COOPERATED AGRI SHOP ON ANDROID APPLICATION WITH SECURE BLOCKCHAIN BASED TRANSACTIONS

A PROJECT REPORT

Submitted by

VASANTH PRASANTH N [211418104304]

HARI KISHORE G [211418104078]

SRIHARIHARAN SHA K K [211418104261]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MAY 2022

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"COOPERATED AGRI SHOP ON ANDROID APPLICATION WITH SECURE BLOCKCHAIN BASED TRANSACTIONS"** is the bonafide work of "**VASANTH PRASANTH N (211418104304), HARI KISHORE G (211418104078), SRIHARIHARAN SHA K K (211418104261)"** who carried out the project work under my supervision.

SIGNATURE

Dr.S.MURUGAVALLI M.E.,Ph.D.,
HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NAZARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE

Dr.N.PUGHAZENDI M.E.,Ph.D.,
PROFESSOR

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NAZARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on.....

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION BY THE STUDENT

We "**VASANTH PRASANTH N [211418104304], HARI KISHORE G [211418104078], SRIHARIHARAN SHA K K [211418104261]",** hereby declare that this project report titled **"COOPERATED AGRI SHOP ON ANDROID APPLICATION WITH SECURE BLOCKCHAIN BASED TRANSACTIONS"**, under the guidance of **Dr.N.PUGHAZENDI M.E.,Ph.D.,** is the original work done by me and we have not plagiarized or submitted to any other degree in any university by me.

VASANTH PRASANTH N

HARI KISHORE G

SRIHARIHARAN SHA K K

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI M.A.,Ph.D.,** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR M.E.,Ph.D.,** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.MANI M.E.,Ph.D.,** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr.S.MURUGAVALLI M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my Project Guide & Project Coordinator **Dr.N.PUGHAZENDI M.E.,Ph.D.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

And finally we like to use this opportunity to thank my parents for their support for the successful completion of the project.

VASANTH PRASANTH N

HARI KISHORE G

SRIHARIHARAN SHA K K

ABSTRACT

This is an app that facilitates Farmers and Customers for smooth, effective and newly composed supply chain mechanism which is cooperative in nature and the increase in demand for locally produced food is widely acknowledged by industries, government. This app not only focuses on direct supply of product to customers but also to provide necessary assistance to the farmers. This has seen the rise in interest for small/medium farm product distribution projects in INDIA. Farmers are selling products to middleman for less profit, but lack of knowledge to the farmer we thought of doing an application that can help farmers in multiple ways one of that is sell their own products directly to customer with no brokers and also to provide a platform for the farmer where the produce from the farmer can be sold at best rates, sharing the transport to take the produced products to the markets and to help farmers on various things which based on the cooperative benefits

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ABBREVIATIONS	EXPANSION
GDP	GROSS DOMESTIC PRODUCT
NAM	NATIONAL AGRICULTURE MARKET
SQL	STRUCTURED QUERY LANGUAGE
JDK	JAVA DEVELOPMENT KIT
E-R	ENTITY RELATIONSHIP
XML	EXTENSIBLE MARKUP LANGUAGE
DB	DATABASE
JDBC	JAVA DATABASE CONNECTIVITY
SOAP	SIMPLE OBECT ACCESS PROTOCOL
WSDL	WEB SERVICE DESCRIPTION LANGUAGE

CHAPTER – 1

INTRODUCTION

1. INTRODUCTION

India’s one of the Top priority sector is agriculture. but today the people involved in agriculture experience problem in their day-to-day life due to extreme poverty. In India about 15% of GDP comes from agriculture but the employers in this work are 50% of our working population. In India income generation by farm is been a key issue. Due to lack of awareness of modern technology or advanced techniques and unstable supply chain management leads to Farm poverty. This Technology and supply chain infusion is easy when it is in cooperative structure. Despite of the hard work by the farmer in today's agriculture market its urged by the middleman’s which leads to poverty. Middleman's role in marketing farm product is necessary to remove in order to provide direct sales between farmers and customers. This application will guide farmer in all aspects, current value of their products, profit of their products sold, learning new farming techniques for more production of their products, agricultural compensation schemes provided by the government creating a platform for the farmers for mutual help and providing various material to famers in subsidy rates. This app ensure farmers for maximum profitability by using direct farmer to customer supply chain management. This optimization app allows for better communication between the farmer and the customer. The initial aim of this of this app is to help to bring cooperative structure to one of the labour intensive job.

Here in this app we use Blockchain technology to ensure safe and secure transactions between customers and farmers use of this technologies is easy to keep records for the products sold by the farmers which is used as a reference for future purposes like

keeping records of the crops and their profit margin in various seasons. Crop management system in their land.

This app has some extraordinary tools to access some tool that could never imagine. There are many similar apps are existing but this app is differs by providing shop keeper exclusive for farmers where they can purchase their needs like Weedicide, Pesticide, Essential tools, seeds, manure, etc. and also shopkeeper has the data of the farmers. In fact there are similar applications identified and several mobile applications have been developed for livestock management, direct consumer producer supply chain several other that appeared as mobile applications for effective management of farm land

CHAPTER – 2

LITERATURE SURVEY

2. LITERATURE SURVEY

- **Santosh G. Karkhile Et al**[1]

In this paper researcher given a entire idea about develop a mobile phone based solution that helps in farm management, leads to agricultural yield improvement and helps in farm maintenance. Researcher explain that traditional farming tolerated unexpected environment where as, Modern farming provide expected environment by weather forecasting. Traditional farming requires large amount of labor and different activities for conducting farming. Alternatively Modern farming does not require huge amount of labor as the mobile, machines and new technology take care of the whole thing. This mobile application provides real time weather information, news and market prices at diverse locations and all information is provided in local languages. So, all the outcomes of researcher application are aid farmer to improve their agriculture to yield more earnings. author expand the System Architecture for the farmer app which include different operations like registration of farmers Weather forecasting, News and feeds, Multiple language support, Market trading.

- **Suporn Pongnumkul Et al** [2]

This research represents reviews on Smartphone applications that use Smartphone built-in sensors to give agricultural solutions. According to agriculture function applications are categorized. Researcher literature review describe different types of agriculture application like farming applications, farm management applications, information system applications and extension service applications. Various functionality in farming make simple using this application like Disease

Detection and Diagnosis, Soil Study ,Crop Water Needs Estimation, HR Management, Information System Applications ,
Extension Service Applications This review paper focus that GPS and cameras are the most trendy sensors used in the smart phone application for farming.

- **Alcardo A. Barakabitze Et al** [3]

In this paper researcher explores how a extensive range of Information and Communication Technologies (ICTs) accessible in Agricultural Research Institutes (ARIs)and how farming researchers make effective use of a wide range of ICT tools allied to ,crop variety, land use, irrigation, soil nutrients requirement, weather report, pest and disease control, awareness about crops, pollution control, and new farming techniques.

- **K. Lakshmisudha Et al**[4]

Author represents wireless sensor networks which can help bring about a great revolution in automating agriculture field. This research project makes plant monitoring process easy as well as reduced human effort in farming day to day activity. User can produce customized environment to the plants. This application provides most favorable growth conditions using
different sensors.

- **Shubham Sharma Et al** [5]

In this paper author explain software application which is essentially for sustainable growth of farmers. A lot of time farmer is confused to get decision regarding selection of pesticide, fertilizer and specific time to do particular farming actions. So to minimize such type of problem this application is very useful for farmers. Fertilizer schedule is registered for various crops. Based on sowing date of

crop, farmers get reminders about use of fertilizer as per plan. Additional advice are also given based on soil type, climatic condition etc. This system merges modern Internet technique and mobile communication systems with GPS for proficient and smooth farming.

- **Hemlata Channe Et al** [6]

In this research the proposed architecture of multidisciplinary model is shown which consists of the five modules: 1) Sensor Kit Module. 2) Mobile App Module. 3) Agro Cloud Module. 4) Big-Data Mining, Analysis and Knowledge Building Engine Module. 5) Government &Agro Banks UI In second module researcher explores uses of Mobile applications for farmers. Researcher focus on main three part a. UI for farmer b. UI for agro marketing agency c. UI for agro vendors including fertilizer. By this module all the agriculture related entities are linked together, this model also make possible supply of harvested crops to the agro marketing agencies and different agriculture products and services from agro venders can get by farmers on this app. This model also facilitates estimates of total production per crop in region wise and state wise, total fertilizer requirements. This will be helpful to keep the cost of agricultural products in control. Through notifications farmers also informed about current schemes for agriculture.

- **Shailaja Patil Et al** [7]

In this paper researcher explores how different mobile phone application and precision agriculture services have impacted the farmer's life in their agricultural activities. Android apps offer proficient functionality to be grownup with technology. In the ground like precision agriculture farmers get extra benefits from the mobile apps which are developed for the agriculture monitoring purpose and vital information exchange.

- **Dr.A.V.Senthil Kumar Et al** [8]

In this Journal paper it says about Collaboration is the basis of the three initiatives studied, which has been identified as important in relation to ensuring the social and economic sustainability of farming in the four study regions. In the context of transition management theory, these three initiatives represent new forms of collaboration and can be identified as „socio-technical" innovations. Collaboration is commonly understood as "working together with somebody in order to produce or achieve something". More specifically agricultural cooperation can be defined as "forms of working together in a regulated manner

CHAPTER – 3

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In the existing the system the farmers and the consumer communication is carried out straightly. Farmers sell their products in markets, farmers markets, and vegetable mandi and so on. In this system the farmers go directly to sell their products. Currently, the farmer goes to nearest market handover his product to a particular agent, agent ask the farmer to visit the market after a specific time to collect the cash earned out of the sold product. Agent sells the product to another agent or a dealer at the cost of that market. Every Agent tries to cuts his commission out of that. There is no way for farmer to know about the deal and the exact amount at which their product was sold. There is no transparency.

3.1.1 Disadvantages of Existing System

- The agricultural products from the farmers are at low price, while it reaches consumer.
- Consumer can't get organic food and equal price from the farmers.
- There is a direct contact between the farmer and retail seller or whole seller, thus consumer gets third grade products.

3.1.2 Similar Applications

- **National Agriculture Market or eNAM**

  National Agriculture Market or eNAM is an online trading platform for agricultural commodities in India. The market sells the produce directly taken from the farmer by taking commission from them. More than 100 commodities including staple food grains, vegetables and fruits are currently listed in its list of

commodities available for trade. On the ENAM platform, farmers can opt to trade directly on their own through the mobile app or through registered commission agents.

- **Big Basket**

  Big Basket is the largest online food and grocery store of the country. They are taking vegetable from various farmers and store them and sell the produce directly to the customer through home delivery. They sells product like grains, pulses and other daily needs through their application.

3.1.3 Advantages and Disadvantages of Similar application

Sr.no	Similar App	Advantages	Disadvantages
1	National Agriculture Market	Large number of buyer is buying from them and have fresh	The farmer has to pay the commission to sell produce
2	Big Basket	Sell fresh product to the customer at home	Preserved or Stored food. No benefit to farmer

TABLE 3.1 ADVANTAGES AND DISDAVANTAGES OF SIMILAR APPLICATIONS

3.2 PROPOSED SYSTEM

The proposed work is helpful for farmers to know about the consumer requirement, will act as unique interface of schemes and have better communication between them. In this system the consumer can post their product requirements and needs after the farmer can see the request from consumer, and then they send reply for their queries and needs. And also the proposed system is required to automate the stock maintenance activities with less human effort. In addition, communication among farmers/consumers is made through the web application itself. There is no need of phone calls to request the items. And in this system we have a special feature called shopkeeper where farmers can buy the good for the agriculture for an subside rates

3.2.1 Advantages of Proposed System

- By providing the communication between farmer and consumer the middle man inference can be reduced.
- This will results in great profit for the farmers and also consumers.
- The burden of the manual work is reduced because of the computerization of the system.

3.3 FEASIBILITY STUDY

3.3.1 Introduction

In recent years, India's domestic fresh food prices have increased faster compared to consumer prices. Therefore, how to use mobile E-commerce platform featuring intelligence, networking and informatization to build a mobile E-Commerce platform for fresh agricultural products which integrates traditional E-

Commerce, logistics, commerce, finance and other related enterprises can be used for reference to solve the problem of the circulation of agricultural products in India, and to promote the sustainable development of agriculture and realize the agricultural modernization.

3.3.2 Economic Feasibility

Being Mobile application it will have no cost. It reduces paper and printing cost and also the usage of paper is declined. And helps to store the date for future purpose. most small/medium farm product distribution services need upfront investment for trucks, staff, branding, storage etc. which could done as a whole in cooperative unit and in a lot of cases are not profitable for the first year or two. Based on the goals of the community and the business plan. Funding can include grants loans and investments from the cooperative unit involved there. From these it's clear that this project is financially feasible.

3.3.3 Technical feasibility

This project is mobile based application. The main technologies that are associated with project are

- Eclipse
- NetBeans
- MySQL
- JDK

Each of the technologies are freely available and the technical skills required are manageable. Time limitations of the product and the ease of implementing using these technologies are synchronized.

This requires less bandwidth to transfer data that stored in database, because in this we don't transfer any multimedia file. From this it is clear that our project is technically feasible.

3.4 HARDWARE REQUIREMENTS

3.4.1 Laptops (or) Personal Computer

Laptops (or) Personal computer are used as a admin.

FIG.3.1 ADMIN DEVICE

3.4.2 Android Mobile

Android Mobile in which the app is going to run

FIG.3.2 ANDROID MOBILE

3.5 SOFTWARE REQUIREMENTS

3.5.1 Eclipse

Eclipse IDE used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. It is the most popular IDE for Java development. It is an IDE used in computer programming. It contains a base workspace and an extensible plug-in system for customizing the environment. It is the second-most-popular IDE for Java development

FIG.3.3 ECLIPSE

3.5.2 NetBeans

NetBeans is an integrated development environment (IDE) for Java. NetBeans allows applications to be developed from a set of modular software components called modules

Netbeans runs on Windows, macOS, Linux and Solaris. In addition to Java development, it has extensions for other languages like PHP, C, C++, HTML5, and JavaScript. Applications based on NetBeans, including the NetBeans IDE, can be extended by third party developers.

FIG.3.4 NETBEANS

3.5.3 SQL yog

SQLyog is a GUI tool for the RDBMS MySQL. It is developed by Webyog, Inc., based in Bangalore, India, and Santa Clara, California. SQLyog is being used by more than 30,000 customers worldwide and has been downloaded more than 2,000,000 times.

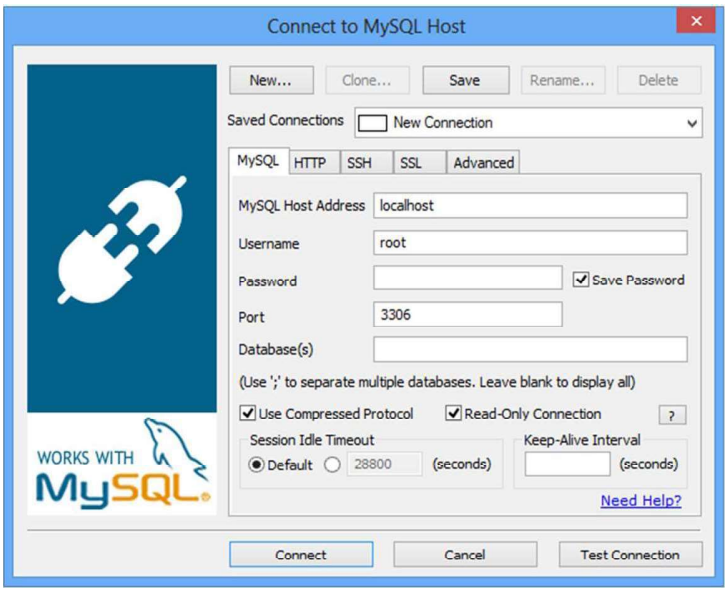


FIG.3.5 SQL yog

CHAPTER – 4

SYSTEM DESIGN

4. SYSTEM DESIGN

4.1 E-R DIAGRAM

ER stands for Entity Relationship. This diagrams display the relationship of various characters of the application (farmer, shopkeeper, admin, user) and it explain the structure of the whole process. this diagram can be made using three basic concepts, attributed, relationships, and entities.

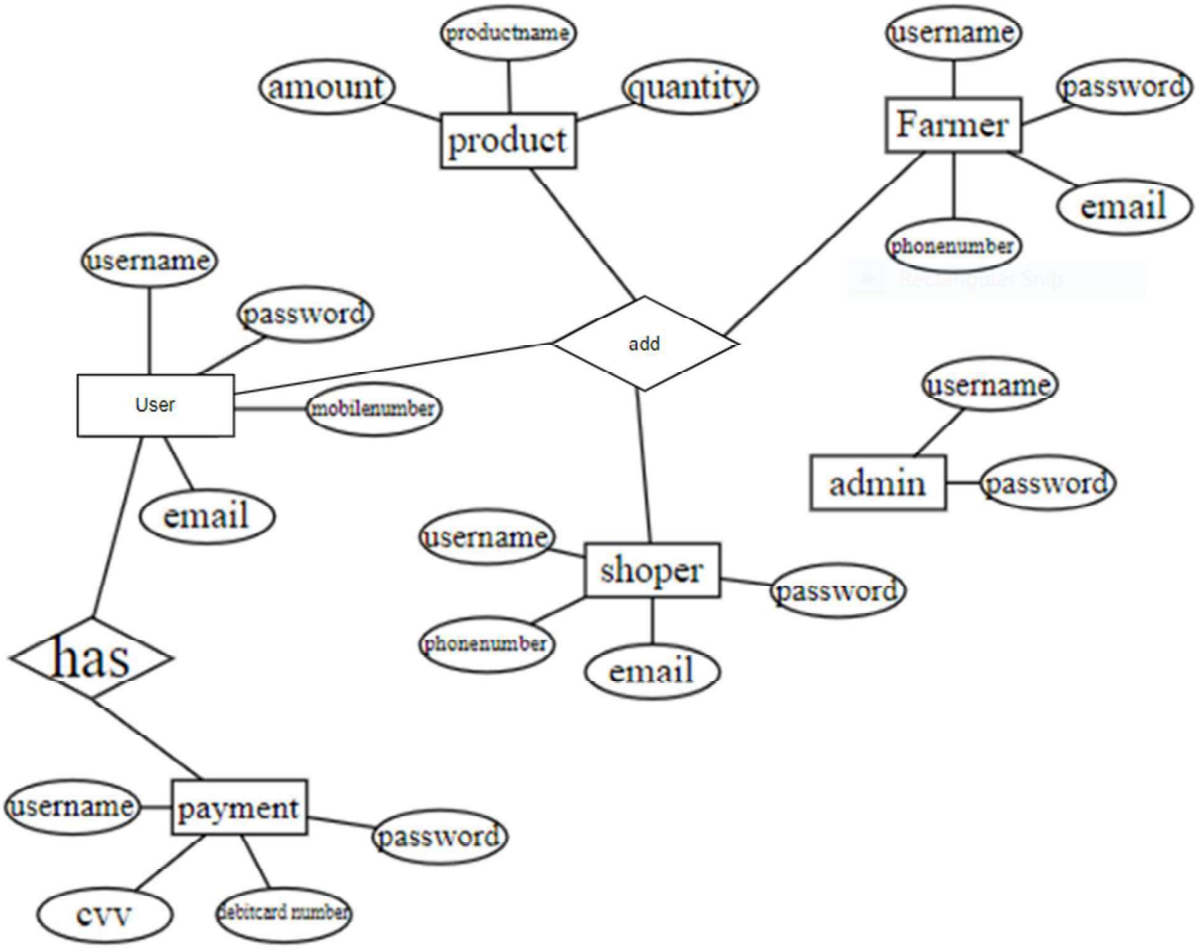


FIG 4.1 E-R DIAGRAM

4.2 UML DIAGRAM

4.2.1 Use Case Diagram

This use case diagram shows all interactions between the user, farmer, shopkeeper, admin and it tells the algorithm used in our application. It is developed in the early stages of the process.

Below is the Use case diagram for our application

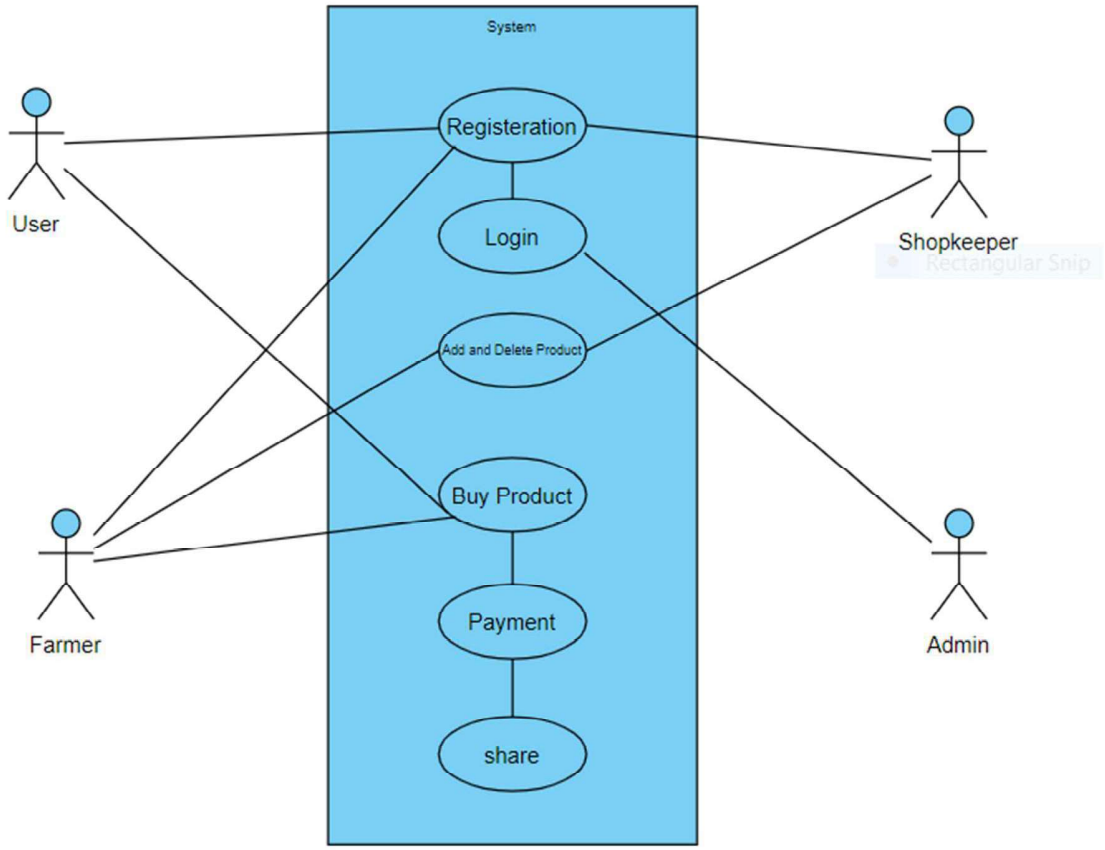


FIG 4.2 USE CASE DIAGRAM

4.2.2 Activity Diagram

This diagram that represents the order of all activities by farmer and user is called the activity diagram. It shows the workflow between different activities that take place in the whole process. However, these are not exactly flowcharts but are similar.

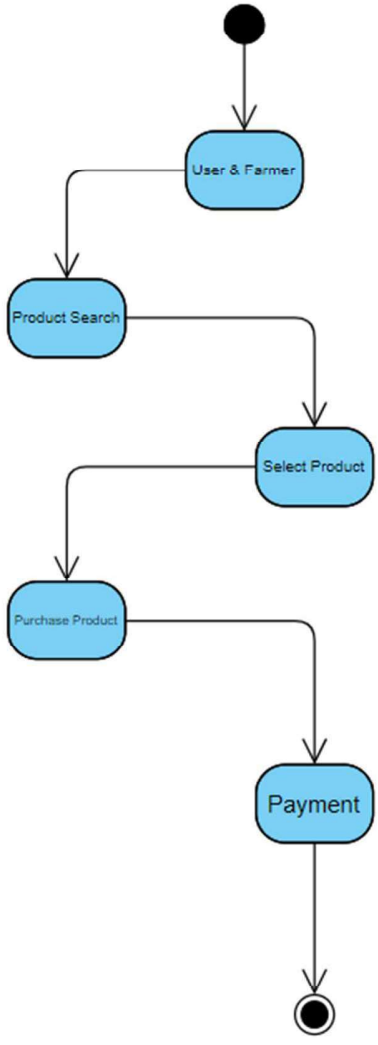


FIG 4.3 ACTIVITY DIAGRAM

4.2.3 Data Flow Diagram Level 0

This is basically a contextual diagram, also referred to as a “context diagram”. It 23 only represents the top level or the 0 Level in the whole process. it gives an abstraction kind of view and shows the whole system as a single process and its relationship to externality.

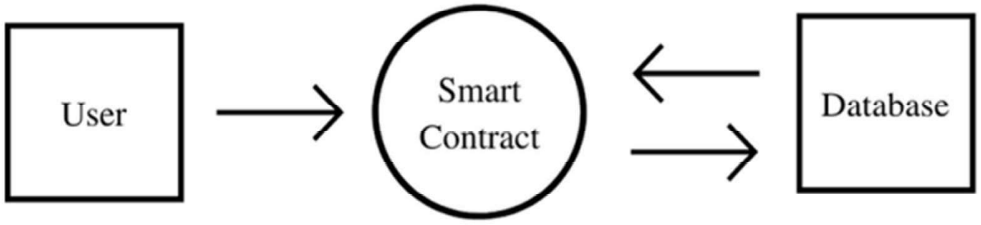


FIG 4.4 DATA FLOW DIAGRAM

4.2.4 Class Diagram

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system.

the admin contains datas like farmer details, products which is connected with user which has address,email and password. Product Details has name, image in binary form, address that needs to be transferred and other details

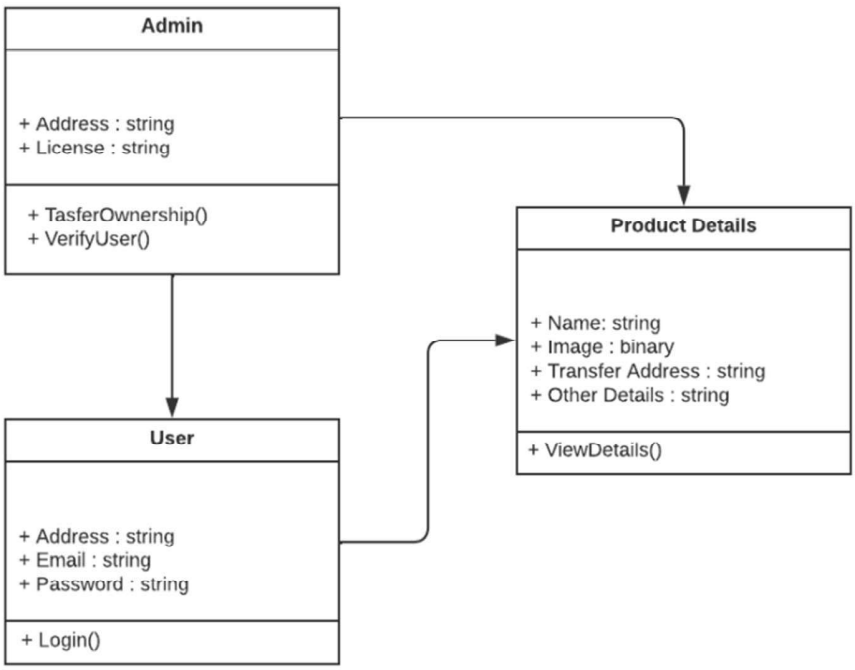


FIG 4.4 CLASS DIAGRAM

CHAPTER – 5

SYSTEM ARCHITECTURE

5. SYSTEM ARCHITECTURE & MODULE DESCRIPTION

5.1 SYSTEM ARCHITECTURE

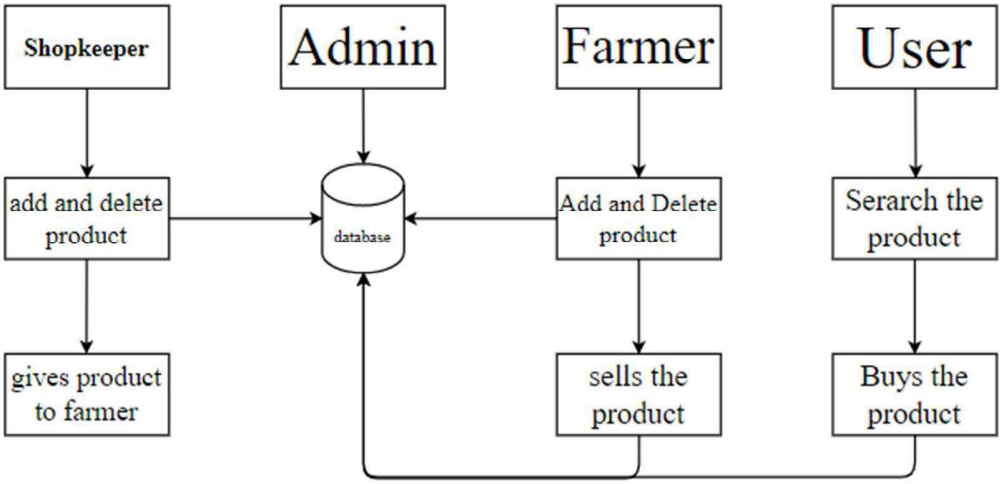


FIG.5.1 SYSTEM ARCHITECTURE

In this application user, farmer, shopkeeper register themselves with their basic details entered by the them these details will be like name, Phone number, email importantly location because it's cooperative in nature which is used to connect near-by farmers.

In this application user wants to buy a product then he searches for the product which he wants. If he finds it then he will go to the description and seller location and make a call to him then he will buy the product. If he doesn't want to buy a product he will check the product review and leave the application.

The farmer who registered in this application gets approval of the admin to check the genuine of the farmers after the approval of the admin only he/she can able to sell the products here while registering itself farmer link his bank account so that

he will directly get benefited from the user and the money transactions will be fully based on blockchain technology which ensures safe and secure transactions.

Shopkeeper who register gets the approval of the admin so that he/she will be genuine and the products need for the farmer

Keeping the records of the goods that are sold also good for both administrator and farmer to know the trend and analysis of the product and plan according to that. These data are stored in the database which is fully carried out by admin and the data is shared to farmer if he needs.

5.2 MODULE DESCRIPTION

5.2.1 User Interface

In our android application we are using the XML programming language. User can register and login process which is fetch from DB. The user can see the near by farmer products list using list view array adapter. In array adapter it storing all the elements which comes from DB. After that the user can able to book the product, using the app wallet.

FIG.5.2 USER INTERFACE

5.2.2 Admin

In admin module, he can see all the details. Admin have a web page for controlling and accessing the details. The page we design for the admin using JSP html and CSS. And fetch the details from DB through JDBC and Servlets.

FIG 5.3 ADMIN

5.2.3 Server

Server will running continuously because then only we can access the DB from Android application. Server will receive the request from SOAP protocol and receive the request with help of WSDL. WSDL is help response the client request which is communicate with DB. The server Communicate with DB with the help of JDBC connection.

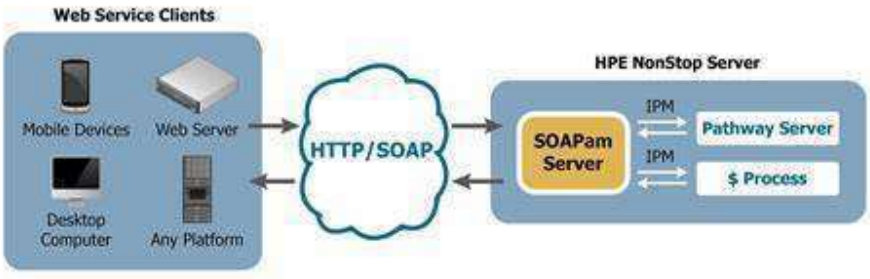


FIG 5.4 SOAP PROTOCOL

5.2.4 Database

In this module we have implement the MYSQL queries for storing and retrieving the data from DB. DB it will store all the details about app such user, admin, product, location details. All the details will store depends on the table which is allocated.

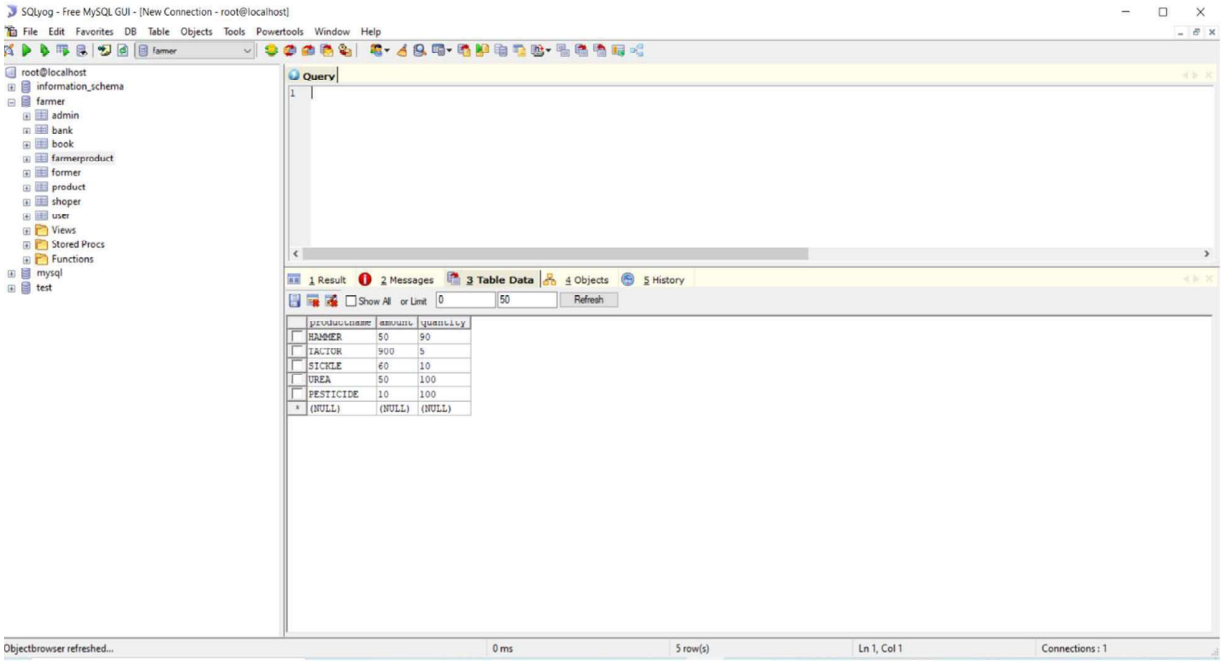


FIG 5.5 DATABASE

5.2.5 Webservice

In this module we implement the soap protocol for sending the client request to server. Using SOAP protocol we need access the Internet so we have to given the internet permission on Andoridmanifest.xml file. Here we are using KSOAP jar files for accessing the SOAP protocol properties and functions and carry the client request to the server.

CHAPTER – 6

SYSTEM

IMPLEMENTATION

6. SYSTEM IMPLEMENTATION

6.1 SAMPLE CODING

callservice.java

```
package com.example.agriproductshop;

import java.io.IOException;

import org.ksoap2.SoapEnvelope;

import org.ksoap2.serialization.SoapObject;

import org.ksoap2.serialization.SoapPrimitive;

import org.ksoap2.serialization.SoapSerializationEnvelope;

import org.ksoap2.transport.HttpTransportSE;

import org.xmlpull.v1.XmlPullParserException;

public class callservice {

                private static String NAMESPACE="http://newpackage/";

                private static String
URL="http://192.168.29.145:8080/Formerservice/NewWebService?WSDL";

        public static String loginService(String s1, String s2,String method) {

                // TODO Auto-generated method stub

                String restex=null;

                SoapObject soap=new SoapObject(NAMESPACE, method);

                soap.addProperty("username",s1);

                soap.addProperty("password",s2);
```

```
            SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(soap);
            HttpTransportSE http=new HttpTransportSE(URL);
            try {
                  http.call(NAMESPACE+method, envelope);
                  SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                  restex=primitive.toString();
            } catch (IOException e) {
                  e.printStackTrace();
                  return "error";
            } catch (XmlPullParserException e) {
                  e.printStackTrace();
                  return "error";
            }
            return restex;
      }
      public static String registerService(String s1, String s2, String s3,
                  String s4, String method) {
            // TODO Auto-generated method stub
            String restex=null;
            SoapObject soap=new SoapObject(NAMESPACE, method);
            soap.addProperty("username",s1);
```

```
                soap.addProperty("password",s2);
                soap.addProperty("email",s4);
                soap.addProperty("mobile",s3);
                SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
                envelope.setOutputSoapObject(soap);
                HttpTransportSE http=new HttpTransportSE(URL);
                try {
                        http.call(NAMESPACE+method, envelope);
                        SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                        restex=primitive.toString();
                } catch (IOException e) {
                        e.printStackTrace();
                        return "error";
                } catch (XmlPullParserException e) {
                        e.printStackTrace();
                        return "error";
                }
                return restex;
        }
        public static String getDeviceList1(String product, String method) {
                // TODO Auto-generated method stub
                String restex=null;
```

```
            SoapObject soap=new SoapObject(NAMESPACE, method);
            soap.addProperty("productname",product);
            SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(soap);
            HttpTransportSE http=new HttpTransportSE(URL);
            try {
                  http.call(NAMESPACE+method, envelope);
                  SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                  restex=primitive.toString();
            } catch (IOException e) {
                  e.printStackTrace();
                  return "error";
            } catch (XmlPullParserException e) {
                  e.printStackTrace();
                  return "error";
            }
            return restex;
      }
      public static String bloginService(String s1, String s2,String method) {
            // TODO Auto-generated method stub
            String restex=null;
            SoapObject soap=new SoapObject(NAMESPACE, method);
```

```
            soap.addProperty("username",s1);
            soap.addProperty("password",s2);
            SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(soap);
            HttpTransportSE http=new HttpTransportSE(URL);
            try {
                  http.call(NAMESPACE+method, envelope);
                  SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                  restex=primitive.toString();
            } catch (IOException e) {
                  e.printStackTrace();
                  return "error";
            } catch (XmlPullParserException e) {
                  e.printStackTrace();
                  return "error";
            }
            return restex;
      }
      public static String bregisterService(String s1, String s2, String s3,
                  String s4, String method) {
            // TODO Auto-generated method stub
            String restex=null;
```

```
            SoapObject soap=new SoapObject(NAMESPACE, method);
            soap.addProperty("username",s1);
            soap.addProperty("password",s2);
            soap.addProperty("accountno",s4);
            soap.addProperty("cvv",s3);
            SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(soap);
            HttpTransportSE http=new HttpTransportSE(URL);
            try {
                  http.call(NAMESPACE+method, envelope);
                  SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                  restex=primitive.toString();
            } catch (IOException e) {
                  e.printStackTrace();
                  return "error";
            } catch (XmlPullParserException e) {
                  e.printStackTrace();
                  return "error";
            }
            return restex;
      }
      public static String bookingservice(String s1, String s2, String name,
```

```
                String productname, String quantity, String cost, String method) {
        String restex=null;
        SoapObject soap=new SoapObject(NAMESPACE, method);
        soap.addProperty("accountno",s1);
        soap.addProperty("cvv",s2);
        soap.addProperty("username",name);
        soap.addProperty("productname",productname);
        soap.addProperty("quantity",quantity);
        soap.addProperty("cost",cost);
        SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
        envelope.setOutputSoapObject(soap);
        HttpTransportSE http=new HttpTransportSE(URL);
        try {
                http.call(NAMESPACE+method, envelope);
                SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                restex=primitive.toString();
        } catch (IOException e) {
                e.printStackTrace();
                return "error";
        } catch (XmlPullParserException e) {
                e.printStackTrace();
                return "error";
```

```
            }


            return restex;
        }
        public static String floginService(String s1, String s2,String method) {
            // TODO Auto-generated method stub
            String restex=null;
            SoapObject soap=new SoapObject(NAMESPACE, method);
            soap.addProperty("username",s1);
            soap.addProperty("password",s2);
            SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(soap);
            HttpTransportSE http=new HttpTransportSE(URL);
            try {
                http.call(NAMESPACE+method, envelope);
                SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                restex=primitive.toString();
            } catch (IOException e) {
                e.printStackTrace();
                return "error";
            } catch (XmlPullParserException e) {
                e.printStackTrace();
```

```
                return "error";
            }
            return restex;
        }
        public static String fregisterService(String s1, String s2, String s3,
                String s4, String method) {
            // TODO Auto-generated method stub
            String restex=null;
            SoapObject soap=new SoapObject(NAMESPACE, method);
            soap.addProperty("username",s1);
            soap.addProperty("password",s2);
            soap.addProperty("email",s4);
            soap.addProperty("mobile",s3);
            SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(soap);
            HttpTransportSE http=new HttpTransportSE(URL);
            try {
                http.call(NAMESPACE+method, envelope);
                SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                restex=primitive.toString();
            } catch (IOException e) {
                e.printStackTrace();
```

```
                return "error";


            } catch (XmlPullParserException e) {
                e.printStackTrace();
                return "error";
            }
            return restex;
        }
        public static String getDeviceList2(String product, String method) {
            // TODO Auto-generated method stub
            String restex=null;
            SoapObject soap=new SoapObject(NAMESPACE, method);
            soap.addProperty("productname",product);
            SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(soap);
            HttpTransportSE http=new HttpTransportSE(URL);
            try {
                http.call(NAMESPACE+method, envelope);
                SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                restex=primitive.toString();
            } catch (IOException e) {
                e.printStackTrace();
```

```
                return "error";


            } catch (XmlPullParserException e) {
                e.printStackTrace();
                return "error";


            }
            return restex;
        }
        public static String addproductService(String s1, String s2, String s3,
                String method) {
            // TODO Auto-generated method stub
            String restex=null;
            SoapObject soap=new SoapObject(NAMESPACE, method);
            soap.addProperty("productname",s1);
            soap.addProperty("amount",s2);
            soap.addProperty("quantity",s3);
            SoapSerializationEnvelope envelope=new
SoapSerializationEnvelope(SoapEnvelope.VER11);
            envelope.setOutputSoapObject(soap);
            HttpTransportSE http=new HttpTransportSE(URL);
            try {
                http.call(NAMESPACE+method, envelope);
```

```
                SoapPrimitive primitive =(SoapPrimitive) envelope.getResponse();
                restex=primitive.toString();
            } catch (IOException e) {
                e.printStackTrace();
                return "error";
            } catch (XmlPullParserException e) {
                e.printStackTrace();
                return "error";
            }
            return restex;
        }
}
```

MainActivity.java

```
package com.example.agriproductshop;
import android.app.TabActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.widget.TabHost;
import android.widget.TabHost.TabSpec;
@SuppressWarnings("deprecation")
public class MainActivity extends TabActivity  {
```

```
        TabHost TabHostWindow;
        Intent inn;
        @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
         TabHostWindow = (TabHost)findViewById(android.R.id.tabhost);
                    TabSpec tab1 = TabHostWindow.newTabSpec("Former");
                    TabSpec tab2 = TabHostWindow.newTabSpec("People");
                    tab1.setIndicator("Former");
                    tab1.setContent(new Intent(this,Farmerhome.class));
                    tab2.setIndicator("People");
                    tab2.setContent(new Intent(this,Peoplelogin.class));
                    TabHostWindow.addTab(tab1);
                    TabHostWindow.addTab(tab2);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

Farmer.java

```
package com.example.agriproductshop;

import android.os.Bundle;

import android.app.Activity;

import android.content.Intent;

import android.view.Menu;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.Toast;

public class Farmer extends Activity {

	Button add,search;

	String username;

	@Override

	protected void onCreate(Bundle savedInstanceState) {

		super.onCreate(savedInstanceState);

		setContentView(R.layout.activity_farmer);

		username = getIntent().getStringExtra("username");

		Toast.makeText(getApplicationContext(), username,
Toast.LENGTH_LONG).show();

		add = (Button)findViewById(R.id.button1);

		search = (Button)findViewById(R.id.button2);

		add.setOnClickListener(new OnClickListener() {
```

```
		@Override
		public void onClick(View arg0) {
			// TODO Auto-generated method stub
			Intent in = new Intent(getApplicationContext(), Addproduct.class);
			startActivity(in);
		}
	});
	search.setOnClickListener(new OnClickListener() {
		@Override
		public void onClick(View arg0) {
			// TODO Auto-generated method stub
			Intent in = new Intent(getApplicationContext(),
FProductsearch.class);
			in.putExtra("username", username);
			startActivity(in);
		}
	});
}
@Override
public boolean onCreateOptionsMenu(Menu menu) {
	// Inflate the menu; this adds items to the action bar if it is present.
	getMenuInflater().inflate(R.menu.farmer, menu);
	return true;
```

```
        }
}
```

Productlist.java

```
package com.example.agriproductshop;
import android.os.Bundle;
import android.app.ListActivity;
import android.content.Intent;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.AdapterView.OnItemClickListener;
public class Productlist extends ListActivity  {
        String username,productname,itemlist;
        String item;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                username = getIntent().getStringExtra("username");
                productname = getIntent().getStringExtra("productname");
```

```
itemlist=getIntent().getStringExtra("list");

Toast.makeText(getApplicationContext(), username,
Toast.LENGTH_LONG).show();

Toast.makeText(getApplicationContext(), productname,
Toast.LENGTH_LONG).show();

Toast.makeText(getApplicationContext(), itemlist,
Toast.LENGTH_LONG).show();

String eventname[] = itemlist.split("@");

setListAdapter(new ArrayAdapter<String>(this, R.layout.activity_productlist,
eventname));

ListView ls = getListView();

ls.setTextFilterEnabled(true);

ls.setOnItemClickListener(new OnItemClickListener() {

    @SuppressWarnings("unused")

    @Override

    public void onItemClick(AdapterView<?> arg0, View view, int arg2,

            long arg3) {

        // TODO Auto-generated method stub

        item = ((TextView)view).getText().toString();

        String[] parts=item.split("\n");

        String Product_name=parts[0].replace("Productname ", "");

        String[] Product_name1=parts[0].split("-");

        String Amount=parts[1].replace("Amount: ", "");

        String quantity=parts[2].replace("Quantity: ", "");
```

```
                                Intent inn=new
Intent(getApplicationContext(),Amountgeneration.class);
                                inn.putExtra("username", username);
                                inn.putExtra("productname", Product_name);
                                inn.putExtra("amount", Amount);
                                inn.putExtra("quantity", quantity);
                                startActivity(inn);
                            }
                        });
                }
}
```

Register.java

```
package com.example.agriproductshop;
import android.os.AsyncTask;
import android.os.Bundle;
import android.app.Activity;
import android.content.Intent;
import android.telephony.gsm.SmsManager;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
```

```
import android.widget.EditText;
import android.widget.Toast;
@SuppressWarnings("deprecation")
public class Register extends Activity {
        EditText user,pass,phno,mail;
         Button reg;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_register);
                user = (EditText)findViewById(R.id.username);
       pass = (EditText)findViewById(R.id.password);
       phno = (EditText)findViewById(R.id.phoneno);
       mail = (EditText)findViewById(R.id.email);
      reg = (Button)findViewById(R.id.register);
      reg.setOnClickListener(new OnClickListener() {
                        @Override
                        public void onClick(View arg0) {
                                // TODO Auto-generated method stub
                                ActiveTask1 task1=new ActiveTask1();
                                task1.execute();
                        }
                });
```

```
}
private class ActiveTask1 extends AsyncTask<String,Void,Void> {
        String s1=user.getText().toString();
        String s2=pass.getText().toString();
        String s3=phno.getText().toString();
        String s4=mail.getText().toString();
        String res=null;
        @Override
        protected void onPreExecute() {
                Toast.makeText(getApplicationContext(), "Please wait.....",
Toast.LENGTH_LONG).show();
        }
        @Override
        protected Void doInBackground(String... params) {
                res=callservice.registerService(s1,s2,s3,s4,"Register");
                return null;
        }
protected void onPostExecute(Void result) {
        if(res.equals("success")){
        Intent intent=new Intent(getBaseContext(),MainActivity.class);
        startActivity(intent);
        try
{
```

```
                SmsManager sms=SmsManager.getDefault();
            sms.sendTextMessage(s3, null, "Your Account Has successfully Registered",
null, null);
            Toast.makeText(getApplicationContext(), "SMS Sent Successfully",
Toast.LENGTH_LONG).show();
        }
        catch(Exception e)
        {
            e.printStackTrace();
            Toast.makeText(getApplicationContext(), "SMS Sent Failed",
Toast.LENGTH_LONG).show();
        }
                        }
        else{
            Toast.makeText(getApplicationContext(), "SMS Sent Failed",
Toast.LENGTH_LONG).show();
        }
        }
        }
        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
            // Inflate the menu; this adds items to the action bar if it is present.
            getMenuInflater().inflate(R.menu.register, menu);
            return true; }}
```

CHAPTER - 7

TESTING

7. TESTING

7.1 TESTING OBJECTIVES

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.2 TYPES OF TESTS

7.2.1 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration.

7.2.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent.

7.2.3 Functional Testing

Functional testing provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

7.2.4 System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing predriven process links and integration points.

7.2.5 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirement.

7.3 TEST CASES

S.NO	ACTIONS	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	TEST RESULT
1	Admin Login	ADMIN	Response 200	Response 200	PASS
2	Shopkeeper Login	HARI	Response 200	Response 200	PASS
3	Add product	Product Details	Item Added	Item Added	PASS
4	Farmer Login	VASA	Response 200	Response 200	PASS

5	Add product	Product Details	Item Added	Item Added	PASS
6	User Login	HARSHA	Item Added	Item Added	PASS
7	Payment	Payment Details	Response 200	Response 200	PASS

TABLE 7.1 TEST CASES

CHAPTER – 8

CONCLUSION AND FUTURE ENCHANCEMENT

8 CONCLUSION & FUTURE ENCHANCEMENT

CONCLUSION

Android application connects the near-by farmers and help mutually for the increased production by sharing the resources our aim is not only to provide direct supply of their good but also to provide necessary assistance to farmers.

In this application shopkeepers provides items like tractor, harvester, manure, fertilizer, etc, to farmers and providing finical assistance. This application ensure more user friendly.

FUTURE ENHANCEMENT

Creating the app in native language would be a great addition because it will be easily understandable for both the user and farmer. SMS verification of the user and Blockchain transaction which needs several permissions are half way done in my project.

Doing more testing for this app so that errors can be neglected and work perfectly

APPENDICES

APPENDICES

A.1 SCREENSHOTS

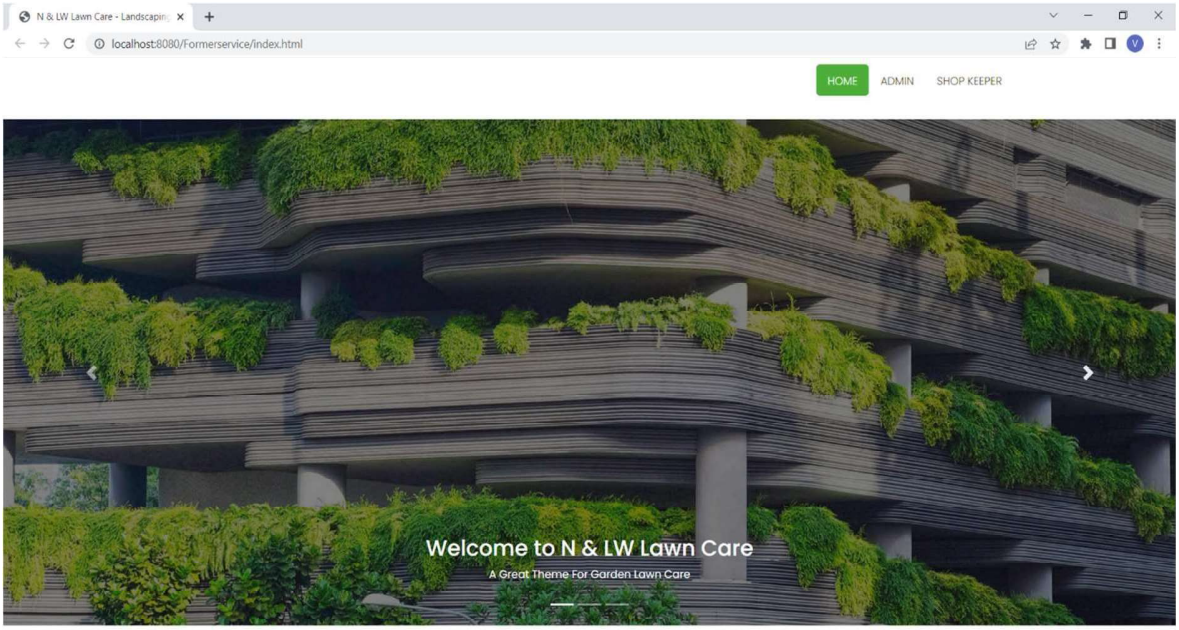


Figure A.1 – Admin and Shopkeeper Welcome page

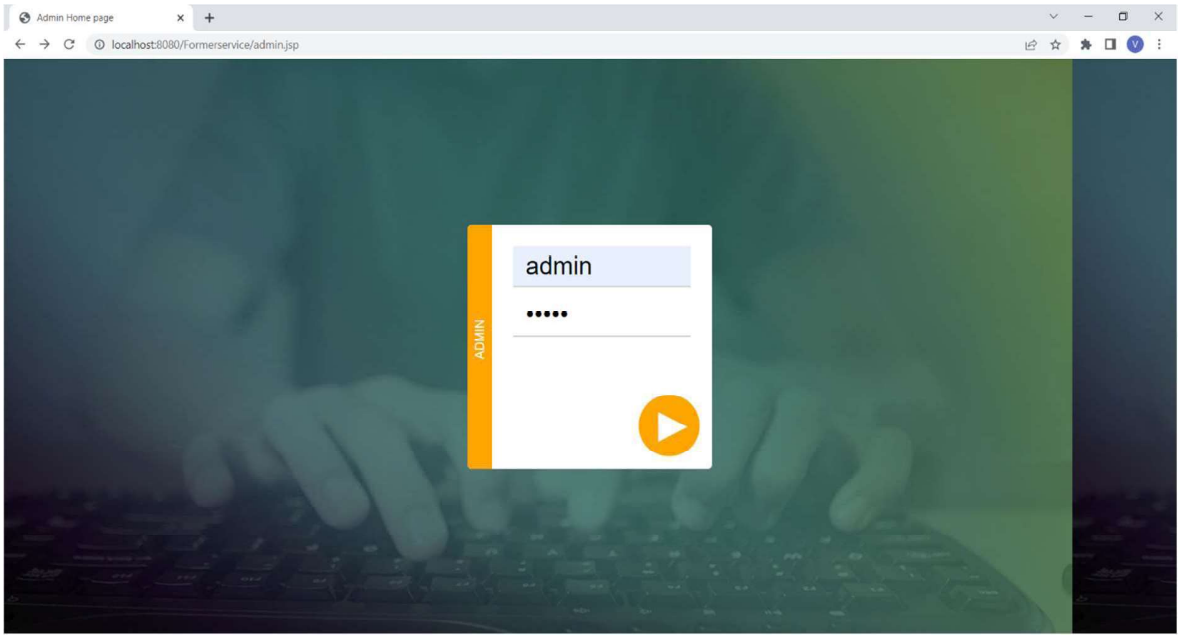


Figure A.2 – Admin Login

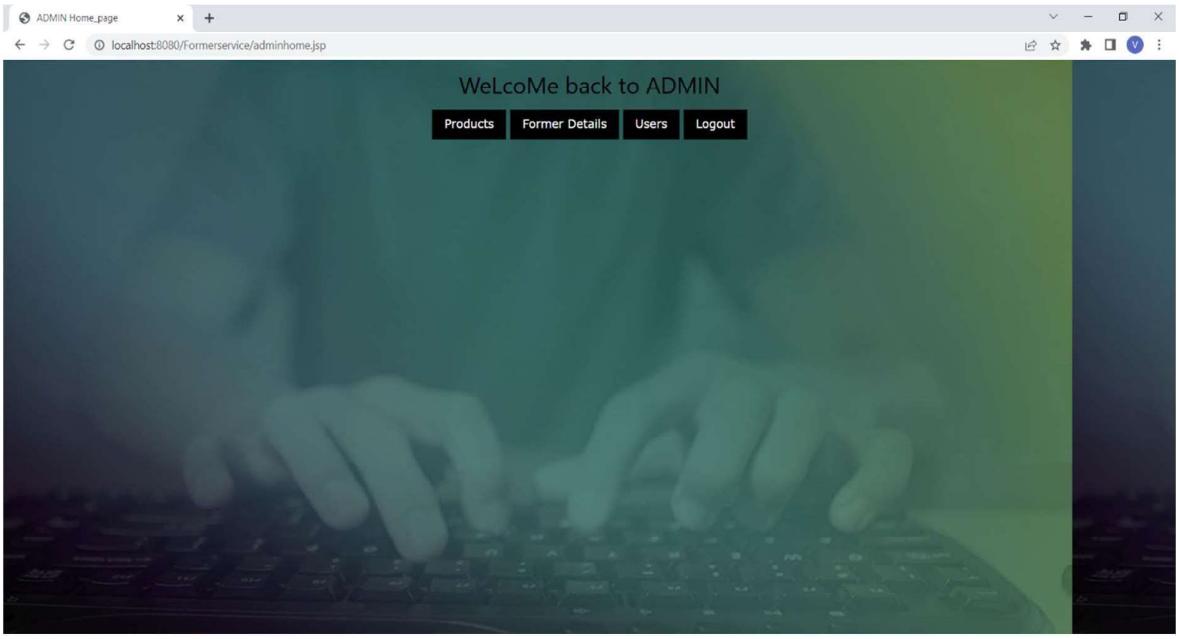


Figure A.3 – Admin page

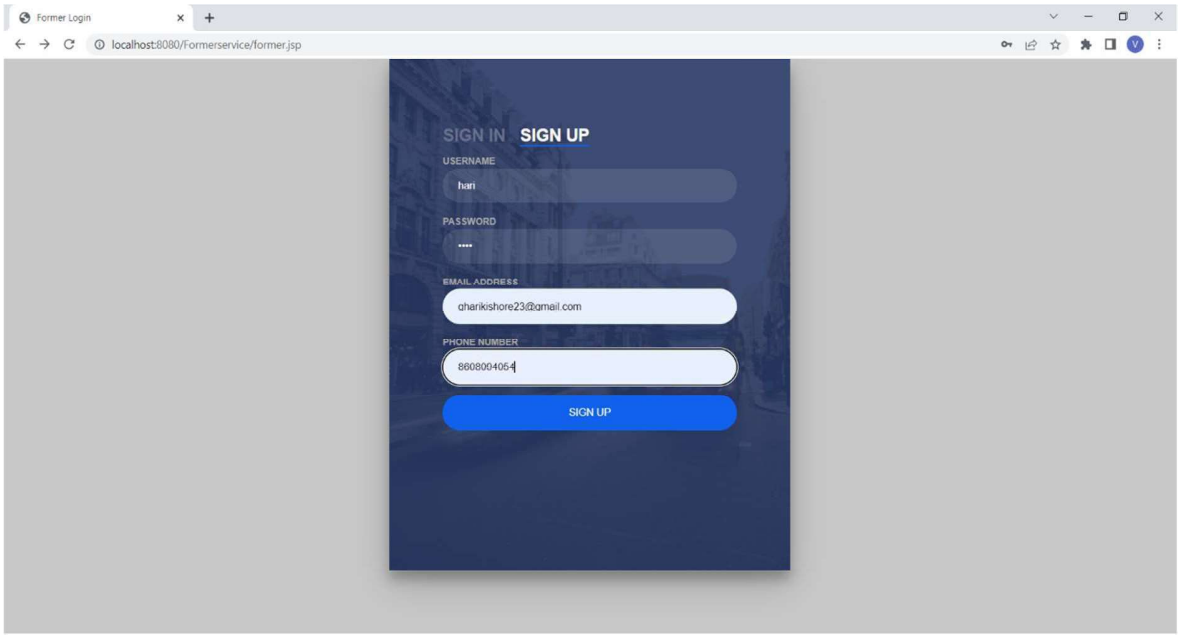


Figure A.4 – Shopkeeper Registration

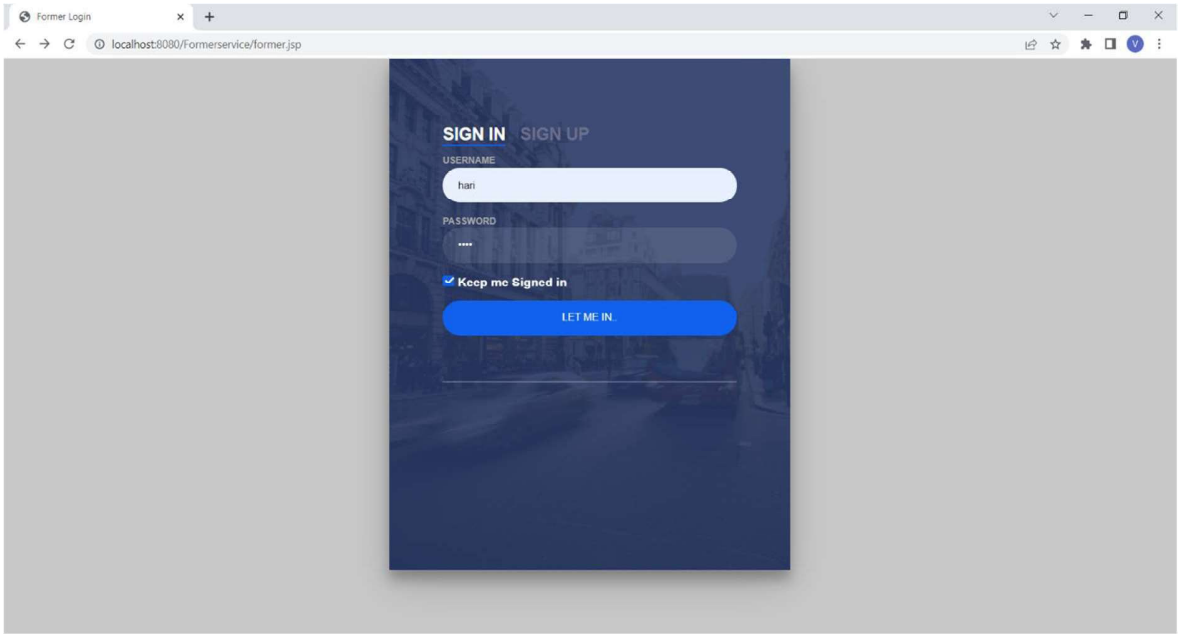


Figure A.5 – Shopkeeper Login

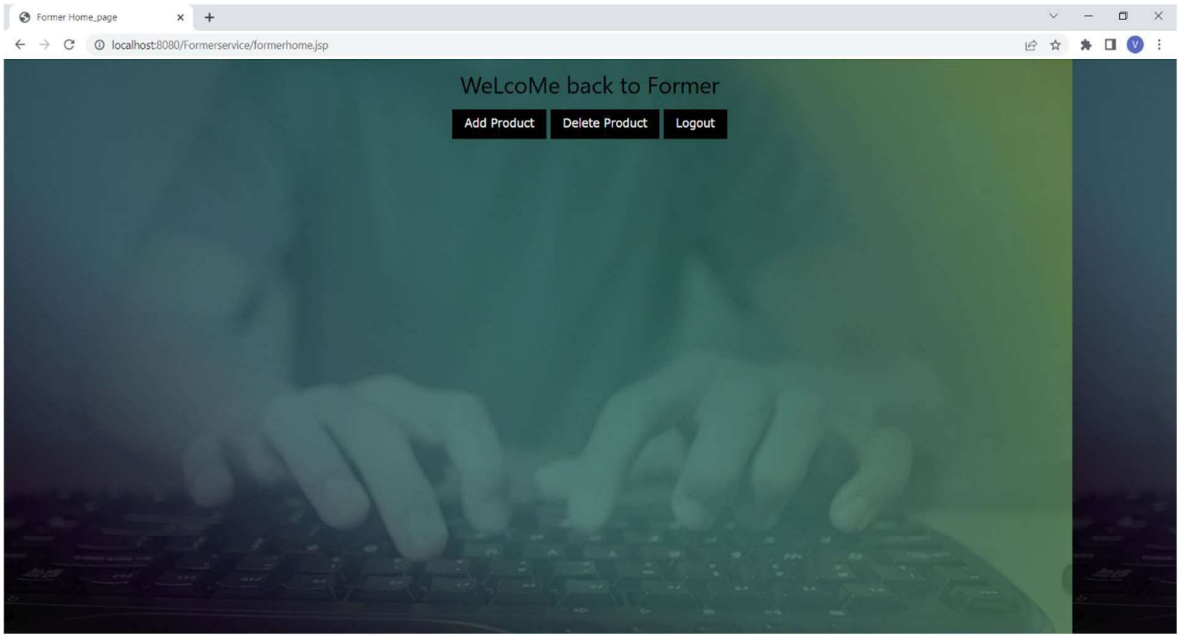


Figure A.6 – Shopkeeper Page

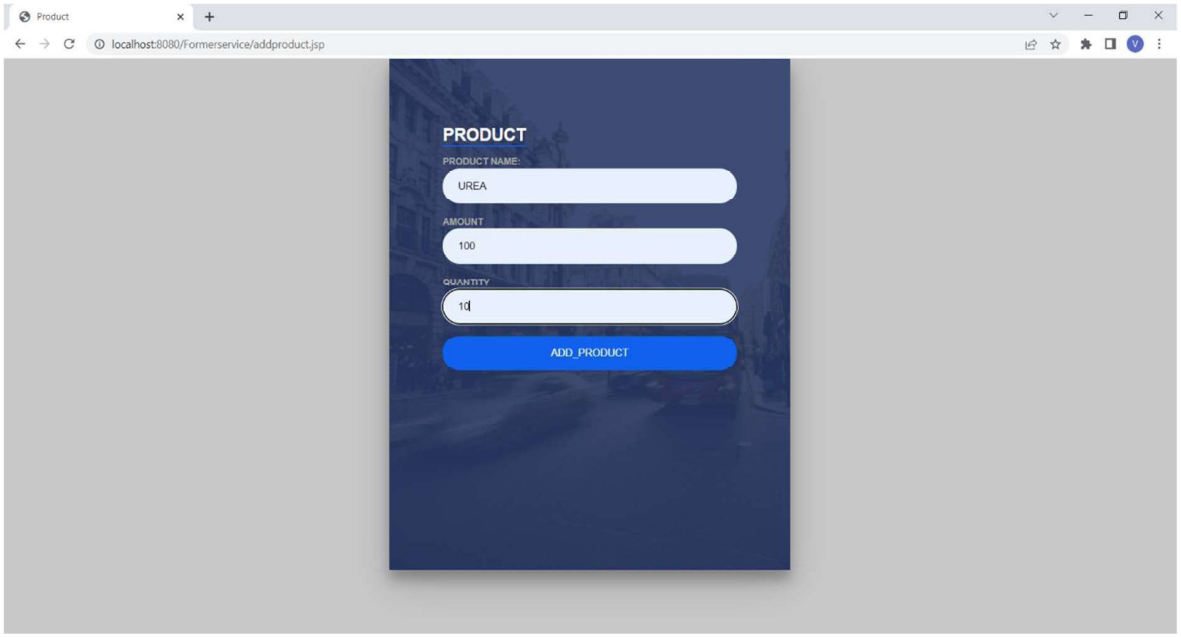


Figure A.7 – Shopkeeper adding items

Figure A.8 – Shopkeeper deleting items

Figure A.9 - Farmer Registeration

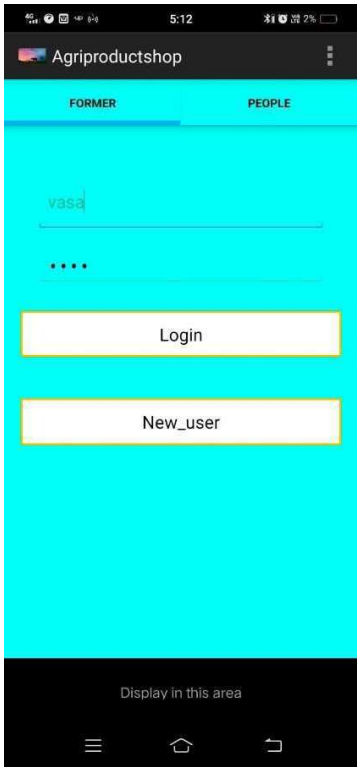


Figure A.10 – Farmer Login

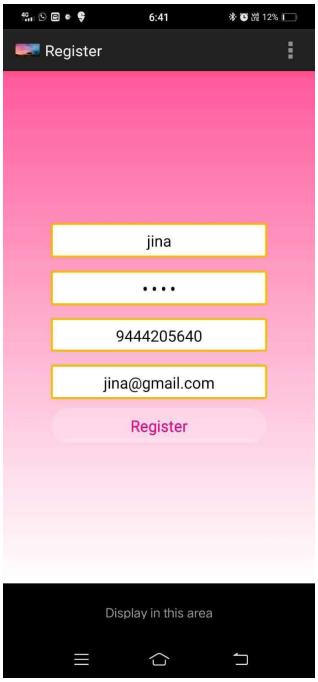


Figure A.11 – User Registeration

Figure A.12 – User Login

Plagiarism Scan Report

Report Generated on: May 21,2022

Plagiarised

Unique

Total Words:	639
Total Characters:	3989
Plagiarized Sentences:	0
Unique Sentences:	29 (100%)

Content Checked for Plagiarism

ABSTRACT

This is an app that facilitates Farmers and Customers for smooth, effective and newly composed supply chain mechanism which is cooperative in nature and the increase in demand for locally produced food is widely acknowledged by industries, government. This app not only focuses on direct supply of product to customers but also to provide necessary assistance to the farmers. This has seen the rise in interest for small/medium farm product distribution projects in INDIA. Farmers are selling products to middleman for less profit, but lack of knowledge to the farmer we thought of doing an application that can help farmers in multiple ways one of that is sell their own products directly to customer with no brokers and also to provide a platform for the farmer where the produce from the farmer can be sold at best rates, sharing the transport to take the produced products to the markets and to help farmers on various things which based on the cooperative benefits

INTRODUCTION

India's one of the Top priority sector is agriculture. but today the people involved in agriculture experience problem in their day-to-day life due to extreme poverty. In India about 15% of GDP comes from agriculture but the employers in this work are 50% of our working population. In India income generation by farm is been a key issue. Due to lack of awareness of modern technology or advanced techniques and unstable supply chain management leads to Farm poverty. This Technology and supply chain infusion is easy when it is in cooperative structure. Despite of the hard work by the farmer in today's agriculture market its urged by the middleman's which leads to poverty. Middleman's role in marketing farm product is necessary to remove in order to provide direct sales between farmers and customers. This application will guide

farmer in all aspects, current value of their products, profit of their products sold, learning new farming techniques for more production of their products, agricultural compensation schemes provided by the government creating a platform for the farmers for mutual help and providing various material to famers in subsidy rates. This app ensure farmers for maximum profitability by using direct farmer to customer supply chain management. This optimization app allows for better communication between the farmer and the customer. The initial aim of this of this app is to help to bring cooperative structure to one of the labour intensive job.

Here in this app we use Blockchain technology to ensure safe and secure transactions between customers and farmers use of this technologies is easy to keep records for the products sold by the farmers which is used as a reference for future purposes like keeping records of the crops and their profit margin in various seasons. Crop management system in their land.

This app has some extraordinary tools to access some tool that could never imagine. There are many similar apps are existing but this app is differs by providing shop keeper exclusive for farmers where they can purchase their needs like Weedicide, Pesticide, Essential tools, seeds, manure, etc. and also shopkeeper has the data of the farmers. In fact there are similar applications identified and several mobile applications have been developed for livestock management, direct consumer producer supply chain several other that appeared as mobile applications for effective management of farm land

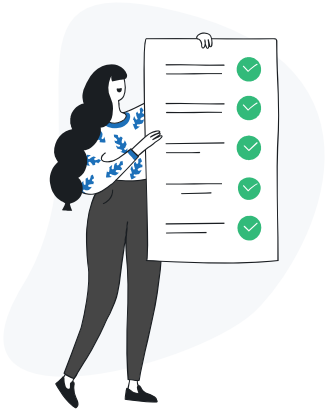
CONCLUSION

This android application connects the near-by farmers and help mutually for the increased production by sharing the resources our aim is not only to provide direct supply of their good but also to provide

necessary assistance to farmers too.

FUTURE ENHANCEMENT

Creating the app in native linguistic language would be a great addition because it will be easily understandable for both the user and farmer. SMS verification of the user and Blockchain transaction which needs several permissions are half way done in my project.



No Plagiarism Found

REFERENCES

REFERENCES

[1] K. Lakshmisudha and Swathi Hegde "Smart Precision based Agriculture using Sensors" International Journal of Computer Applications (0975 –8887) Volume 146 No.11, July 2016

[2] Hemlata Channe and Sukhesh Kothari "Multidisciplinary Model for Smart Agriculture using Internet of-Things (IoT), Sensors, Cloud-Computing, Mobile-Computing & Big-Data Analysis" Int.J. Computer Technology & Applications, Vol 6(3),374-382 ISSN:2229-6093

[3] Shailaja Patil and Anjali R.Kokate "Precision Agriculture: A Survey" International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064 Index Copernicus Value (2013): 6.14 Impact Factor (2015): 6.391

[4] Shubham Sharma, Viraj Patodkar, Sujit Simant, Chirag hah Prof.Sachin Godse "E-Agro Android Application"(Integrated Farming Management Systems for sustainable development of farmers) International Journal of Engineering Research and General Science Volume 3, Issue 1, January-February, 2015 ISSN 2091-2730

[5] Sotiris Karetsos, Constantina Costopoulou, Alexander Sideridis "Developing a smart phone app for m-government in agriculture"Journal of Agricultural Informatics.2014 Vol. 5, No. 1.

[6] Shitala Prasad1, Sateesh K.Peddoju2 and Debashis Ghosh3,"Agro Mobile: A Cloud-Based Framework for Agriculturists on Mobile

Platform" International Journal of Advanced Science and Technology Vol.59, (2013), pp.41-52

[7] M. V. Bueno-Delgado , J. M.Molina-Martínez , R. Correoso-Campillo , P. Pavón-Mariño"Ecofert: An Android application for the optimization of fertilize cost in fertigationq Computers and Electronics in Agriculture www.elsevier.com/locate/compag

[8] .T.Poovarasan and Dr.A.V.Senthil kumar "Farmer Consumer direct interaction app" International journal of research in computer applications and robotics (ijrcar) vol.8 Issue 3 pg 15-21 March 2020

WEB REFERENCE

https://www.agritechtomorrow.com/article/2018/10/farmers-aregrowingcomfortable-withmobile-apps/11056

http://agricoop.nic.in/programmes-schemes-listing

https://agricoop.nic.in/hi/programmes-schemes-listing

http://sfacindia.com/FPOS.aspx

BOOK REFERENCE

Cay S. Horstmann and Gary Cornell, "Core Java™, Volume I – Fundamentals" 8th Edition, Prentice Hall, 2007.

Cay S. Horstmann and Gary Cornell, "Core Java, Vol. 2: Advanced Features", 8th Edition, Prentice Hall, 2008