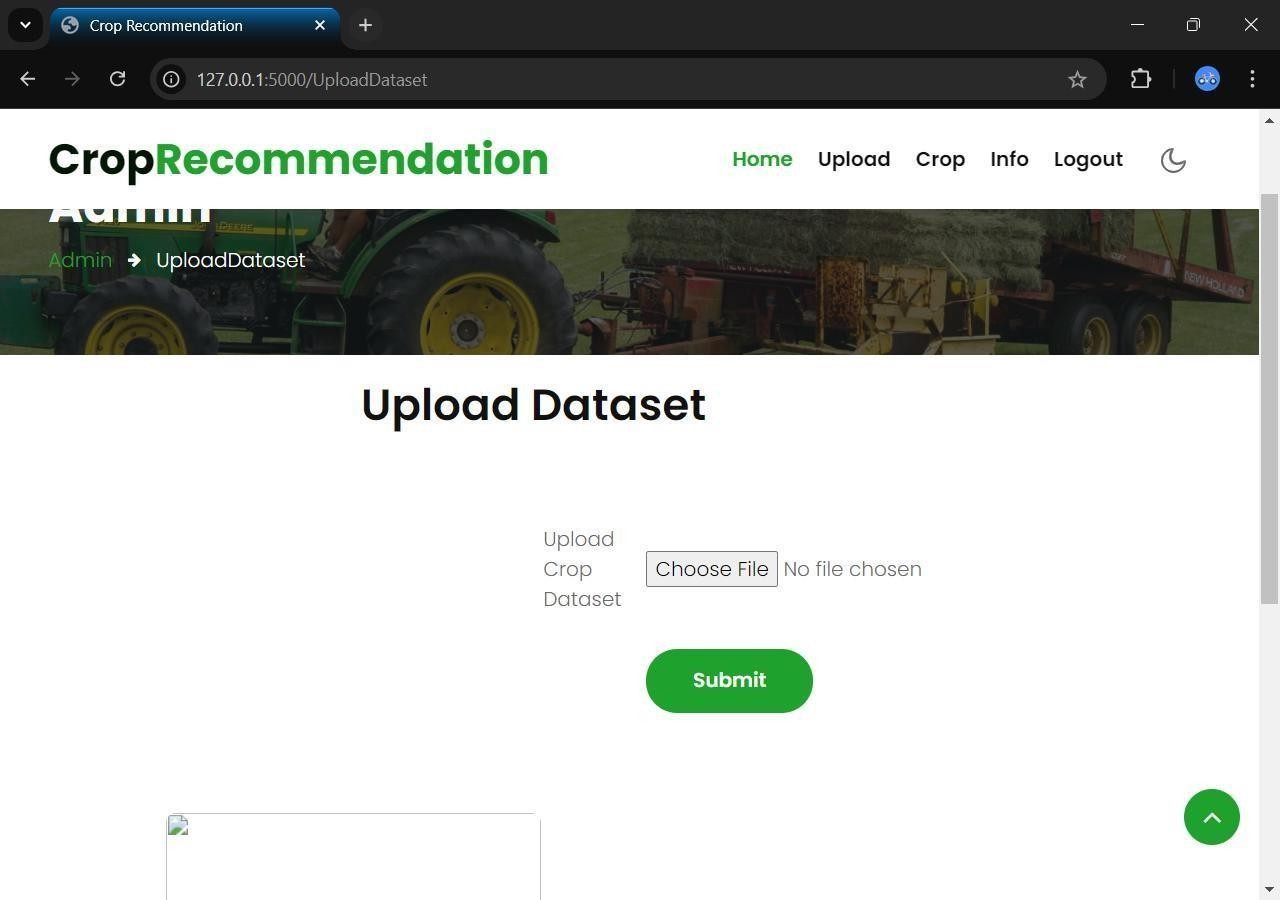
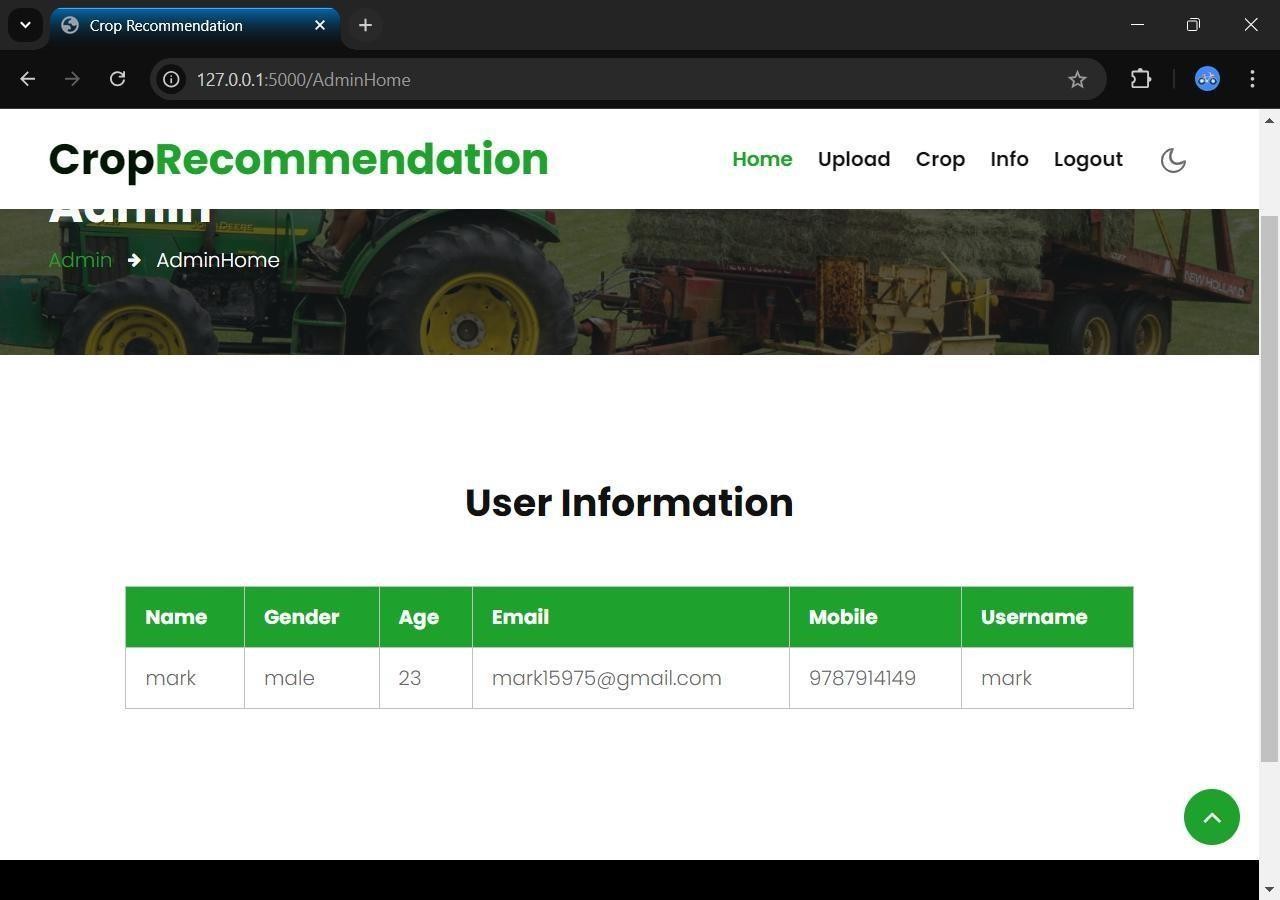
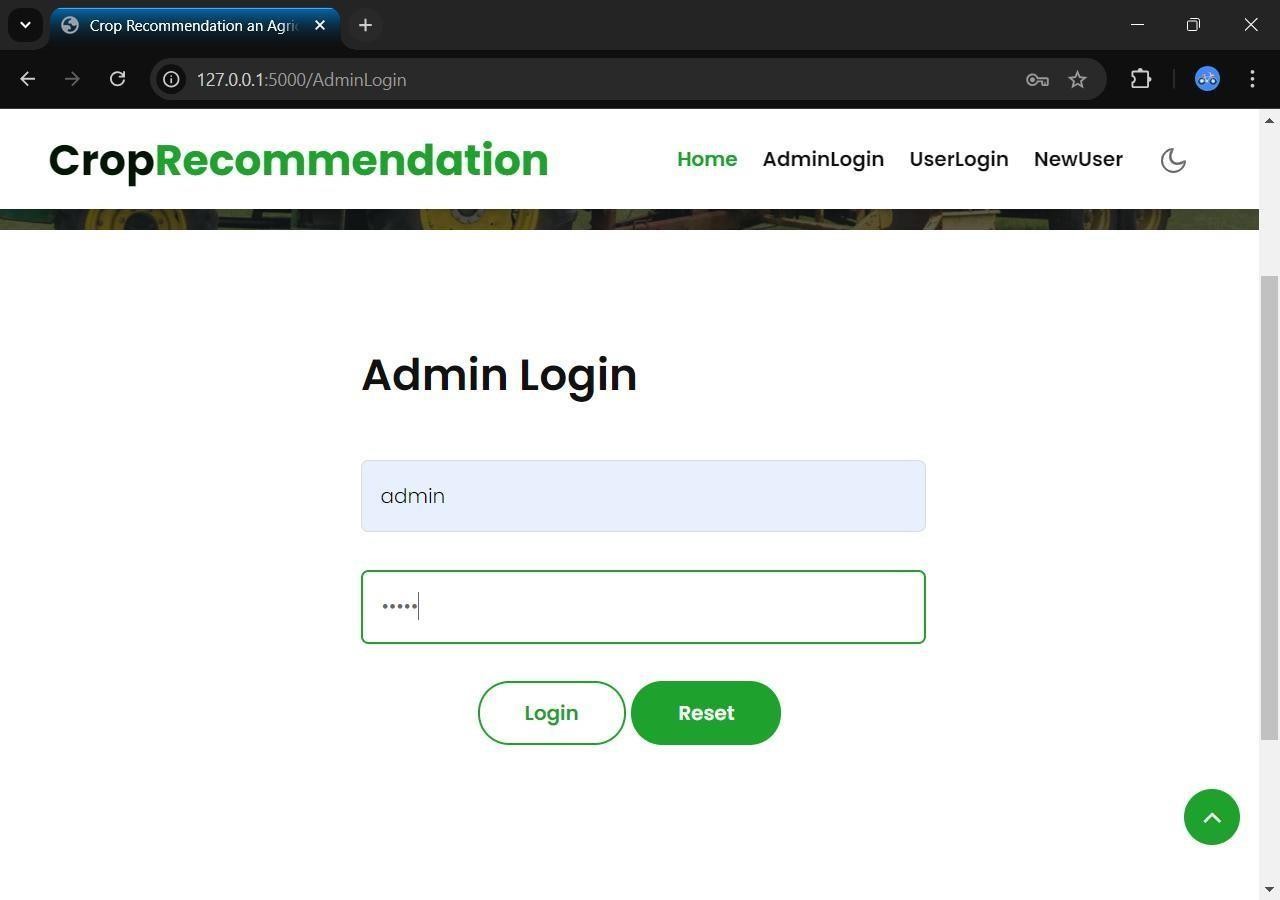
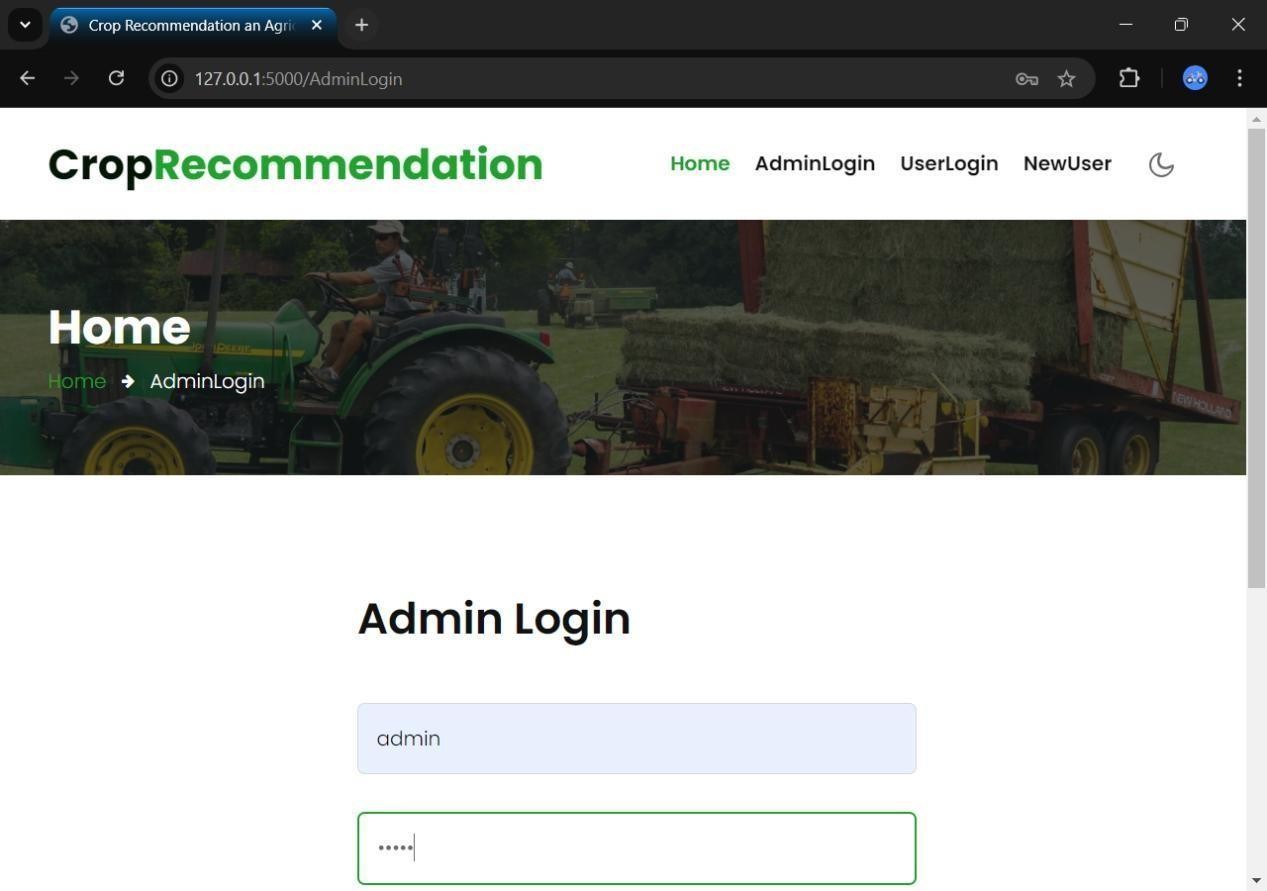
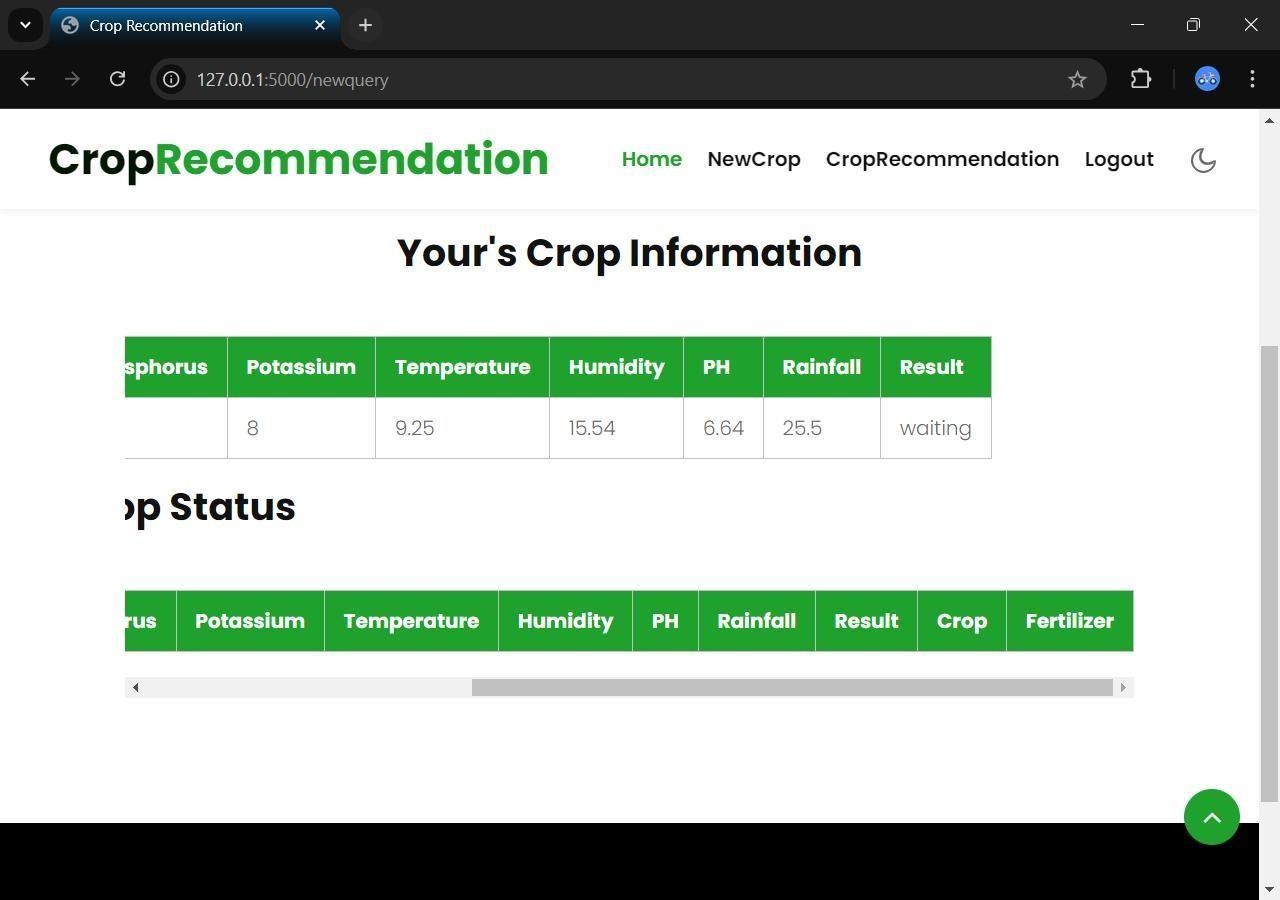
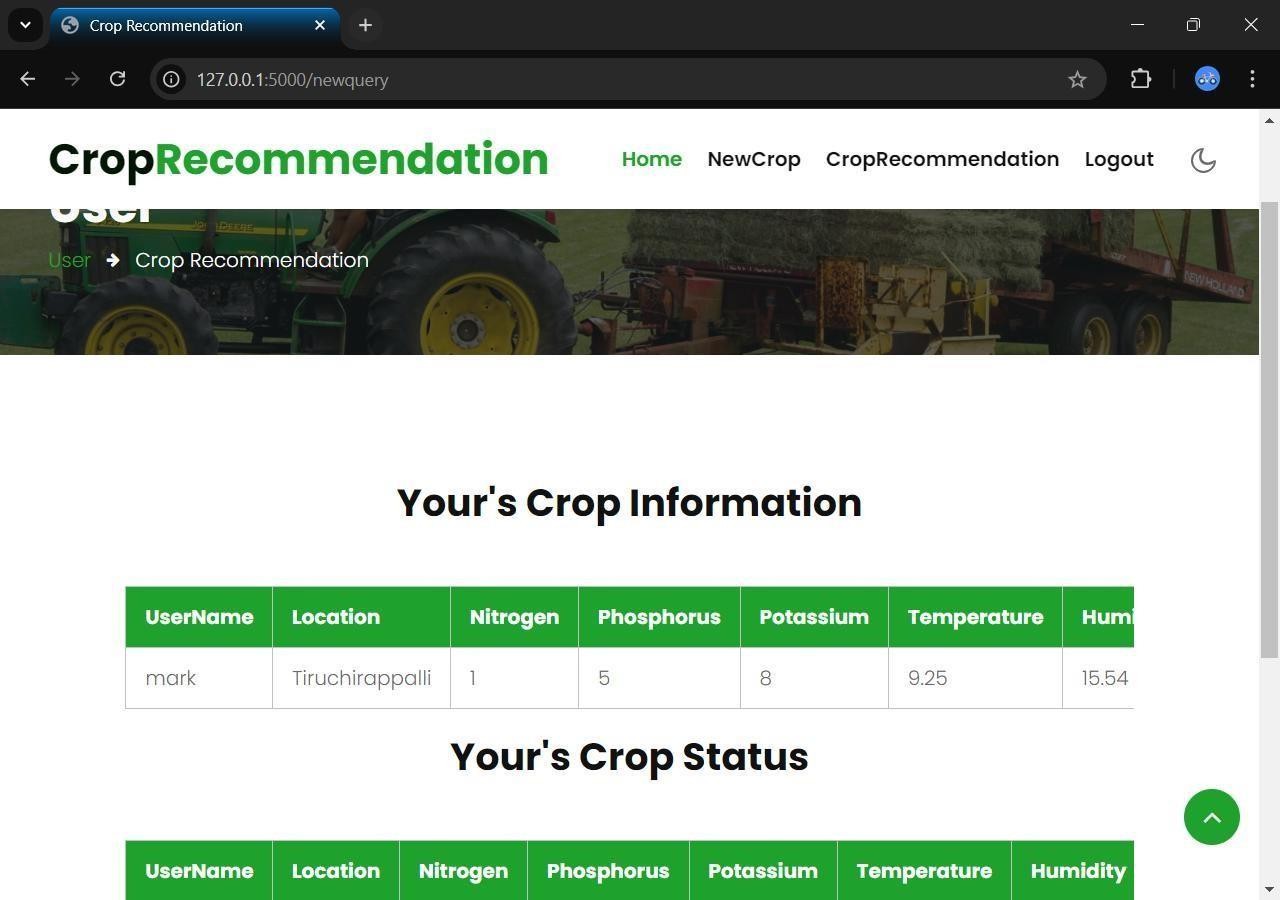
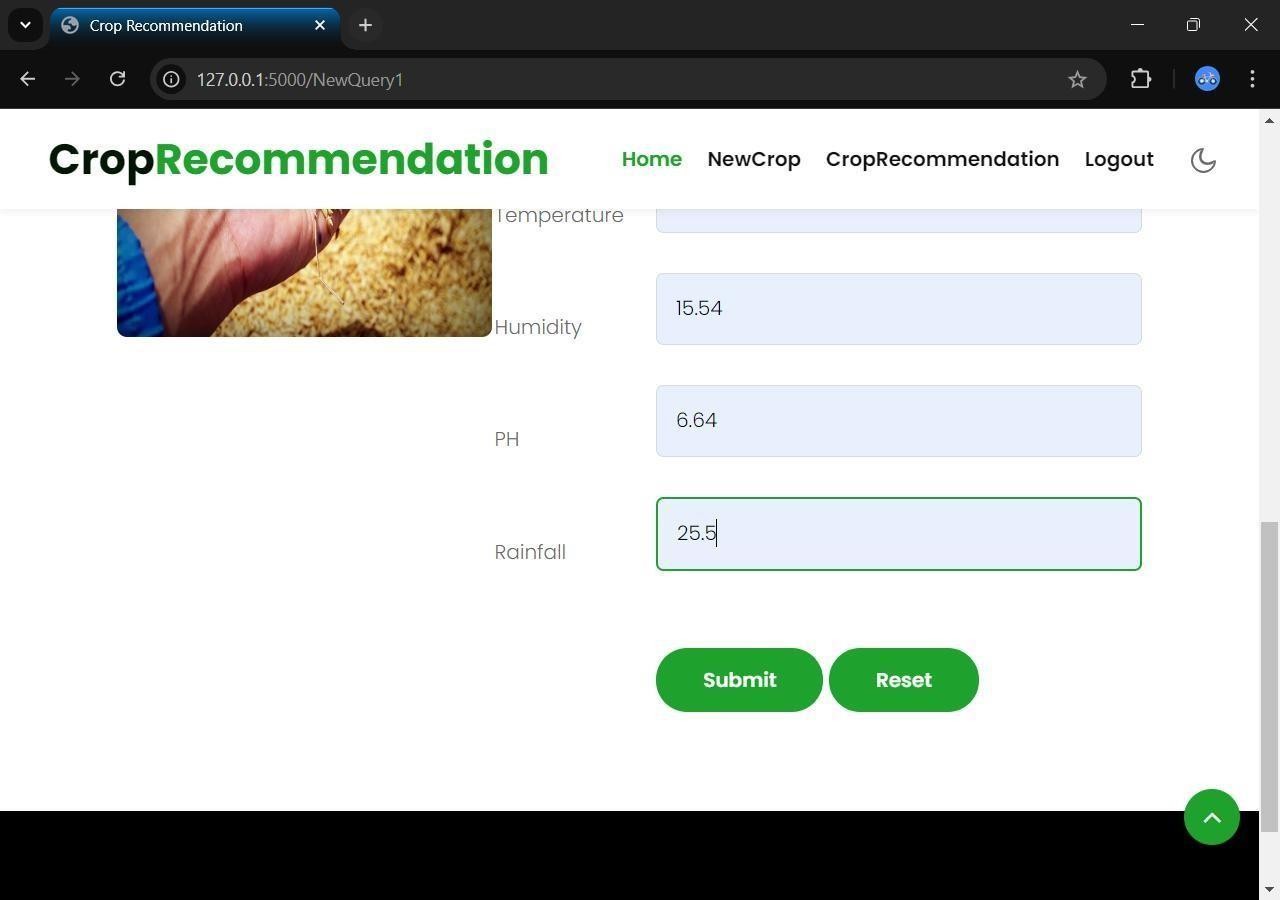
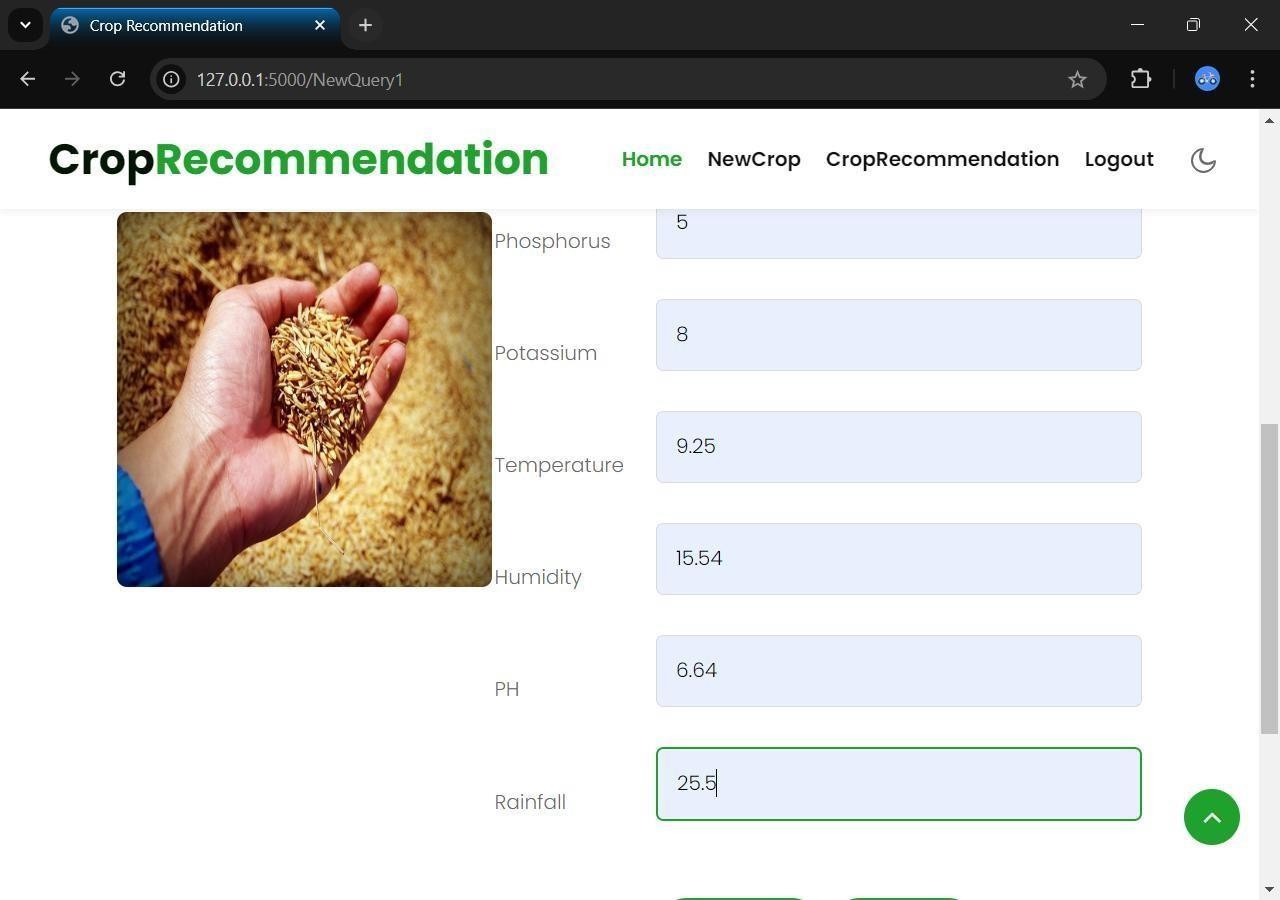
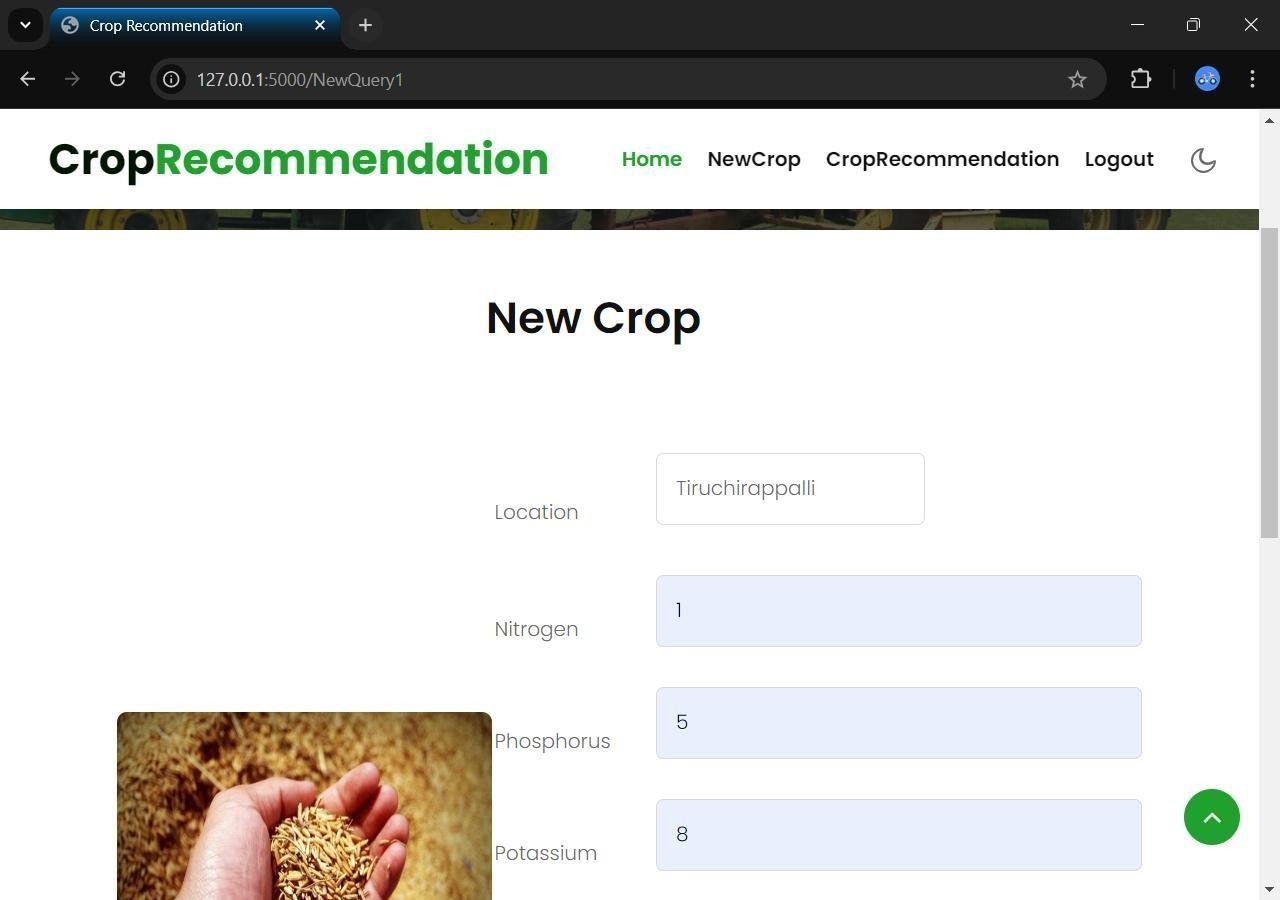
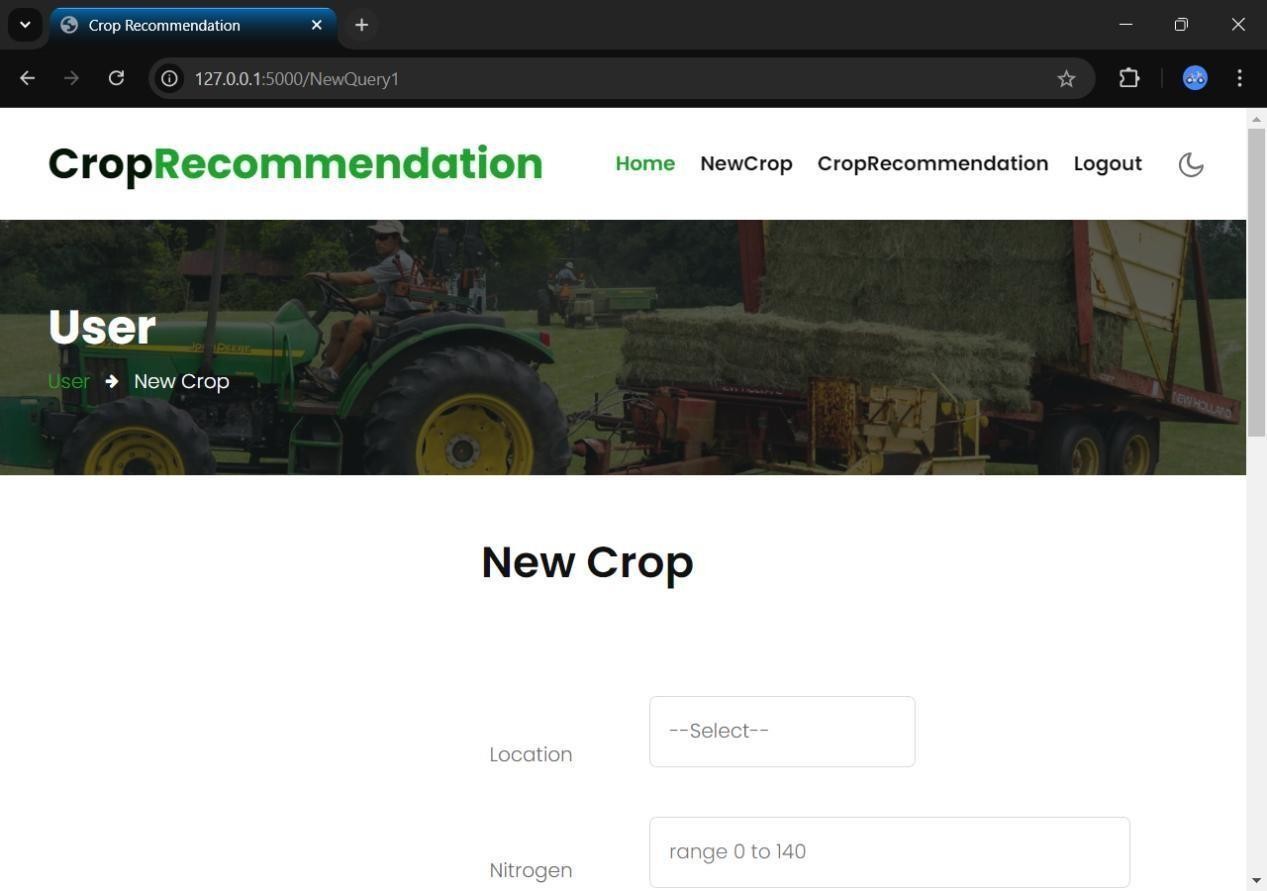
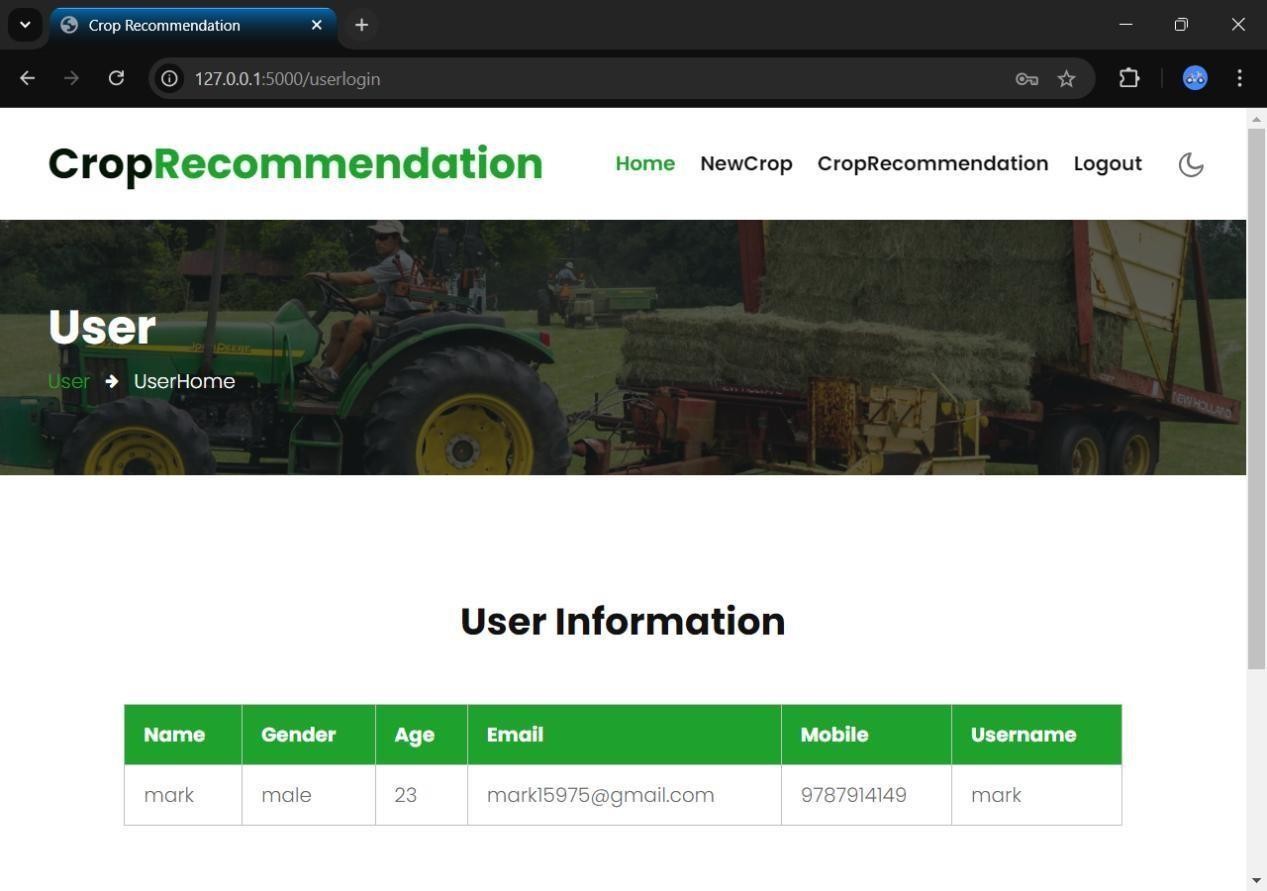
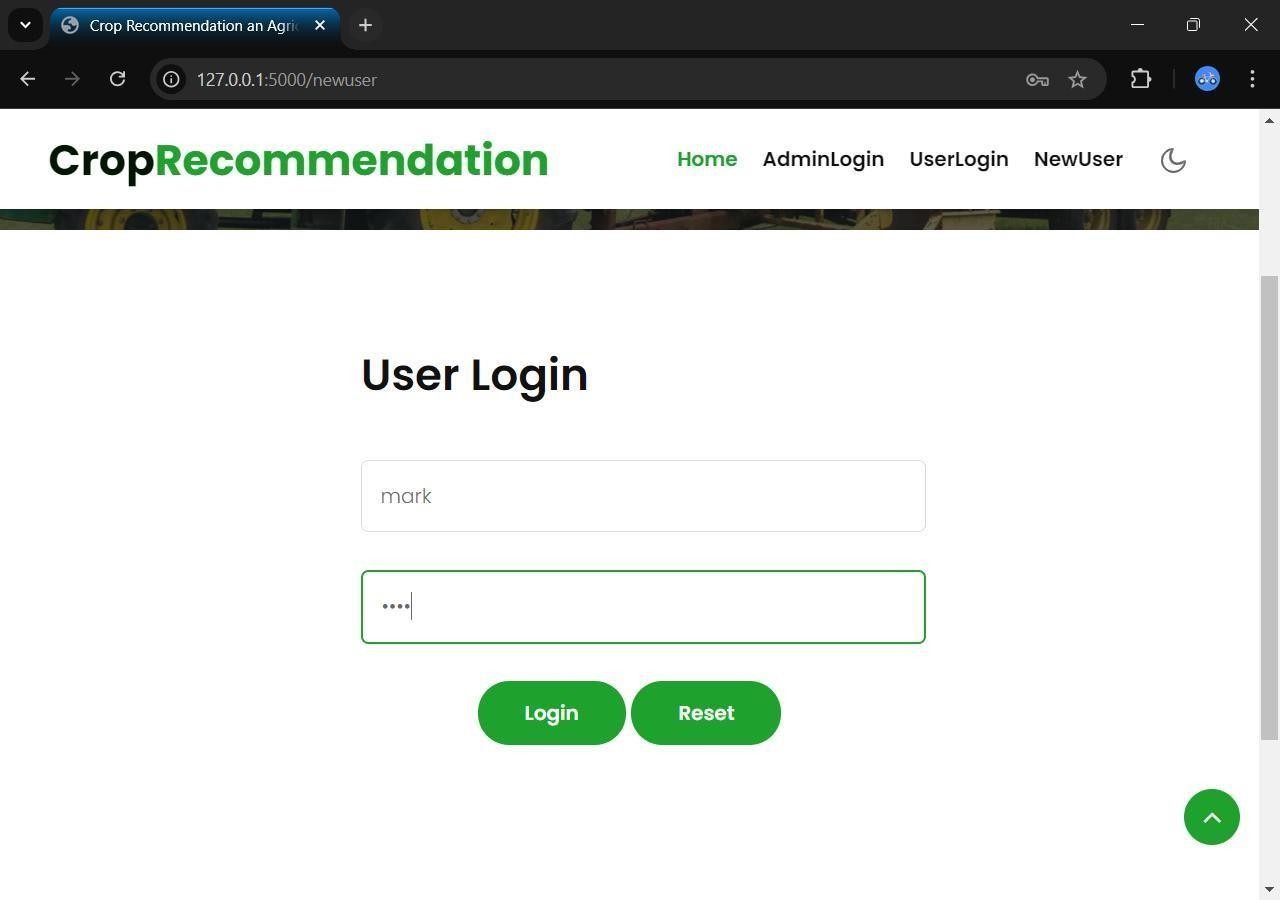
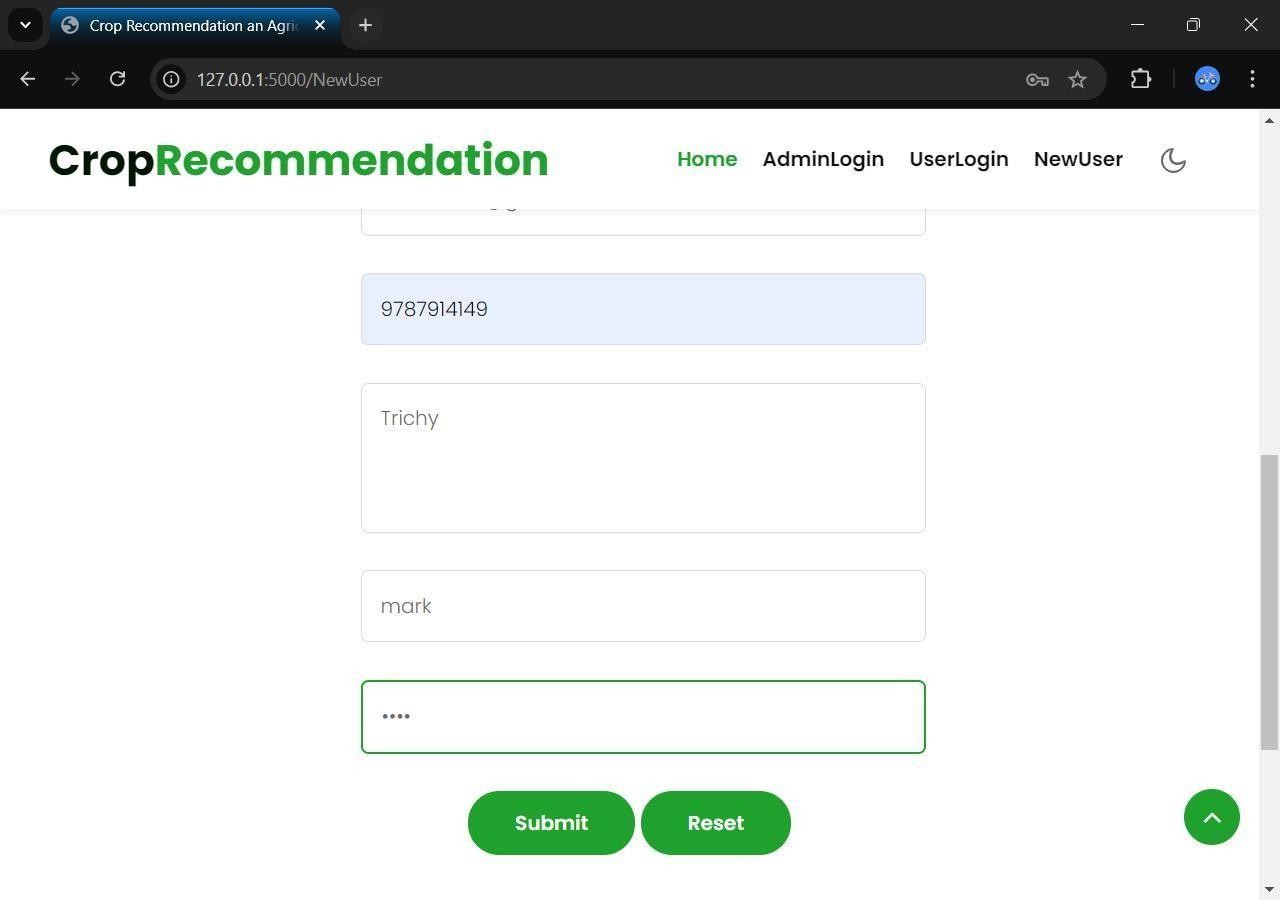
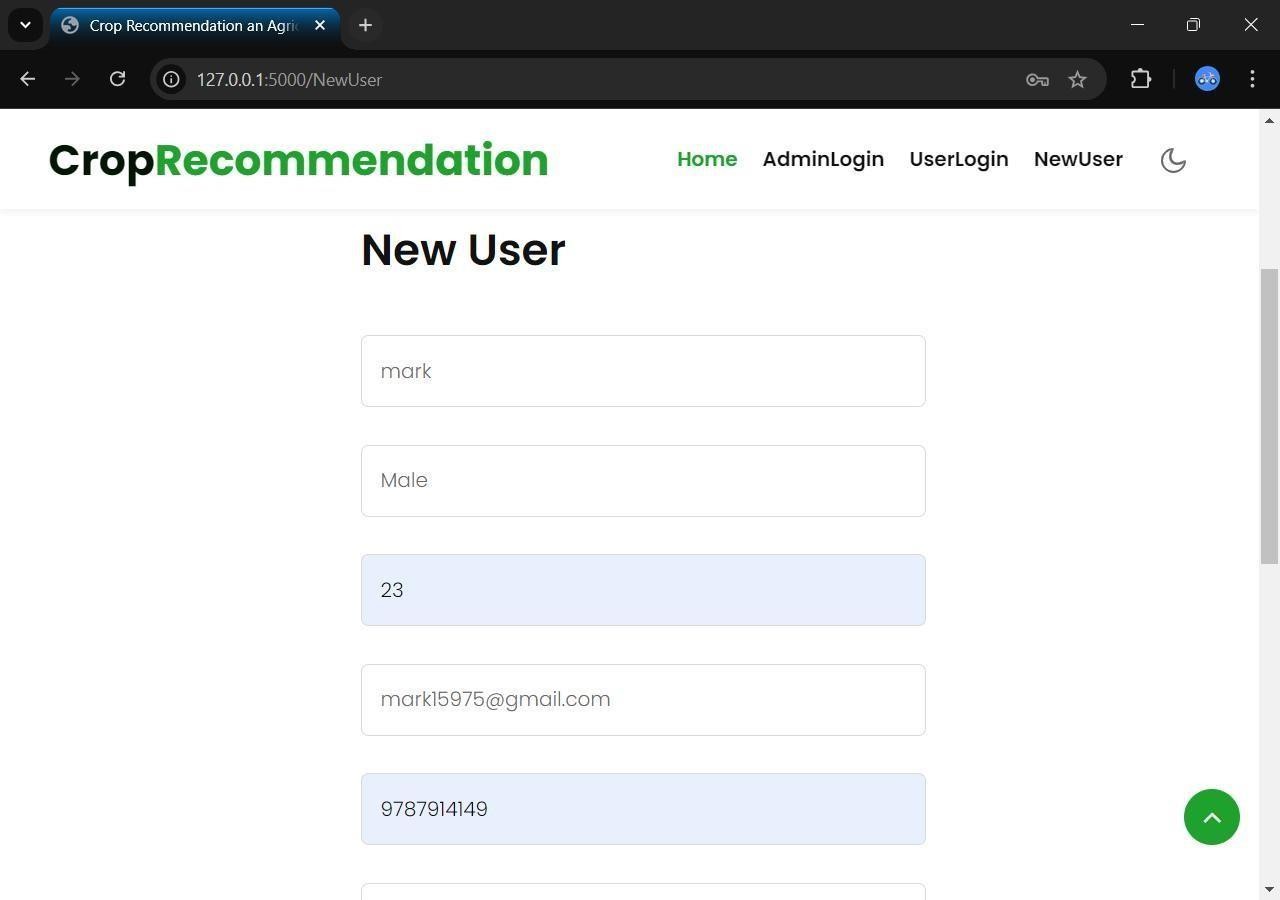
**import** requests

requests.post( **"**[**http://sms.creativepoint.in/api/push.json?apikey=6555c521622c1&route=tran**](http://sms.creativepoint.in/api/push.json?apikey=6555c521622c1&route=tran) **ssms&sender=FSSMSS&mobileno="**+ targetno + **"&text=Dear customer your msg is "**+ message + **" Sent By FSMSG FSSMSS"**)

**if** name == **' main '**: app.run(debug=**True**, use\_reloader=**True**)



**SCREENSHOTS**



**TESTING**

**TESTING OBJECTIVE**

Testing is a set activity that can be planned and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it is vital success of the system.

Testing Objectives:

There are several rules that can serve as testing objectives, they are

1. Testing is a process of executing a program with the intent of finding an error
2. A good test case is one that has high probability of finding an undiscovered error.
3. A successful test is one that uncovers an undiscovered error.

If testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrates that software functions appear to the working according to the specification, that performance requirements appear to have been met.

There are three ways to test a program

1. For Correctness
2. For Implementation efficiency
3. For Computational Complexity.

Tests used for implementation efficiency attempt to find ways to make a correct program faster or use less storage. It is a code-refining process, which reexamines the implementation phase of algorithm development. Tests for computational complexity amount to an experimental analysis of the complexity of an algorithm or an experimental comparison of two or more algorithms, which solve the same problem.The data is entered in all forms separately and whenever an error occurred, it is corrected immediately. A quality team deputed by the management verified all the necessary documents and tested the Software while entering the data at all levels.

# TYPES OF TESTING

The development process involves various types of testing. Each test type addresses a specific testing requirement. The most common types of testing involved in the development process are:

* + Unit Test
  + Functional Test
  + Integration Test
  + System Test
  + Validation Test

## Unit Testing:

The first test in the development process is the unit test. The source code is normally divided into modules, which in turn are divided into smaller units called units. These units have specific behavior. The test done on these units of code is called unit test. Unit test depends upon the language on which the project is developed. Unit tests ensure that each unique path of the project performs accurately to the documented specifications and contains clearly defined inputs and expected results.

## Functional Testing:

Functional test can be defined as testing two or more modules together with the intent of finding defects, demonstrating that defects are not present, verifying that the module performs its intended functions as stated in the specification and establishing confidence that a program does what it is supposed to do.

## Integration Testing:

In integration testing modules are combined and tested as a group. Modules are typically code modules, individual applications, source and destination

applications on a network, etc. Integration Testing follows unit testing and precedes system testing. Testing after the product is code complete. Betas are often widely

distributed or even distributed to the public at large in hopes that they will buy the final product when it is released.

## System Testing

System testing is defined as testing of a complete and fully integrated software product. This testing falls in black-box testing wherein knowledge of the inner design of the code is not a pre-requisite and is done by the testing team. It is the final test to verify that the product to be delivered meets the specifications mentioned in the requirement document. It should investigate both functional and non-functional requirements.

## Validation Testing

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements. Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfills its intended use when deployed on appropriate environment.

# MAINTENANCE

**SOFTWARE DESCRIPTION**

## Frond End: Python

Python is an interpreted high-level programming language for general- purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. In July 2018, Van Rossum stepped down as the leader in the language community. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit Python Software Foundation. Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach. While offering choice in coding methodology, the Python philosophy rejects exuberant syntax (such as that of Perl) in favor of a simpler, less-cluttered grammar. As Alex Martelli put it: "To describe something as 'clever' is not considered a compliment in the Python culture."Python's philosophy rejects the Perl "there is more than one way to do it" approach to language design in favour of "there should be one—and preferably only one— obvious way to do it".

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of CPython that would offer marginal increases in speed at the cost of clarity.[ When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. CPython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter. An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name a tribute to the British comedy group Monty Python and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard for and bar.

A common neologism in the Python community is pythonic, which can have a wide range of meanings related to program style. To say that code is pythonic is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called un pythonic. Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as Pythonists, Pythonistas, and Pythoneers. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead,

when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Python’s initial development was spearheaded by Guido van Rossum in the late 1980s. Today, it is developed by the Python Software Foundation. Because Python is a multiparadigm language, Python programmers can accomplish their tasks using different styles of programming: object oriented, imperative, functional or reflective. Python can be used in Web development, numeric programming, game development, serial port access and more.

There are two attributes that make development time in Python faster than in other programming languages:

1. Python is an interpreted language, which precludes the need to compile code before executing a program because Python does the compilation in the background. Because Python is a high-level programming language, it abstracts many sophisticated details from the programming code. Python focuses so much on this abstraction that its code can be understood by most novice programmers.
2. Python code tends to be shorter than comparable codes. Although Python offers fast development times, it lags slightly in terms of execution time. Compared to fully compiling languages like C and C++, Python programs execute slower. Of course, with the processing speeds of computers these days, the speed differences are usually only observed in benchmarking tests, not in real-world operations. In most cases, Python is already included in Linux distributions and Mac OS X machines.

## Back End: My SQL

MySQL is the world's most used open source [relational database](http://en.wikipedia.org/wiki/Relational_database_management_system) [management system](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS) as of 2008 that run as a server providing multi-user access to a number of databases. The MySQL development project has made its [source code](http://en.wikipedia.org/wiki/Source_code) available under the terms of the [GNU General Public License,](http://en.wikipedia.org/wiki/GNU_General_Public_License) as well as under a variety of [proprietary](http://en.wikipedia.org/wiki/Proprietary_software) agreements. MySQL was owned and sponsored by a single [for-profit](http://en.wikipedia.org/wiki/Business) firm, the [Swedish](http://en.wikipedia.org/wiki/Sweden) company [MySQL AB,](http://en.wikipedia.org/wiki/MySQL_AB) now owned by [Oracle](http://en.wikipedia.org/wiki/Oracle_Corporation) [Corporation.](http://en.wikipedia.org/wiki/Oracle_Corporation)

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack—LAMP is an acronym for "[Linux,](http://en.wikipedia.org/wiki/Linux) [Apache,](http://en.wikipedia.org/wiki/Apache_HTTP_Server) MySQL, [Perl](http://en.wikipedia.org/wiki/Perl)/[PHP](http://en.wikipedia.org/wiki/PHP)/[Python](http://en.wikipedia.org/wiki/Python_%28programming_language%29)." [Free-](http://en.wikipedia.org/wiki/Free_software) [software](http://en.wikipedia.org/wiki/Free_software)-open source projects that require a full-featured database management system often use MySQL.For commercial use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: [TYPO3](http://en.wikipedia.org/wiki/TYPO3), [Joomla,](http://en.wikipedia.org/wiki/Joomla) [Word Press,](http://en.wikipedia.org/wiki/WordPress) [phpBB,](http://en.wikipedia.org/wiki/PhpBB) [MyBB,](http://en.wikipedia.org/wiki/MyBB) [Drupal](http://en.wikipedia.org/wiki/Drupal) and other software built on the [LAMP](http://en.wikipedia.org/wiki/LAMP_%28software_bundle%29) software stack. MySQL is also used in many high-profile, large- scale [World Wide Web](http://en.wikipedia.org/wiki/World_Wide_Web) products, including Wikipedia, Google(though not for searches), [ImagebookTwitter,](http://en.wikipedia.org/wiki/Facebook) [Flickr,](http://en.wikipedia.org/wiki/Flickr) [Nokia.com,](http://en.wikipedia.org/wiki/Nokia) and [YouTube.](http://en.wikipedia.org/wiki/YouTube)

## Inter images

MySQL is primarily an RDBMS and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, MySQL Workbench is actively developed by Oracle, and is freely available for use.

## Graphical

The official MySQL Workbench is a free integrated environment developed by MySQL AB, that enables users to graphically administer MySQL databases and visually design database structures. MySQL Workbench replaces the previous package of software, [MySQL GUI Tools.](http://en.wikipedia.org/wiki/MySQL_GUI_Tools) Similar to other third-party packages, but still considered the authoritative MySQL frontend, MySQL Workbench lets users manage database design & modeling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).MySQL Workbench is available in two editions, the regular free and open source Community Edition which may be downloaded from the MySQL website, and the proprietary Standard Edition which extends and improves the feature set of the Community Edition.

# CONCLUSION

We presented a deep learning approach for crop prediction, which demonstrated superior performance in Crop Challenge using large datasets of products. The approach used deep neural networks to make crop datasets such as soil and textual datasets. In conclusion, deep learning models offer a promising solution for predicting crop yields based on environmental variables such as temperature, pH, rainfall, and soil data. By using neural networks to analyse large and complex datasets, these models can identify patterns and relationships that would be difficult or impossible for humans to discern. By training the model on historical data and then using it to make predictions on new data, farmers and researchers can gain valuable insights into which crops are most likely to thrive under certain environmental conditions. However, there are still some challenges to overcome, such as the need for high-quality and diverse data, the difficulty of interpreting complex neural networks, and the potential for bias and errors in the training data. Overall, deep learning holds great promise for revolutionizing the field of crop prediction and helping to feed a growing global population

# FUTURE ENHANCEMENT

This project describes crop yield prediction ability of the algorithm. In future we can determine the efficient algorithm based on their accuracy metrics that will helps to choose an efficient algorithm for crop prediction based on soil images

# APPENDICES

## Appendix A: Glossary of Terms

* **MLP (Multi-Layer Perceptron)**: A type of artificial neural network used for classification and regression tasks, including crop recommendation based on environmental factors.
* **CNN (Convolutional Neural Network)**: A deep learning algorithm especially effective in image recognition, used in this project to analyze soil images for crop prediction.
* **Data Preprocessing**: The process of cleaning and preparing raw data for analysis, removing noise, and handling missing values to improve model accuracy.
* **Crop Prediction**: Forecasting crop productivity and yield based on multiple parameters like soil composition, climate, and historical data.

## Appendix B: Software and Tools Used

* **Python**: Primary programming language for this project, with libraries for machine learning and data visualization.
* **PyCharm**: Integrated Development Environment (IDE) used for coding in Python.
* **MySQL**: Database management system used for storing datasets and query information.
* **Flask**: Web framework used for deploying the web interface of the crop recommendation system.

## Appendix C: Dataset Sources

* **Kaggle**: Datasets for crop yield, soil quality, and environmental factors are sourced from Kaggle, including temperature, pH, and nitrogen levels.
* **Weather Data**: Historical weather data integrated into the model to provide accurate crop predictions.

# REFERENCES

**BOOK REFERENCES**

1. Heinold, Brian. "A practical introduction to Python programming." (2021).
2. Kneusel, Ronald T. Practical deep learning: A Python-based introduction. No Starch Press, 2021.
3. Dhruv, Akshit J., Reema Patel, and Nishant Doshi. "Python: the most advanced programming language for computer science applications." Science and Technology Publications, Lda (2021): 292-299.
4. Sundnes, Joakim. Introduction to scientific programming with Python. Springer Nature, 2020.
5. Hill, Christian. Learning scientific programming with Python. Cambridge University Press, 2020.

# WEBSITE REFERENCES

1. https://medium.com/javarevisited/10-free-python-tutorials-and-courses-from- google-microsoft-and-coursera-for-beginners-96b9ad20b4e6
2. https://[www.bestcolleges.com/bootcamps/guides/learn-python-free/](http://www.bestcolleges.com/bootcamps/guides/learn-python-free/)
3. https://[www.programiz.com/python-programming](http://www.programiz.com/python-programming)
4. https://realpython.com/
5. https://[www.codecademy.com/learn/learn-python](http://www.codecademy.com/learn/learn-python)