

*A Project Report on*

**AgroFlo**

Submitted in partial fulfilment for the award of the degree of

**B.Tech (CSE)**

*By*

**Varun Mulchandani - 18BCE2507**

**Prateek Sharma - 18BCI0215**

**Saksham Gupta - 18BCI0273**

**Under the guidance of  
Dr. Sweta Bhattacharya  
Assistant Professor (Senior)**



**SCHOOL OF COMPUTER SCIENCE & ENGINEERING**

**April 2020**

## **Abstract:**

The purpose of this project is to build an application to enable farmers and customers to make sale transactions efficiently. Ridden by various scams over time, the plight of the Indian farmer has only been getting worse. It has thus become essential to remove them from the picture and this is the main purpose of this project.

AgroFlo helps the farmer and the customer, by eliminating the middleman and making sure both the parties be it customer or the farmer have a seamless experience while either buying or selling items at a rate that is just and reasonable. AgroFlo helps revolutionize the agricultural sector.

## **Acknowledgement**

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and our classmates. We would like to extend my sincere thanks to all of them.

We are highly indebted to our Professor Dr. Sweta Bhattacharya Ma'am for her guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

## **1. Introduction**

### **1.1 Motivation**

Farmers in rural areas are often exploited by middlemen and are offered lesser prices for their produce than what the middlemen sell it for. On top of that, since the transaction with the middlemen is on an informal and usually undocumented basis, it cannot be used as proof to get loan from banks for the following crop season. The project aims to create a platform on which the consumer places an order for a certain amount of goods and the order is assigned to a farmer, or a group of farmers, who have that amount of produce up for sale. The priority of assignment of the order will be based on the income of the farmers from the app, and on a humanitarian basis, the order will be given to the farmer who has the required amount, and has earned the least in the current timeframe. Loans by banks can also be given based on transactions in the app and the credibility score generated using that. The loans taken by the farmer can also be used as another metric to assign priority to the farmer in

receiving orders, i.e. farmers who have the highest loan will be given a higher priority. Consumers can also rate the quality of the produce sold by the farmer. Farmers who are on the platform but cannot access a smartphone can receive orders using text messaging (SMS). Crops may also be suggested to the farmers based on soil analysis dataset.

## **1.2 Aim of the proposed work**

Ridden by various scams over time, the plight of the Indian farmer has only been getting worse. Numerous new articles highlight the middlemen taking advantage of farmers throughout the nation. It has thus become essential to remove them from the picture and this is the main focus of our project.

This project is applicable to farmers and customers all across the country as long as they abide to certain rules and regulations for fair transitions.

The deliverable product will be in the form of a mobile application which provides separate interfaces for the farmer and the customer to communicate and make transactions in an efficient manner along with other functionalities to engage activity. This includes recommendation for farmers to decide on the crops to grow based on conditions, providing an unbiased rating scale and creating a system to recommend farmers with low sales over a period of time to bridge the gap between the rich and the poor as much as possible.

The only criteria to be met is that the farmer and the customer have some communication device [preferably a smart phone with internet connectivity for making transactions easier]. This is the only constraint that may limit the scalability of this project.

The project scope does not involve us interfering between the transactions between the farmer and the customer in any direct way. All issues must be dealt with between the people at the two ends of transactions.

## **1.3 Objective(s) of the proposed work**

- The project aims to provide a platform for the farmer to directly list their produce which the customer can buy, thus eliminating the need to have a middleman.
- The application also aims to recommend the type of crop to be grown in the given season to the farmer based on soil analysis data.
- It will also serve as a log of income for the farmer, which can be used as a proof to banks while availing loans.

- The platform will act as an interface that gives a higher priority to the farmers who have made a smaller income in the current season.

## 1.4 Report Organisation

Abstract

Acknowledgement

1. Introduction
  1. 1.1.Motivation
  2. 1.2.Aim of the proposed Work
  3. 1.3.Objective(s) of the proposed work
  4. 1.4.Report Organization
2. Literature Survey
  1. 2.1.Survey of the Existing Models/Work
  2. 2.2.Summary/Gaps identified in the Survey
3. Proposed System Requirements Analysis and Design
  1. 3.1.Introduction
  2. 3.2.Requirement Analysis
    1. 3.2.1.Stakeholder Identification
    2. 3.2.2.Functional Requirements
    3. 3.2.3.Non Functional Requirements
    4. 3.2.4.System Requirements
      1. 3.2.4.1.H/W Requirements(details about Application-Specific Hardware)
      2. 3.2.4.2.S/W Requirements(details about Application-Specific Software)
    5. 3.2.5.Work breakdown structure
    6. 3.2.6.Pert chart (with task table)
    7. 3.2.7.Gantt Chart
4. Design of the Proposed System
  1. 4.1.Introduction
  2. High level Design (Framework, Architecture or Module for the Proposed System(with explanation))
    1. 4.2.1.Architecture design (choose the appropriate pattern with justification)
    2. 4.2.2.Architecture diagram (explanation)
    3. 4.2.3.UI design
  3. Detailed Design (ER Diagram/UML Diagram/Mathematical Modeling)
    1. 4.3.1.ER Diagram
    2. 4.3.2.UML diagram (Use case, class, Statechart, Activity and interaction diagrams)
5. Implementation and Testing (Snap shots with description)
  1. 5.1.Implementation details (snapshots)
  2. 5.2.Testing
    1. 5.2.1.Types of Testing
    2. 5.2.2.Testcases (for all modules as per the template)
6. Conclusion, Limitations and Scope for future Work
7. References

## **2. Literature Survey:**

### **2.1 Survey of Existing models that work:**

- a) SMARTCROP: Farmers can post their products and attract more buyers which will save precious time and money. They can get better profits than normal. According to the market prices, the product entries can be removed or re-activated by the users. As these portals reduce middleman buyers and sellers can privately chat to negotiate prices safely.
- b) DIGITAL MANDI INDIA: Digital Mandi India is a mobile application that helps an end user to check the latest Indian agricultural commodities mandi prices across different states and cities. An end user can browse through various commodity categories and states categories. It has a simple flow to reach the selected commodities mandi price making it user friendly. The data is synced from the Indian government portal Agmarknet.nic.in — Powered by NIC.
- c) AGRI MARKET: Agri Market is a user-friendly agriculture application to find market price of crops much easier for the user with the help of Global Positioning System (GPS). The application lets a user identify the market price of the crops within 50Kms of the device's location
- d) MANDI TRADES: Mandi trades is an app where you can find mandi prices, price alert, and food produced on demand. Commodity Prices & Farm Produce data sourced from Govt. of India.
- e) Crop recommendation system for precision agriculture: To recommend crops to farmers who are often unaware of what they have to grow based on their conditions. This research paper uses Random Forests and Naive Bayes to recommend crops to the farmers.
- f) Plant Disease Detection Using Image Processing: This research paper utilises image processing techniques like image segmentation and feature extraction to identify diseases in plants.

### **2.2 Summary/Gaps identified:**

A lot of these apps perform the basic functionality, which is enabling a platform for both farmers and consumers to participate in sales without the middlemen abusing the former and creating a hassle free environment.

Gap: A major gap/loophole in this environment is that the farmers are not given recommendations/feedback about the crops that have to be grown given their budget/demographic, nor does it put a system in place to check the growth of diseased crops in case the farmer is unfamiliar with the types of diseases that crops have.

Our software enables both these applications.

For Plant Disease detection, the above approach utilises image processing techniques as opposed to deep learning ones which can accelerate the rate of learning as well as produce results with higher accuracy.

### **3. Proposed system requirements Design and analysis**

#### **3.1 Introduction**

The Agro management system maintains information on the farmers and the customers. This includes the sales a farmer has made, the quality of crops sold, the prices for the crops and the transactions that have taken place over time. This would also include the priorities for the farmers.

This is a high priority project as farmers suffer a lot on a daily basis because of middlemen. Farmer suicide rate in India has ranged between 1.4 and 1.8 per 100,000 total population. This amounts to about 10 deaths per day and has been increasing since 2005.

#### **3.2 Requirement Analysis**

##### **3.2.1 Stakeholder Identification**

The stakeholders in this project are:

1. Developers
2. Customers
3. Farmers
4. Government Agencies
5. Logistics Partners
6. SMS service provider

##### **3.2.2 Function Requirements:**

1. Users must be able insert their details during sign up.
2. Users must enter their phone number and password to sign in.
3. Credentials are validated.
4. Error message if invalid credentials.
5. User selects farmer based on priority if the user is a customer. Selects the amount and type of crop required.
6. User agrees or disagrees to incoming order if the user is a farmer.

7. Customer gets notified if the order is rejected or accepted.
8. Payment through UPI/Net banking/Card - TBD.

### **3.2.3 Non Functional Requirements**

#### **Performance Requirements**

1. The database should contain all the tables in at least third normal form (3NF) to increase efficiency.
2. A bulk SMS handler also needs to be part of the package so that communication in the absence of the internet is possible.
3. A registration portal for farmers to register without a mobile phone must be integrated.

#### **Safety Requirements**

1. The database has to be backed up regularly and a solution like RAID 5 as it provides protection against data loss as well as parity checking and increased read and write speeds than data stored normally.
2. A good transactional model (ACID) must be used for the database as in its barest essence, the project is an e-commerce application.

#### **Security Requirements**

1. The security for the user data will be provided by the database itself.
2. For communication between the client devices (both buyers and farmers) secure communication methods (https) will be used to protect user information like payment method information and user logins.
3. The user passwords will be stored as salted hashes in the database.

#### **Software Quality Attributes**

1. Availability: The software will contain both an offline and online method to list produce for the farmer (considering that farmers are more likely to not have internet)
2. Usability: The application must have an easy to use interface as it is intended to be used by farmers in rural areas as well.
3. Portability: The application does not use much of the storage of the device it is installed on considering low end phones that will be using it as well.

### **3.2.4 System Requirements**

### 3.2.4.1 Hardware Requirements:

To run our application on a device the device must be running at least Android version 5.0 and should have a minimum RAM of 512 Mb with an active/stable Internet connection at the time of usage.

### External System Requirements:

#### User Interfaces:

1. Front End Software-XML and Java
2. Back End Software-Firebase DB

#### Hardware Interfaces

1. Android Phone or a device running android 6.0 or above
2. A phone which supports data access and has a working internet connection with a free space of at least 12 Mb

### 3.2.4.2 Software Requirements

Following are the software used in Agroflo:

Operating System	Android 6.0 as it is user friendly and has the best support
Database	To save the farmer details, customer details and the orders placed by every individual we have used Firebase
Java	To make the application we have used Java as it provides a more interactive support and has wide availability of COTS software for making the application more interesting

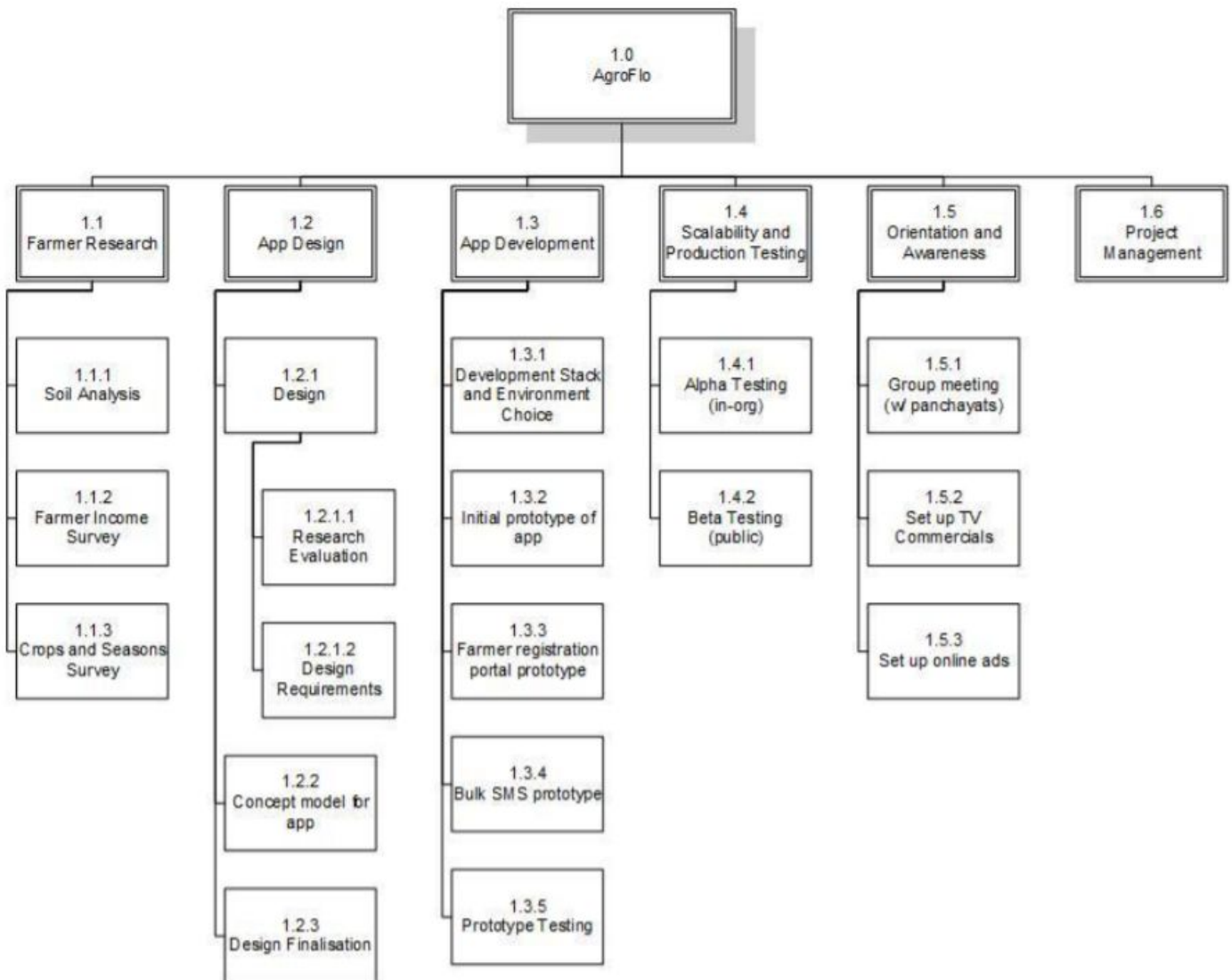
#### Communication Interfaces

The project supports all types of android devices. We are using simple electronic forms for getting the data of the users and store it on the database.

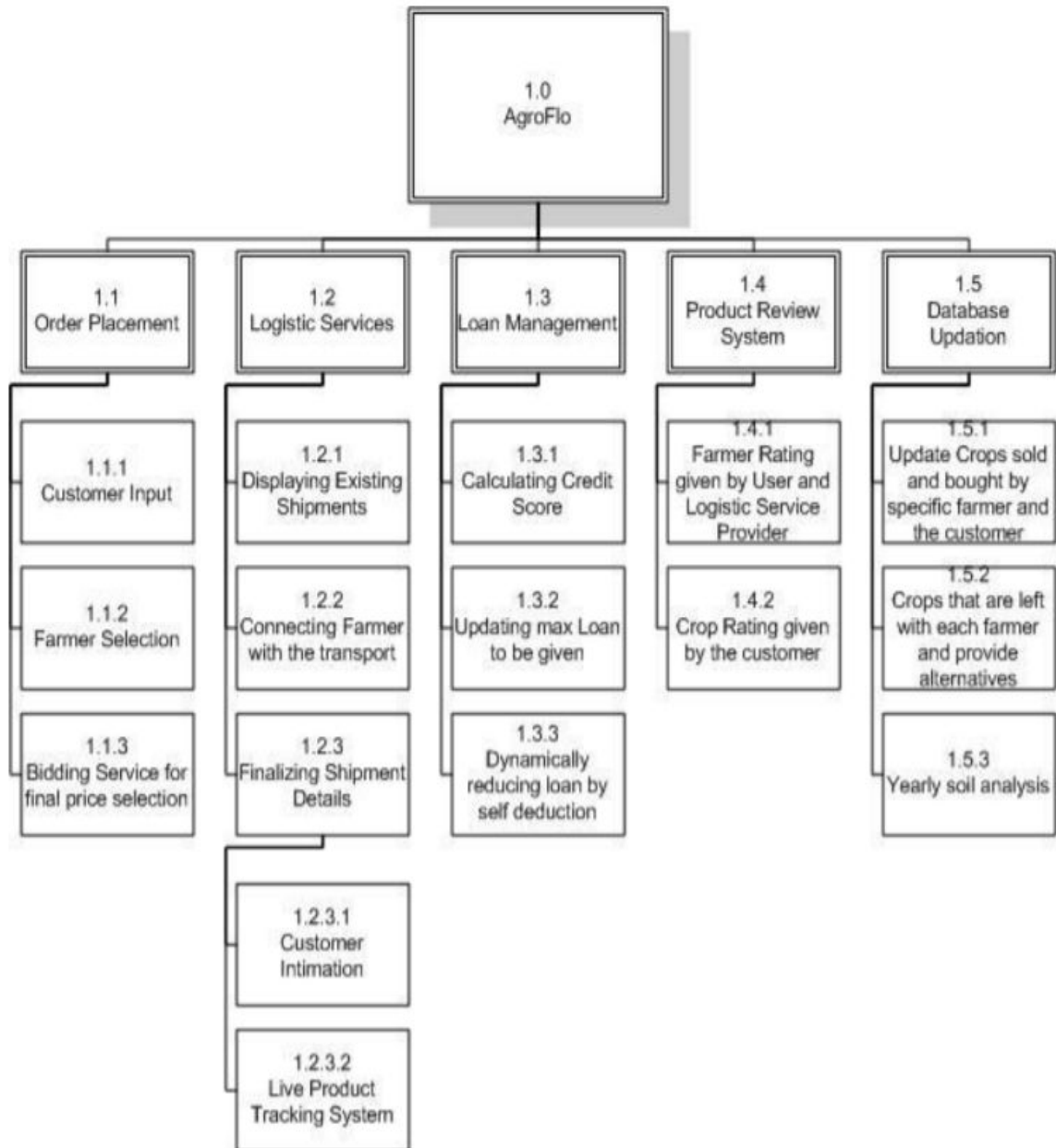


### 3.2.5 Work Breakdown Structure

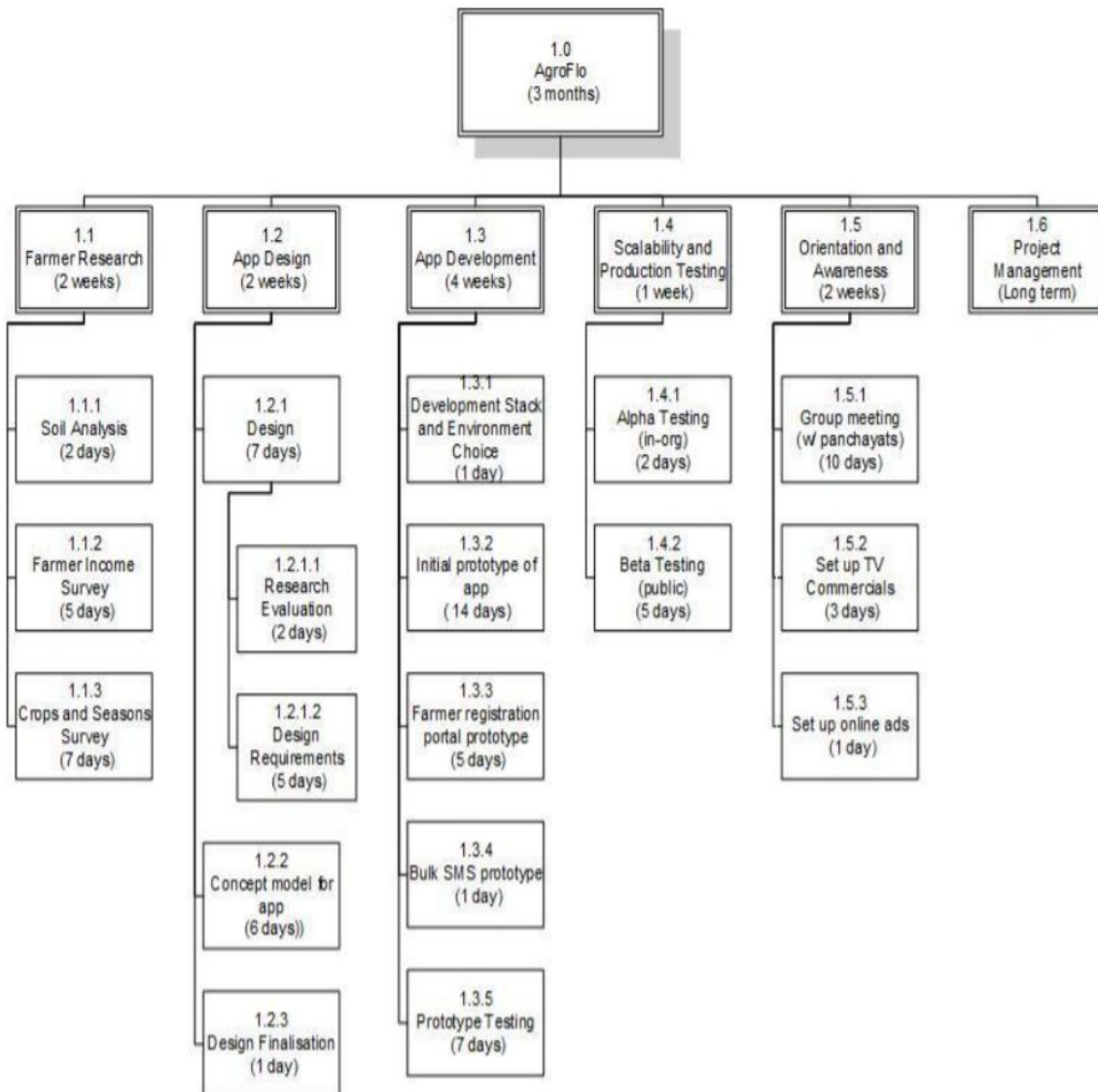
#### Verb Oriented WBS:



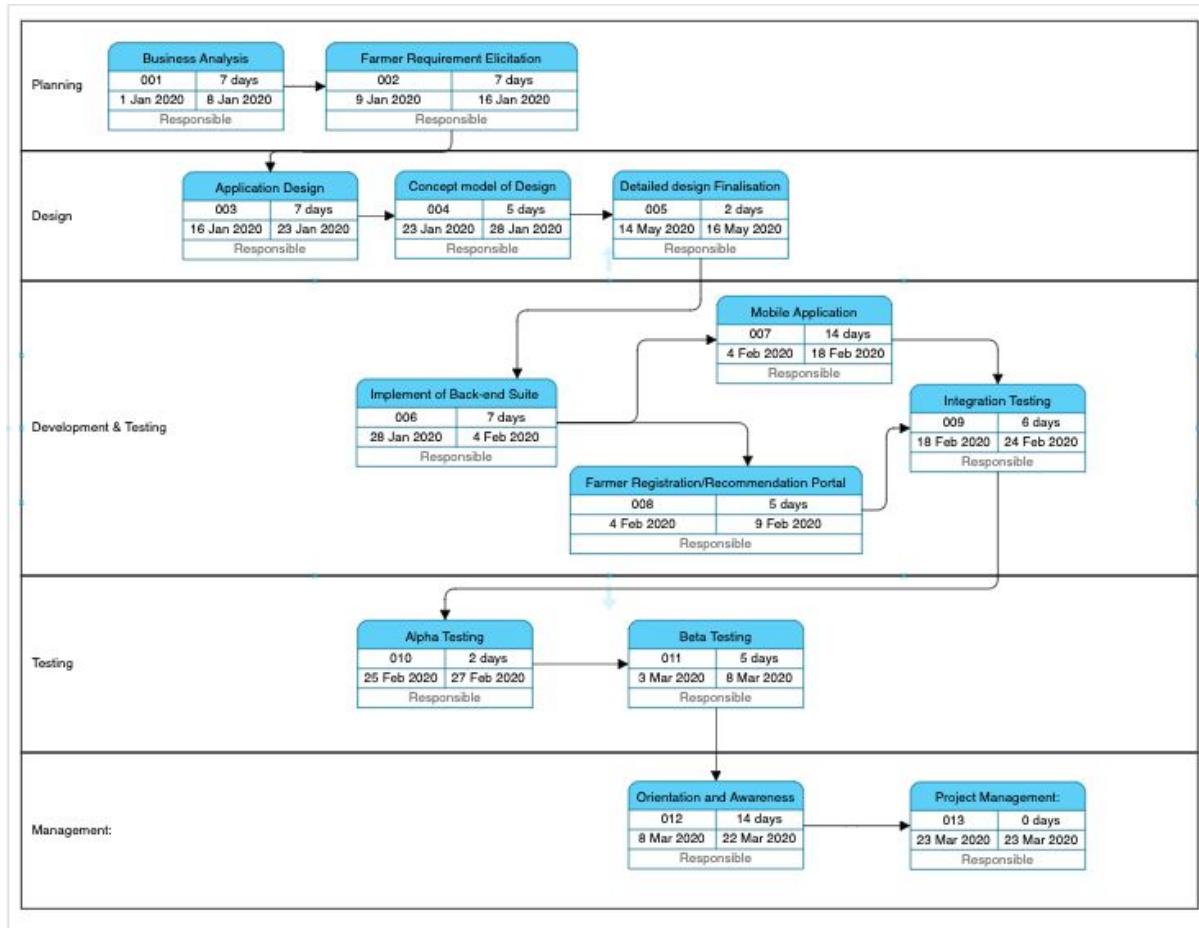
Noun Oriented WBS:



## Time phased WBS:



### 3.2.6 Pert chart (with task table):

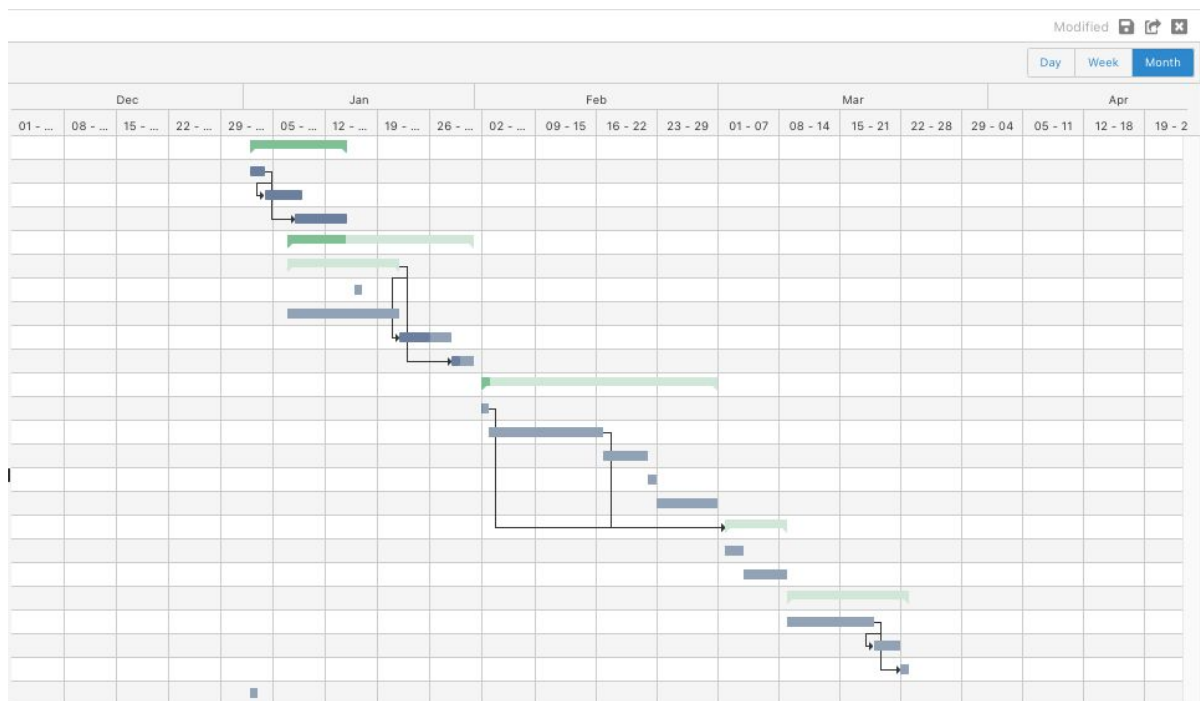


Task	Prerequisite Tasks	Time to perform
1	-	7 days
2	1	7 days
3	1,2	7 days
4	1,2,3	5 days
5	1,2,3,4	2 days
6	1,2,3,4,5	7 days
7	1,2,3,4,5,6	14 days
8	1,2,3,4,5,6	5 days
9	1,2,3,4,5,6,7,8	6 days
10	1,2,3,4,5,6,7,8,9	2 days
11	1,2,3,4,5,6,7,8,9,10	5 days
12	1,2,3,4,5,6,7,8,9,10,11	14 days
13	1,2,3,4,5,6,7,8,9,10,11,12	-

### 3.2.7 Gantt Chart

Software Gantt Chart

<div> <div>+ Add Task</div> <div>Remove Task</div> </div>			
ID	Title	Start Time	End Time
1	Business Analysis and Farmer Research	01/01/2020	01/15/2020
6	Soil Analysis	01/01/2020	01/03/2020
7	Farmer income survey	01/03/2020	01/08/2020
8	Crops and Seasons survey	01/08/2020	01/15/2020
2	Application Design	01/07/2020	02/01/2020
10	Design	01/07/2020	01/22/2020
19	Research Evaluation	01/15/2020	01/17/2020
20	Design Requirements	01/07/2020	01/22/2020
11	Concept Model for App	01/22/2020	01/29/2020
13	Design Finalisation	01/29/2020	02/01/2020
3	Application Development	02/01/2020	02/29/2020
14	Development Stack and Environment choice	02/01/2020	02/02/2020
15	Initial prototype of Application	02/02/2020	02/16/2020
21	Farmer registration/recommendation Portal Prototype	02/16/2020	02/21/2020
22	Bulk SMS Prototype	02/21/2020	02/22/2020
23	Prototype Testing	02/22/2020	02/29/2020
4	Scalability and Production Testing	03/01/2020	03/08/2020
24	Alpha Testing	03/01/2020	03/03/2020
25	Beta Testing	03/03/2020	03/08/2020
5	Orientation and Awareness	03/08/2020	03/22/2020
16	Group meeting with Panchayats	03/08/2020	03/18/2020
17	Set up TV commercials	03/18/2020	03/21/2020
18	Set up online ads	03/21/2020	03/22/2020
26	New task	01/01/2020	01/02/2020



## **4. Design of the Proposed System**

### **4.1 Introduction**

### **4.2 High Level Design (Framework, Architecture or Module for the Proposed System(with explanation))**

#### **4.2.1 Architecture design**

The application is primarily made up of 3 parts:

The Android app is made in java, and is distributed as a signed APK file that can be installed on any android device having version higher than or Android 5. It stores data such as Farmer and Customer login details and order information on a Google Firebase noSQL database. This makes it necessary for an internet connection for the app to function.

The crop recommendation algorithm uses a K-Nearest Neighbours machine learning algorithm to recommend crops to the farmers. The given algorithm has been altered with different parameters and checked for speed along with the accuracy. It is deployed on a website as of now.

The plant disease prediction algorithm uses a Deep Learning 2D Convolutional Neural Network with a Dense Network and a Softmax function for multi-label classification.

#### **4.2.2 Architecture description**

We've used a K-Nearest Neighbours machine learning algorithm to recommend crops to the farmers. We've altered the given algorithm with different parameters and checked the speed along with the accuracy.

We switched from the default distance metric which is the Manhattan distance metric to the Euclidean distance metric, this was a much better choice than before and helped us get a better run time as Euclidean distance computation for less number of features is much faster. We also changed the number of neighbours we were looking at to get better results. Although the accuracy was the same for the 2 approaches but the time complexity was highly reduced. Final Parameters = Euclidean Distance with 4 neighbours.

We linked the Machine Learning Model with a front end using Flask backend and the pickle file of the model.

The Dataset used for the Disease recognition problem was taken from PlantVille's website, which consists of a set of color images, each under a class(disease).

Results Achieved:

Train Accuracy - 94%

Test/Validation Accuracy - 78%.

### 4.2.3 UI design

Screenshot of Web Page which gives recommendations:

Input:

State - Karnataka

Cost per Hectare - 16000 Rs

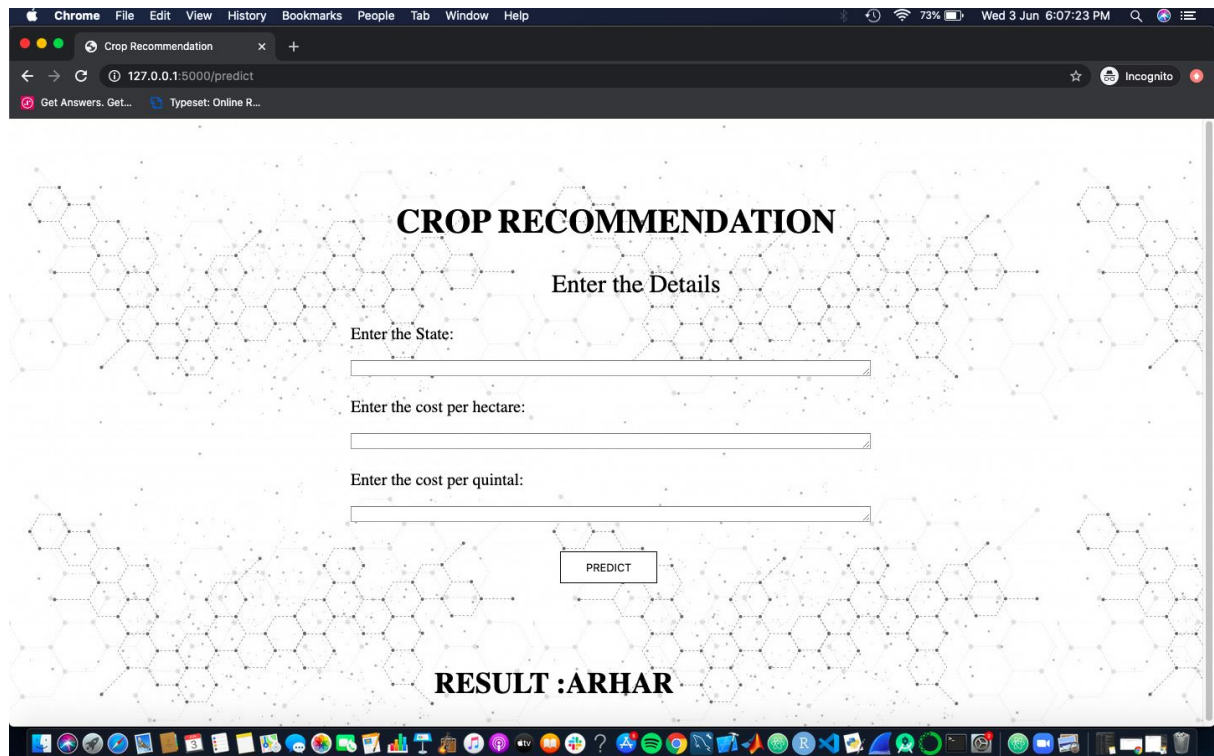
Cost per Quintal - 2000 Rs

The screenshot shows a web browser window with the title 'Crop Recommendation'. The page has a background pattern of hexagons and dots. The main heading is 'CROP RECOMMENDATION' in bold. Below it is the sub-heading 'Enter the Details'. There are three input fields: 'Enter the State:' with 'Karnataka' entered, 'Enter the cost per hectare:' with '16000' entered, and 'Enter the cost per quintal:' with '2000' entered. A black button with the text 'PREDICT' is located below the input fields. Below the button is the text 'RESULT :'. The browser's address bar shows '127.0.0.1:5000'. The browser's status bar shows 'Wed 3 Jun 6:07:16 PM' and '73%' battery. The browser's taskbar shows various application icons.

Result:

The farmer is recommended to grow Arhar.



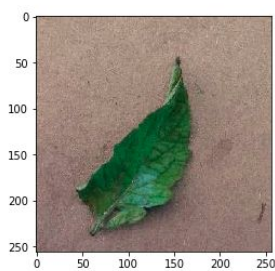


Screenshot of Prediction:

Input image: A tomato plant suffering from mosaic virus(leaf):

```
[25]: img = (x_test[0] * 255).astype(np.uint8)
      plt.imshow(img)
```

```
Out[25]: <matplotlib.image.AxesImage at 0x7f21a01731d0>
```



Output:



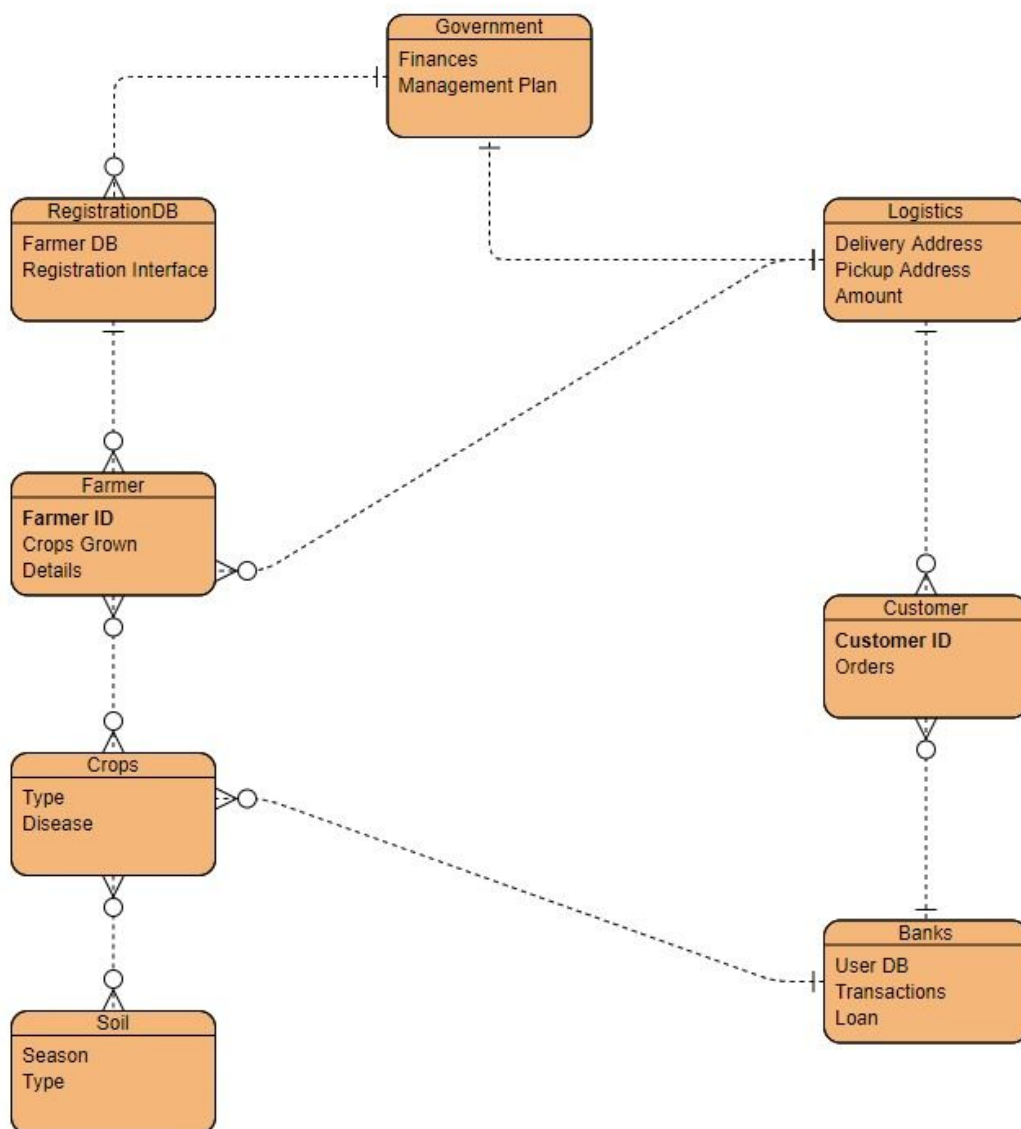
```
[26]: n = 0
sample = x_test[n]
sample = np.array([sample])
res = label_binarizer.classes_[(np.where(model.predict([sample])[0] == np.argmax(model.predict([sample])[0])))]
print(res[0])
```

Tomato\_\_Tomato\_mosaic\_virus

+ Code + Markdown

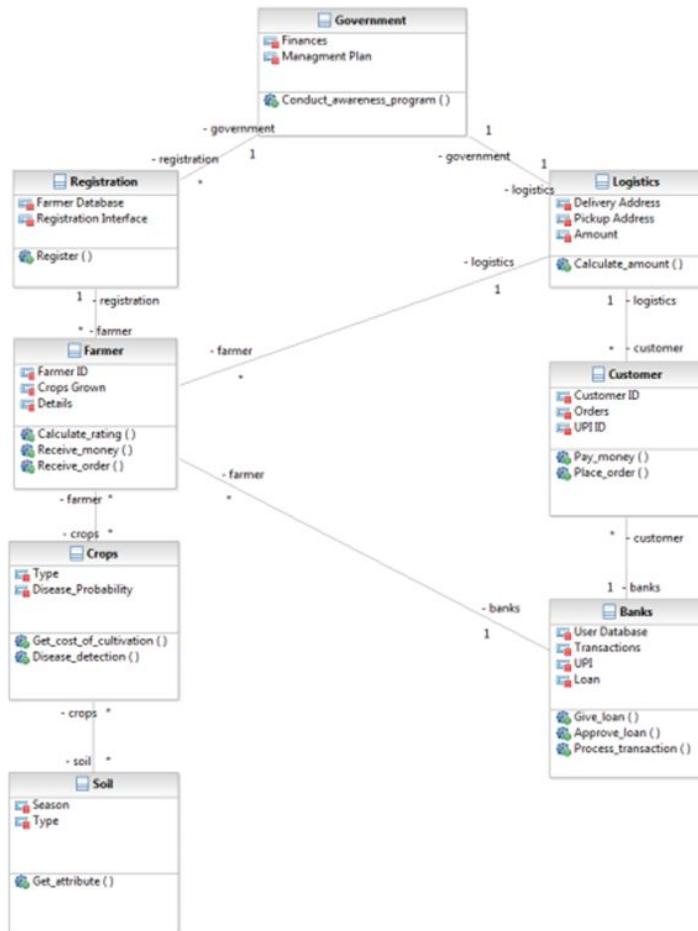
## 4.3 Detailed Design

### 4.3.1 ER Diagram

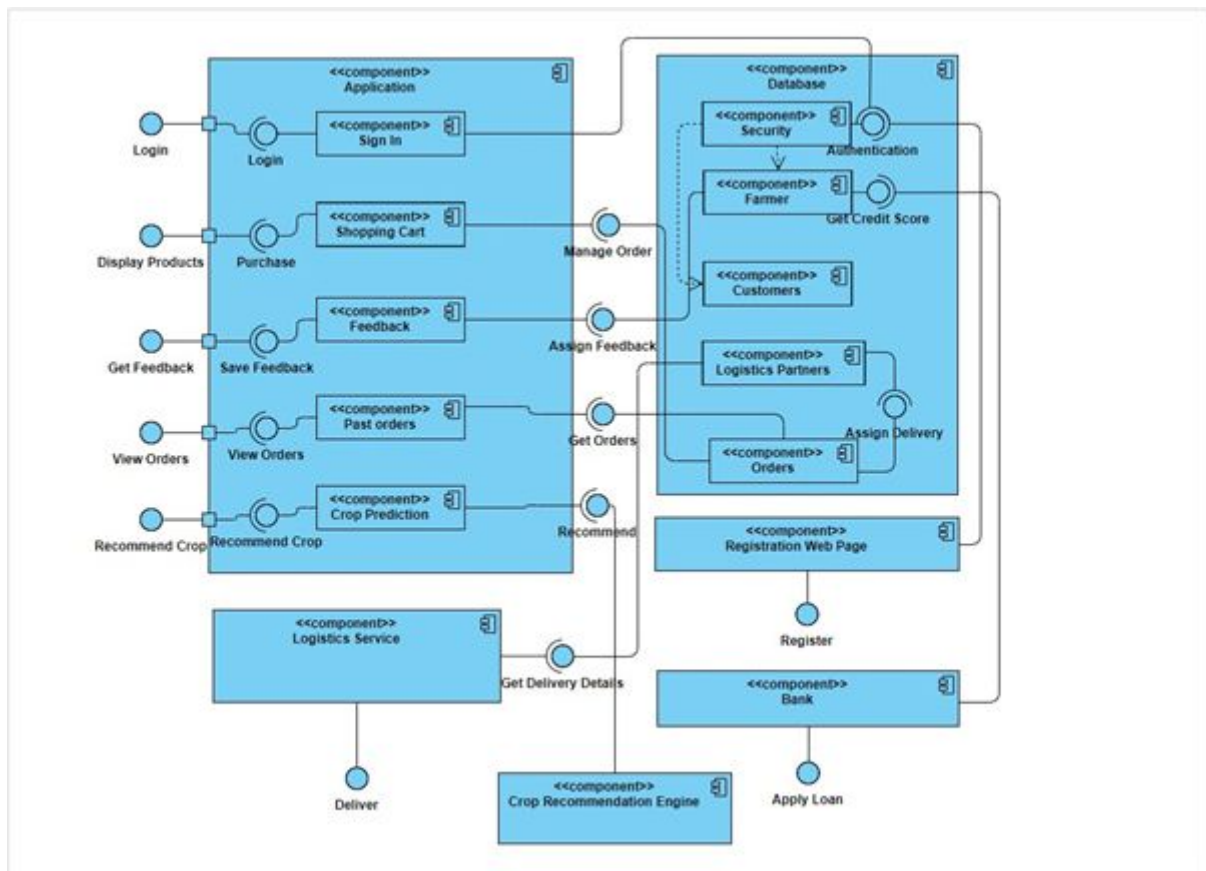


## 4.3.2 UML diagrams

### Class Diagram:

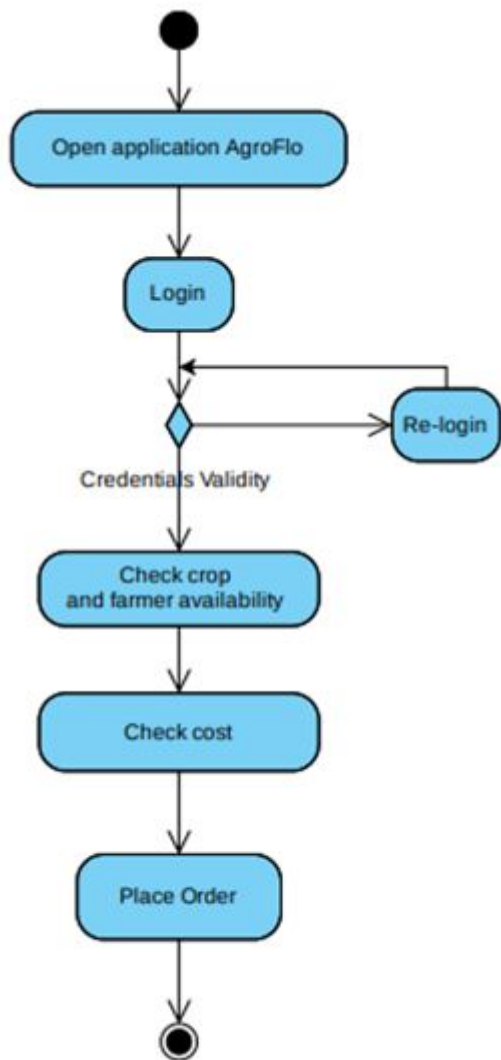


## Component Diagram:

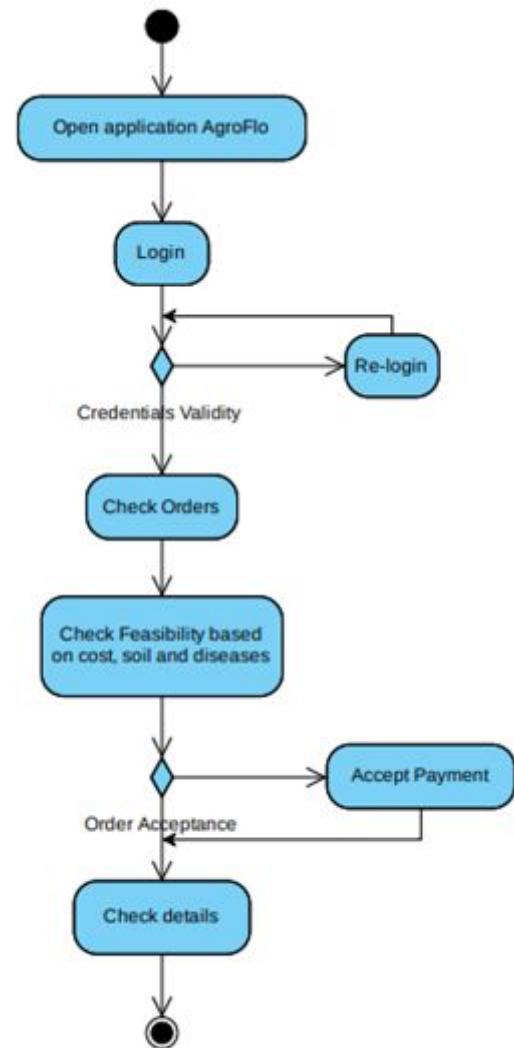


## Activity Diagram:

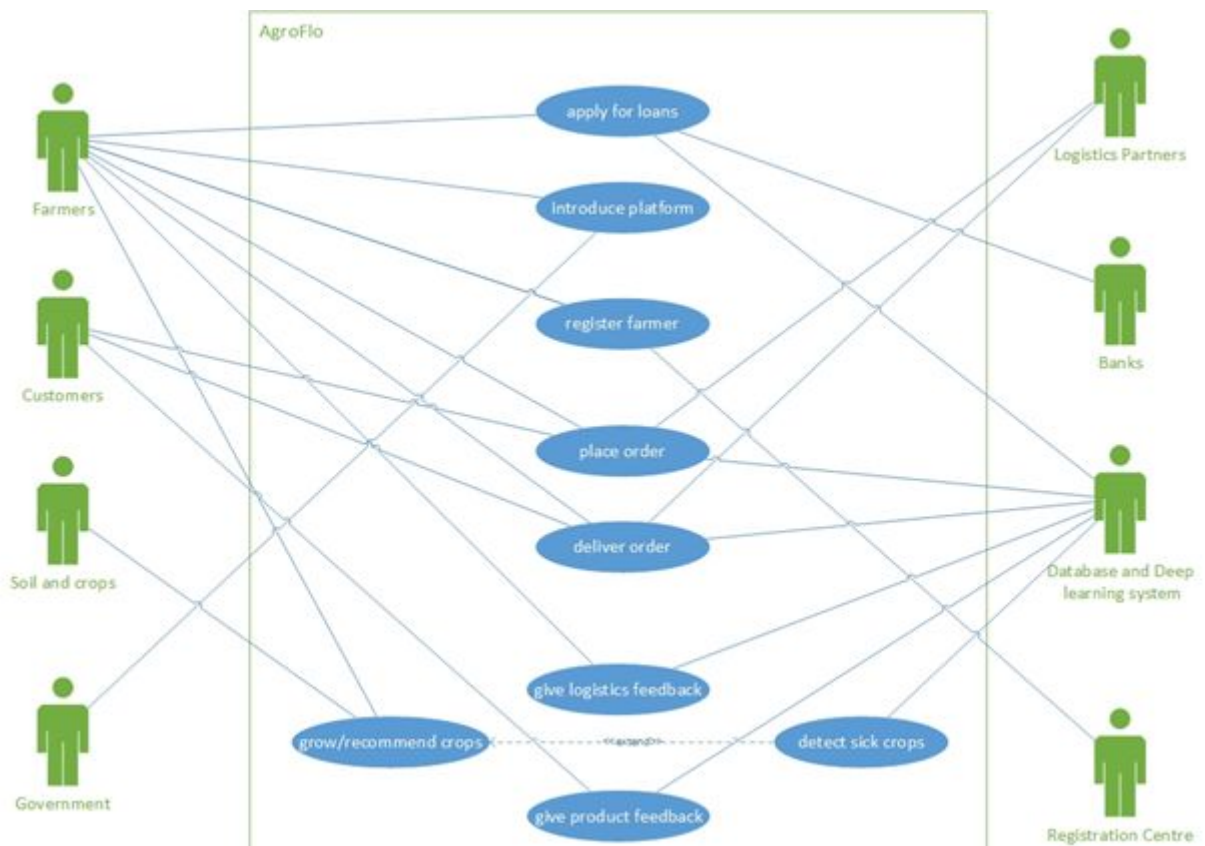
### Customer:



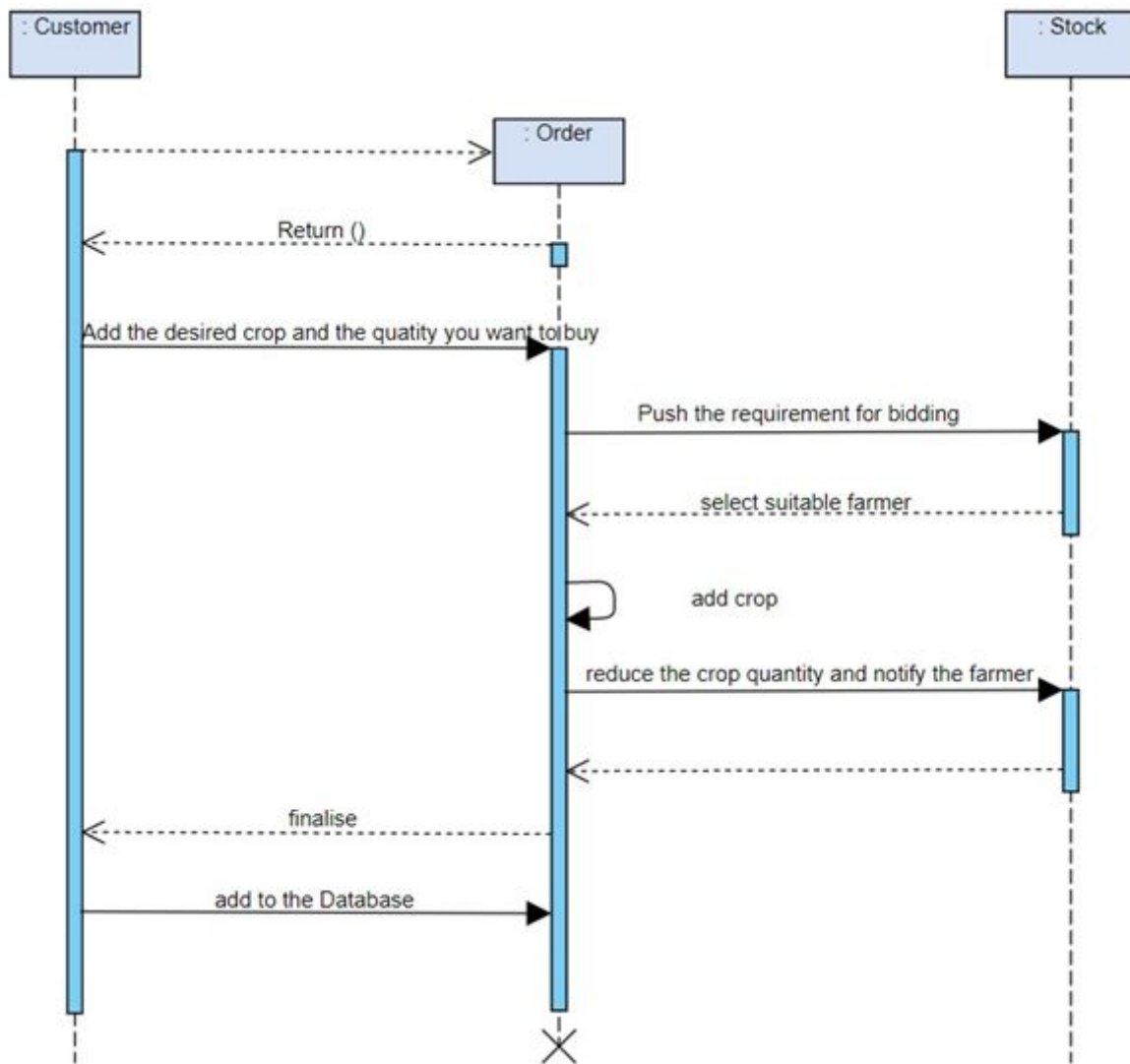
### Farmer:



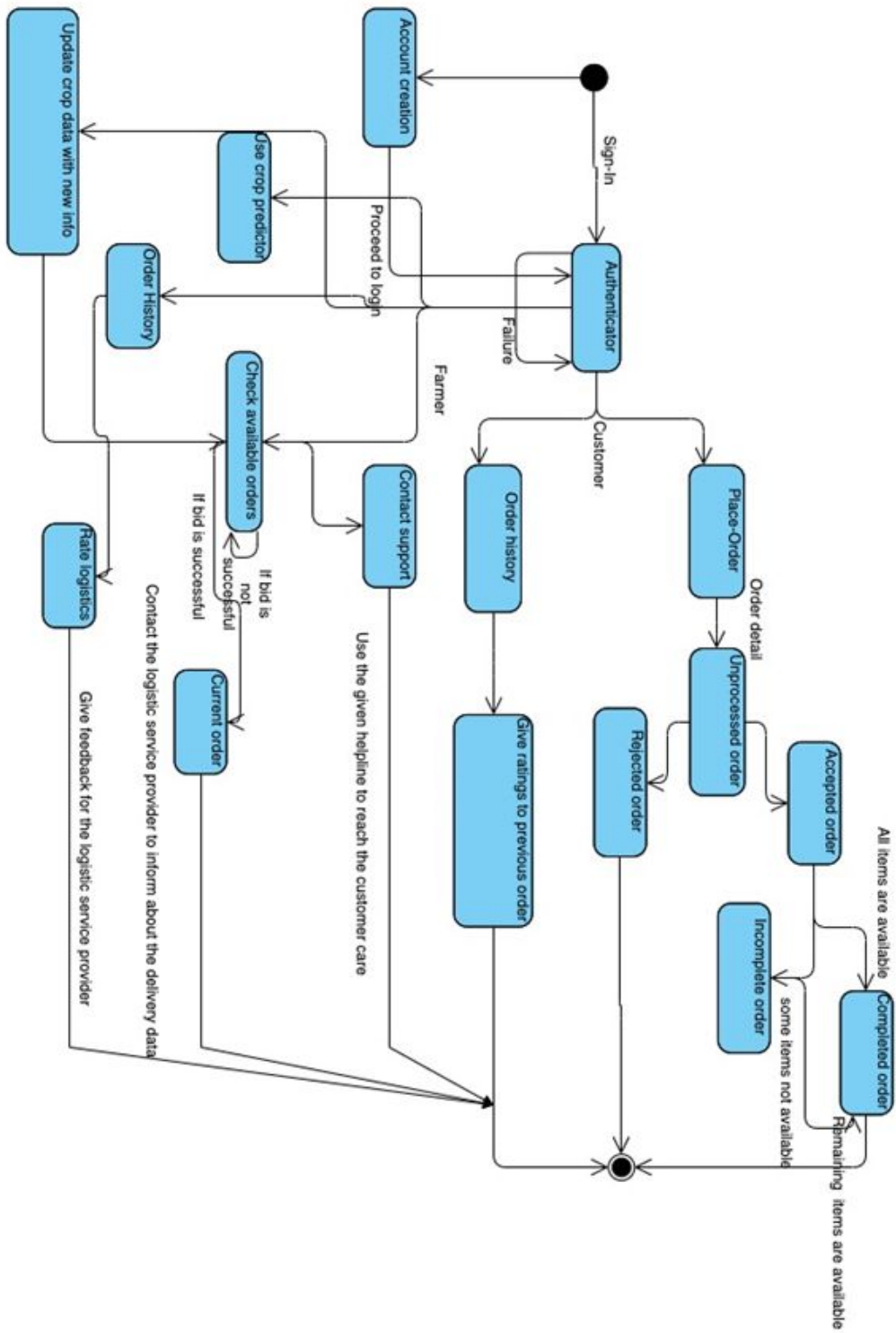
## Use Case Diagram:



### Sequence Diagram:



## State Machine Diagram:



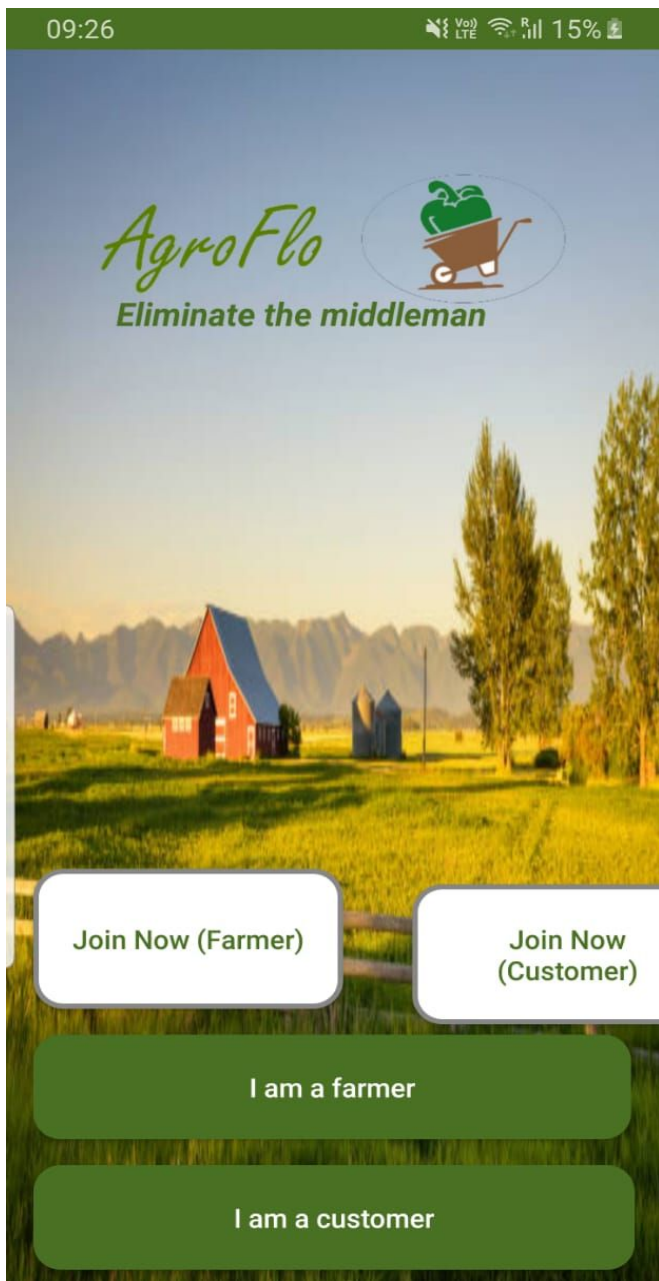
## 5. Implementation and Testing

The Implementation for our project has been carried out in 3 parts

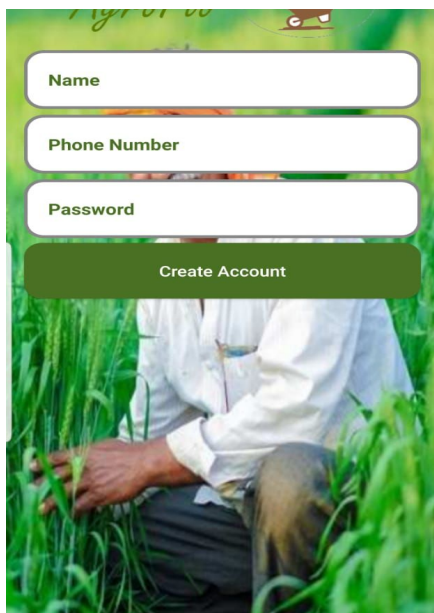
### **PART 1:** The Android Application

This is the first screen of our app which presents two options for the farmer

Join Now or Sign Up







Sign-Up form for farmer and customer. The form is overlaid on a background image of a farmer in a field. The form includes three input fields: Name, Phone Number, and Password, followed by a green 'Create Account' button.

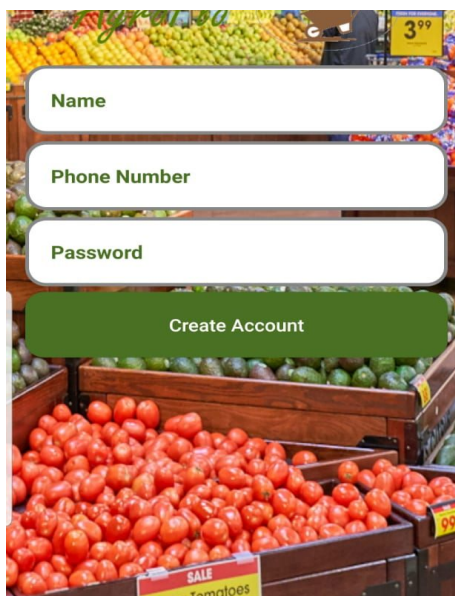
**Name**

**Phone Number**

**Password**

**Create Account**

Sign-Up page for farmer and Customer



Sign-Up form for farmer and customer. The form is overlaid on a background image of a grocery store. The form includes three input fields: Name, Phone Number, and Password, followed by a green 'Create Account' button.

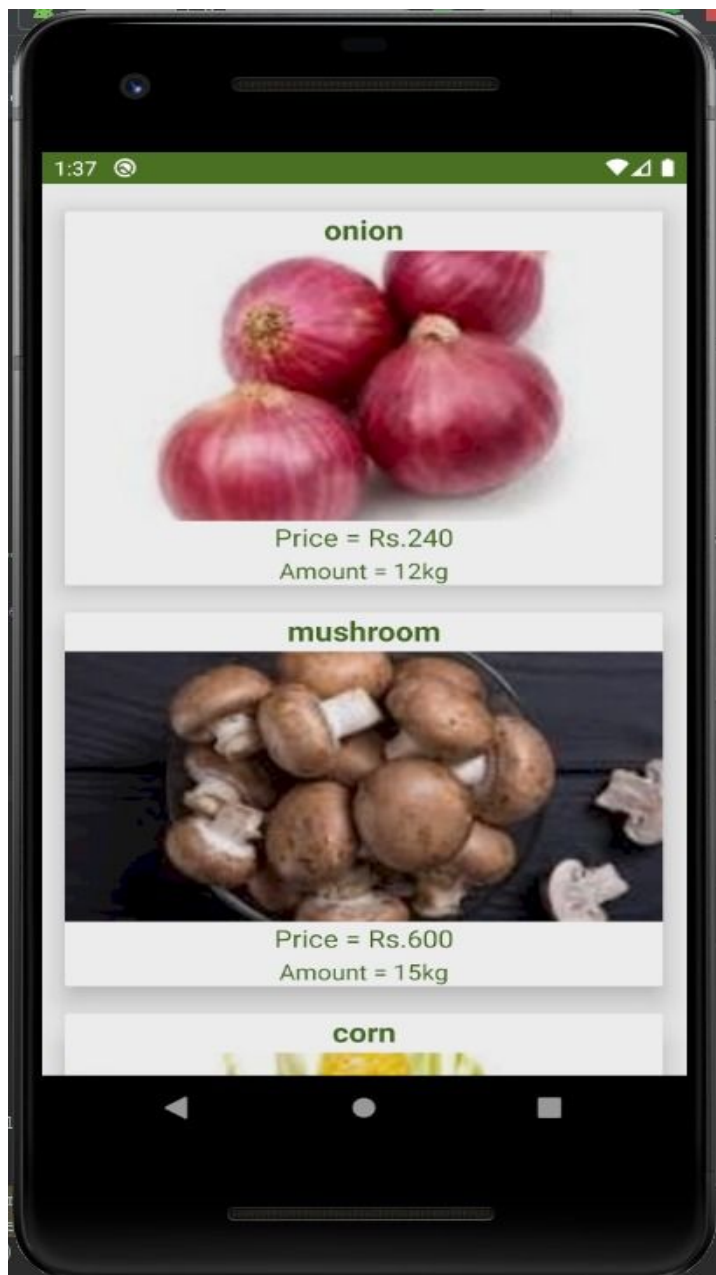
**Name**

**Phone Number**

**Password**

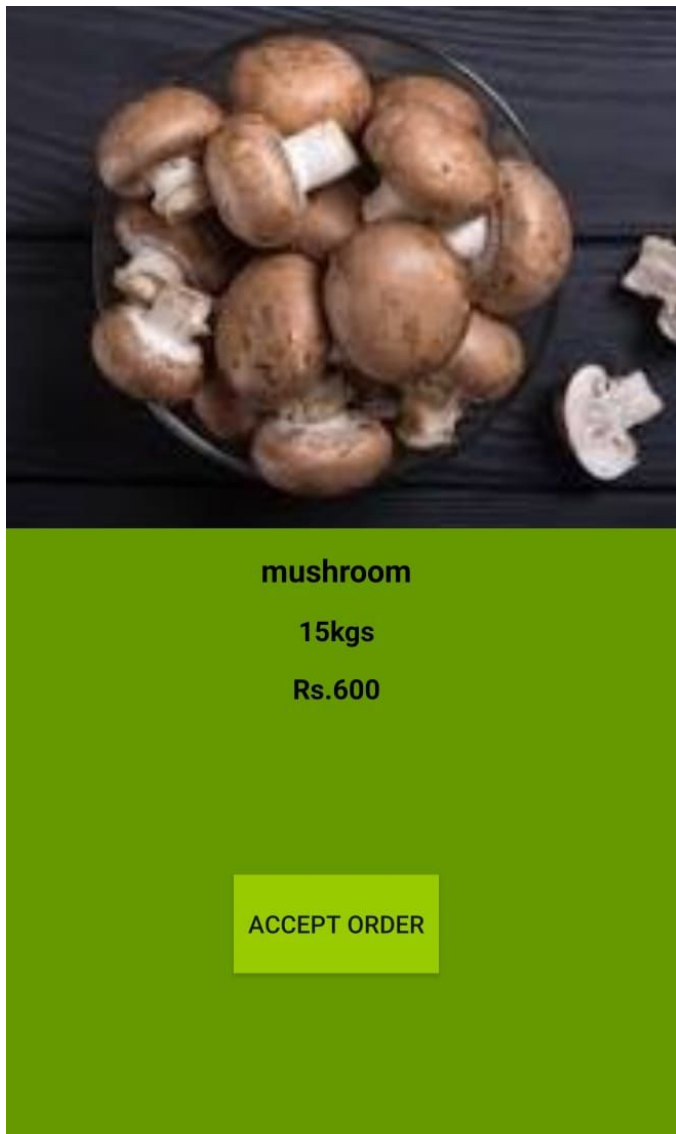
**Create Account**

On Signing In the farmers and customers see separate screens



Screen visible to a customer

Screen Visible to a farmer



The farmer has to choice to accept an order or not worry about it the list has several orders

## **PART 2: ML Model**

These images show how we have implemented the ML model

- 1) Loading the Data:

```
[ ] 1 import pandas as pd
    2 from sklearn.neighbors import KNeighborsClassifier
    3 from sklearn import preprocessing
```

```
[ ] 1 data = pd.read_csv('/content/crop_data.csv')
```

	Crop	State	Cost of Cultivation (₹/Hectare)	A2+FL	Cost_per_hectare	Cost_per_quintal	Yield (Quintal/ Hectare)
0	ARHAR	Uttar Pradesh		9794.05	23076.74	1941.55	9.83
1	ARHAR	Karnataka		10593.15	16528.68	2172.46	7.47
2	ARHAR	Gujarat		13468.82	19551.90	1898.30	9.59
3	ARHAR	Andhra Pradesh		17051.66	24171.65	3670.54	6.42
4	ARHAR	Maharashtra		17130.55	25270.26	2775.80	8.72

Since the state feature has string type data and KNN requires numerical input, we encode those values to categorical data.

### Encoded States

$$\begin{bmatrix} 11 & 4 & 2 & 0 & 6 & 6 & 8 & 0 & 2 & 3 & 9 & 5 & 11 & 6 & 0 & 4 & 0 & 10 & 2 & 6 & 1 & 4 & 9 & 11 \\ 0 & 7 & 9 & 4 & 0 & 6 & 11 & 7 & 12 & 8 & 0 & 5 & 9 & 11 & 2 & 3 & 11 & 4 & 0 & 6 & 10 & 5 & 8 & 11 \\ 9 \end{bmatrix}$$

- Manhattan distance with 3 neighbours takes the least amount of time. Note: The hyper/parameters used will give similar results.

```

CPU times: user 2.1 ms, sys: 15 μs, total: 2.11 ms
Wall time: 4.6 ms

```

### 3) Fitting the data to the Models:

```
[ ] 1 %%time
2 #Distance measure - euclidian distance
3 #Number of neighbours - 4
4 niegh2 = KNeighborsClassifier(n_neighbors = 4, p = 2)
5 niegh2.fit(x,y)
```

CPU times: user 1.4 ms, sys: 0 ns, total: 1.4 ms  
Wall time: 1.31 ms

```
[ ] 1 %%time
2 #Distance measure - manhattan distance
3 #Number of neighbours - 3
4 niegh3 = KNeighborsClassifier(n_neighbors = 3, p = 1)
5 niegh3.fit(x,y)
```

CPU times: user 1.4 ms, sys: 0 ns, total: 1.4 ms  
Wall time: 3.28 ms

```
[ ] 1 %%time
2 #Distance measure - euclidian distance
3 #Number of neighbours - 3
4 niegh4 = KNeighborsClassifier(n_neighbors = 3, p = 2)
5 niegh4.fit(x,y)
```

CPU times: user 1.51 ms, sys: 99 µs, total: 1.61 ms  
Wall time: 1.36 ms

#### 4) Predictions:

```
<> [ ] 1 %%time
2 predicted_crop = niegh1.predict([[encode, cost_hectare, cost_quintal]])
3 print(predicted_crop)
```

['ARHAR']  
CPU times: user 2.64 ms, sys: 103 µs, total: 2.74 ms  
Wall time: 3.59 ms

```
[ ] 1 %%time
2 predicted_crop = niegh2.predict([[encode, cost_hectare, cost_quintal]])
3 print(predicted_crop)
```

['ARHAR']  
CPU times: user 919 µs, sys: 993 µs, total: 1.91 ms  
Wall time: 5.1 ms

```
[ ] 1 %%time
2 predicted_crop = niegh3.predict([[encode, cost_hectare, cost_quintal]])
3 print(predicted_crop)
```

['ARHAR']  
CPU times: user 1.72 ms, sys: 0 ns, total: 1.72 ms  
Wall time: 1.48 ms

```
[ ] 1 %%time
2 predicted_crop = niegh4.predict([[encode, cost_hectare, cost_quintal]])
3 print(predicted_crop)
```

['ARHAR']  
CPU times: user 1.88 ms, sys: 0 ns, total: 1.88 ms  
Wall time: 1.75 ms

Output of the ML model deployed on the Web Application:

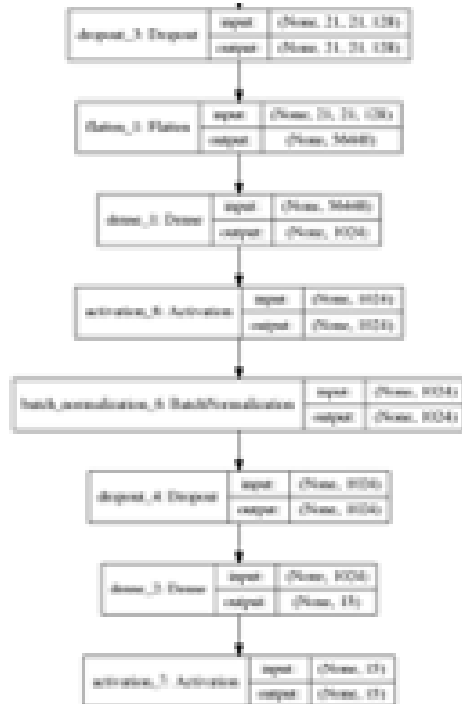
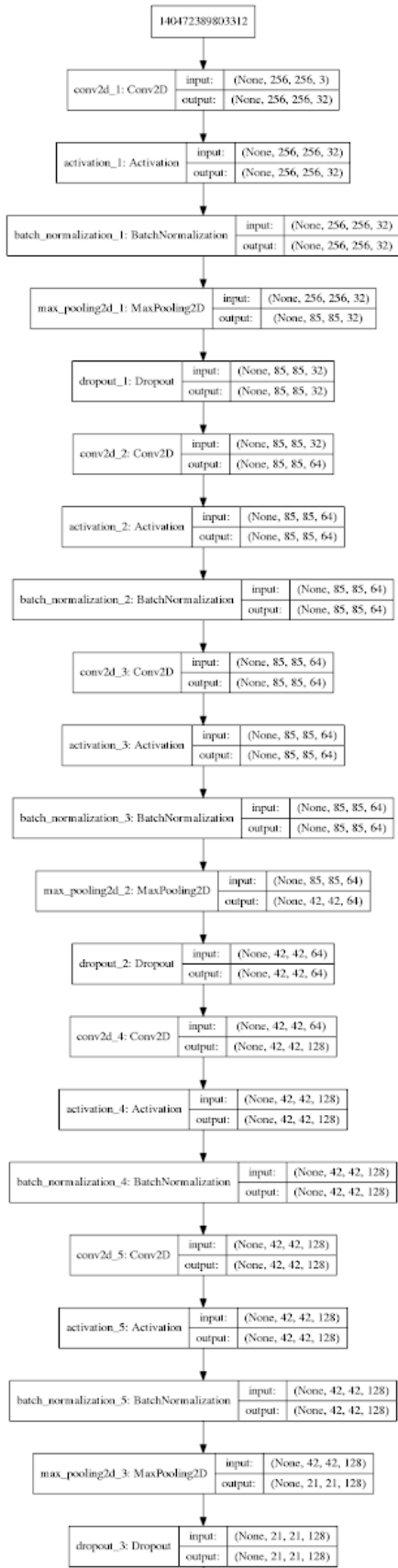
The screenshot shows a web browser window with the title 'Crop Recommendation'. The address bar shows the URL '127.0.0.1:5000/predict'. The page has a background pattern of hexagons and dots. The main heading is 'CROP RECOMMENDATION'. Below it is the sub-heading 'Enter the Details'. There are three input fields: 'Enter the State:', 'Enter the cost per hectare:', and 'Enter the cost per quintal:'. Below these fields is a button labeled 'PREDICT'. At the bottom, the result is displayed as 'RESULT :ARHAR'.

### PART 3:Deep learning Model

The deep learning model used for plant disease detection consists of the following parts:

- 1) A Convolutional Neural Network
- 2) A dense network appended on top of the Convolutional Neural Network.
- 3) Linear Layers with 'ReLU' activation function are added between the network for learning.
- 4) The final layer of the network consists of a Softmax function for multiclass classification. It gives us a probability distribution of the possible classes. Class with the highest probability is the class of the image.

The architecture can be better understood by the following graph:



Training the Deep learning Model:

The following are the mean accuracies after training for 25 epochs:

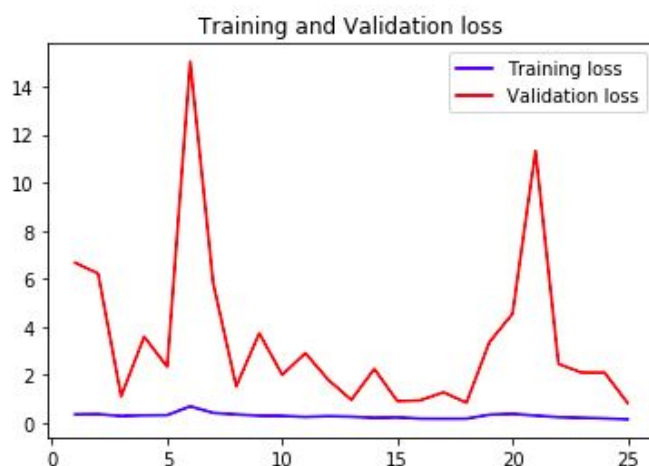
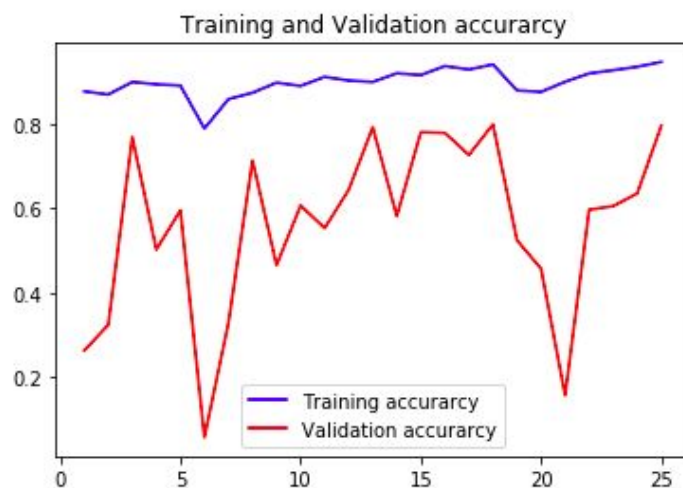
Train Accuracy : 94%

```
Epoch 25/25  
73/73 [=====] - 32s 443ms/step - loss: 0.1592 - acc: 0.9489 - val_loss: 0.8395 - val_acc: 0.7970
```

Test/Validation Accuracy : 79%

```
[INFO] Calculating model accuracy  
591/591 [=====] - 1s 2ms/step  
Test Accuracy: 79.695431492252
```

Graphs:



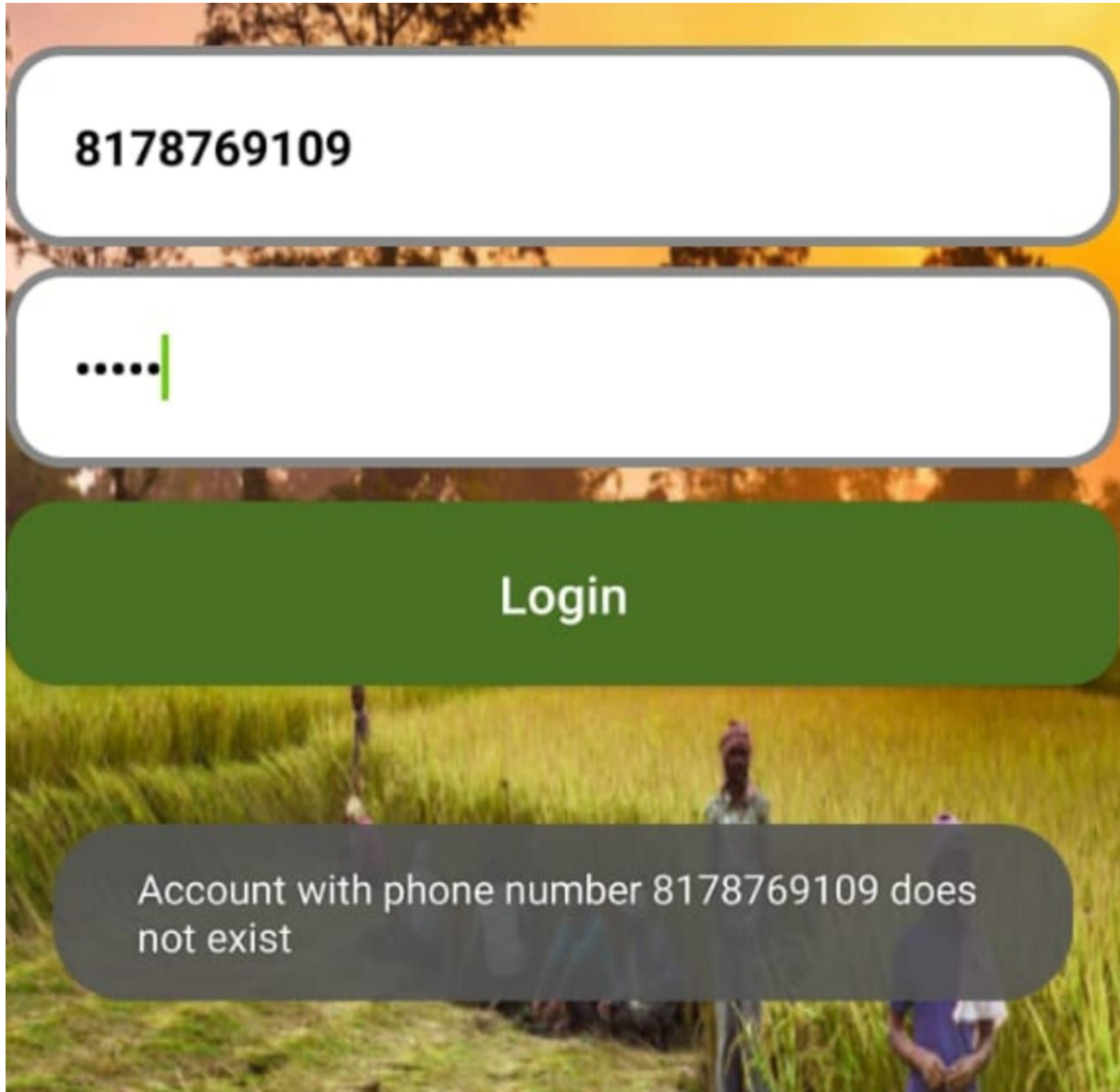
In the above image, the fluctuation of the validation loss while training can be due to an imbalanced dataset, but finally comes back to the global minima after training.



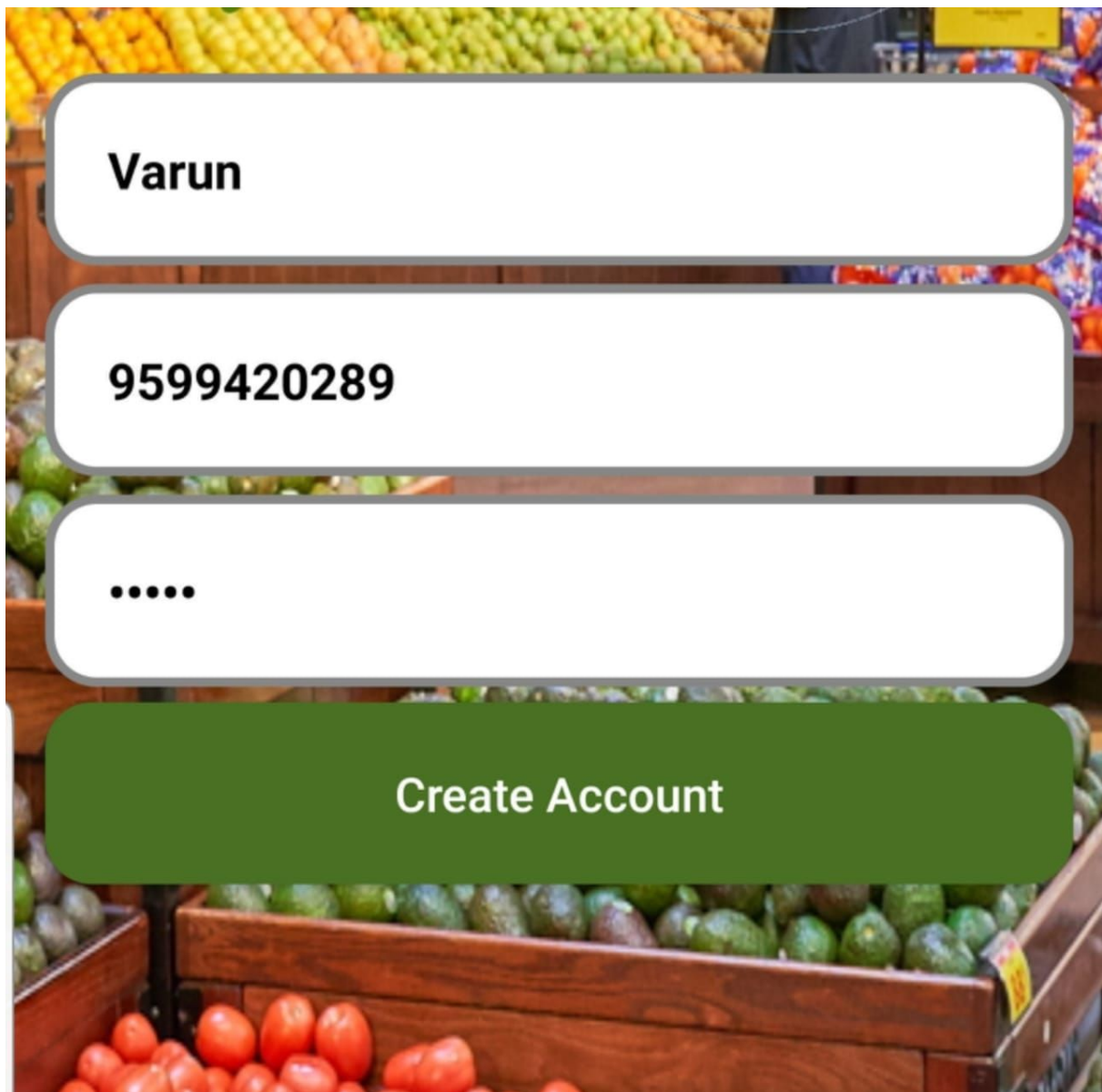
### Testing:

Unit Testing has only been done for the android app as software testing for the Machine learning models is out of scope for this project

Test 1:Checking if app stores and uses login/signup information properly



Test 2: Testing for the integrity of the data submitted to the app



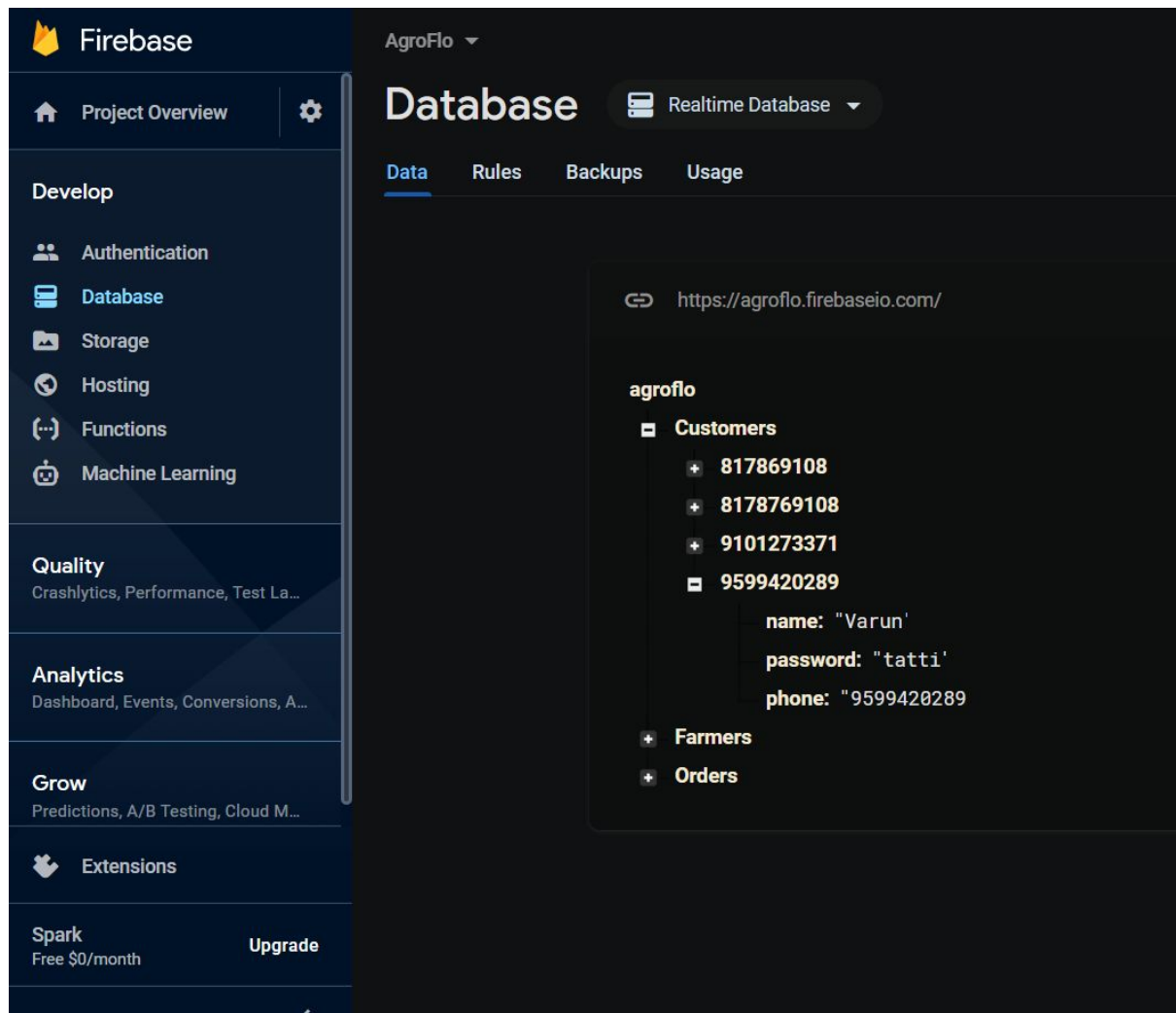
**Varun**

**9599420289**

.....

**Create Account**

Input



output-The data is securely stored

## 6. Conclusion, Limitations, and Future Works

The project helped us to make AgroFlo an app that eliminates Middle-men and helps farmers and customers to interact directly. The App performs well, is scalable, and has multiple use cases with several features that makes the app easy to use. We have integrated ML and DL models to help the farmer because what good is tech until used for a good cause. Our app is a one stop solution for every small farmer and every customer who wants to get the best products without paying the exorbitant prices. AgroFlo ensures that a farmer gets timely and complete payment for all the crops that they sell and they are never in a struggle to sell more crops and get more money.

Some advantages of AgroFlo are as follows:

- The app helps to connect the farmer to the customer directly and eliminates the middleman. This enables farmers to earn more and avoid abuse by said middlemen.
- Advantageous for bulk purchasers with high demands like cloud kitchens and restaurants.
- Customers can place orders in an instant and with no hassle. This enables them to get fresh items and support local farmers.
- Payments to the farmer are always provided as soon as orders are placed and all transactions are noted.
- The process of selling crops now becomes easier for the farmer as potential customers are only one click away.
- Provides user/region specific crop recommendations to farmers.
- Gives farmers the ability to check if their produce is disease infested or not, through accurate deep learning models.

No App/Product comes without flaws and our app has some flaws, although all these flaws can be addressed, but addressing them right now was outside the scope of the project work.

Drawbacks and planned future improvements include:

- Lack of security of personal information as all data is stored on Google Firebase, without hashing/salting of the passwords and other sensitive info. This can be resolved by using the Java MessageDigest library to hash sensitive information before sending it to the database.
- The second drawback is scalability as this is ideal only for small marketplaces and the lack of technology and internet services in rural India limits the use of application only to cities or places with good network services. A solution for this is to setup offline centers for farmers and use services like SMS instead of Internet based applications in such areas
- Logistic services are required to deliver the goods from a farmer to the customers. This can prove expensive if there are locations where only individual users reside. Thus, the project will be initially deployed in areas with high concentration of farmers or areas with many customers to gain the trust of logistics companies in the business model.

## 7. References

[1]GeeksforGeeks:

<https://www.geeksforgeeks.org/>

[2]Keras API

<https://keras.io/>

[3]Tensorflow Documentation

<https://www.tensorflow.org/>

[4]Firebase for Android

<https://firebase.google.com/>

[5]Dataset for Deep learning Model

<https://plantvillage.psu.edu/>

[6]Other Applications

<https://medium.com/@sumana.T/7-best-apps-for-e-market-platforms-for-farmers-324dd668270b>

[7]Dataset for Recommendation system

<https://www.kaggle.com/srinivas1/agriculture-crops-production-in-india>