Institute of Science and Technology

TRIBHUVAN UNIVERSITY

AMRIT SCIENCE CAMPUS

Lainchaur, Kathmandu

Project Report on

Crop Recommendation System

With integrated crop cure and marketplace functionality

Prepared Under the Supervision of

Rakesh Kumar Bachchan

Lecturer

Department of Computer Science and Information Technology

Amrit Science Campus

Submitted By:

Ashish Khatri 23123/076

Kritan Sitaula 23149/076

Nischal Kafle 23165/076

Submitted To:

Tribhuvan University

Institute of Science and Technology

Date: ___________________

AMRIT CAMPUS

Affiliated to Tribhuvan University

SUPERVISOR'S RECOMMENDATION

I hereby recommend that this project prepared under my supervision by Ashish Khatri (23123/076), Kritan Sitaula (23149/076), and Nischal Kafle(23165/076) entitled "Crop Recommendation System – with integrated cure recommendation and marketplace" in partial fulfillment of requirement of the degree of BSc. In Computer Science and Information Technology (B.Sc. CSIT) be processed for evaluation.

Mr. Rakesh Kumar Bachan

Supervisor

AMRIT CAMPUS

Affiliated to Tribhuvan University

CERTIFICATE OF APPROVAL

This is to certify that the project report entitles "Crop Recommendation System – with integrated Cure recommendation and marketplace" submitted by:

Ashish Khatri	23123/076
Kritan Sitaula	23149/076
Nischal Kafle	23165/076

are bonafide students of this institute and the work has been carried out by them under the supervision of Mr. Rakesh Kumar Bachhan and it is approved for the partial fulfillment of the requirement of Tribhuvan University, for the award of the degree of B.Sc. Computer Science and Information technology.

HOD, Deparment of CSIT
Amrit Campus

Supervisor
Mr. Rakesh Kumar Bachan

Internal Examiner
Amrit Campus

External Examiner
IOST, Tribhuvan University

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to all those who have contributed to the completion of this project report.

First and foremost, we extend our heartfelt thanks to Mr. Rakesh Kumar Bachhan, our project supervisor, for his invaluable guidance, support, and mentorship throughout the duration of this project. His expertise, encouragement, and constructive feedback have been instrumental in shaping the direction and success of our work.

We are deeply thankful to our faculty members and academic advisors for their unwavering support and encouragement throughout the project. Their insights, expertise, and encouragement have been invaluable in guiding us through the various stages of the project.

We are grateful to our friends and family members for their unwavering support, encouragement, and understanding during the challenging periods of this project. Their words of encouragement and belief in our abilities have been a source of strength and motivation.

Last but not least, we express our gratitude to all the individuals and organizations who have directly or indirectly contributed to the success of this project. Your support and collaboration have been invaluable, and we are deeply appreciative of your contributions.

Thank you all for being part of this journey and for your invaluable support and encouragement.

ABSTRACT

Our project presents the development of a comprehensive software solution aimed at optimizing various stages of the farming cycle. Our software encompasses functionalities like soil analysis, crop selection, procurement, plant care recommendations, and direct-to-consumer marketing.

The first stage of the farming cycle involves users providing input on soil nutrients (Nitrogen, Phosphorous, Potassium, pH) and climatic conditions (temperature, humidity, rainfall). Using a Random Forest classification algorithm, the software predicts suitable crops for cultivation under the given conditions, allowing farmers to make informed decisions regarding crop selection. Subsequently, farmers can directly order seeds online through the software platform.

In the next stage, the software assists farmers in managing plant health by providing recommendations based on soil nutrient levels and crop type. This includes guidance on fertilization and pest management strategies tailored to the specific needs of the crops.

Finally, the software facilitates direct-to-consumer marketing by enabling farmers to sell their harvested produce online. Through an integrated e-commerce platform and powerful Content Management System (CMS), farmers can manage their online store and facilitate transactions with consumers.

Overall, the software solution aims to empower farmers with data-driven decision-making tools, streamline agricultural processes, and enhance market connectivity, ultimately contributing to increased productivity, sustainability, and profitability within the agricultural sector.

Table of Contents

List of Tables

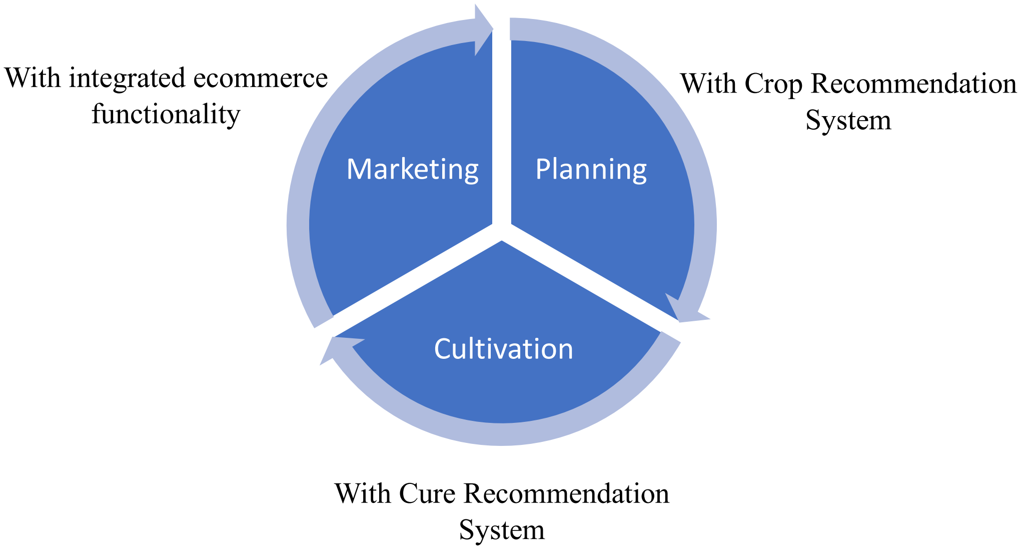
List of Figures

Chapter 1: Introduction

1.1 Introduction

For modernizing agriculture, we need to make the fusion of technology and traditional farming practices. We are introducing the software tailored to address the diverse needs of farmers by data-driven decision-making and integration of agricultural processes. Our college project aims to bridge the gap between traditional farming methodologies and technology through the development of an innovative software solution.

Our software is engineered to address three key stages of the farming cycle: planning, cultivation, and marketing.



1. The first stage of our software allows users to input data regarding soil nutrients, such as Nitrogen, Phosphorous, Potassium, and pH levels, alongside climatic parameters including temperature, humidity, and rainfall patterns. Utilizing a Random Forest classification algorithm, the software intelligently analyzes this data to recommend suitable crops for cultivation, tailored to the specific conditions of the farm using huge dataset. Furthermore, our software also simplifies online ordering of seeds for the recommended crops.
2. Our software extends support to farmers during the cultivation phase by providing plant care recommendations. By inputting soil nutrient levels and the name of the cultivated crop, farmers can receive tailored suggestions for optimal fertilization and pest management strategies, optimizing crop health and productivity.
3. Moreover, upon harvesting, our software facilitates direct online sales of farm produce to consumers through a user-friendly e-commerce platform. By cutting out intermediaries and enabling direct-to-consumer transactions, farmers can establish stronger connections with consumers, enhance market reach, and command better prices for their produce.

1.2 Problem Statement

Current agricultural practices suffer various problems. Among which, lack of expertise in crop selection, lack of expertise during cultivation and lack of market to sell produce are some. We have tried to address these issues in our software with three different software solution integrated in one package. A unified software solution is needed to integrate soil analysis, crop selection, procurement, plant care recommendations, and direct-to-consumer marketing, addressing these challenges and empowering farmers with streamlined processes and enhanced market connectivity.

1.3 Objectives

The objectives of the project can be stated as:

1. To develop the software solution to address planning, cultivation and marketing of farm produce.
2. To promote data driven decision in agricultural practices.
3. To motivate farmers in ecommerce through powerful ecommerce tools and CMS.
4. To blend traditional farming practices and modern technological tools.

1.4 Scope and Limitation

1.4.1 Scope

As a web-based system, the system will be easily accessible to those with a device supporting a browser and an Internet Connection. The software will allow users to input soil nutrient data (Nitrogen, Phosphorous, Potassium, pH) and climatic conditions (temperature, humidity, rainfall), which will be processed using advanced algorithm. The software will facilitate direct online ordering of recommended crop seeds from authorized suppliers, ensuring timely access to quality inputs for farmers. Upon cultivation, the software will provide personalized plant care recommendations based on soil nutrient levels (N, P, K) and crop name, helping farmers in optimizing crop health and productivity. Farmers will be empowered to sell their harvested produce directly to consumers through an online platform integrated with a robust Content Management System (CMS), facilitating seamless transactions and market connectivity.

1.4.2 Limitation

Some of the limitation of this software solution is listed below:

1. The accuracy of crop recommendation and cure recommendation highly relies on the input data regarding NPK, pH, temperature, humidity and rainfall. Variability and inadequacy in data quality highly impacts the accuracy.
2. Cure recommendation generated by this system are based on generalized guidelines and may not address unique pest or disease specific to crop or region.
3. System doesn't take account for specific regional or microclimatic variations that could influence crop suitability and management practices.
4. Raising technological literacy among farmers, making digital infrastructure accessible to farmers and comprehensive support and training to them are challenging.

1.5 Development Methodology

For this project we adopted XP Methodology (Extreme Programming). With a small team of three, pair programming can be particularly effective. We worked together in pairs, alternating roles between writing code and reviewing, thereby promoting knowledge sharing, collaboration, and higher code quality.

We practiced Continuous Integration (CI) by frequently integrate our work in shared git repository (link: https://github.com/CSIT-final-year-project/Godmel). We practiced TDD (Test Driven Development) and ensure software meets specified requirements and remains maintainable over the time of the project.

Chapter 2: Background Study and Literature Review

2.1 Background Study

Machine learning is an application of artificial intelligence (AI) that gives systems the ability to automatically learn and evolve from experience without being specially programmed by the programmer. The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The main aim of machine learning is to allow computers to learn automatically and adjust their actions to improve the accuracy and usefulness of the program, without any human intervention or assistance.

2.1.1 Supervised and Unsupervised Learning

Machine learning techniques can be broadly categorized into the following types:

Supervised learning takes a set of feature/label pairs, called the training set. From this training set the system creates a generalized model of the relationship between the set of descriptive features and the target features in the form of a program that contains a set of rules. The objective is to use the output program produced to predict the label for a previously unseen, unlabelled input set of features, i.e. to predict the outcome for some new data. Data with known labels, which have not been included in the training set, are classified by the generated model and the results are compared to the known labels. This dataset is called the test set. The accuracy of the predictive model can then be calculated as the proportion of the correct predictions the model labeled out of the total number of instances in the test set.

Unsupervised learning takes a dataset of descriptive features without labels as a training set. In unsupervised learning, the algorithms are left to themselves to discover interesting structures in the data. The goal now is to create a model that finds some hidden structure in the dataset, such as natural clusters or associations. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system does not figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data. Unsupervised learning can be used for clustering, which is used to discover any inherent grouping that are already present in the data. It can also be used for association problems, by creating rules based on the data and finding relationships or associations between them.

For our system we are using supervised learning algorithm.

2.2 Literature Review

[1] In paper “Smart Management of Crop Cultivation using IoT and Machine Learning” by Archana Gupta, Dharmil Nagda, Pratiksha Nikhare, Atharva Sandbhor, author has proposed a smart system that can assist farmers in crop management by considering sensed parameters (temperature, humidity) and other parameters (soil type, location of farm, rainfall) that predicts the most suitable crop to grow in that environment.

[2] In paper “Crop Recommendation on Analyzing Soil Using Machine Learning” by Anguraj.Ka, Thiyaneswaran.Bb, Megashree.Gc, Preetha Shri.J.Gd, Navya.Se, Jayanthi. Jf, author has proposed new technologies include the use of Internet of Things (IOT) and Machine Learning. The real time data from the field area can be collected using IOT system. The collected data from the field area is fed to the trained model. The trained model then makes the predictions using the data. The result produced by the model greatly helps is sowing the suitable crops in the particular field area.

[3] In Reference paper “Classification of Soil and Crop Suggestion using Machine Learning Techniques”, by A. Mythili, determines a model is proposed for predicting the soil type and

suggest a suitable crop that can be cultivated in that soil. The model has been tested using various machine learning algorithms such as KNN, SVM and logistic regression. The accuracy of the present model is maximum than the existing models.

[4] In “IOT based Crop Recommendation, Crop Disease Prediction and Its Solution” Rani Holambe, Pooja Patil, Padmaja Pawar, Saurabh Salunkhe, Mr. Hrushikesh Joshi, proposed a system would assist the farmers in making an informed decision about which crop to grow depending on a variety of environmental and geographical factors. The ML and IoT based suggestions will significantly educate the farmer and help them minimize costs and make strategic decisions by replacing intuition and passed-down knowledge with far more reliable data-driven ML models. This allows for a scalable, reliable solution to an important problem affecting hundreds of millions of people.

2.3 Study of Existing Work

Different e-commerce platform has been developed in Nepal to sell farmer’s good online such as Krishi COOP Bazaar. Also, different machine learning model has been developed in for prediction. Different work such as Regularized Greedy Forest to see an appropriate crop sequence at a given time stamps. Other research used historical records of meteorological data as training set where model is trained to spot climate that are deterrent for the assembly of crops which efficiently predicts the yield of crops on the idea of monthly weather patterns.

Our system implements both e-commerce functionalities and machine learning models for crop recommendation and soil analysis. We also believed that not only either soil nutrients or weather data should be taken consideration for training the model but all the appropriate parameters such as temperature, rainfall, geography and soil condition to predict crop suitability.

Chapter 3: System Analysis

3.1 Requirement Analysis

Functional Requirements:

Our system allows three kinds of user privileges. One is admin, another farmers and other is customer. The functionality available for each privileges are represented with Use case diagram below.

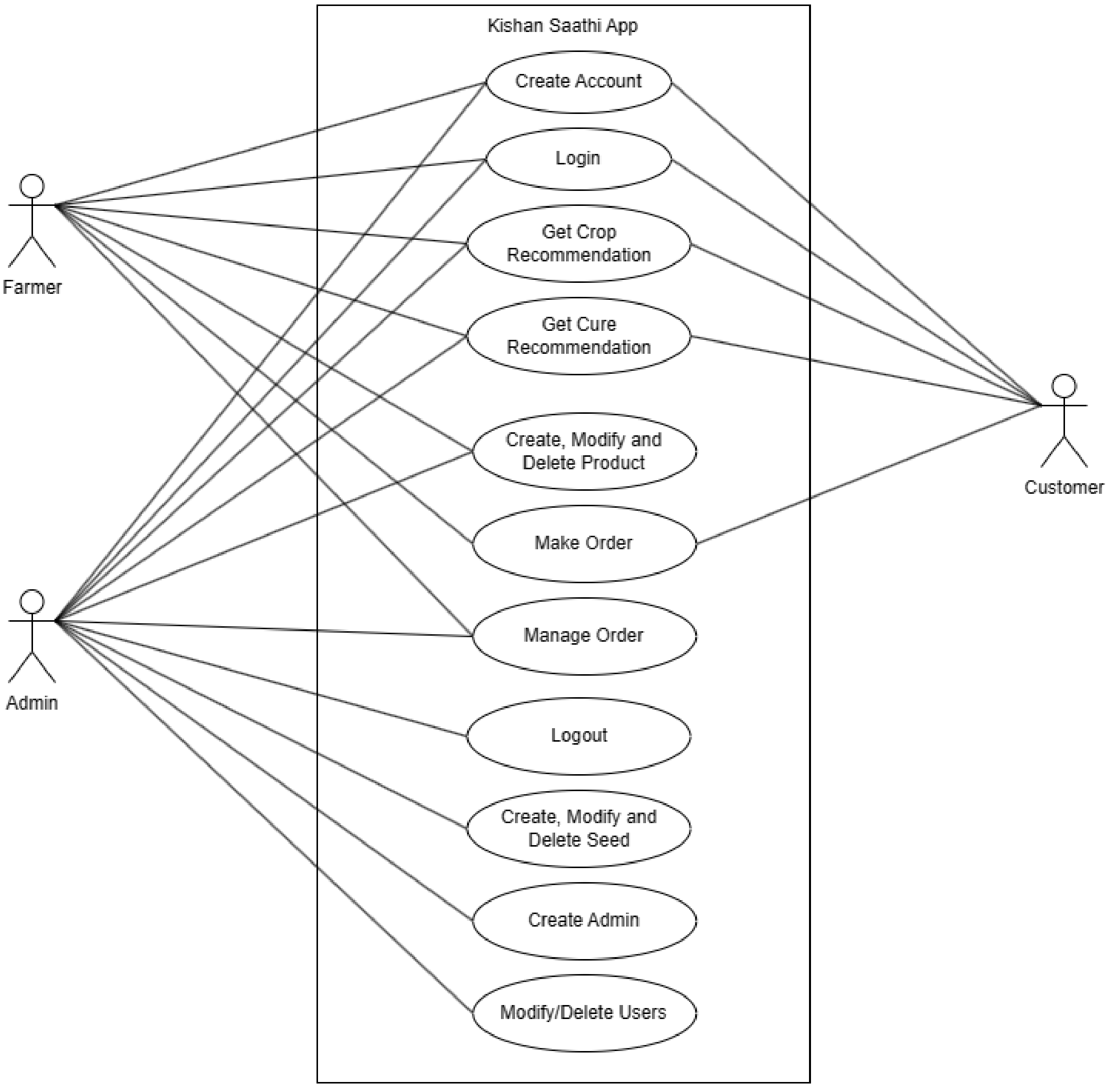


Figure 1: Use Case Diagram

- Admin
  1. Use 'Crop Recommendation System" and "Crop Cure Recommendation System"
  2. Create seeds that is recommended for purchase to any user who obtains prediction of the crop
  3. Modify/Delete the created seed
  4. Create farm products to list in market place from where any user can make an order
  5. Modify/delete products
  6. Create another admin
  7. Modify/Delete users

8. Manage Orders

- Farmer
    1. Use ‘Crop Recommendation System” and “Crop Cure Recommendation System”
    2. Make order for seeds of crop recommended by crop recommendation system
    3. Create farm products to list in market place from where any user can make an order
    4. Modify/delete products
    5. Modify/Delete users
    6. Manage Orders
- Customer
    1. Create Account
    2. Make orders for seeds or farm products

Hardware & Software Requirements

For this project, we require laptop with support of HTML5, CSS, and other implementation tools required for the project. The minimum system requirements for the laptop include: i5 11^{th} generation, 16GB RAM, 512 SSD, Windows 11 OS.

The software requirements for the project includes powerful IDE that is Visual Studio Code, Jupyter Notebook for training models, MongoDB for database, Microsoft edge browser, postman for API testing and many other open source programs. Draw.io and MS Word are also needed fro documentation.

Non-Functional Requirements

The non-functional requirements of our software are listed below:

1. Secured
2. Easily accessible
3. Responsive Design
4. User Friendly
5. High Performance

3.2 Feasibility Analysis

A feasibility study is conducted to assess the practicality and viability of implementing the proposed software solution for the farming cycle. This study evaluates various aspects, including technical, economic, operational, and scheduling feasibility, to determine whether the project is feasible and worth pursuing.

1. Technical Feasibility:
   This software solution encompasses complex functionalities such as soil nutrient analysis, crop classification, e-commerce integration, and data management. We have rich set of historical data regarding crop nature based on soil nutrients data. We are well familiar with classification algorithms. We are capable of training the model and integrate into webapp along with developing ecommerce functionality in web app. So, the project is technically feasible.
2. Economic Feasibility:
   Software requirement for development of this project is available free for community development. However, hardware requirement might be high to setup IOT as we might need NPK sensor for soil nutrient, moisture sensor for moisture measurement in soil, Arduino board, etc. But as for now this project is just a prototype so we are going to take all these parameters through user and weather data through online free APIs like open weather map. So, for prototyping, this project is economically feasible.

3. Operational Feasibility:
   Operational feasibility assessment indicates that the software solution can be effectively implemented within the existing agricultural ecosystem. User acceptance testing and stakeholder engagement activities needs to be done before putting it into practice.
4. Scheduling Feasibility
   The project expands over a period of about 4 months. The project schedule is feasible. Schedule plan is shown in the Gantt chart below.

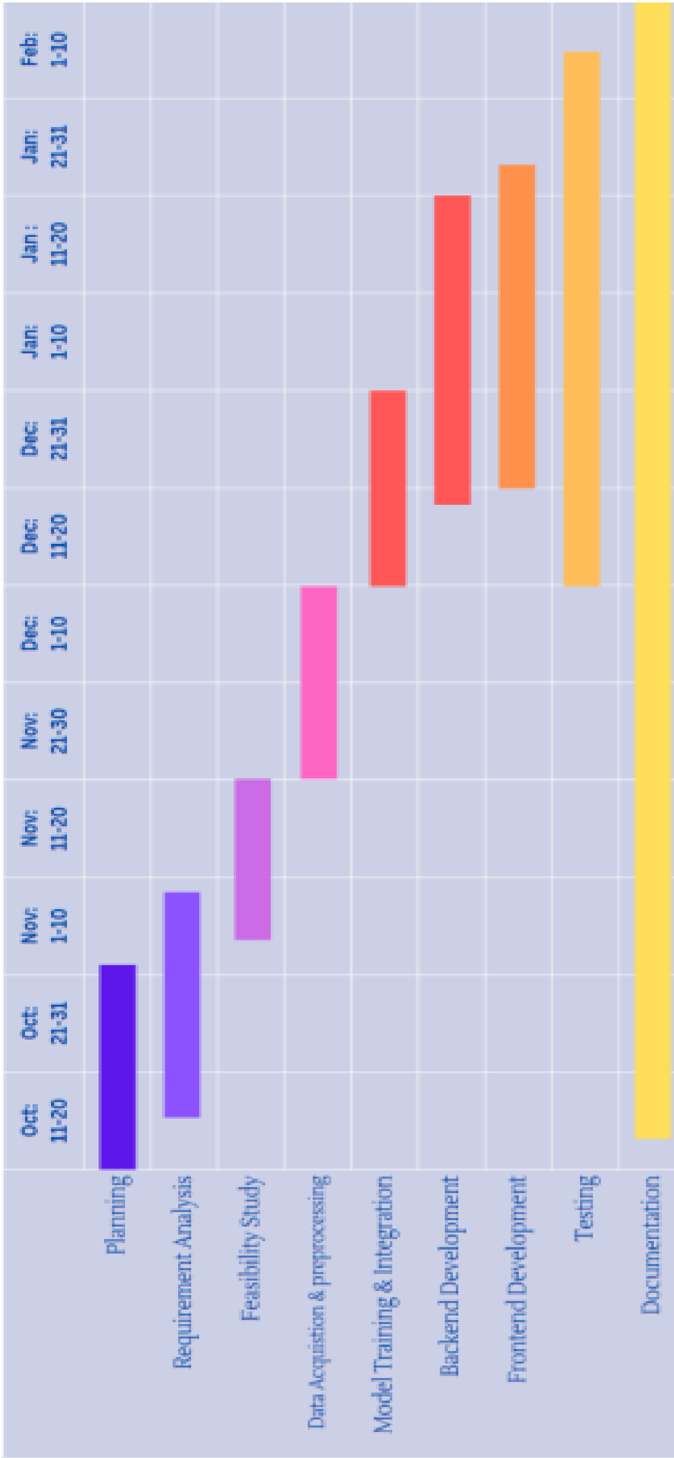


Figure 2: Gantt Chart

3.3 Analysis

Activity Diagram

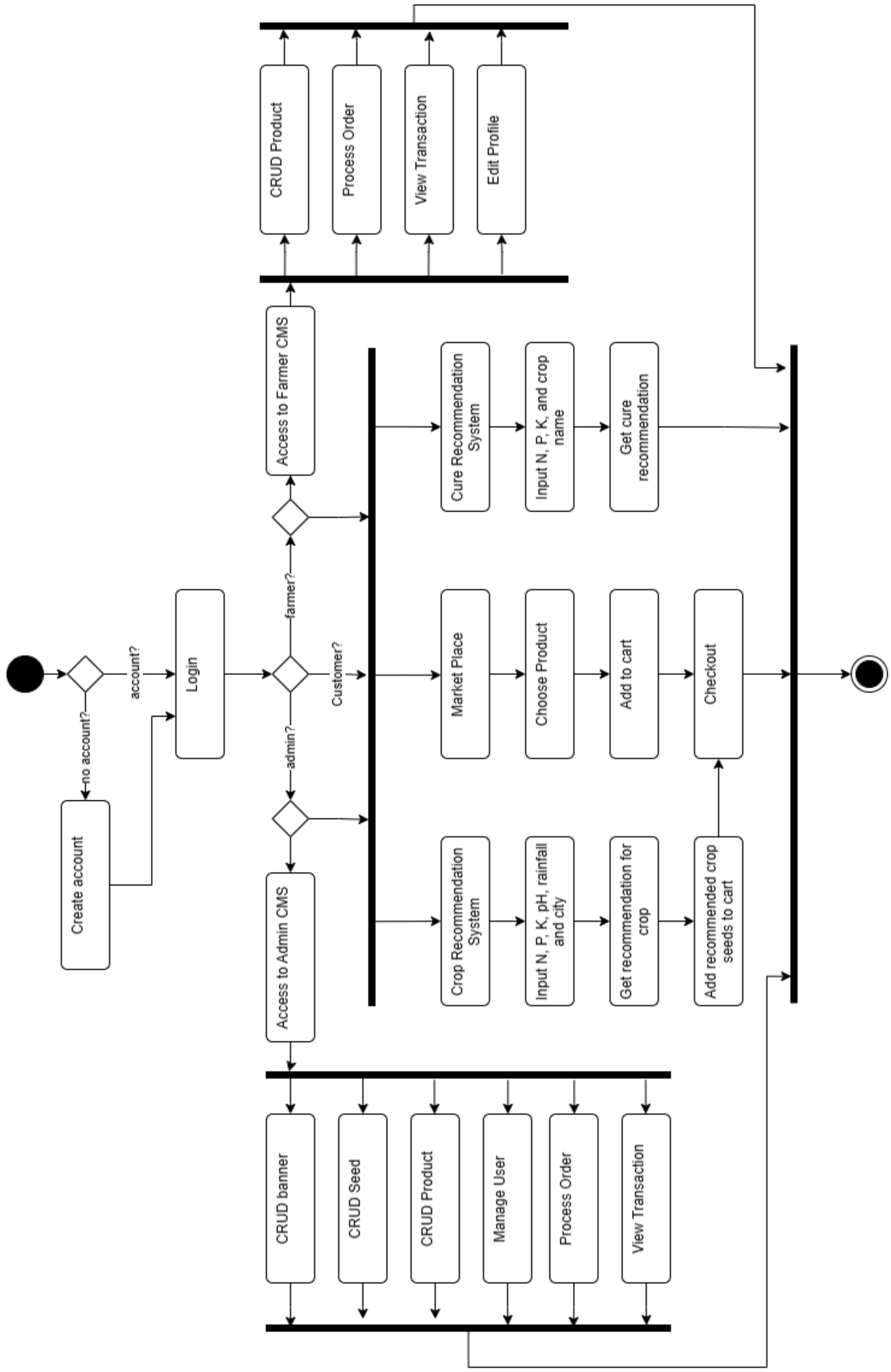


Figure 3: Activity Diagram

ER Diagram

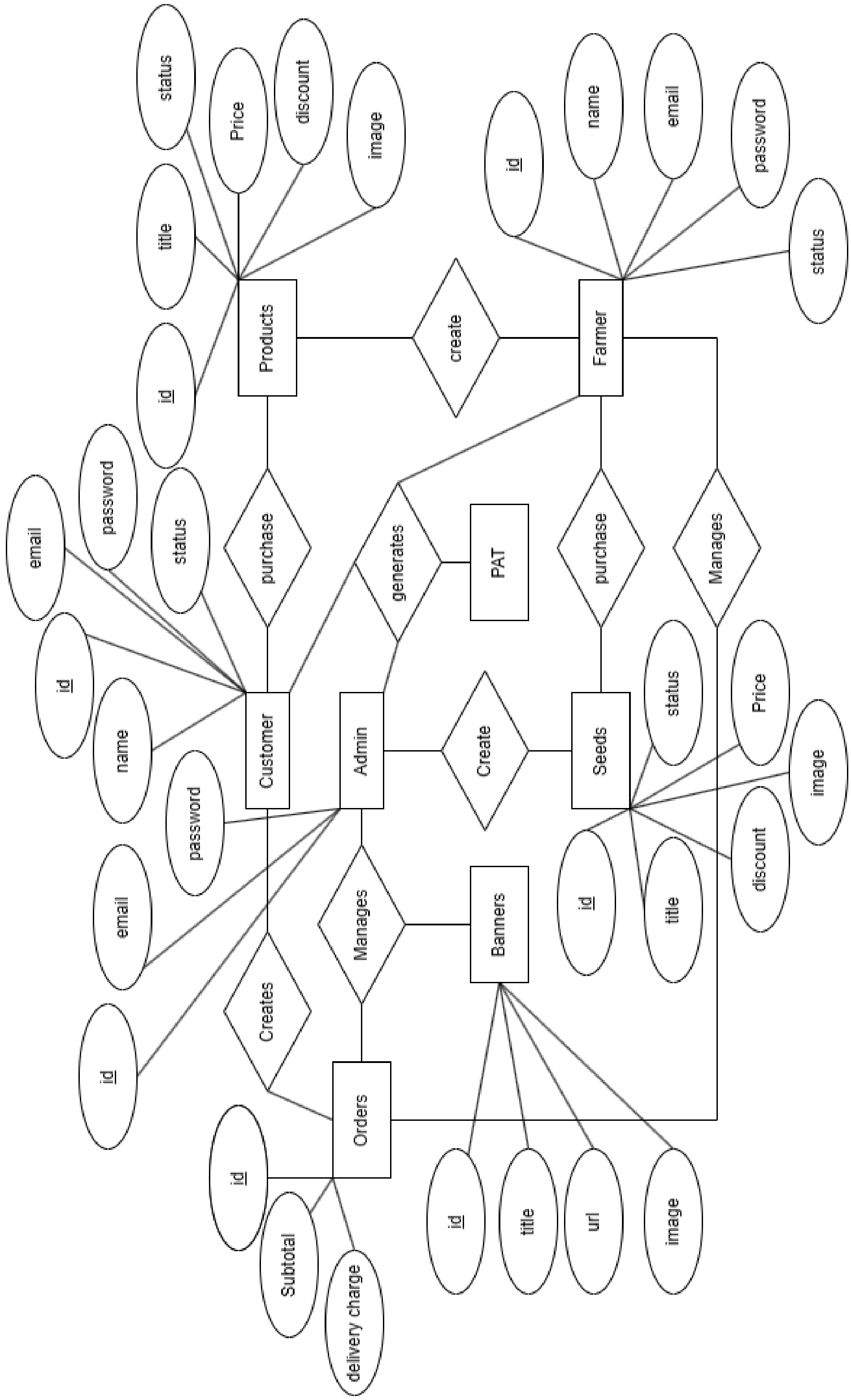


Figure 4: ER Diagram

Figure 4: ER Diagram

Data Flow Diagram

1. **DFD 0**

The context level DFD describes the whole system. Below DFD of the Kishan Saathi App shows three users can operate the system; Admin, Farmer and Customer.

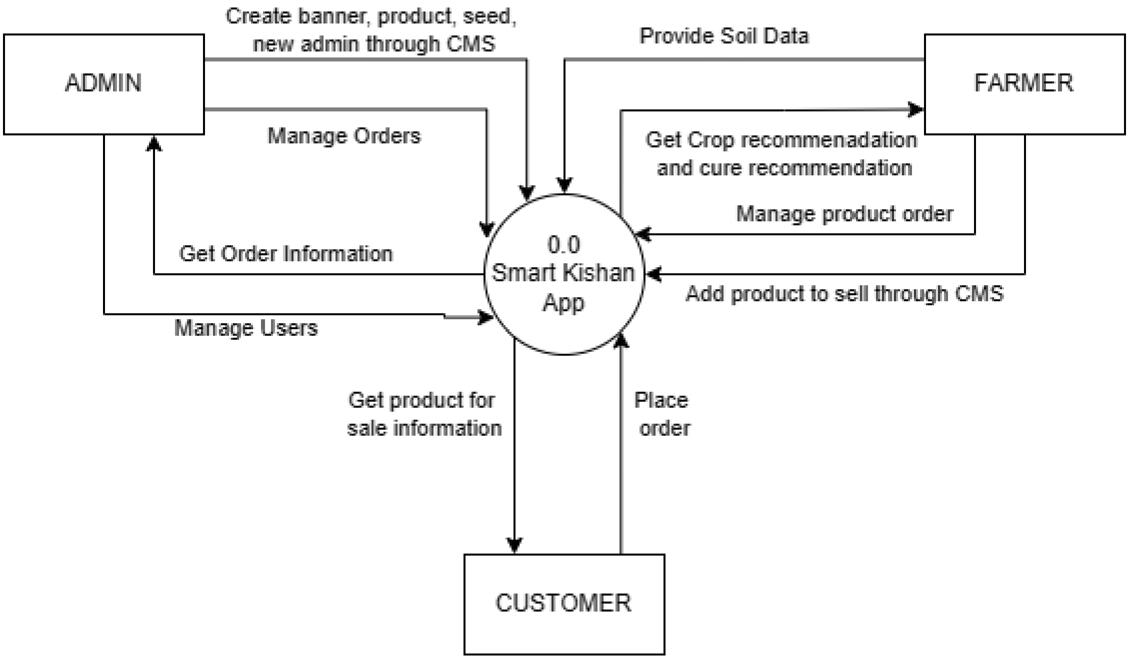


Figure 5: DFD Level 0

2. **DFD 1 for Admin**

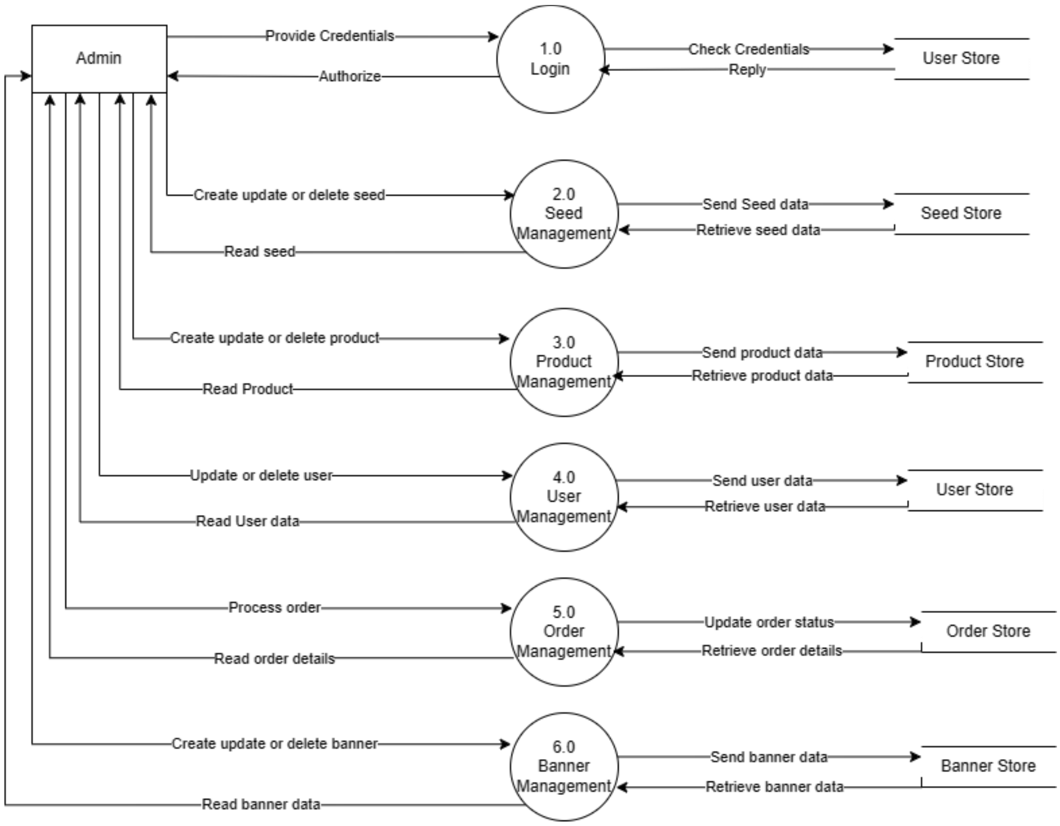


Figure 6: DFD level 1 for admin

3. DFD 1 for Farmer and customer

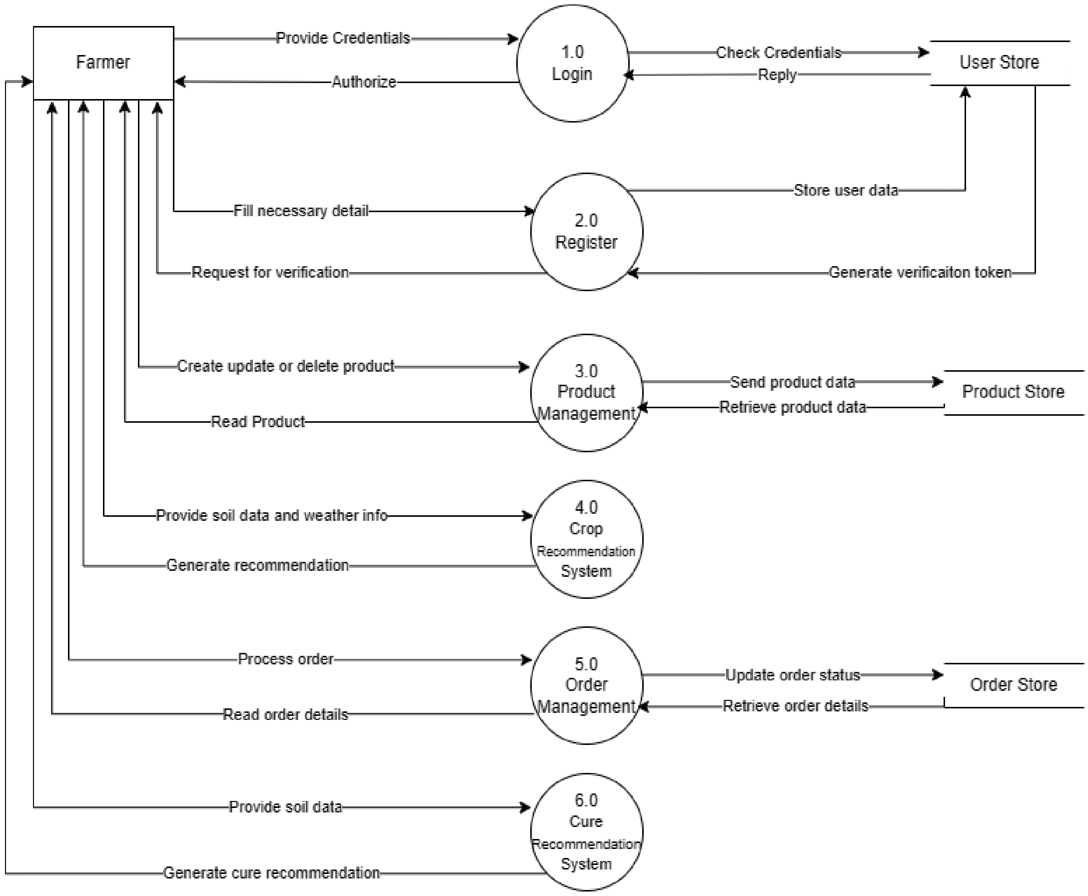


Figure 7: DFD1 for farmer

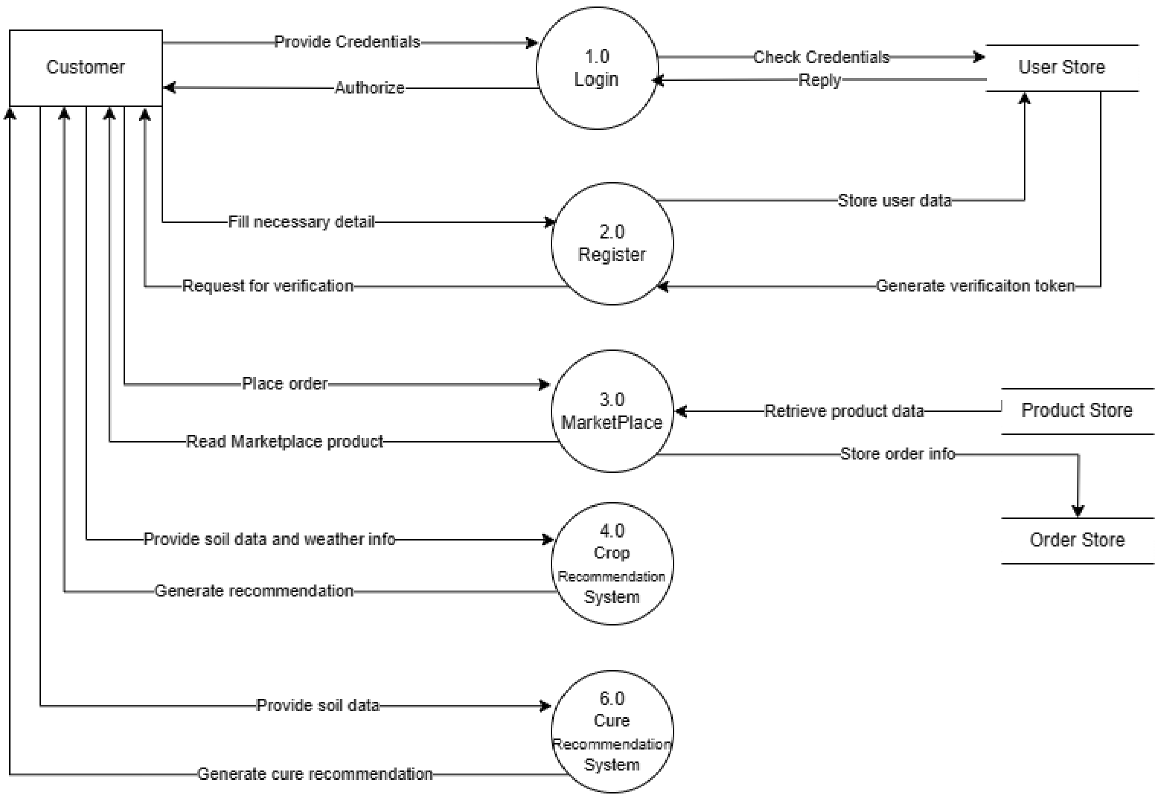


Figure 8: DFD 1 for customer

Chapter 4: System Design

4.1 Design

Database Design

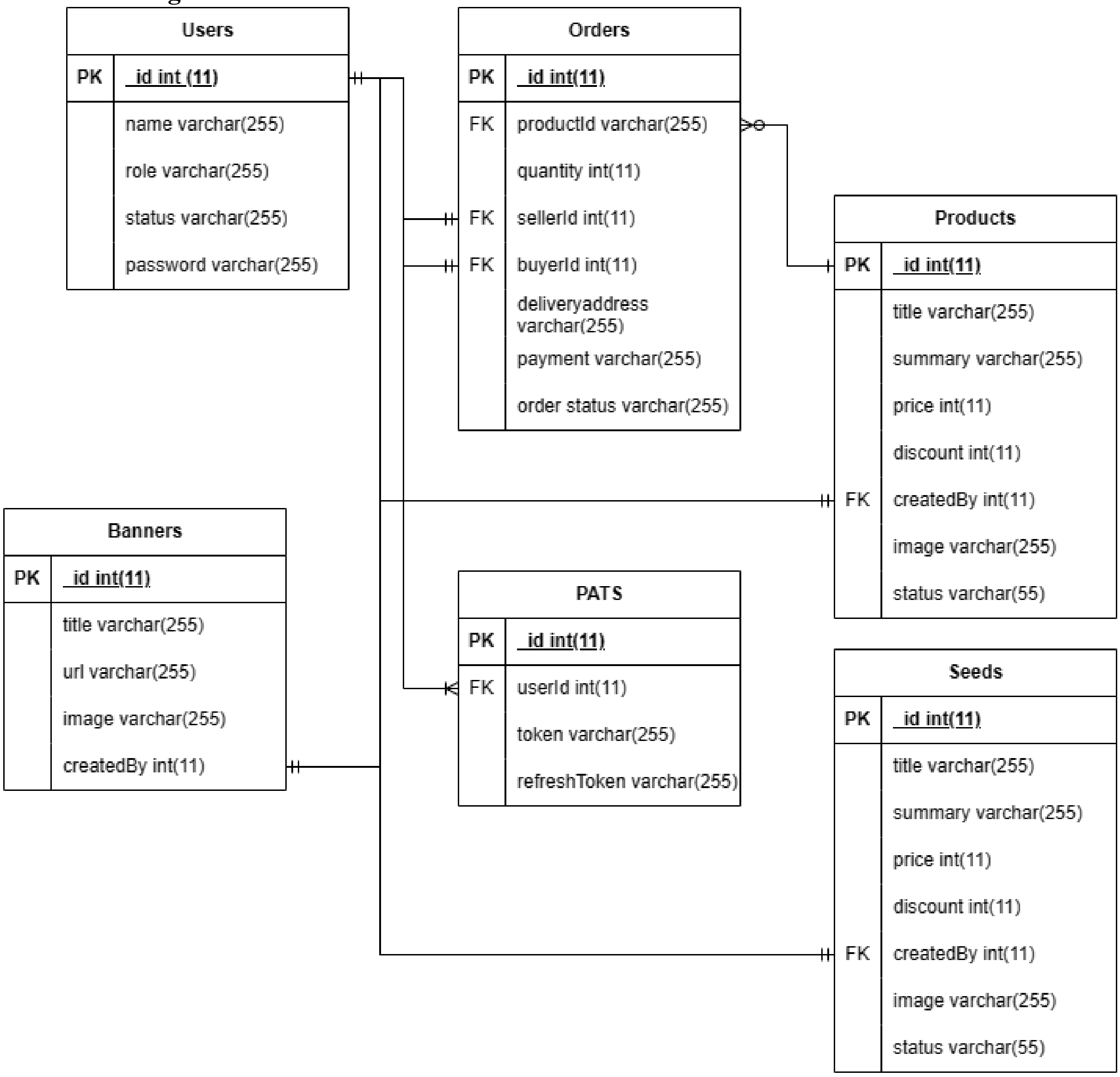


Figure 9: Database Design

Forms Design

For our system, following forms will be necessary:

1. Login and Registration form
2. Crop recommendation form
3. Cure Recommendation form
4. Checkout form
5. Banner, Seed, Product, User creation form in CMS

The respective design of forms are shown below:

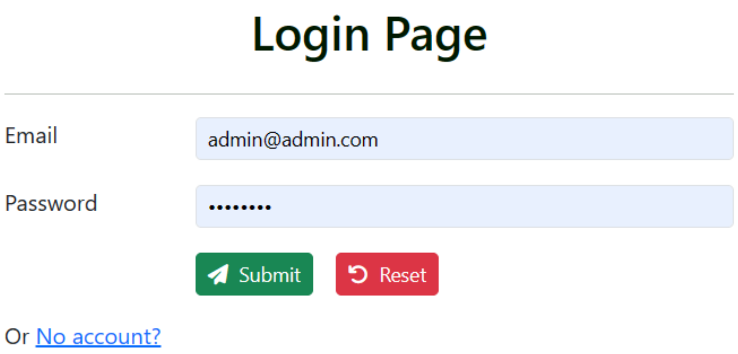


Figure 10: Login form

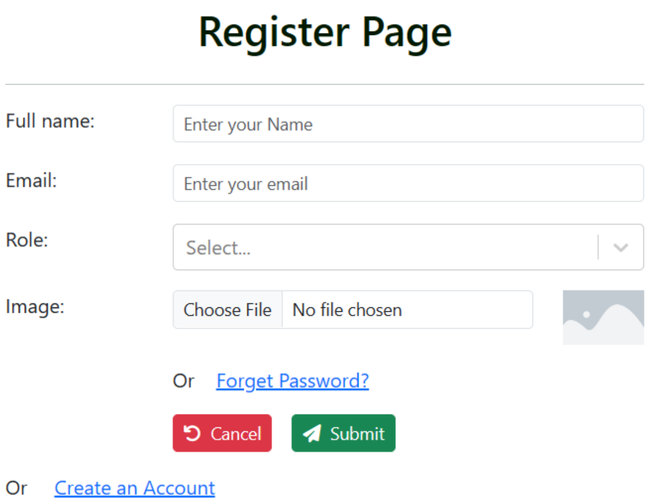


Figure 11: Registration Form

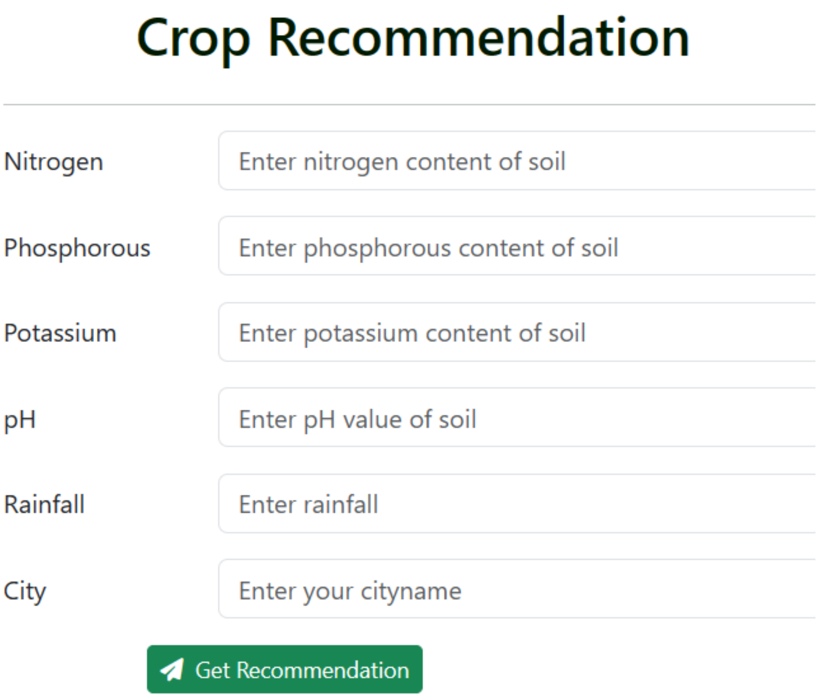


Figure 12: Crop Recommendation form

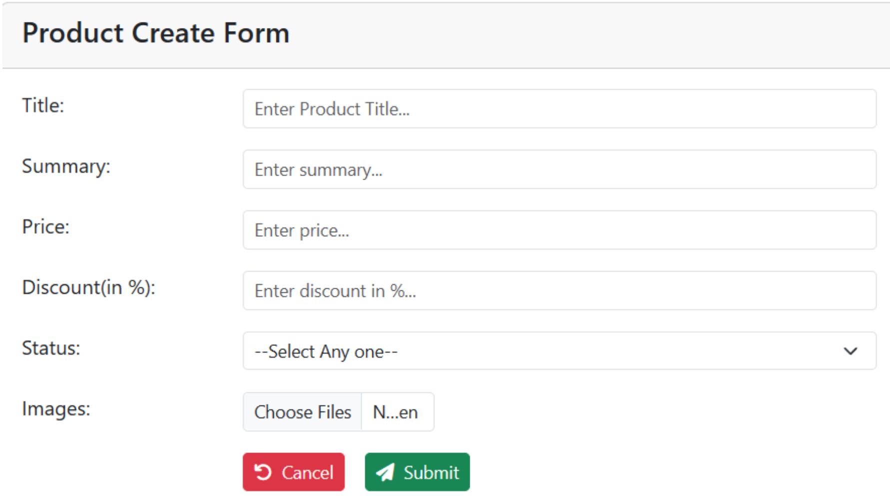


Figure 13: Create form

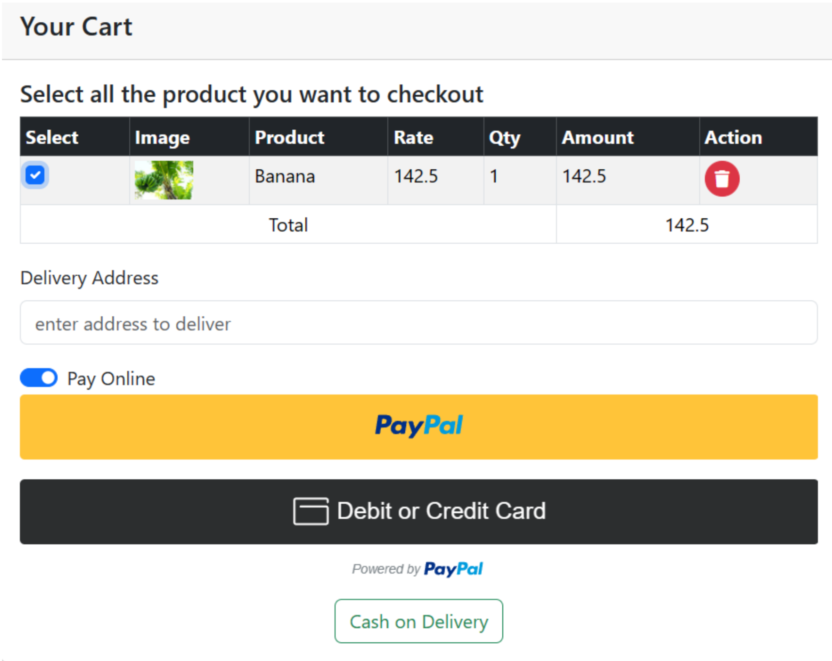


Figure 14: Checkout form

Dialog Design

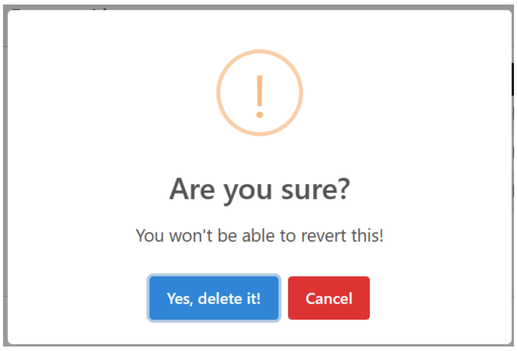


Figure 15: Dialog box

Interface design

There are two different interface design. One for landing pages. Other for Content Management System.

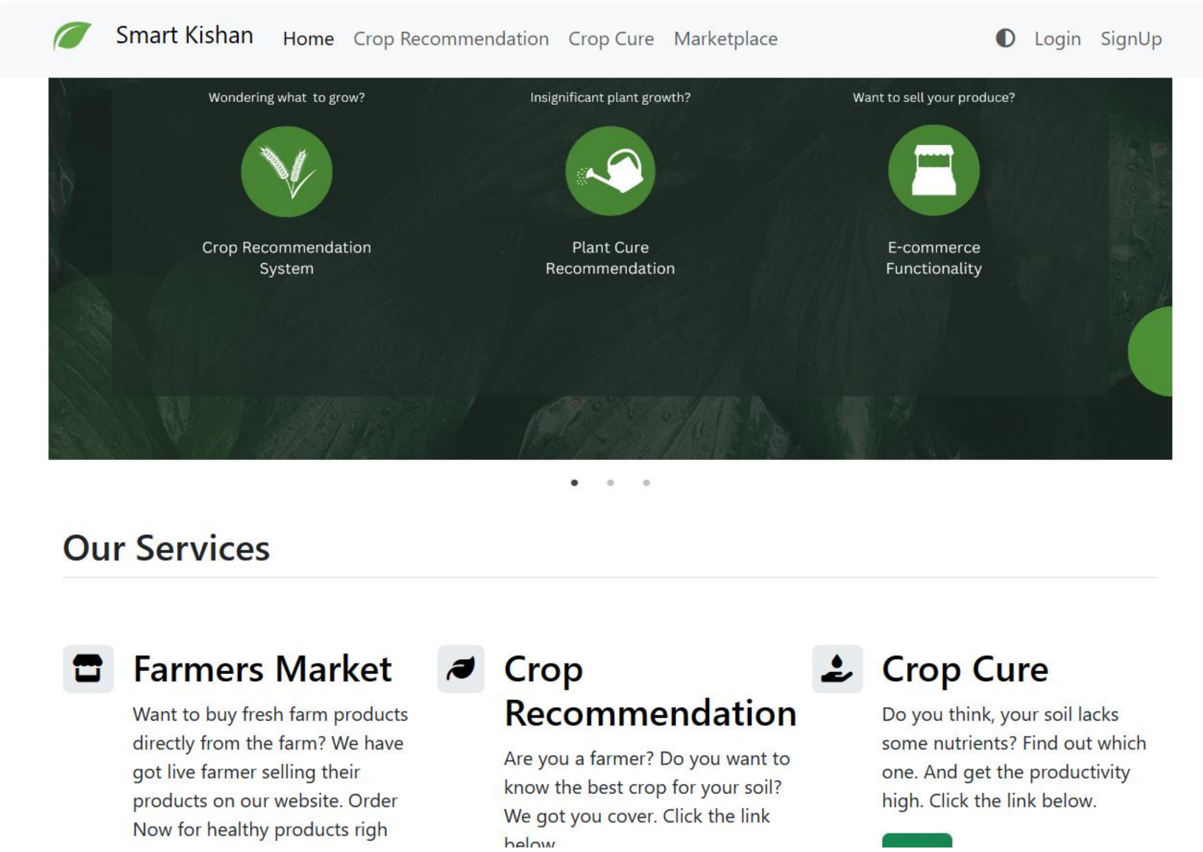


Figure 16: Landing Page Interface

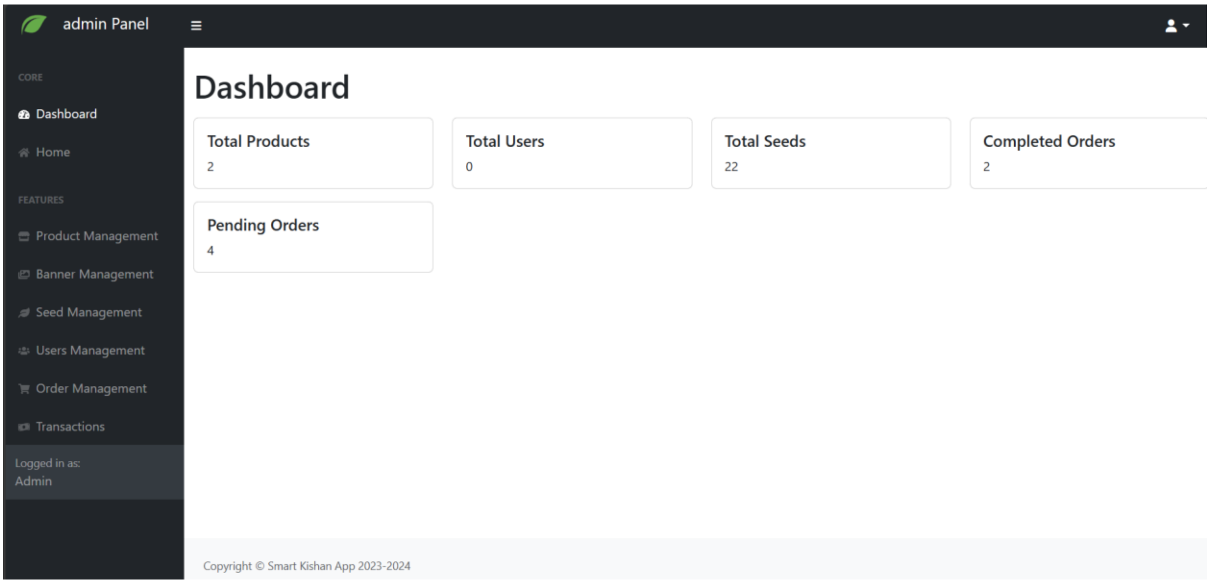


Figure 17: Content Management System Interface

4.2 Algorithm Detail

Random Forest Classifier

The Random Forest or Random Decision Forest is a supervised Machine learning algorithm used for classification, regression, and other tasks using decision trees. Random Forests are particularly well-suited for handling large and complex datasets, dealing with high-dimensional feature spaces, and providing insights into feature importance. This algorithm's ability to maintain high predictive accuracy while minimizing overfitting makes it a popular choice across various domains, including finance, healthcare, and image analysis, among others.

The Random Forest classifier creates a set of decision trees from a randomly selected subset of the training set. It is a set of decision trees (DT) from a randomly selected subset of the training set and then it collects the votes from different decision trees to decide the final prediction.

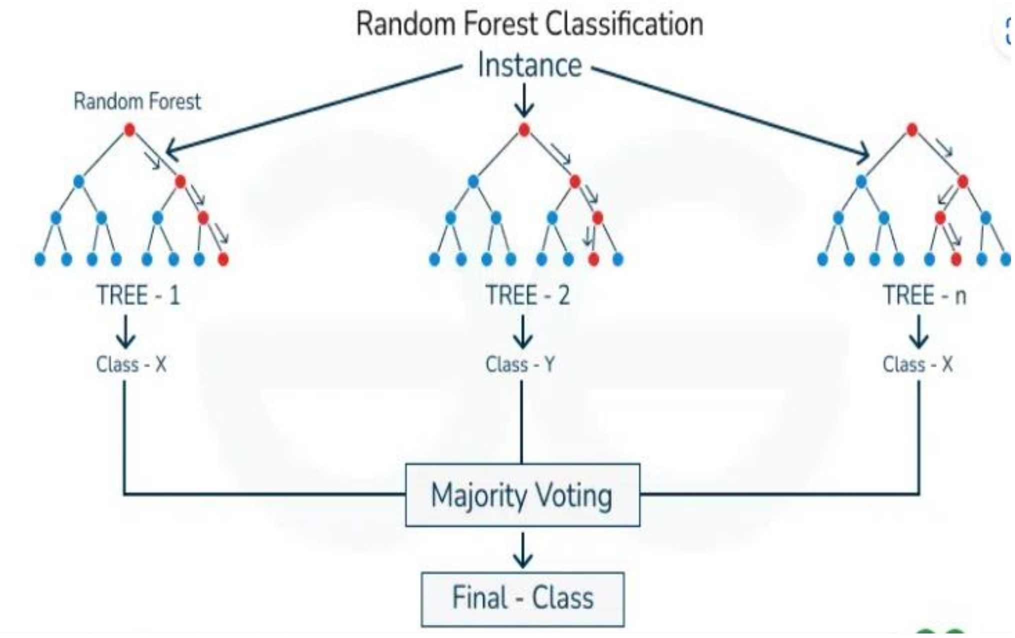


Figure 18: Random Forest Classification

Each decision tree in the random forest is constructed using a subset of the training data and a random subset of features introducing diversity among the trees, making the model more robust and less prone to overfitting.

During the training phase, each tree is built by recursively partitioning the data based on the features. At each split, the algorithm selects the best feature from the random subset, optimizing for information gain or Gini impurity. The process continues until a predefined stopping criterion is met, such as reaching a maximum depth or having a minimum number of samples in each leaf node.

For creating Decision Tree, we used CART Algorithm. CART stands for Classification and Regression Tree Algorithm.

CART Algorithm

Classification and Regression Trees (CART) is a decision tree algorithm that is used for both classification and regression tasks. It is a supervised learning algorithm that learns from labelled data to predict unseen data.

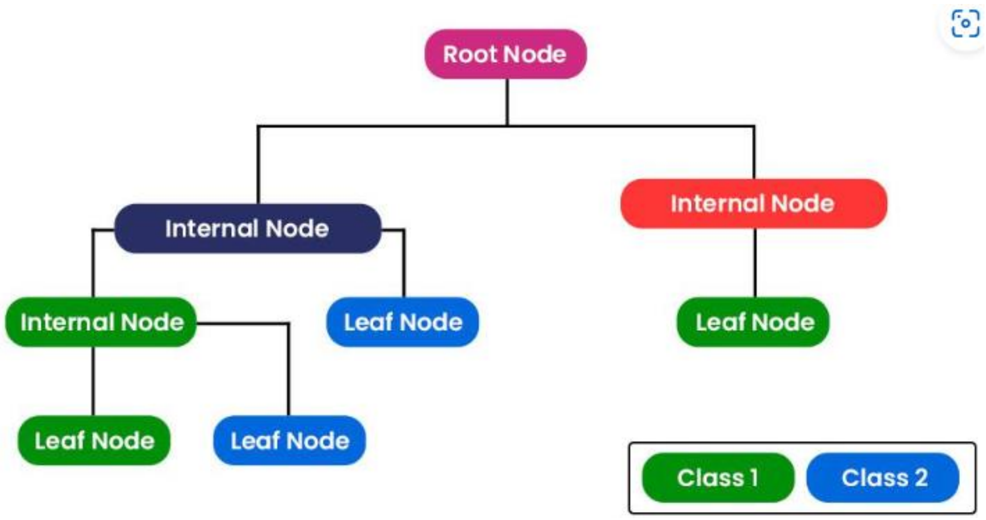
Tree structure: CART builds a tree-like structure consisting of nodes and branches. The nodes represent different decision points, and the branches represent the possible outcomes of those decisions. The leaf nodes in the tree contain a predicted class label or value for the target variable.

Splitting criteria: CART uses a greedy approach to split the data at each node. It evaluates all possible splits and selects the one that best reduces the impurity of the resulting subsets. For classification tasks, CART uses Gini impurity as the splitting criterion. The lower the Gini impurity, the more pure the subset is. For regression tasks, CART uses residual reduction as the splitting criterion. The lower the residual reduction, the better the fit of the model to the data.

Pruning: To prevent overfitting of the data, pruning is a technique used to remove the nodes that contribute little to the model accuracy. Cost complexity pruning and information gain pruning are two popular pruning techniques. Cost complexity pruning involves calculating the cost of each node and removing nodes that have a negative cost. Information gain pruning involves calculating the information gain of each node and removing nodes that have a low information gain.

The CART algorithm works via the following process:

- The best-split point of each input is obtained.
- Based on the best-split points of each input in Step 1, the new "best" split point is identified.
- Split the chosen input according to the "best" split point.
- Continue splitting until a stopping rule is satisfied or no further desirable splitting is available.



CART algorithm uses Gini Impurity to split the dataset into a decision tree. It does that by searching for the best homogeneity for the sub nodes, with the help of the Gini index criterion.

Chapter 5: Implementation and Testing

5.1 Implementation

5.1.1 Tools Used

1. Visual Studio Code:
   We used VS Code IDE for development of the project. Visual Studio Code (VS Code) is a free source-code editor developed by Microsoft for Windows, Linux, and macOS. It offers a wide range of features for developers, including support for debugging, syntax highlighting, intelligent code completion, code refactoring, and Git integration.
   One of the key features of Visual Studio Code is its extensibility, which allows developers to customize and enhance their coding experience through various extensions available in the Visual Studio Code Marketplace. These extensions can add support for additional programming languages, provide new themes, integrate with other tools and services, and more.

2. Front End Tools:
   For designing front end, we used ReactJS. ReactJS is a JavaScript library developed by Facebook for building user interfaces (UIs) for web applications. It's often used for creating interactive and dynamic UI components in single-page applications (SPAs) and mobile applications.
   We used React bootstrap to develop frontend components. React Bootstrap is a library that combines the power of React components with the styling and components of the Bootstrap framework. Bootstrap is a popular front-end framework for building responsive and mobile-first websites and web applications. React Bootstrap allows developers to use Bootstrap's UI components in their React applications, leveraging the advantages of both technologies.
   Additionally, we used other tools for maintaining aesthetics of the software like font-awesome, styled components, slick-carousel, react-toastify, etc.

3. Back End Tools:
   The backend of the project can be divided into two parts. One is machine learning backend and other is backend of ecommerce functionality.

   Backend of machine learning is built on python and integrated into flask application. Python is widely favored for machine learning due to its simplicity, extensive library support, vibrant community, flexibility, and seamless integration with other technologies. Its straightforward syntax makes it accessible to both beginners and experienced developers, enabling them to focus on solving machine learning problems rather than grappling with complex language intricacies. Python boasts a rich ecosystem of libraries tailored for data manipulation and machine learning tasks, such as NumPy, Pandas, and scikit-learn, facilitating efficient data preprocessing, model building, and evaluation.

   Backend of ecommerce functionality was built on NodeJS runtime environment using Express framework. Node.js is a runtime environment that allows developers to run JavaScript on the server-side. Express.js is a web application framework for Node.js, designed for building web applications and APIs.When used together, Node.js with Express provides a powerful platform for creating server-side applications in JavaScript. Express simplifies the process of handling HTTP requests and responses, routing, middleware integration, and more, while Node.js enables high-performance, non-blocking I/O operations, making it ideal for building scalable and efficient server-side applications.

Additionally, JWT was integrated for authorization. Multer was used for file uploading and handling in server side.

4. Database:
MongoDB database was used as database. MongoDB is a popular open-source NoSQL database management system that stores data in flexible, JSON-like documents. It's known for its scalability, flexibility, and performance, making it suitable for a wide range of applications, from small startups to large enterprises.

5.1.2 Implementation Details of Modules

1. For Crop Recommendation System

Below is a screenshot of first five data of data set:

	N	P	K	temperature	humidity	ph	rainfall	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

The function to read the dataset is as:

```
def read_csv(filename):

  data = []

  with open(filename, 'r') as file:

    header = next(file)  # Skip the header row

    for line in file:

      data.append(line.strip().split(','))

  return data
```

Crop label Dictionary to convert string label into integer:

```
crop_dict = {
   'rice': 1,
   'maize': 2,
   'jute': 3,
   'cotton': 4,
   'coconut': 5,
   'papaya': 6,
   'orange': 7,
   'apple': 8,
   'muskmelon': 9,
   'watermelon': 10,
   'grapes': 11,
   'mango': 12,
   'banana': 13,
   'pomegranate': 14,
   'lentil': 15,
   'blackgram': 16,
   'mungbean': 17,
   'mothbeans': 18,
   'pigeonpeas': 19,
   'kidneybeans': 20,
   'chickpea': 21,
   'coffee': 22
}
```

This dictionary is used to create a new column for converting string label into numeric value by the line of code: y=np.array([crop_dict[label] for label in y])

The features and labels are preprocessed by following function:

```
def preprocess_data(data):

  data_array = np.array(data)

  X = data_array[:, :-1].astype(float)

  y = data_array[:, -1]

  # Convert labels to integers using crop_dict

  y = np.array([crop_dict[label] for label in y])

  return X, y
```

Now the data is splitted into training and testing data. Train size was was taken 0.8 part of size of features and test size was 0.2 part.

```
train_size = int(0.8 * len(features))
  X_train, X_test = features[:train_size], features[train_size:]
  y_train, y_test = labels[:train_size], labels[train_size:]
```

Now we create random forest classifier's object

```
rf_classifier = RandomForest(n_estimators=100)
```

The equivalent class for random forest algorithm is

```
class RandomForest:
  def __init__(self, n_estimators=100, max_depth=None, sample_size=None):
    self.n_estimators = n_estimators
    self.max_depth = max_depth
    self.sample_size = sample_size
    self.estimators = []

  def fit(self, X, y):
    for _ in range(self.n_estimators):
      tree = DecisionTree(max_depth=self.max_depth)
      if self.sample_size:
        indices = np.random.choice(len(X), size=self.sample_size, replace=True)
        X_sample = X[indices]
        y_sample = y[indices]
      else:
        X_sample = X
        y_sample = y
      tree.fit(X_sample, y_sample)
      self.estimators.append(tree)

  def predict(self, X):
    predictions = np.array([estimator.predict(X) for estimator in self.estimators])
    return np.apply_along_axis(lambda x: np.bincount(x).argmax(), axis=0, arr=predictions)
```

Then we call the fit function in random forest classifier's object to train our split train data

```
rf_classifier.fit(X_train, y_train)
```

This fit function creates object of Decision tree. The related Decision Tree Class is:

```
class DecisionTree:
    def __init__(self, max_depth=None):
        self.max_depth = max_depth

    def fit(self, X, y):
        self.tree = self._grow_tree(X, y)

    def _grow_tree(self, X, y, depth=0):
        num_samples, num_features = X.shape
        num_classes = len(np.unique(y))

        # Stopping criteria
        if (self.max_depth is not None and depth >= self.max_depth) or num_classes == 1:
            return {'class': np.bincount(y).argmax(), 'num_samples': num_samples, 'depth': depth}

        # Find the best split
        best_feature = None
        best_threshold = None
        best_gini = np.inf
        for feature in range(num_features):
            thresholds = np.unique(X[:, feature])
            for threshold in thresholds:
                left_indices = np.where(X[:, feature] <= threshold)[0]
                right_indices = np.where(X[:, feature] > threshold)[0]
                if len(left_indices) == 0 or len(right_indices) == 0:
                    continue
                gini = self._gini_impurity(y[left_indices], y[right_indices])
                if gini < best_gini:
                    best_feature = feature
                    best_threshold = threshold
                    best_gini = gini

        if best_gini == np.inf:
            return {'class': np.bincount(y).argmax(), 'num_samples': num_samples, 'depth': depth}

        # Split the dataset
        left_indices = np.where(X[:, best_feature] <= best_threshold)[0]
        right_indices = np.where(X[:, best_feature] > best_threshold)[0]
        left_tree = self._grow_tree(X[left_indices, :], y[left_indices], depth + 1)
        right_tree = self._grow_tree(X[right_indices, :], y[right_indices], depth + 1)

        return {'feature': best_feature, 'threshold': best_threshold,
                'left': left_tree, 'right': right_tree, 'num_samples': num_samples, 'depth': depth}

    def _gini_impurity(self, left_y, right_y):
        p_left = len(left_y) / (len(left_y) + len(right_y))
        p_right = len(right_y) / (len(left_y) + len(right_y))
        return p_left * self._gini(left_y) + p_right * self._gini(right_y)

    def _gini(self, y):
        if len(y) == 0:
            return 0
        p = np.bincount(y) / len(y)
        return 1 - np.sum(p ** 2)
```

After that fit function of Random Forest call fit function of Decision which will create decision trees from the training set. After that all the decision trees are appended in the fit function of Random Forest. Now the model is ready to make predictions. We need new conditions taken as input from the user to make prediction.

```
predictions = rf_classifier.predict(condn)
```

The condn parameter is supply via user input as:

```
@ app.route('/crop-predict', methods=['POST'])
def crop_prediction():
    title = 'Crop Recommendation Module'
    print("hello")
    if request.method == 'POST':
        N = int(request.form['nitrogen'])
        P = int(request.form['phosphorous'])
        K = int(request.form['potassium'])
        ph = float(request.form['ph'])
        rainfall = float(request.form['rainfall'])
        # temperature = float(request.form['temperature'])
        # humidity = float(request.form['humidity'])

        city = request.form.get("city")

        if weather_fetch(city) is not None:
            temperature, humidity = weather_fetch(city)
        else:
            return json.dumps({"error": "City not found"})

        feature_list = [N, P, K, temperature, humidity, ph, rainfall]
        single_pred = np.array(feature_list).reshape(1, -1)

        prediction = crop_recommendation(single_pred)

        response = {
            "result": prediction[0],
            "message": "Crop Predicted"
        }

        return json.dumps(response)
```

2. For Cure Recommendation System

We have a dataset which contains suitable value of N, P, and K for different crops. Below is the screenshot of few data.

	A	B	C	D	E
1		Crop	N	P	K
2	0	rice	80	40	40
3	3	maize	80	40	20
4	5	chickpea	40	60	80
5	12	kidneybea	20	60	20
6	13	pigeonpea	20	60	20
7	14	mothbean	20	40	20

Now, we are taking input of crop name and value of N, P, K from user as:

```
crop_name = str(request.form['cropname'])

  N = int(request.form['nitrogen'])

  P = int(request.form['phosphorous'])

  K = int(request.form['potassium'])
```

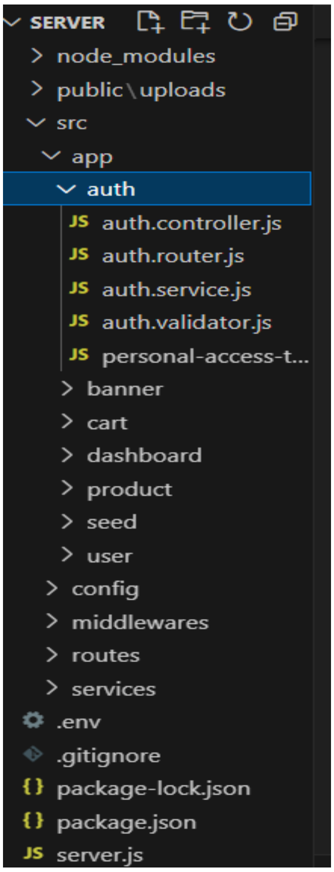
We extract suitable value of N, P, K as nr, pr, kr from file. Then we find n, p, k which is the difference between actual N, P, K and suitable nr, pr, kr respectively. Based on that we can determine which of the parameter is abundance or low in soil.

```
df = pd.read_csv('Data/fertilizer.csv')

  nr = df[df['Crop'] == crop_name]['N'].iloc[0]
  pr = df[df['Crop'] == crop_name]['P'].iloc[0]
  kr = df[df['Crop'] == crop_name]['K'].iloc[0]

  n = nr - N
  p = pr - P
  k = kr - K
  temp = {abs(n): "N", abs(p): "P", abs(k): "K"}
  max_value = temp[max(temp.keys())]
  if max_value == "N":
    if n < 0:
      key = 'NHigh'
    else:
      key = "Nlow"
  elif max_value == "P":
    if p < 0:
      key = 'PHigh'
    else:
      key = "Plow"
  else:
    if k < 0:
      key = 'KHigh'
    else:
      key = "Klow"
```

We create a dictionary of cure for each case of high and low. We markup the dictionaries content and dumps the final cure recommendation.

```
response = Markup(str(fertilizer_dic[key]))
  res = {
        "result": response,
        "message": "Recommended cure"
     }

  return json.dumps(res)
```

3. For ecommerce functionality:
No major algorithm was used in this functionality. An express app was setup in server file and the system was broken down into modules; auth, banner, cart, dashboard, product, seed, and user. MVC pattern was followed. Folder were organized into modular architecture. Each folder of modules contain file for router, controller, service validator and model. Common middleware, routes and config are stored in separate folders. Project variable are controlled from '.env' file.



5.2 Testing

5.2.1 Test Cases for Unit Testing

Table 1: Unit Testing of Authentication and Authorization

Test	Description	Test Step	Expected Result	Status
TA1	User Registration test	Go to sign up page, fill registration form	After filling registration form, user will get verification link at specified email	pass
TA2	Verification	Check verification email and click on the link provided to verify	If verified, user will redirect to password set page	pass
TA3	Set Password	In password set page, create password. Only password with Uppercase, Lowercase, Number, Symbol and at least 8 characters are allowed.	Password should be stored in database in encrypted form and user should be redirect into login page.	pass
TA4	Login	Provide login credential in login page.	For Admin, redirect in ADMIN CMS. For Farmer, redirect in FARMER CMS. For Consumer, redirect in HOMEPAGE.	pass

TA5	Permission Test	Browse Admin Dashboard URL, while login as User	Block Admin Dashboard and redirect in homepage	pass
TA6	Logout	Click on logout	User should be logged out	pass

Table 2: Unit test of Banner Management

Test	Description	Test Step	Expected Result	Status
TB1	Banner Create	Go to Banner management in sidebar of admin panel. Click on create and provide necessary information to create banner.	Banner should be created and displayed in homepage.	pass
TB2	Banner List	Go to Banner Management page	There should be list of all the banners created by logged in user	pass
TB3	Banner Update	Click on the yellow pencil icon button at the last column of banner listing and edit the necessary information	The edited information should be reflected.	pass
TB4	Banner Delete	Click on the red trash icon button at the last column of banner listing	Confirmation dialog should appear and upon confirming the related banner data should be deleted from database	pass

Table 3: Unit Testing of Product module

Test	Description	Test Step	Expected Result	Status
TP1	Product Create	Go to Product management in sidebar of admin panel. Click on create and provide necessary information to create product.	Product should be created and displayed in Marketplace.	pass
TP2	Product List	Go to Product Management page	There should be list of all the products created by logged in user	pass
TP3	Product Update	Click on the yellow pencil icon button at the last column of product listing and edit the necessary information	The edited information should be reflected.	pass
TP4	Product Delete	Click on the red trash icon button at the last column of product listing	Confirmation dialog should appear and upon confirming the related product data should be deleted from database	pass

Table 4: Unit Testing of Seed module

Test	Description	Test Step	Expected Result	Status
TS1	Seed Create	Go to Seed management in sidebar of admin panel. Click on create and provide necessary information to create product.	Seed should be created and displayed as product to add on cart when predicted by crop recommendation system.	pass
TS2	Seed List	Go to Seed Management page	There should be list of all the seeds created by logged in user.	pass
TS3	Seed Update	Click on the yellow pencil icon button at the last column of seed listing and edit the necessary information.	The edited information should be reflected.	pass
TS4	Seed Delete	Click on the red trash icon button at the last column of seed listing.	Confirmation dialog should appear and upon confirming the related seed data should be deleted from database.	pass

Table 5: Unit testing of User Module

Test	Description	Test Step	Expected Result	Status
TU1	Create Admin	Login as admin and go to user management. Click on create admin to create a new admin.	New user with role admin should be created.	pass
TU2	User List	Go to User Management page	There should be list of all the Users.	pass
TU3	User Update	Click on the yellow pencil icon button at the last column of user listing and edit the necessary information.	The edited information should be reflected.	pass
TU4	User Delete	Click on the red trash icon button at the last column of user listing.	Confirmation dialog should appear and upon confirming the related user data should be deleted from database.	pass

Table 6: Unit testing of cart module

Test	Description	Test Step	Expected Result	Status
TAC1	Product Added to cart	In Marketplace, click add to cart button of any product	A modal for selecting quantity will popped and the product should be added to cart afterward.	pass
TAC2	Product checkout	In cart, select all the product to checkout, enter the delivery address and click checkout	Total should be displayed, and after clicking checkout order should be updated in seller’s profile	pass

Table 7: Unit testing of order module

Test	Description	Test Step	Expected Result	Status
TO1	Order Listing	Go to order management page	All the new order should be listed	pass
TO2	Mark Dispatched	At the last column, click the mark dispatched button	The order should be completed and completed order should be displayed in transaction page	pass

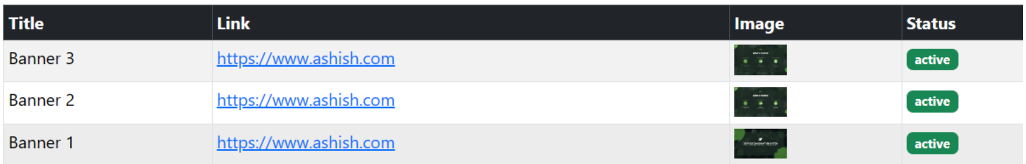
Table 8: Unit testing of transaction module

Test	Description	Test Step	Expected Result	Status
TT1	Transaction Listing	Go to transaction page	All the completed order should be displayed	pass

5.2.2 Test Cases for System Testing

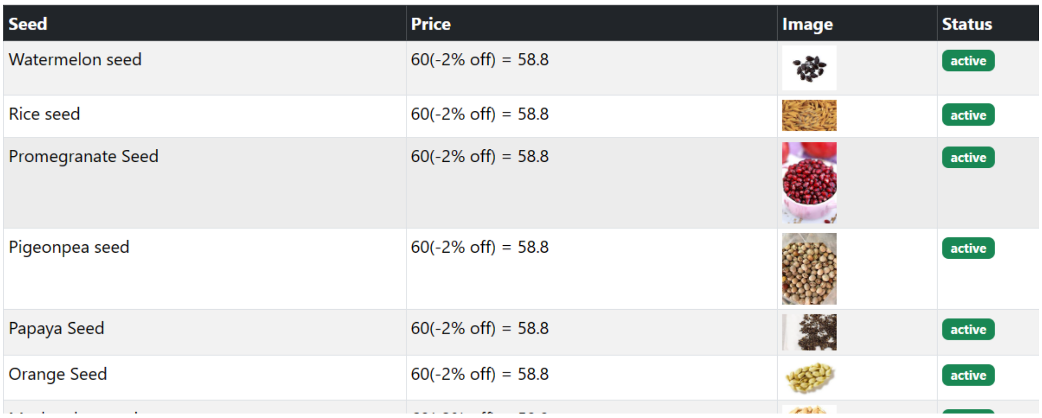
To test the proper functioning of the system, we take a test cases as:

1. We logged in as admin user and created new banner, seeds, product and new admin account from default admin account. Changes is expected to reflect.
   Following is the result. Few Banners, Seeds, Products and new admin is created from admin CMS.
   Banner:.



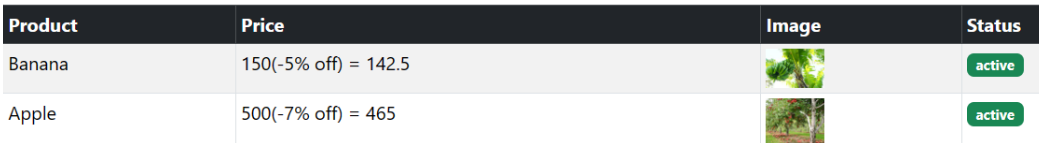
Title	Link	Image	Status
Banner 3	https://www.ashish.com		active
Banner 2	https://www.ashish.com		active
Banner 1	https://www.ashish.com		active

Seed:



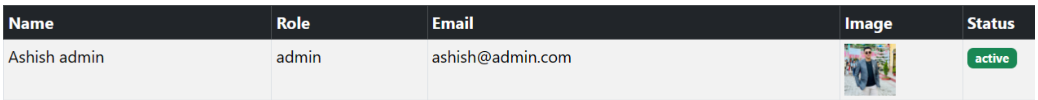
Seed	Price	Image	Status
Watermelon seed	60(-2% off) = 58.8		active
Rice seed	60(-2% off) = 58.8		active
Promegranate Seed	60(-2% off) = 58.8		active
Pigeonpea seed	60(-2% off) = 58.8		active
Papaya Seed	60(-2% off) = 58.8		active
Orange Seed	60(-2% off) = 58.8		active

Product:



Product	Price	Image	Status
Banana	150(-5% off) = 142.5		active
Apple	500(-7% off) = 465		active

New Admin:



Name	Role	Email	Image	Status
Ashish admin	admin	ashish@admin.com		active

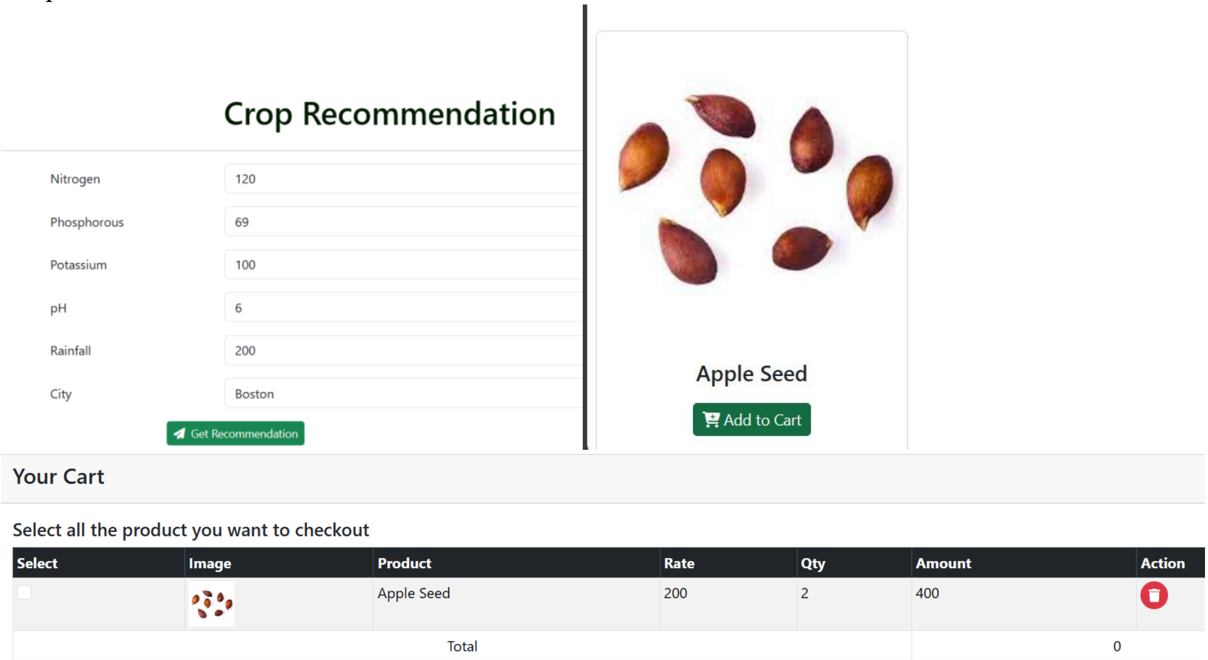
2. A new farmer user is signed up with the required credentials. Then the user first uses the crop recommendation system. A random data will be provided to the crop recommendation form. To get the crop recommendation. User must be able to order the recommended crop from the recommendation. The order must be shown in admin panel. Then farmer user will test cure recommendation system. A test data will be fed and the result for curing the crop should be obtained. From farmer CMS, we will create few products, which must be displayed in Farmer Marketplace.

   Result:

   New Farmer account:

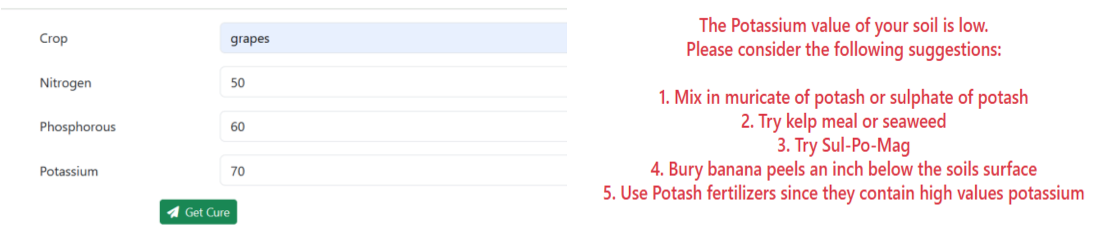

```
_id: ObjectId('65fbdd2c5d671f7478414108')
name : "Ashish Farmer"
email : "ashish@farmer.com"
role : "farmer"
token : null
image : "1711004972545_8930.jpg"
status : "active"
password : "$2a$10$AMmoQhjbaWMjJ/BNbTvi7eSLifM.zb9KL6.PFY5XKWUf8JlDmKBm."
resetToken : null
resetExpiry : null
createdAt : 2024-03-21T07:09:32.558+00:00
updatedAt : 2024-03-21T07:10:03.753+00:00
```


   Crop recommendation:



Select	Image	Product	Rate	Qty	Amount	Action
		Apple Seed	200	2	400	
		Total			0	

   Cure recommendation:



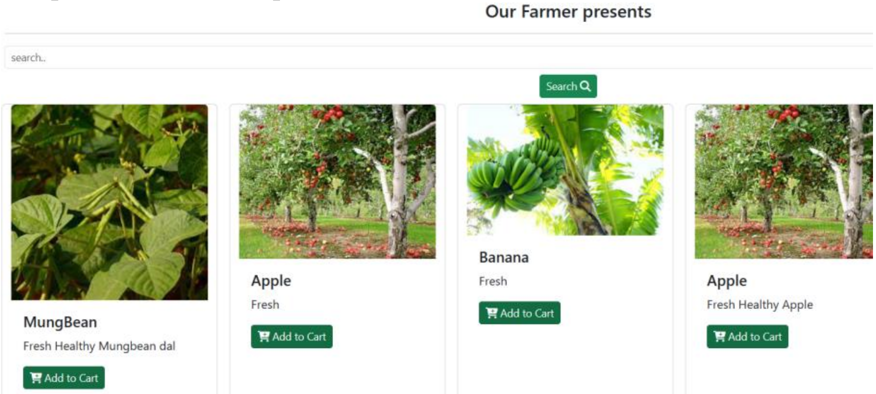
   Product created in farmer account

Product	Price	Image	Status
MungBean	250(-5% off) = 237.5		active

Order list:

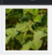
Product	Quantity	Customer	Payment	Amount	Status	Dispatch
MungBean	2	Name: Ashish Customer Contact: ashish@customer.com Delivery at: Chandragiri - 12	paid	475	new	✓ Mark

3. A new customer user is signed up with the required credentials. Then the customer user will navigate to farmer market place. Customer user must be able to place order. The order should be updated in farmer's panel.

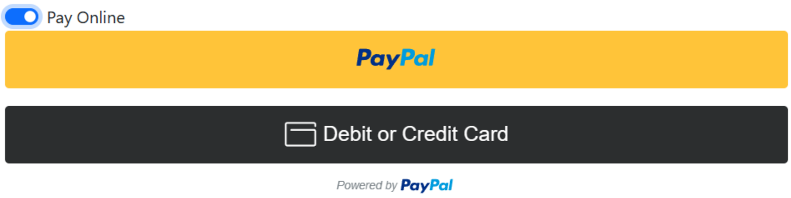


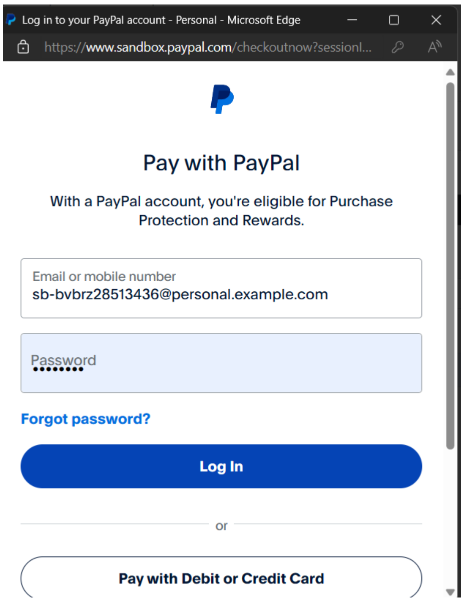
Your Cart

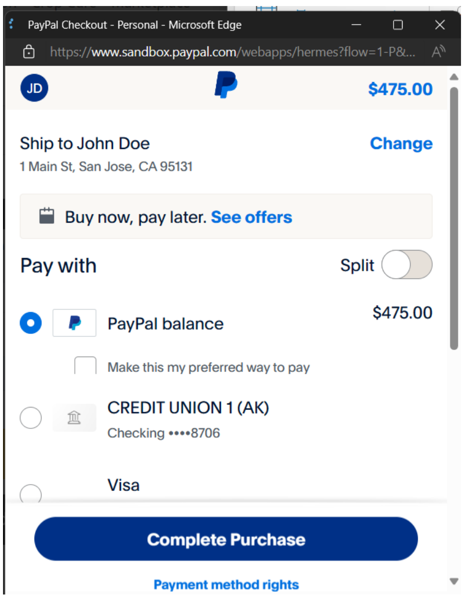
Select all the product you want to checkout

Select	Image	Product	Rate	Qty	Amount	Action
☑		MungBean	237.5	2	475	
		Total			475	

4. Additionally, paypal integration is also tested while ordering.







5.3 Result Analysis

Analysis of crop recommendation system:

From above user input data, system receive following feature list:
"feature_list": [120, 69, 100, -1.03, 53, 6.0, 200.0]
Where correspondence mapping is:
feature_list = [N, P, K, temperature, humidity, ph, rainfall]

The feature_list is fed into the model which has 100 decision tree. Each decision tree derive it's own conclusion. The conclusion with highest majority which is "apple" is resulted.

```
{"feature_list": [120, 69, 100, -1.12, 53, 6.0, 200.0], "message": "Crop
    Predicted", "result": "Apple"}
```

The confidence level of the certainity of the prediction is calculated as

```
Random Forest  with accuracy : 0.9954545454545455
```

Analysis of cure recommendation system:

Provided feature: ['grapes', 50, 60, 70]
In data set, suitable value of n, p, k is as follows:

63	grapes	20	125	200

The module calculate nr, pr, kr as:
nr = 50-20=30
pr = 60-125=-65
kr = 70-200 = -130
By taking absolute value of each, maximum value is kr. So potassium content is needed to be cure first. Since kr<0, we can assume potassium content is low in the soil. The corresponding dictionary recommendation for low potassium is as follows:

```
'Klow': """The Potassium value of your soil is low.
<br/>Please consider the following suggestions:

<br/><br/>1. Mix in muricate of potash or sulphate of potash
<br/>2. Try kelp meal or seaweed
<br/>3. Try Sul-Po-Mag
<br/>4. Bury banana peels an inch below the soils surface
<br/>5. Use Potash fertilizers since they contain high values potassium
"""
```

Thus, we get result as:

```
{"message": "Recommended cure", "result": "The Potassium value of your soil is low.\n
Please consider the following suggestions:\n\n

1. Mix in muricate of potash or sulphate of potash\n
2. Try kelp meal or seaweed\n
3. Try Sul-Po-Mag\n
4. Bury banana peels an inch below the soils surface\n
5. Use Potash fertilizers since they contain high values potassium\n "}
```

Chapter 6: Conclusion and Future Recommendation

6.1 Conclusion

The development of the proposed software solution for the farming cycle represents a significant step towards revolutionizing agricultural practices and empowering farmers with data-driven decision-making tools. By seamlessly integrating soil analysis, crop selection, procurement, plant care recommendations, and direct-to-consumer marketing functionalities, the software addresses key challenges faced by farmers and enhances productivity, sustainability, and market access within the agricultural sector.

Through the implementation of advanced algorithms such as the Random Forest classification algorithm, the software effectively predicts suitable crops based on soil nutrients and climatic conditions, enabling farmers to make informed decisions regarding crop selection. The direct online ordering feature further streamlines the procurement process, providing farmers with convenient access to quality seeds tailored to their specific requirements.

Additionally, the software's plant care recommendation module offers valuable guidance to farmers throughout the cultivation process, facilitating optimal fertilization and pest management strategies based on soil nutrient levels and crop type. The integration of e-commerce capabilities enables farmers to sell their produce directly to consumers, eliminating intermediaries and fostering stronger connections between producers and consumers.

6.2 Future Recommendation

In future, the crop recommendation system and cure recommendation system can be automated with used of sensors. NPK sensor is available in the market which measures Nitrogen, Phosphorous and Potassium content of the soil. Soil moisture sensor with pH indicator is available in the market to measure humidity and pH value of the soil. Other data can be fetched from different API. The wi-fi module can be attach in Arduino board along with other sensor to get real time data from the field and get real time recommendation without the need of inputting soil data by user. This will not only made the process easier but also provide real time data.

Other future recommendations are as follows:

a. Enhanced Data Analytics: Continuously improve the software's data analytics capabilities by incorporating additional factors such as soil texture, crop history, and environmental factors to further enhance crop prediction accuracy and recommendation precision.
b. Mobile Application Development: Develop a mobile application version of the software to provide farmers with on-the-go access to essential features and functionalities, enhancing usability and convenience.
c. Expansion of Market Connectivity: Explore opportunities to expand the software's market connectivity beyond direct-to-consumer sales, such as partnerships with local markets, restaurants, and food delivery services to broaden the reach and impact of farmers' produce.
d. Integration with IoT Devices: Incorporate Internet of Things (IoT) devices such as soil sensors, weather stations, and drones to enable real-time data collection and monitoring, facilitating proactive decision-making and resource optimization.
e. Community Engagement and Training: Organize workshops, training sessions, and community outreach programs to educate farmers on the benefits and usage of the software, fostering adoption and maximizing its impact on agricultural communities.
f. Continuous Improvement and Updates: Implement a feedback mechanism to gather user feedback and suggestions for software improvements, ensuring ongoing refinement and enhancement of features based on user needs and evolving industry trends.

References

[1] 2019 IEEE “Smart Management of Crop Cultivation using IoT and Machine Learning” Archana Gupta, Dharmil Nagda, Pratiksha Nikhare, Atharva Sandbhor

[2] “Crop Recommendation on Analyzing Soil Using Machine Learning” Anguraj.Ka, Thiyaneswaran.Bb, Megashree.Gc, Preetha Shri.J.Gd, Navya.Se, Jayanthi. Jf, 2020.

[3] “Classification of Soil and Crop Suggestion using Machine Learning Techniques”, A. Mythili, IEEE 2019.

[4] “IOT based Crop Recommendation, Crop Disease Prediction and Its Solution” Rani Holambe, Pooja Patil, Padmaja Pawar, Saurabh Salunkhe, Mr. Hrushikesh Joshi, 2019 IRJET

[5] Kaggle: https://www.kaggle.com/datasets

Appendices

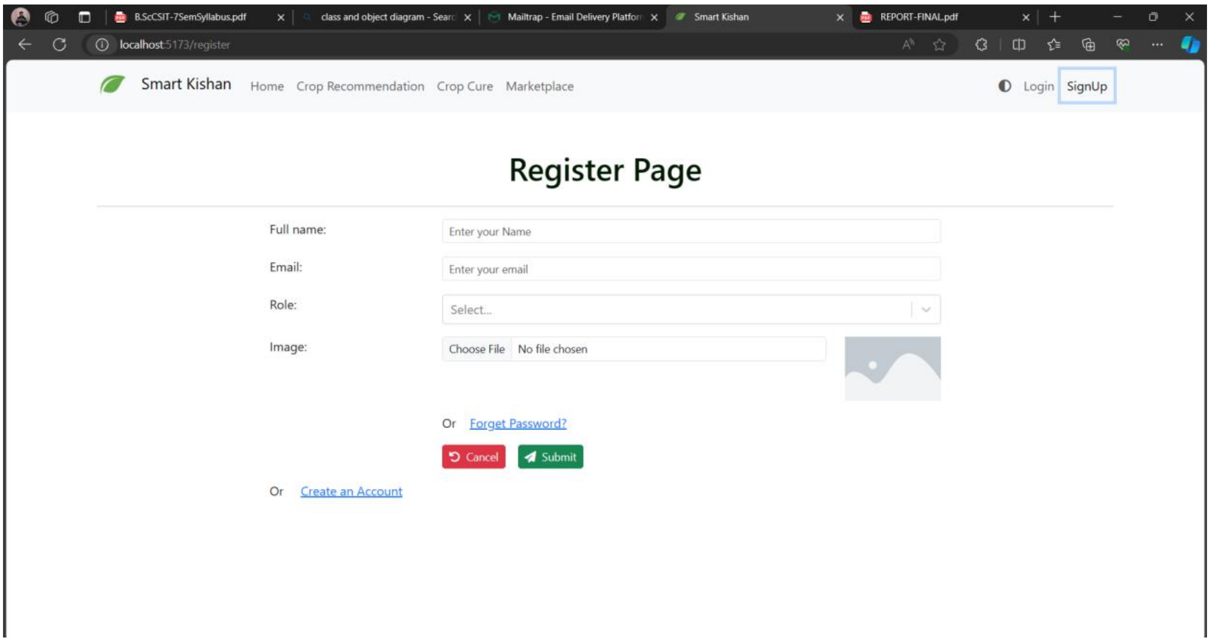


Figure 19: Sign Up page

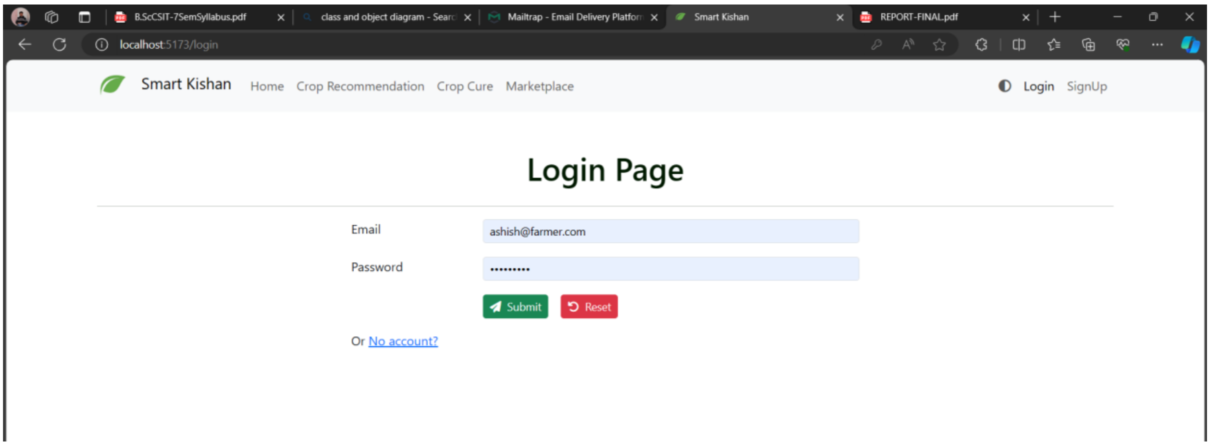


Figure 20: Login Page

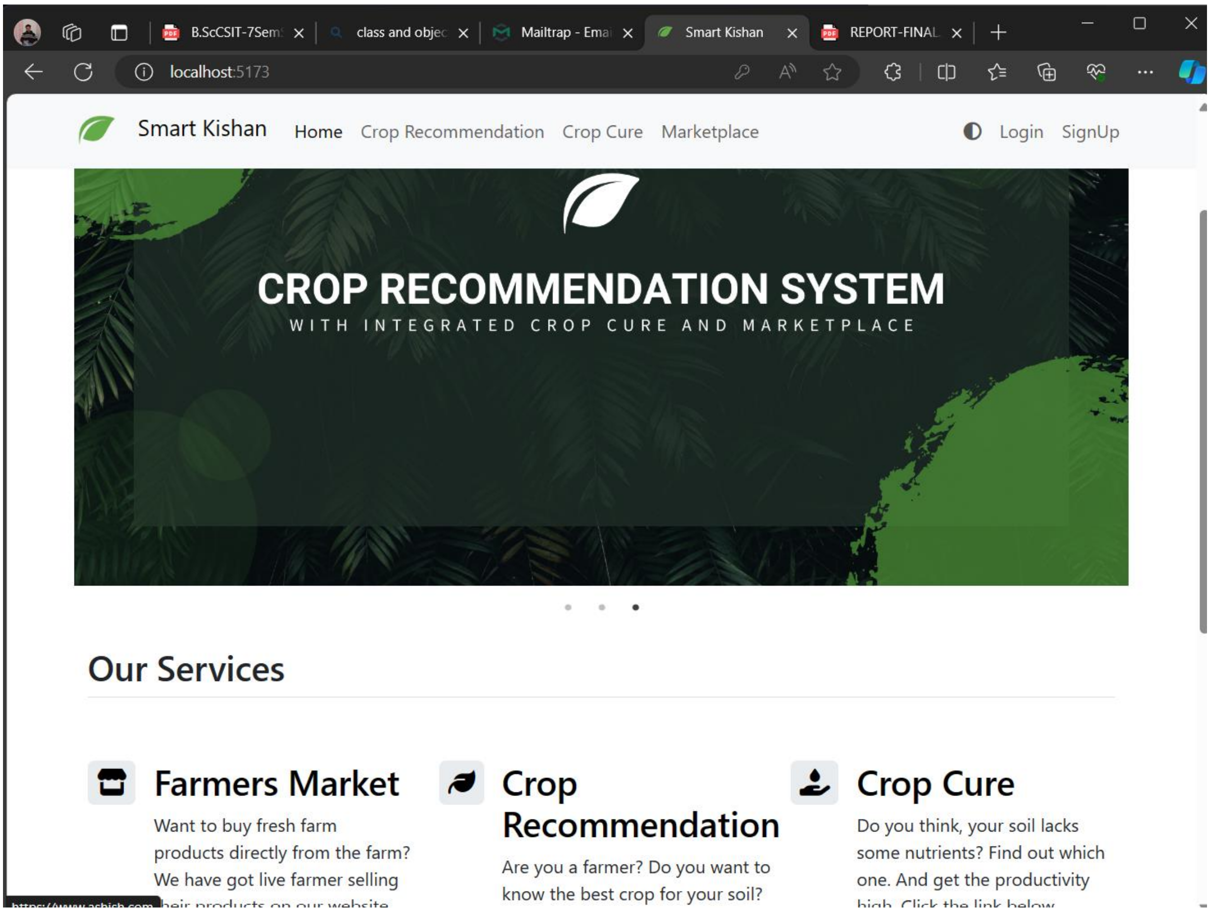


Figure 21: Landing Page

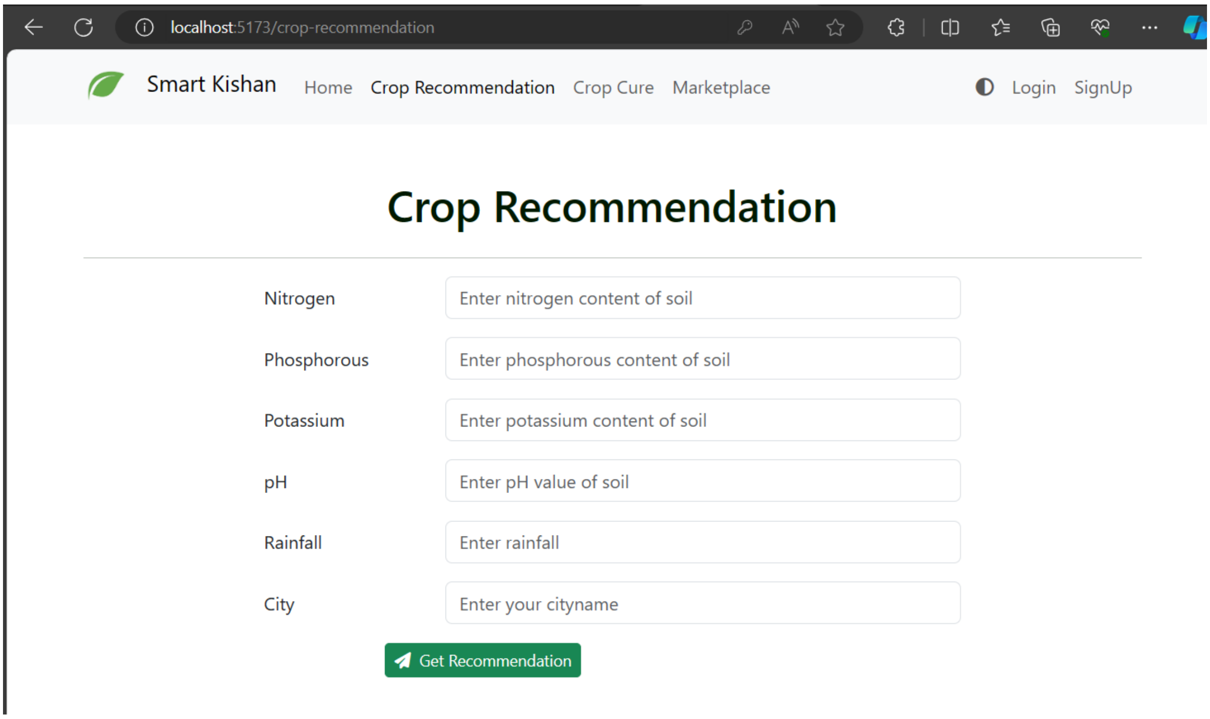


Figure 22: Crop Recommendation Form

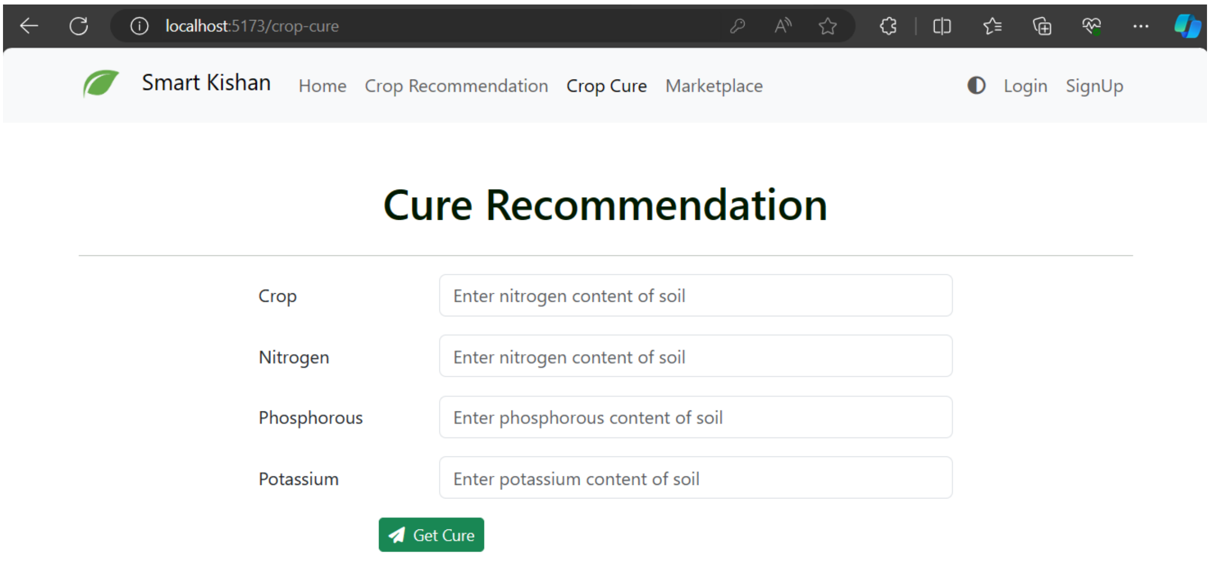


Figure 23: Crop Cure recommendation form

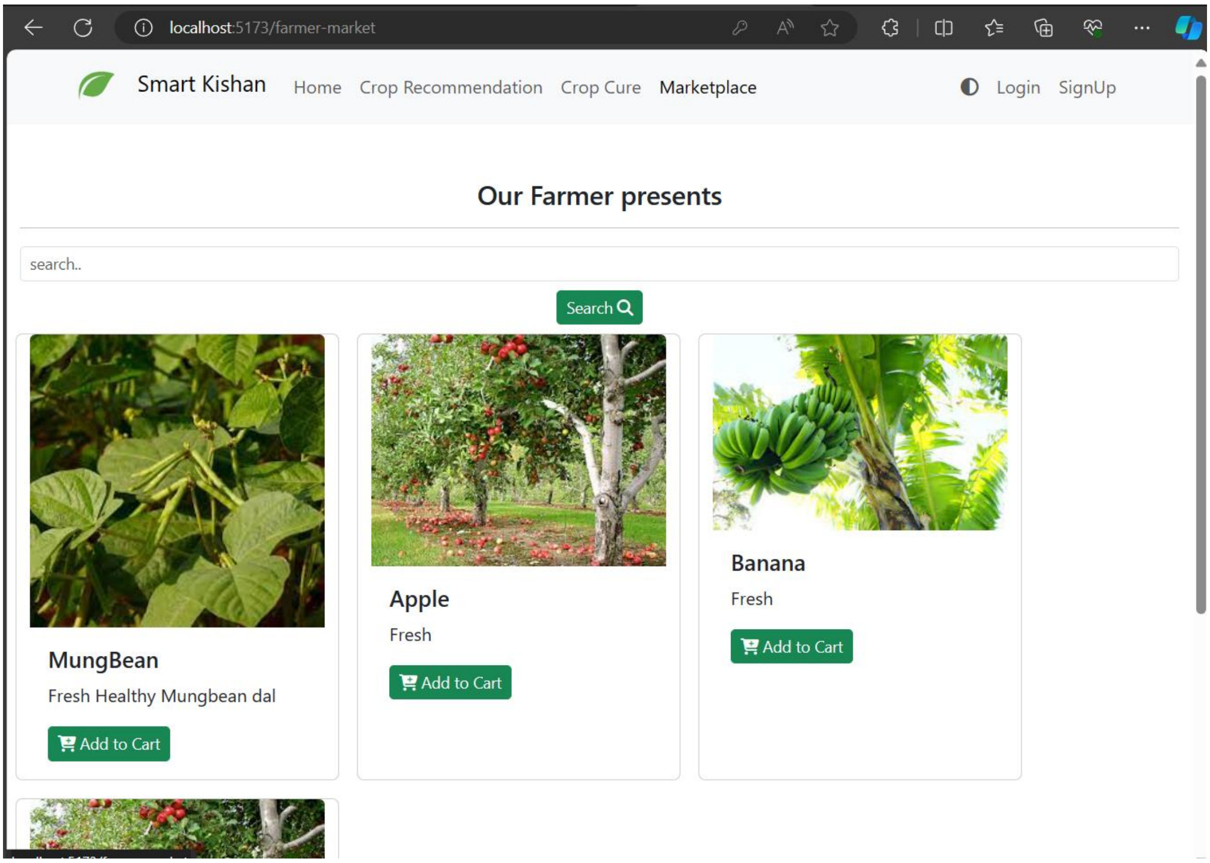


Figure 24: Marketplace

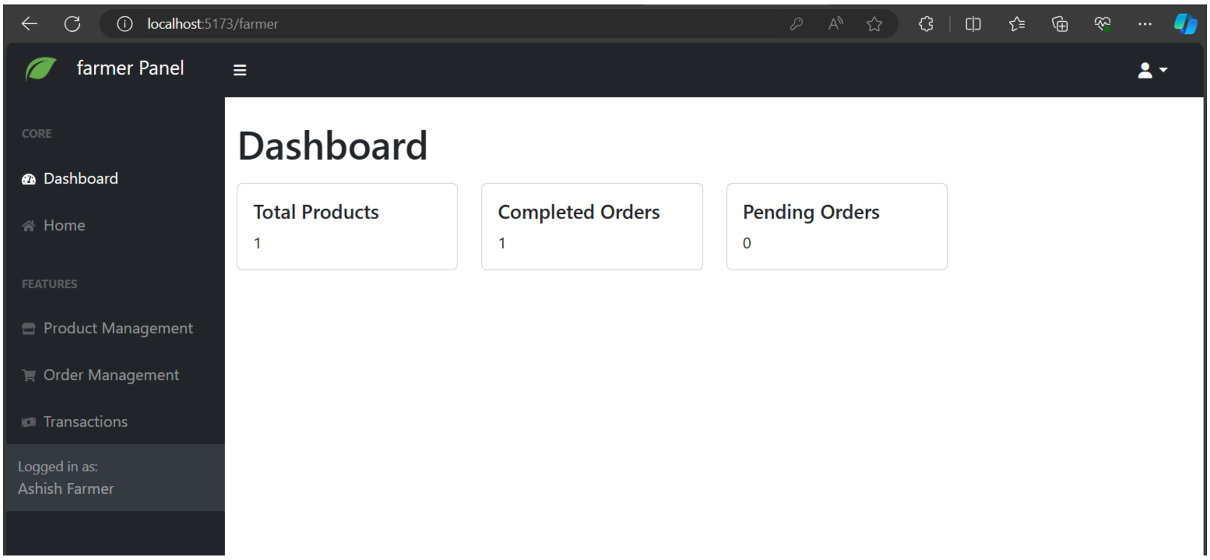


Figure 25: Farmer CMS dashboard

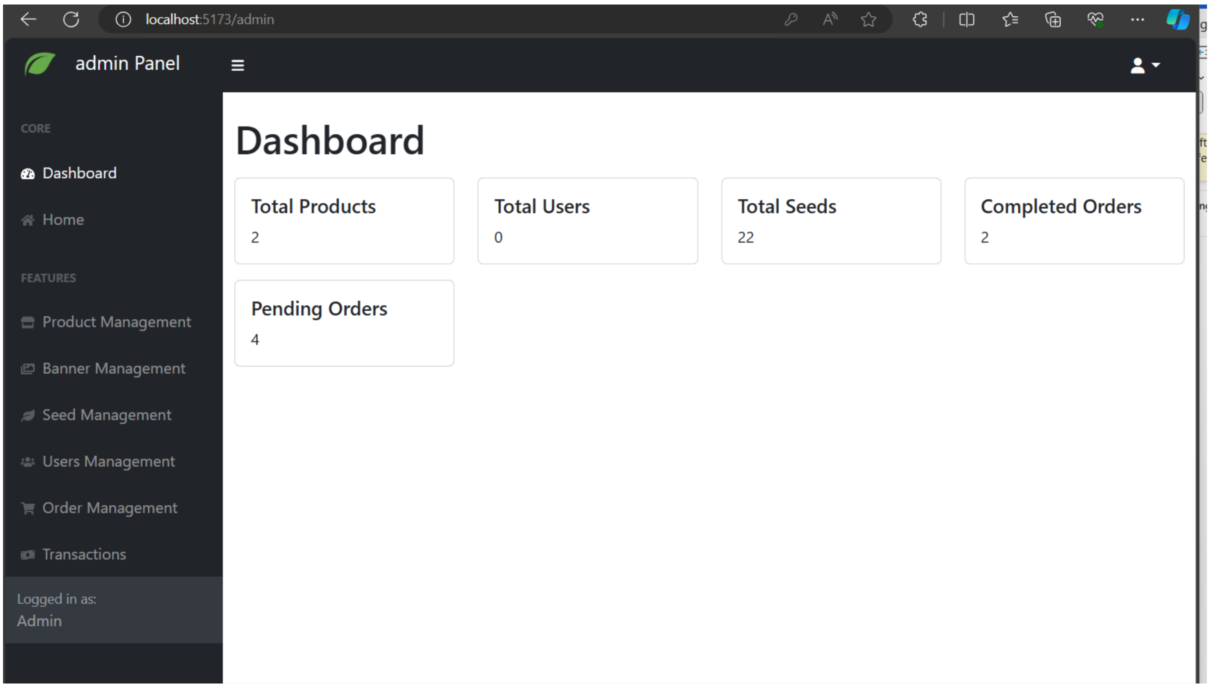


Figure 26: Admin CMS dashboard