

# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JNANA SANGAMA”, BELAGAVI- 590018



## A MINI PROJECT REPORT

ON

## DATABASE MANAGEMENT SYSTEM

Submitted in complete fulfillment of the requirements

For the award of degree of

**Bachelor of Engineering**

In

**Computer Science and Engineering**

By

**Teenu Prathyush**

**[1KS15CS113]**

Under the guidance of

**Ms. Impana K.P.**

**Mrs. RenukaDevi.S**

**Mr. Pradeep Kumar**

**Asst.Prof, Dept of CSE**

**Asst.Prof, Dept. of CSE**

**Asst.Prof, Dept. of CSE**



**Department of Computer Science & Engineering**

**K.S.INSTITUTE OF TECHNOLOGY**

#14, Raghuvanahalli, Kanakapura Main Road, Bengaluru-560109.

2017-2018

# K.S.INSTITUTE OF TECHNOLOGY

#14, Raghuvanahalli, Kanakapura Main Road, Bengaluru-560109.



**Department of Computer Science & Engineering**

## CERTIFICATE

This is to certify that mini project work entitled **“Online Agriculture System”** carried out by **Mr.Teenu Prathyush** bearing USN **1KS15CS113** bonafide student of **K.S. Institute of Technology** in the partial fulfillment for the award of the **Bachelor of Engineering in Computer Science & Engineering** of the **Visvesvaraya Technological University**, Belagavi, during the year 2017. It is certified that all corrections / suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of mini Project work prescribed for the said degree.

**Dr. Rekha.B. Venkatapur**

Prof & HOD, CS&E Department

**Dr. T.V. Govindaraju**

Principal/Director,KSIT

**Ms. Impana K.P.**

Asst.Prof, Dept of CSE

**Mrs. RenukaDevi.S**

Asst.Prof, Dept. of CSE

**Name of the Examiners**

**Signature with date**

1.

2.

# ACKNOWLEDGEMENT

I take this opportunity to thank one and all involved in building this project. Firstly, I would like to thank the college for providing us an opportunity to work on the project.

First and foremost, I thank the management of **K. S. Institute of Technology** for providing all the resources required for the project.

I wish to acknowledge my sincere gratitude to our beloved **Principal, Dr. T.V. Govindaraju** for his encouragement and providing all facilities for the accomplishment of this project.

The project would not have been possible without the support of our beloved Prof & HOD, **Dr.Rekha.B.Venkatapur** , Dept of CS&E.

I am also highly grateful to my project guide, **teacher, designation, Dept of CS&E and teacher, designation, Dept of CS&E** who have been very generous in assisting and supporting, to do this Project “**Online Agriculture System**”, which formally started as just a rough idea and now it is resulted in this.

I also would like to thank all the other teaching and nonteaching staff members who had extended their support and co-operation while bringing up this project.

**Teenu Prathyush**

**1KS15CS113**

# **ABSTRACT**

This is a web based project which is useful for farmers and agricultural students. This is an open discussion portal providing solutions to small farmers and agricultural students. It also provides soil analysis for all regions and suggestions on which fertilizers to use where and how much? And which crop, herb or vegetable to be grown where and in which season? It also helps to make decisions on market and best prices. Information about major crop markets and their current price for the crop will be published daily.

Online feedback can be given by all users. Feedback can be posted by students, farmers and the general public. Feedbacks are directed to the administrator of the system. Information pages should be dynamic so that the administrator can change it and the users only view the information.

# Description of the Project

## Technologies used:

- HTML, CSS, JavaScript for front end view.
- MySQL for database management.
- Java, Servlets, JDBC for connectivity, controlling and model components.

## Users of the system:

- Administrator
- Farmers
- Agriculture students
- Agriculture officers
- General Public

## Modules: It has been modularized into following modules:

- User Module
- Soils and Fertilizers
- Crop Details
- Market Details and Commodity
- Feedback

- 1. User Module :** First, to enter into this system the users has to login to this system. There are 2 types of users in this system.
  - Admin users - Has full access to all the modules of this system.
  - Farmers, Agriculture Students, Agriculture Officers and General Public – Has restricted access. i.e., Normal users have access to some of the modules only.
- 2. Soils and Fertilizers :** This module is used to maintain the various Soils and Fertilizers Details. Admin type of users have full access to this module. Other users have limited access.

This module contains:

- A separate screen should be provided to maintain the Soils and Fertilizers Details. It should provide a way to add, update and delete both the details.
- If a new Soil Information is received it should be added to the System.

- If a new Fertilizer information is received it should be added to the system with the corresponding details like soil name, crop type, crop name etc.

**3. Crop Details :** This module is used to maintain the various details about crops. Admin type of users have full access to this module. Other users have limited access.

This module contains:

- A separate screen should be provided to maintain the Crops Information. It should provide a way to add, update and delete the crop details.
- If a new crop information is received, it should be added to the system with the corresponding details like Soil Name, Crop Type, Crop Name and Season.

**4. Market Details and Commodity :** In this module we can maintain the market details. Admin type of users have full access to this module. Other users have limited access.

This module contains:

- A separate screen should be provided to maintain the market related information. It should provide a way to add, update and delete the market related information.
- Administrator type of user can add, update and delete the commodities in the market.
- He can add the information about new markets into the system.

**5. Feedback :** This module is used to give feedback. This module contains:

- A separate screen should be provided to non-admin users to send feedback about the various crops, soils, fertilizers, and market information to the admin.
- Administrator type of user should be able to view the feedback sent by any user of the system.
- Administrator type of user should also be able to view the date and time of the sent feedback.

**Advantages:** This system is helpful to all the people who study about Agriculture. They can gain valuable information about Soils, Crops, Fertilizers, Commodity, and Markets across India. New information will be updated every day so people can view about the existing state of Agriculture in the country. Feedback given by the users will be taken into account and any problems, which may arise, will be resolved.

# CONTENTS

SL.NO	CHAPTER NAME	PAGE NO.
	ACKNOWLEDGEMENT	I
	ABSTRACT	II
	DESCRIPTION	III
1	Introduction	1
	1.1 About Project	1
	1.2 Working	2
	1.3 Scope	3
	1.4 Importance	3
2	Requirements Analysis	4
	2.1 Software Requirements	4
	2.2 Hardware Requirements	4
	2.3 Functional Requirements	5
	2.4 Non-Functional Requirements	5
3	Design	6
	3.1 Frontend Design	6
	3.1.1 Hypertext Markup Language (HTML)	6
	3.1.2 Cascading Style Sheet (CSS)	6
	3.1.3 JavaScript	6
	3.1.4 Java Server Pages(JSP)	7
	3.2 Backend Design	8
	3.2.1 Schema of the database	8
	3.2.2 E-R diagram of the database	9
	3.2.3 EER diagram of the database	10
4	Implementation	11
	4.1 Backend Implementation	11
	4.2 Frontend Implementation	14
	4.2.1 HTML	14
	4.2.2 CSS	15
	4.2.3 JavaScript	15
	4.3 Connectivity Implementation	16
5	Testing and Snapshots	18
	5.1 Testing	18
	5.2 Types of Testing	18

5.2.1 Unit Testing	18
5.2.2 Integration Testing	18
5.2.3 System Testing	19
5.3 Snapshots	20
5.3.1 Signup and login	20
5.3.2 Add Soil / Remove Soil / Update Soil	22
5.3.3 Add Crop/Add Fertilizer/Add Market/Add Commodity	23
5.3.4 Send Feedback/View Feedback	25
5.3.5 The Home Page	26
5.3.6 Database Snapshots	27
CONCLUSION	29
FUTURE ENHANCEMENTS	30
REFERENCES	31



# CHAPTER 1

## INTRODUCTION

### 1.1 About Project

We start this chapter by defining the very meaning of the word **project**. It can be defined in many ways, one can define it as *a planned piece of work that has a specific purpose and that usually requires a lot of time*.

A Project to be developed will be under influence of many factors such as feasibility, scalability, resources available, resources required, time constraints, requirement for the project.

Any project goes through multiple phases such as project planning, requirements analysis, project design, construction, testing, integrating, deployment and maintenance. These phases can be referred to as life cycle of a project.

Project that is been assigned and expected to be delivered is a database management system with a client - server interaction where the client (user) should be able to request and get access to the database on the server side and add new entries to the database.

The constraints laid by the university on the database management system mini project are as follows:

- For any problem selected, write the ER Diagram, apply ER-mapping rules, normalize the relations, and follow the application development process.
- Make sure that the application should have five or more tables, at least one trigger and one stored procedure, using suitable frontend tool.
- Indicative areas include; health care, education, industry, transport, supply chain,etc.

## 1.2 Working

Any client-server applications require a network to be established. The request and service by the client and the server respectively should happen remotely. The components of a simple client-server system include a client system (which maybe a desktop, phone, any electronic gadget with access to a network), a server and a network which will be able to connect client and server.

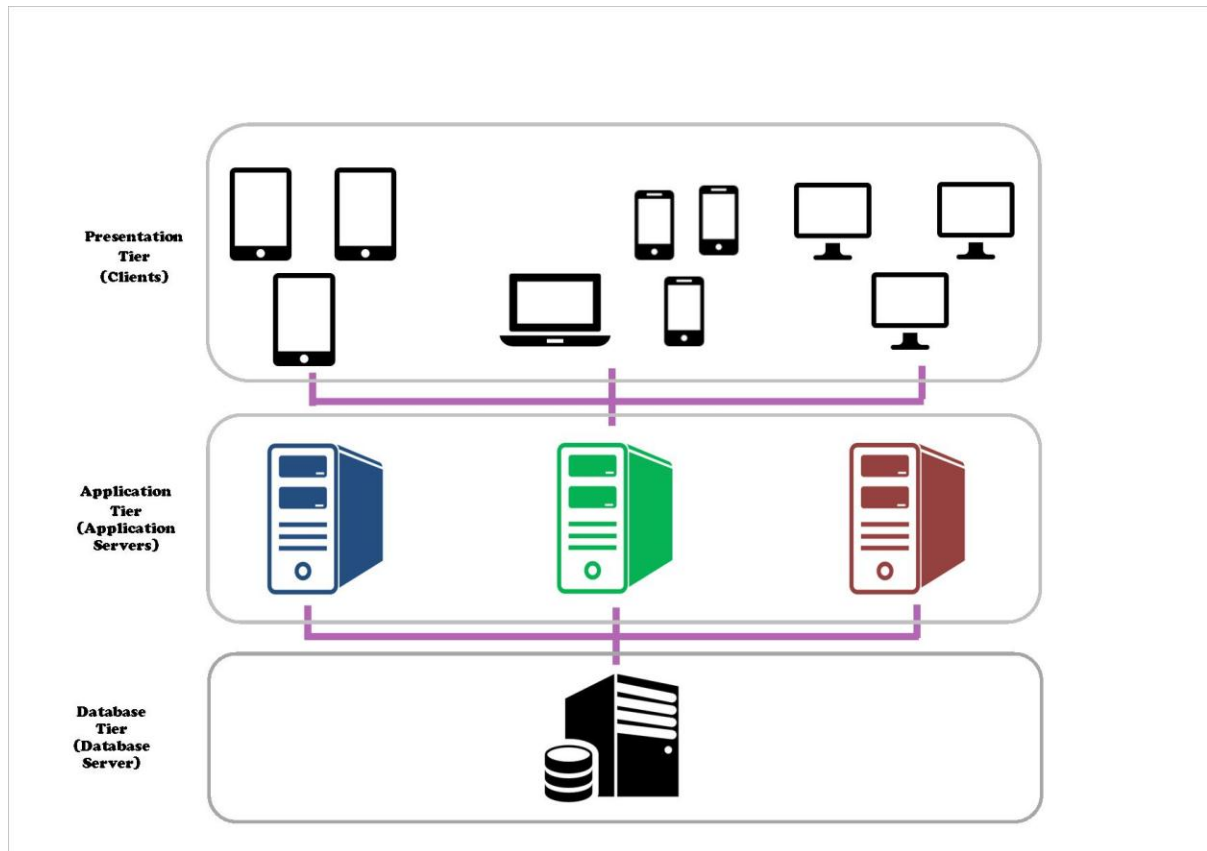


Fig 1.1 Client Server Architecture

The figure 1.1 shows simple client-server architecture. It consists of various clients connected to a server, which will be in turn monitoring the database.

A client sends a request to the server through a network based on agreed protocols as the server receives the request it sends the acknowledgement that it is ready to accept the requests, the client then starts sending the requests which will be computed, stored and retrieved from the database in the server side.

## **1.3 Scope**

A database management system is the root of maintaining, managing, organizing data structures of all scale while over a network or within a system. A database management has a large scale scope and can be used in all many fields which will be a prime factor in implementing mobility and convenience to tasks. It can be used in fields such as feedback system, query, reporting, book keeping, etc. it can be used by sectors such as schools, universities, hospitals, banks, libraries, etc.

## **1.4 Importance**

The first ever database management system was developed four decades ago by IBM and since then the need for data to kept and maintained by database management system has been quite prominent in all sectors and companies. A database management system makes maintaining data more efficiently by features like data abstraction, support of various data types, multiple users, minimizing redundancy, adding relations between the data, sharing of data, Integration of Data, Integration Constraints, Data Security, Backup and Recovery Procedures, Data Independence are some of the features which are not provided by traditional data management system.

Hence one can say they are responsible for making management of data more efficient, secure, reliable, and less prone to data loss.

## CHAPTER 2

# REQUIREMENT ANALYSIS

The requirement analysis specifies the requirements needed to develop a project. In this phase, we collect the requirements needed for designing the project. The requirements collected are then analyzed and carried to the next phase.

## 2.1 Software Requirements

**Software requirements** refers all the aspects required from the software side. These requirements involve software, tools, utilities which are required by the database project.

There are no specific requirements for the project as it is flexible and not necessarily have fixed requirements. The following are the features of the machine with which the project has been built and deployed:

- Windows 7 64-bit Operating System
- MySQL database server v5.7 (64-bit)
- MySQL Workbench v6.3 CE (64-bit)
- Microsoft Visual C++ 2013 (64-bit)
- Netbeans IDE 8.2
- JDK 8u144 (64-bit)
- Oracle GlassFish Server
- Other packages used
  - mysql.jdbc 5.1.23
  - servlet-api

## 2.2 Hardware Requirements

**Hardware requirements** refer to the hardware parts which are required to run the project on a machine. The project doesn't have many restrictions on the requirements and is flexible to run on any machines, preferable a x64 Architecture processor.

The following are the specifications on which the project was built and deployed

- Intel ® Core™ i5-5200 CPU @2.20GHz
- RAM 4 GB
- Peripherals: Keyboard and Mouse
- Display Unit

## 2.3 Functional Requirements

A **functional requirement** defines a function of a system and its components. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. The functional requirement used in the project is:

- Keyboard function

## 2.4 Non-functional Requirements

A **non-functional requirement** is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. Non-functional requirements define how a system is supposed to be. It essentially specifies how the system should behave and that it is a constraint upon the systems behavior.

**Reliability** describes the ability of a system or component to function under stated conditions for a specified period of time. Reliability is defined as the probability of success as the frequency of failures.

**Maintainability** is the ease with which a product can be maintained in order to isolate defects, correct defects or their cause, maximize a product's useful life, maximize efficiency, reliability, and safety.

**Availability** is the probability that a system, at a point in time, will be operational and able to deliver the requested services. It is typically measured as a factor of its reliability - as reliability increases, so does availability.

## CHAPTER 3

# DESIGN

### 3.1 Front-end Design

Frontend Design is the presentation of the project it is the visible part of a project and is necessary to make project simple, beautify, functionality abstraction, display retrieved data.

The following are the Elementals of this project

- HTML
- CSS
- JavaScript
- JSP

#### 3.1.1 Hypertext Markup Language (HTML)

HTML is the standard markup language for creating web pages and web applications. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

#### 3.1.2 Cascading Style Sheets (CSS)

It is a style sheet language used for describing the presentation of a document written in a markup language.<sup>[1]</sup> Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML. CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts.<sup>[3]</sup> This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

#### 3.1.3 JavaScript

It is a high-level, dynamic, weakly typed, prototype-based, multi-paradigm, and interpreted programming language. Alongside HTML and CSS, JavaScript is one of the three core technologies of World Wide Web content production. It is used to make webpages interactive and provide online programs, including video games. The majority of websites employ it, and all modern web browsers support it without the need for plug-ins by means of a built-in JavaScript engine.

### **3.1.4 JavaServer Pages (JSP)**

It is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. Released in 1999 by Sun Microsystems,<sup>[1]</sup> JSP is similar to PHP and ASP, but it uses the Java programming language.

To deploy and run JavaServer Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required. JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, such as HTML, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, contain Java bytecode rather than machine code. Like any other Java program, they must be executed within a Java virtual machine (JVM) that interacts with the server's host operating system to provide an abstract, platform-neutral environment.

## 3.2 Back-end Design

Backend design in this project consists of database design, schema design and ER diagram

### 3.2.1 Schema of the database:

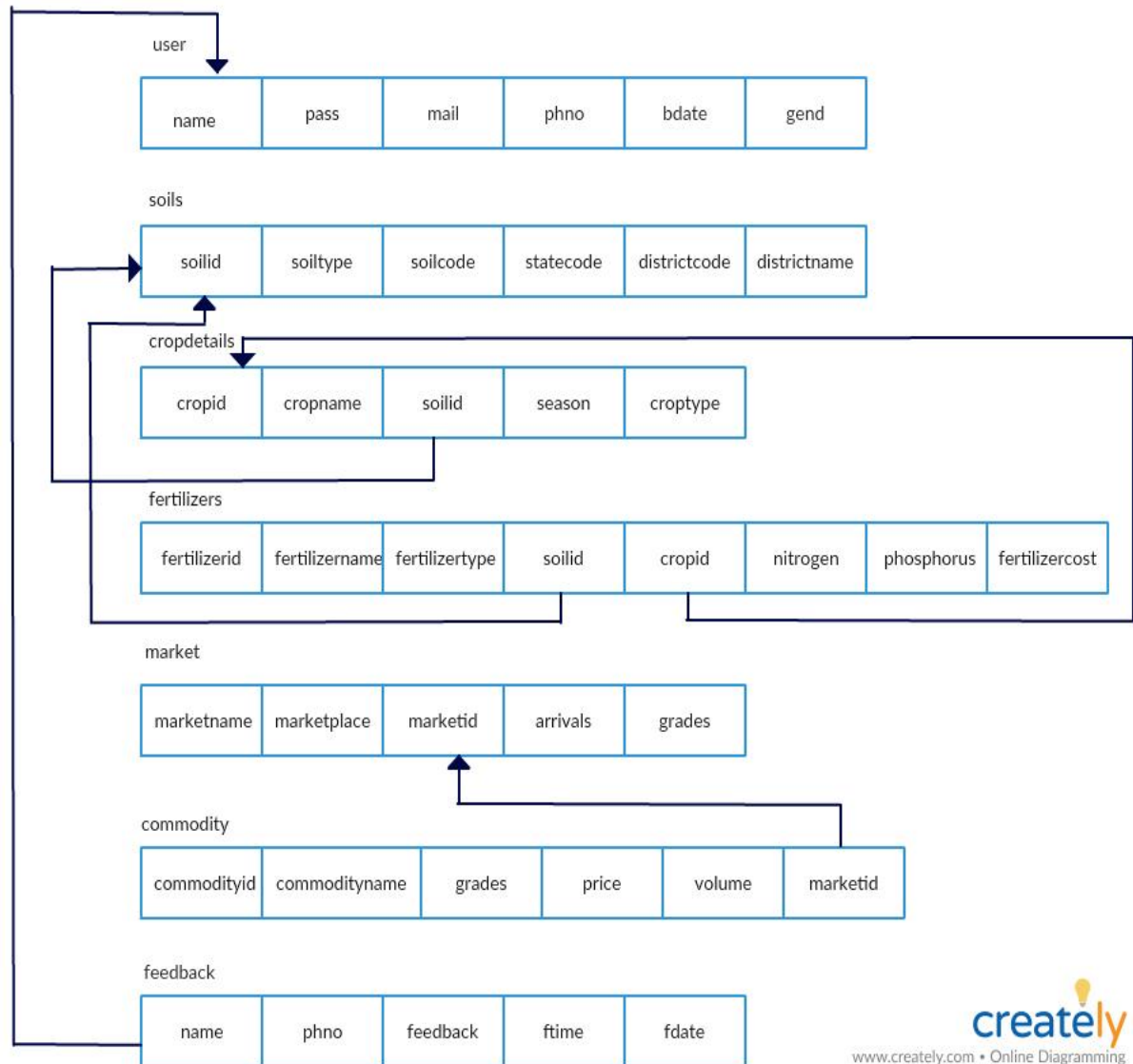


Fig 3.2.1: Schema Diagram



### 3.2.2 ER Diagram of the database:

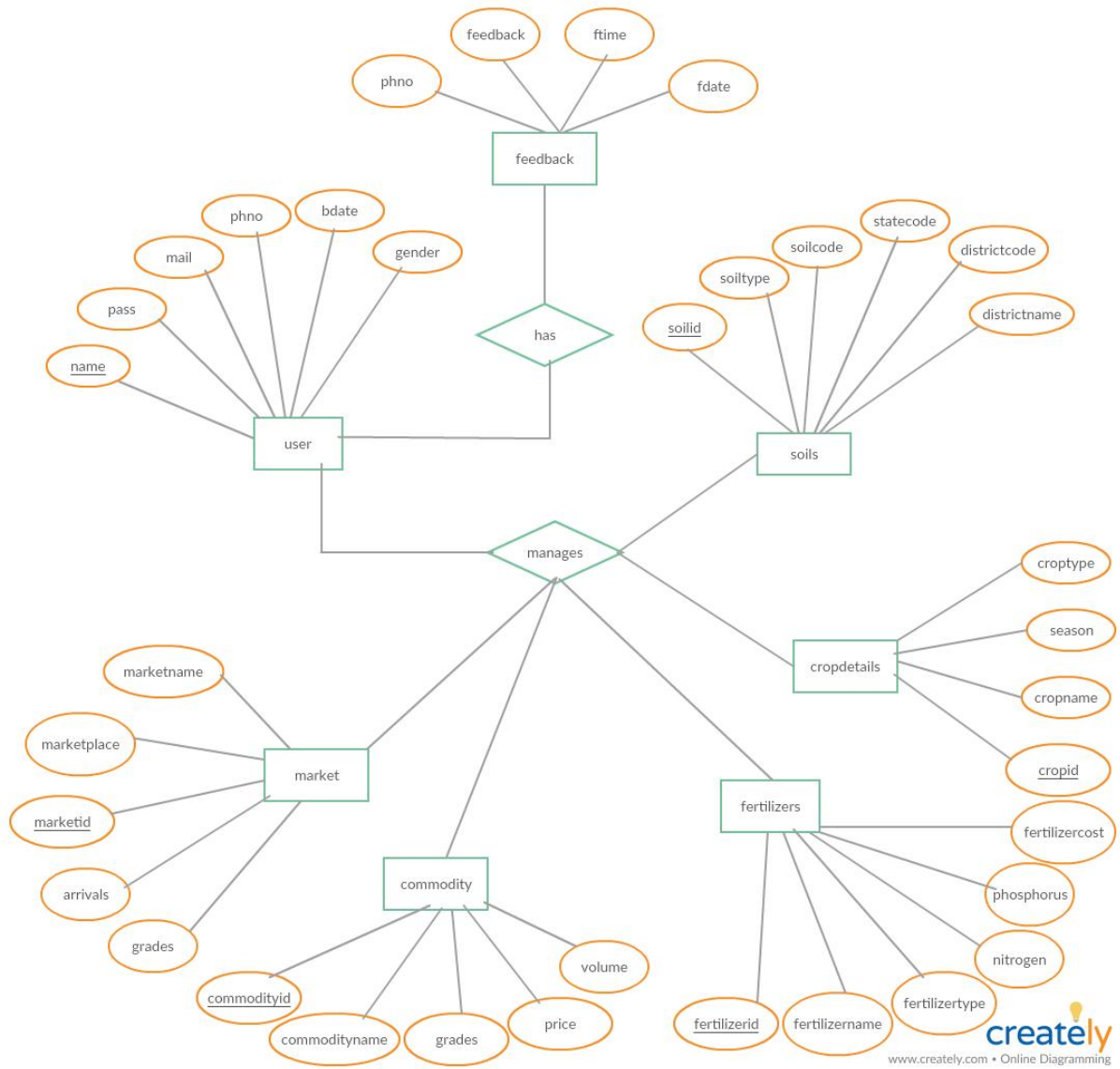
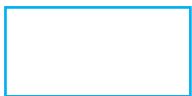


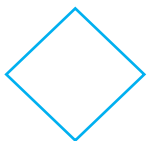
Fig 3.2.2: Entity-Relationship Diagram



: **Attribute** which is used to signify the tuples



: **Entity** which is used to signify the tables



: **Relation** which is used to signify the relation between two entities.

### 3.2.3 EER Diagram of the database:

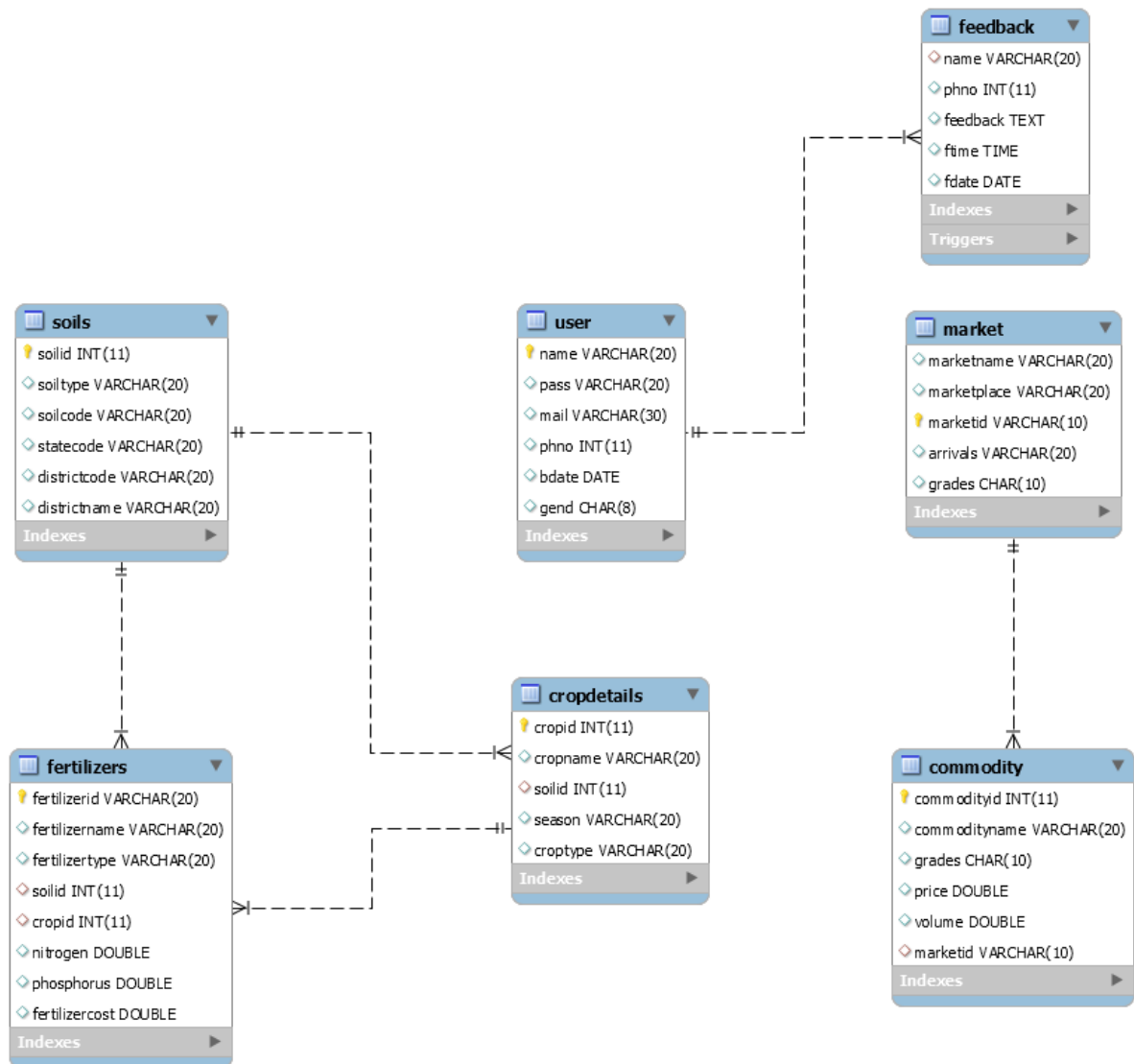


Fig 3.2.3: EER Diagram

## CHAPTER 4

# IMPLEMENTATION

### 4.1 Back-end Implementation

This phase involves creation of database, tables, triggers, stored procedures. The following are the commands used for backend implementation.

```
CREATE SCHEMA `agriculture` ;           -- used to create database
```

```
CREATE TABLE `user` (  
  `name` VARCHAR(20) DEFAULT NULL,  
  `pass` VARCHAR(20) DEFAULT NULL,  
  `mail` VARCHAR(30) DEFAULT NULL,  
  `phno` INT(11) NOT NULL,  
  `bdate` DATE DEFAULT NULL,  
  `gend` VARCHAR(10) DEFAULT NULL,  
  PRIMARY KEY (`phno`)  
)                                     --user table
```

```
CREATE TABLE `soils` (  
  `soilid` int(11) NOT NULL,  
  `soiltype` varchar(20) DEFAULT NULL,  
  `soilcode` varchar(20) DEFAULT NULL,  
  `statecode` varchar(20) DEFAULT NULL,  
  `districtcode` varchar(20) DEFAULT NULL,  
  `districtname` varchar(20) DEFAULT NULL,  
  PRIMARY KEY (`soilid`)  
)                                     --soils table
```

```
CREATE TABLE `cropdetails` (  
  `cropid` int(11) NOT NULL,  
  `cropname` varchar(20) DEFAULT NULL,  
  `soilid` int(11) DEFAULT NULL,  
  `season` varchar(20) DEFAULT NULL,  
  `croptype` varchar(20) DEFAULT NULL,  
  PRIMARY KEY (`cropid`),  
  KEY `soilid` (`soilid`),  
  CONSTRAINT `cropdetails_ibfk_1` FOREIGN KEY (`soilid`) REFERENCES `soils`  
  (`soilid`)  
)                                     --cropdetails table
```

```

CREATE TABLE `fertilizers` (
  `fertilizerid` varchar(20) NOT NULL,
  `fertilizername` varchar(20) DEFAULT NULL,
  `fertilizertype` varchar(20) DEFAULT NULL,
  `soilid` int(11) DEFAULT NULL,
  `cropid` int(11) DEFAULT NULL,
  `nitrogen` double DEFAULT NULL,
  `phosphorus` double DEFAULT NULL,
  `fertilizercost` double DEFAULT NULL,
  PRIMARY KEY (`fertilizerid`),
  KEY `soilid` (`soilid`),
  KEY `cropid` (`cropid`),
  CONSTRAINT `fertilizers_ibfk_1` FOREIGN KEY (`soilid`) REFERENCES `soils`
(`soilid`),
  CONSTRAINT `fertilizers_ibfk_2` FOREIGN KEY (`cropid`) REFERENCES
`cropdetails` (`cropid`)
)
--fertilizers table

```

```

CREATE TABLE `market` (
  `marketname` varchar(20) DEFAULT NULL,
  `marketplace` varchar(20) DEFAULT NULL,
  `marketid` varchar(10) NOT NULL,
  `arrivals` varchar(20) DEFAULT NULL,
  `grades` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`marketid`)
)
--market table

```

```

CREATE TABLE `commodity` (
  `commodityid` int(11) NOT NULL,
  `commodityname` varchar(20) DEFAULT NULL,
  `grades` varchar(20) DEFAULT NULL,
  `price` double DEFAULT NULL,
  `volume` double DEFAULT NULL,
  PRIMARY KEY (`commodityid`)
)
--commodity table

```

```

CREATE TABLE `feedback` (
  `name` varchar(20) DEFAULT NULL,
  `phno` int(11) NOT NULL,
  `feedback` text,
  `ftime` time DEFAULT NULL,
  PRIMARY KEY (`phno`)
)
--feedback table

```

```
-- Trigger ftime
DELIMITER ;;
CREATE trigger ftime before insert on feedback for each row set new.ftime=now();;
DELIMITER ;
```

```
-- Trigger fdate
DELIMITER ;;
CREATE trigger fdate before insert on feedback for each row set new.fdate=now();;
DELIMITER ;
```

```
-- Stored procedure
DELIMITER $$
USE `agriculture`;
DELIMITER ;;
CREATE DEFINER=`root`@`localhost` PROCEDURE `GetSoilInfo`()
BEGIN
SELECT * FROM soils;
END ;;
DELIMITER ;
```

## 4.2 Front-end Implementation

### 4.2.1 HTML

Below is a basic script to create a html page

```
<!DOCTYPE html>
<html>
<head>

<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

preview of the html script is

# This is a Heading

This is a paragraph

The following are some of the tags used in the project.

**<html>:** This tag is used to tell the browser that the following scripts between these tags are HTML type.

**<head>:** The <head> element is a container for all the head elements. The <head> element can include a title for the document, scripts, styles, meta information, and more.

The following elements can go inside the <head> element:

- <title> (this element is required in an HTML document)
- <style>
- <base>
- <link>
- <meta>
- <script>
- <noscript>

**<title>:** This tag is used to give name for the webpage and is placeholder in the web browser's tab.

**<Body>:** This tag acts as a container to hold the formatted texts and paragraphs which is to be displayed.

**<table>** : Defines a table.

**<thead>:** It signifies the table header.

**<th>:** It signifies a header cell in a table.

**<tr>:** It signifies the table row.

**<td>:** it defines cell of a table.

**<tbody>:** groups the body content in a table.

## 4.2.2 CSS

The css element in a html file can be used in three ways, internal, external, inline.

The project uses internal CSS where CSS elements are involved in the **<style>** tag which is enclosed in the **<head>** tag.

Below is an example of internal CSS being implemented.

```
<head>
<style>
.bg{
  width: 33%;
  height: 200px;
}

.navbar {
  overflow: hidden;
  background-color: #4CAF50;
  font-family: Arial;
}
</style>
</head>
```

## 4.2.3 JavaScript

Similar to CSS JavaScript also can be implemented in Three ways, and in the project the JavaScript is implemented using internal JavaScript.

The JS to be applied should be in the **<script>** tag and this tag can be used anywhere in the page, it is preferred to be placed at the bottom of the file.

## 4.3 Connectivity Implementation

The backend and the frontend have to be connected in order to store and retrieve the data in server and client side respectively.

JavaServer Pages(JSP) is used for both actions as it performs actions related to both server and client side. It is more like a tag based language than programming, hence one can say its tag based language which supports elements of java.

**Connecting to database:** mysql-database package is imported to the project and is imported to jsp file using `<%@ page import = "java.sql.*" %>`

The connection is established using the following lines. The server when launched uses the url which is mentioned, driver is imported from the package, user(root) and password(root) are the values of credentials of the database(agriculture).

```
Class.forName("com.mysql.jdbc.Driver"); //Register the JDBC Driver
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/agriculture",
"root", "root"); //Create a Connection
```

**Execute a Query:** This requires using an object of type statement or PreparedStatement for submitting an SQL statement to the database.

```
statement=connection.createStatement();
String sql ="SELECT * FROM soils";
resultSet = statement.executeQuery(sql);
```

The values retrieved from the database is stored in a variable and is to be formatted into a table format the following are the set of instructions.

```
<tr bgcolor="#FFB85F">
    <td><%=resultSet.getString("soilid") %></td>
    <td><%=resultSet.getString("soiltype") %></td>
    <td><%=resultSet.getString("soilcode") %></td>
    <td><%=resultSet.getString("statecode") %></td>
    <td><%=resultSet.getString("districtcode") %></td>
    <td><%=resultSet.getString("districtname") %></td>
</tr>
```



**Storing Database from the client side page:** The data to be stored in database is stored in variable.

```
String soilid = request.getParameter("soilid");
String soiltype = request.getParameter("soiltype");
String soilcode = request.getParameter("soilcode");
String statecode = request.getParameter("statecode");
String districtcode = request.getParameter("districtcode");
String districtname= request.getParameter("districtname");
```

The driver manager then connects to the database using credentials

```
Class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/agriculture",
"root", "root");
```

The data is then inserted into tables of the selected database in the instructions using the following instruction.

```
try{
Statement st = con.createStatement();
st.executeUpdate("insert                                into
soils(soilid,soiltype,soilcode,statecode,districtcode,districtname)values('"+
soilid+"','"+soiltype      + "','"+soilcode      + "','"+statecode      + "','"+districtcode      + "','"+
districtname+"')");
}
catch(Exception ex){
    out.println(ex.getMessage());
}
```

The try-catch statement is to check if the data was stored. The statement has condition such that if it was successful the attributes are set. If there is an error, then an error message is displayed.

The Data gets stored in the database in the server side. The data stored is connected to the server hence upon refresh of the webpage the client can see the new addition of data.

**Clean up the Environment:** You should explicitly close all the database resources relying upon the JVM's garbage collection as follows:

```
resultSet.close()
st.close()
con.close()
```

## **CHAPTER 5**

# **TESTING**

### **5.1 Testing**

Testing is the process of executing a program to find the errors. A good test has the high probability of finding a yet undiscovered error. A test is vital to the success of the system. System test makes a logical assumption that if all parts of the system are correct, then the goal will be successfully achieved.

### **5.2 Types of testing**

- Unit Testing
- Integration Testing
- System Testing

#### **5.2.1 Unit testing**

Here each pages are tested for functionality and format of the webpage individually, the test is done by considering all possible inputs and analyzed for expected output.

If the expected output matches with the test output the unit is said to have passed the test.

#### **5.2.2 Integration testing**

Each tested units are integrated and tested as a single unit and tested. Black box testing takes an external perspective of the test object to derive test cases. These tests can be functional or non-functional, though usually functional. The test designer selects valid and invalid inputs and determines the correct output. There is no knowledge of the test object's internal structure. This method of test design is applicable to all levels of software testing: unit, integration, functional testing, system and acceptance. The higher the level, and hence the bigger and more complex the box, the more one is forced to use black box testing to simplify. While this method can uncover unimplemented parts of the specification, one cannot be sure that all existent paths are tested. Bottom-Up Testing is an approach to integration testing where the lowest level components are tested first, then used to facilitate the testing of higher level components. The process is repeated until the component at the top of the hierarchy is tested. This project has been tested for its working and is found the requirements as mentioned.

### **5.2.3 System Testing**

The different components are integrated to make up the entire system. The testing process is concerned with finding errors, which result from unanticipated interactions between the different components. It is also concerned with validating that the system meets its functional and non – functional requirements.

This testing procedure was carried out iteratively with addition of every new component. The initial component was the creation of characters

## 5.3 SNAPSHOTS

Snapshots are pictorials of the working project on paper as a way to report, log, and brief about some phase which is captured.

### 5.3.1 Signup and Login

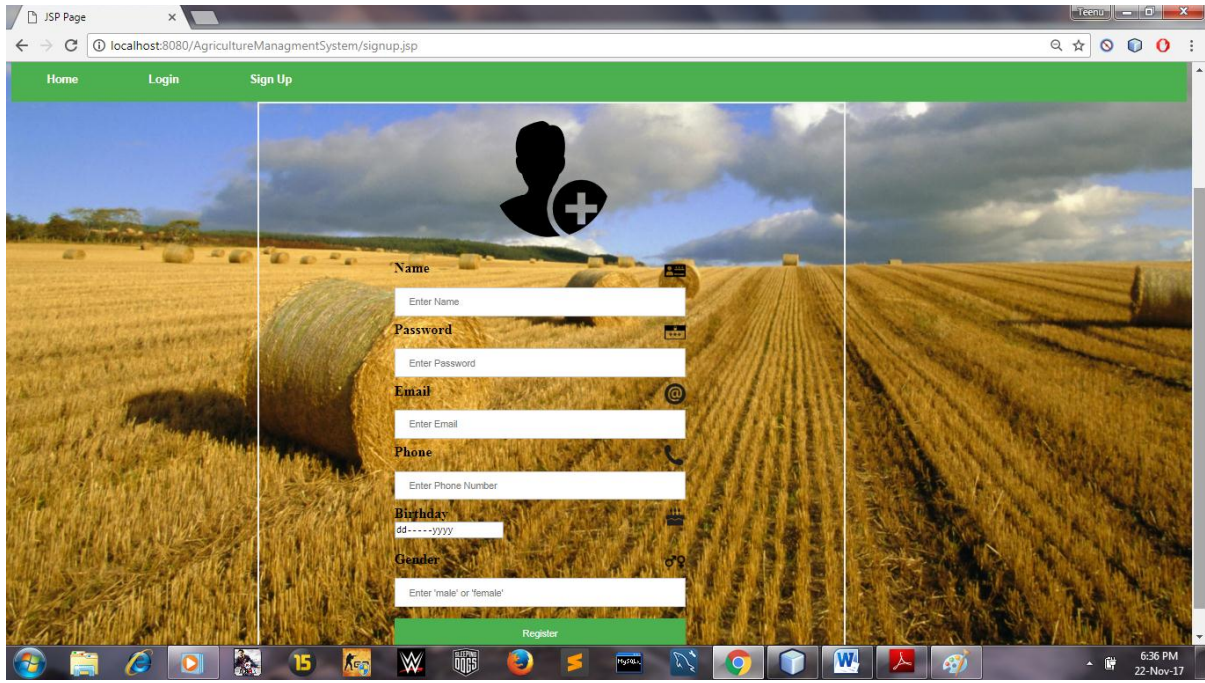


Fig 5.3.1(a): The Signup page

A new user can register with unique a username, password, email, phone number, birthday and gender.

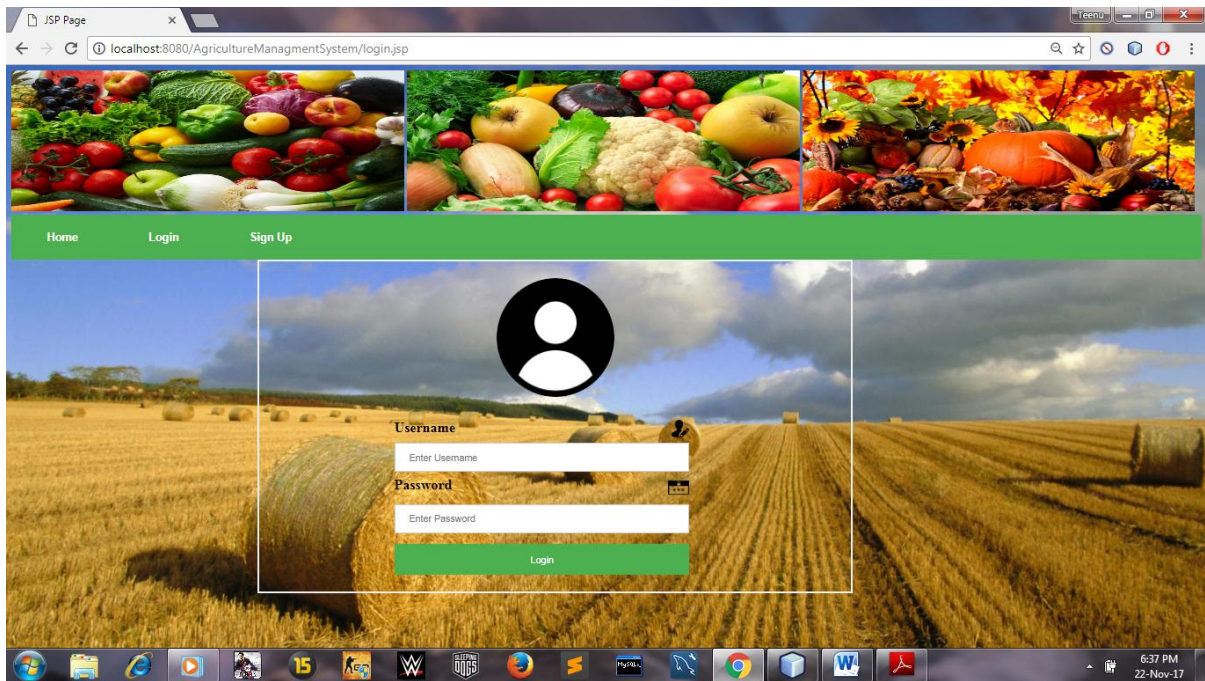


Fig 5.3.1(b): The Login Page

A previously registered user can login through this page with valid username and password.

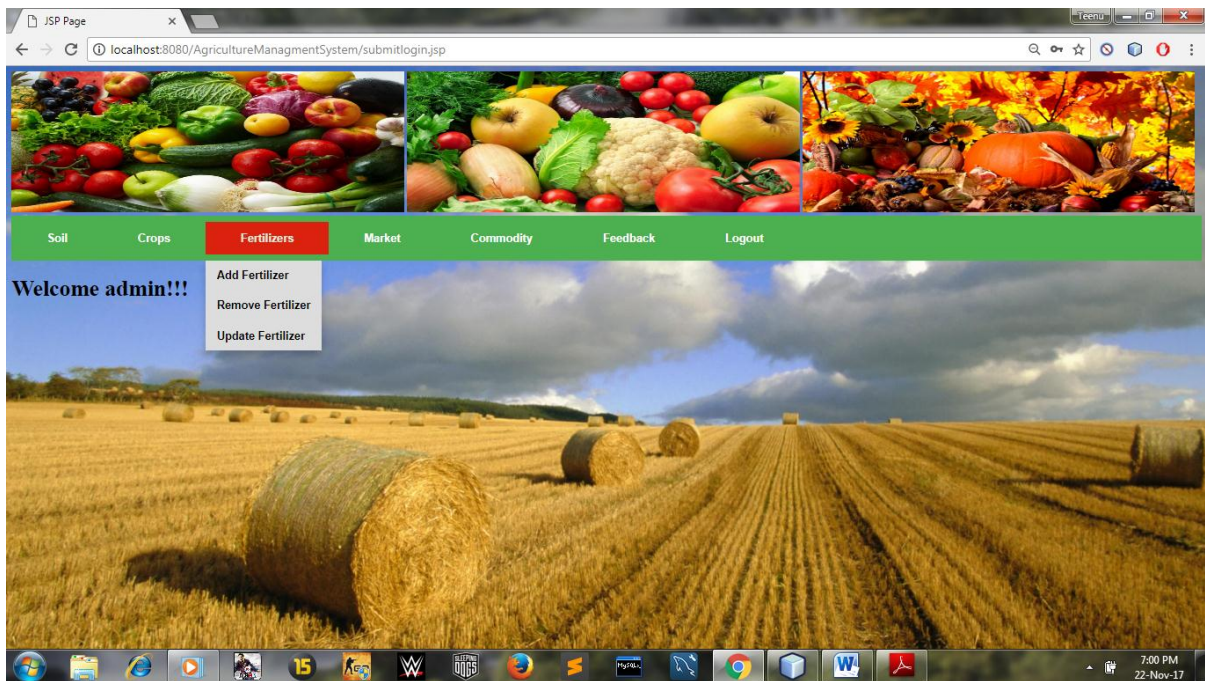


Fig 5.3.1(c): After Login Navigation

The user(admin) is redirected to this page on successful login.



### 5.3.2 Add Soil / Remove Soil / Update Soil

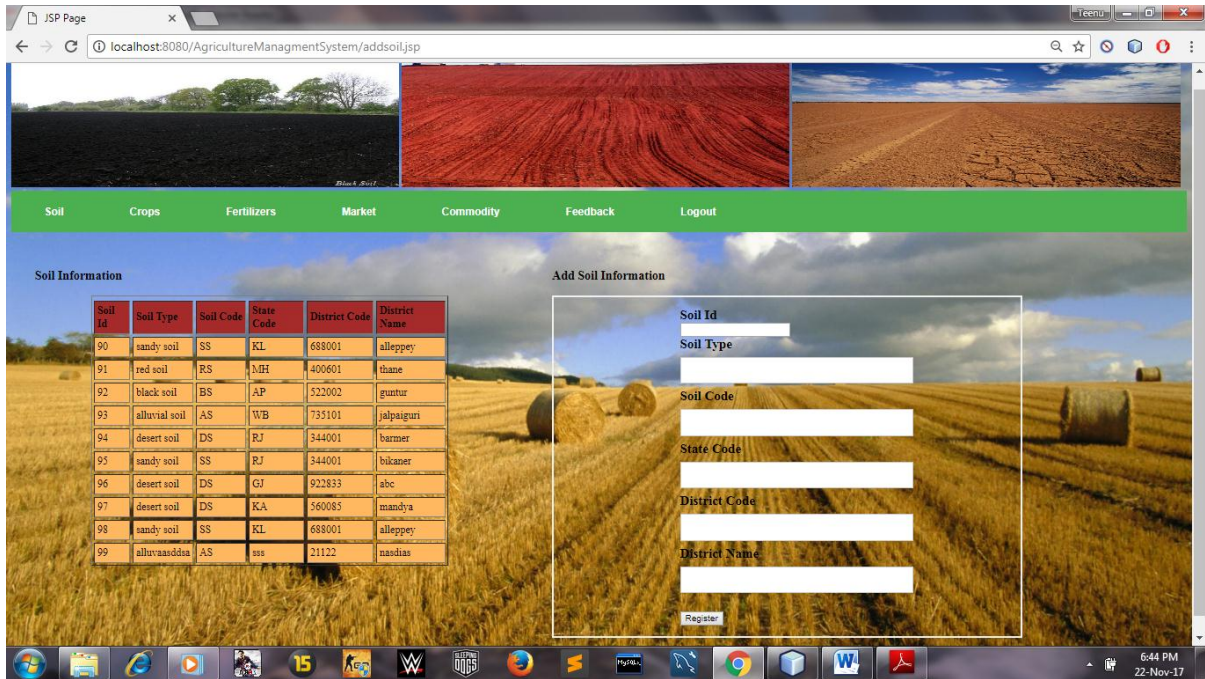


Fig 5.3.2(a): Add Soil

Soil information like soilid,soiltype,soilcode etc can be added in this page

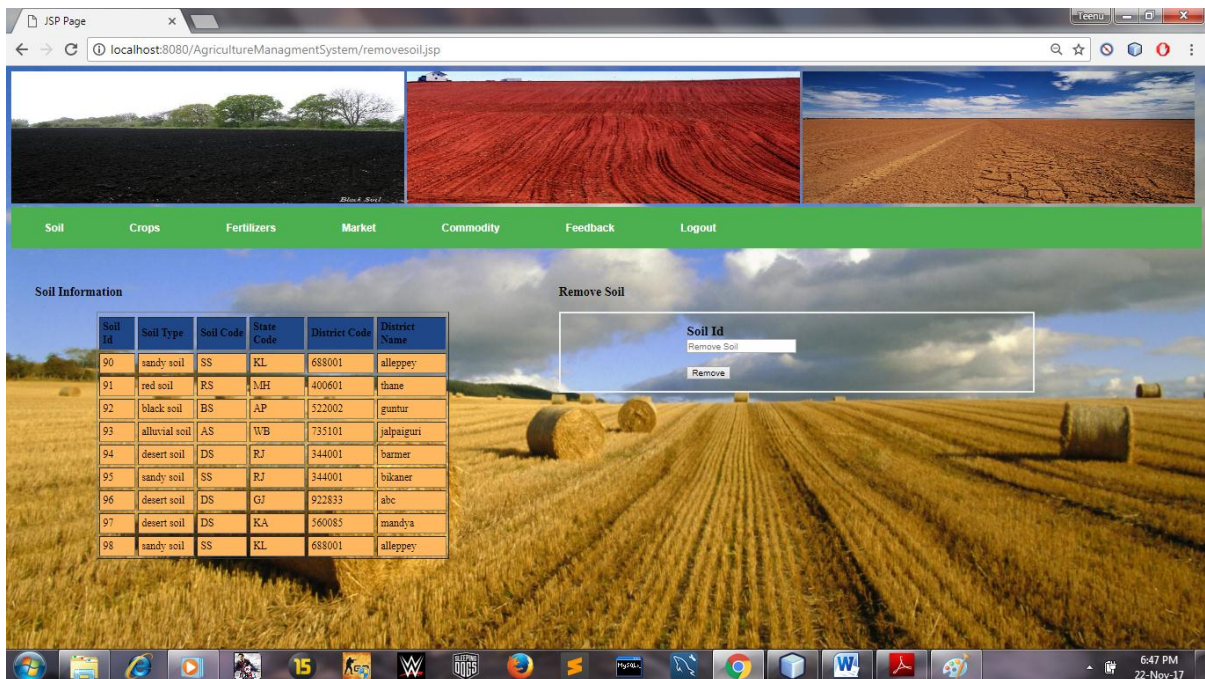


Fig 5.3.2(b): Remove Soil

Soil information can be removed in this page by inputting the soilid



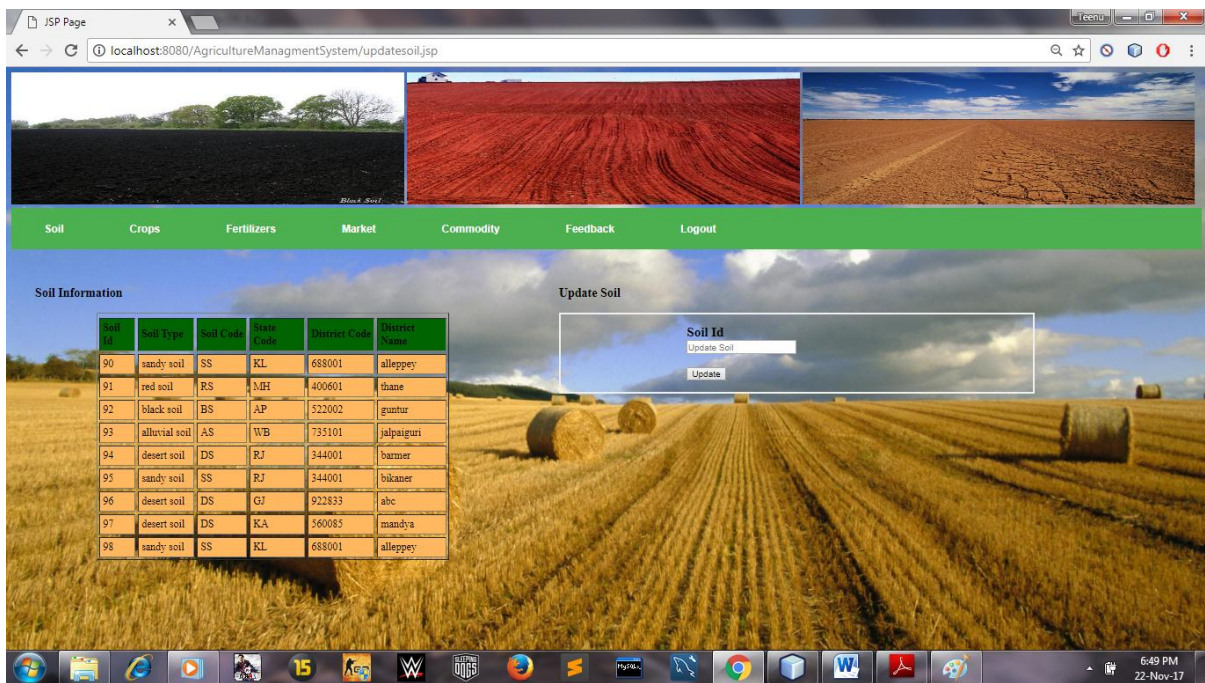


Fig 5.3.2(c): Update Soil

Soil information can be updated in this page by inputting the soilid

### 5.3.3 Add Crop/ Add Fertilizer/ Add Market/ Add Commodity

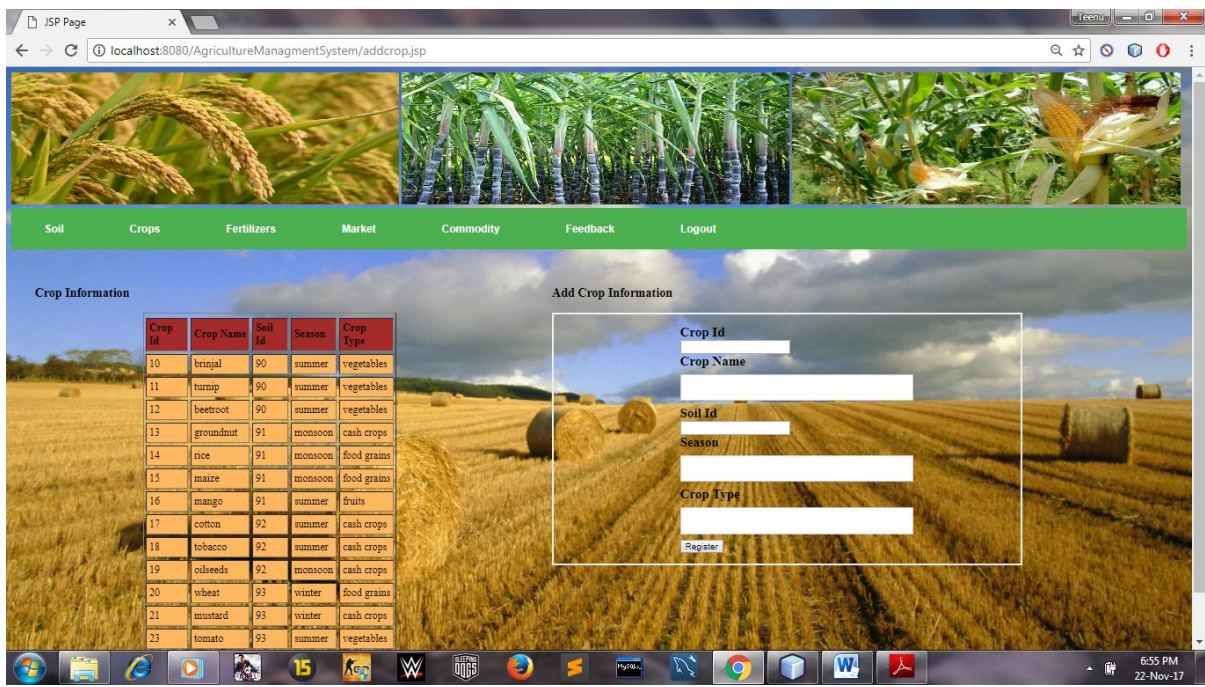


Fig 5.3.3(a): Add Crop

Crop information like cropid, cropname etc can be added in this page



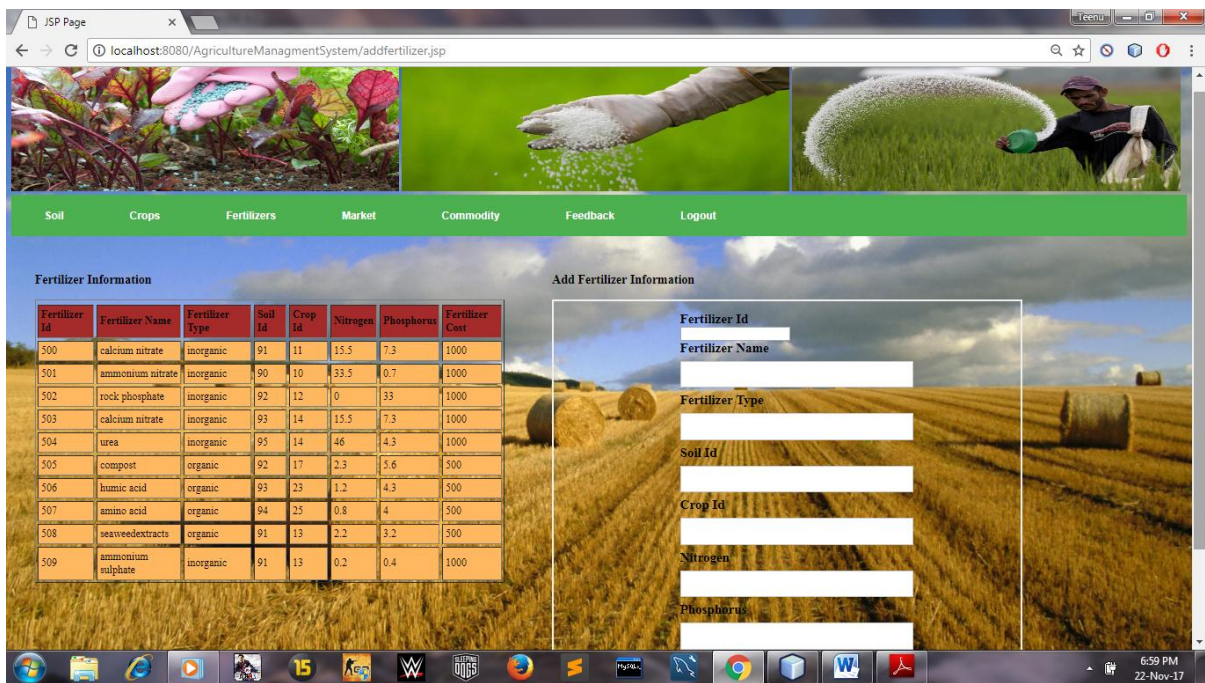


Fig 5.3.3(b): Add Fertilizer

Fertilizer information like fertilizerid,fertilizername,fertilizertype etc can be added in this page

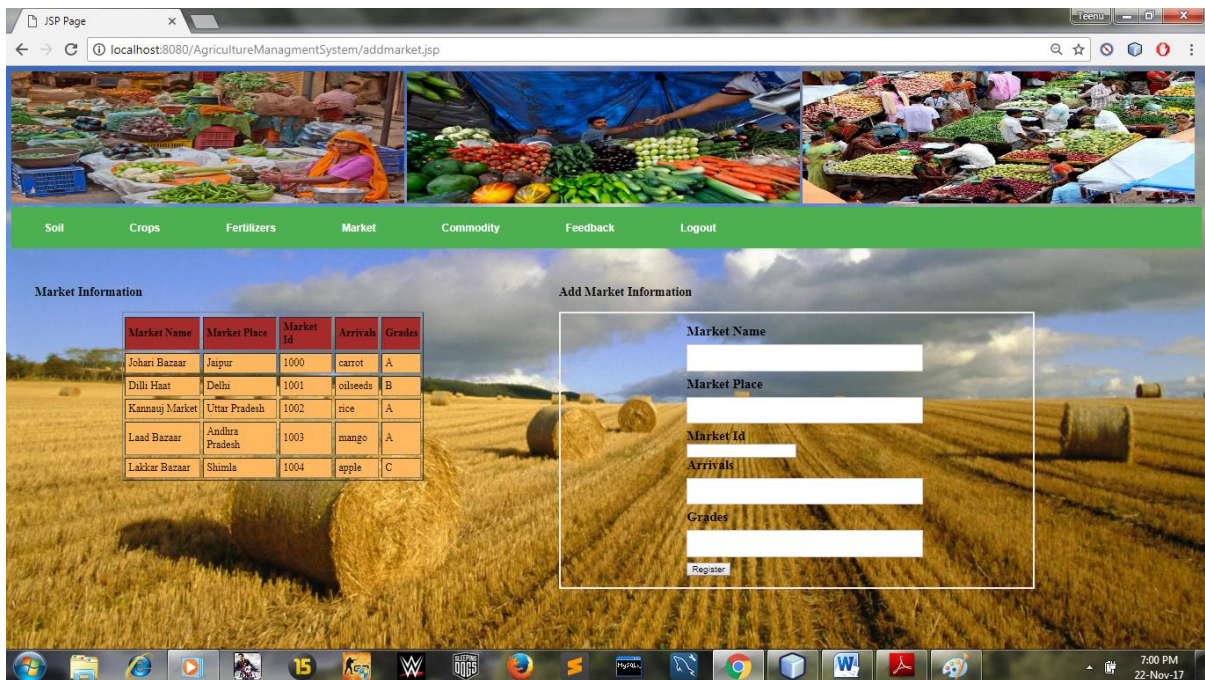


Fig 5.3.3(c): Add Market

Market information like marketname,marketplace,marketid etc can be added in this page.



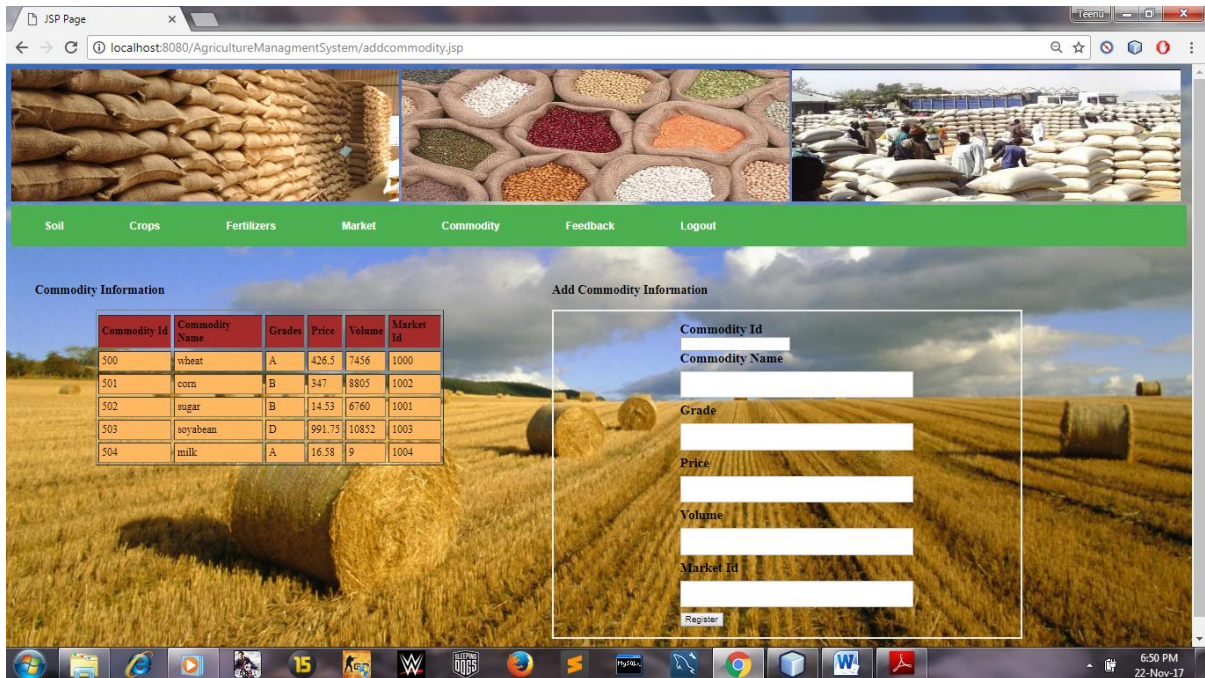


Fig 5.3.3(d): Add Commodity

Commodity information like Commodityname,Grade,Price etc can be added in this page.

### 5.3.4 Send Feedback / View Feedback

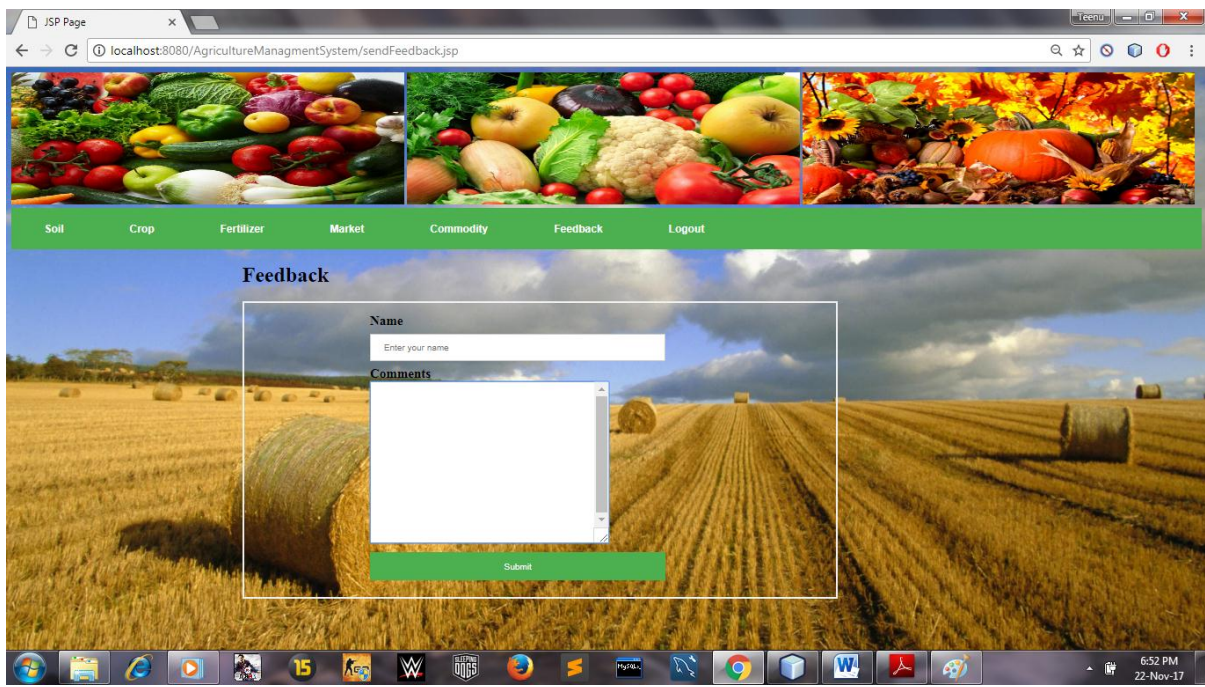


Fig 5.3.4(a): Send Feedback

Other users can send feedback to the administrator of the system.



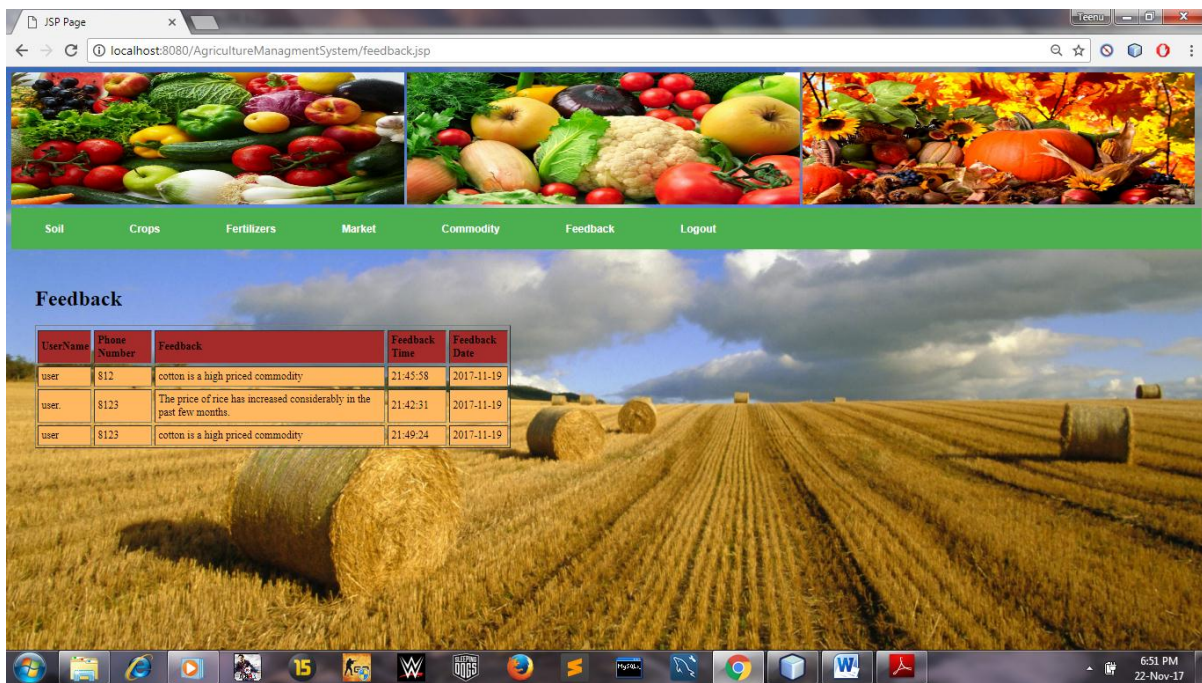


Fig 5.3.4(b): View Feedback

Administrator of the system can view the feedback sent by other users.

### 5.3.5 The Home Page

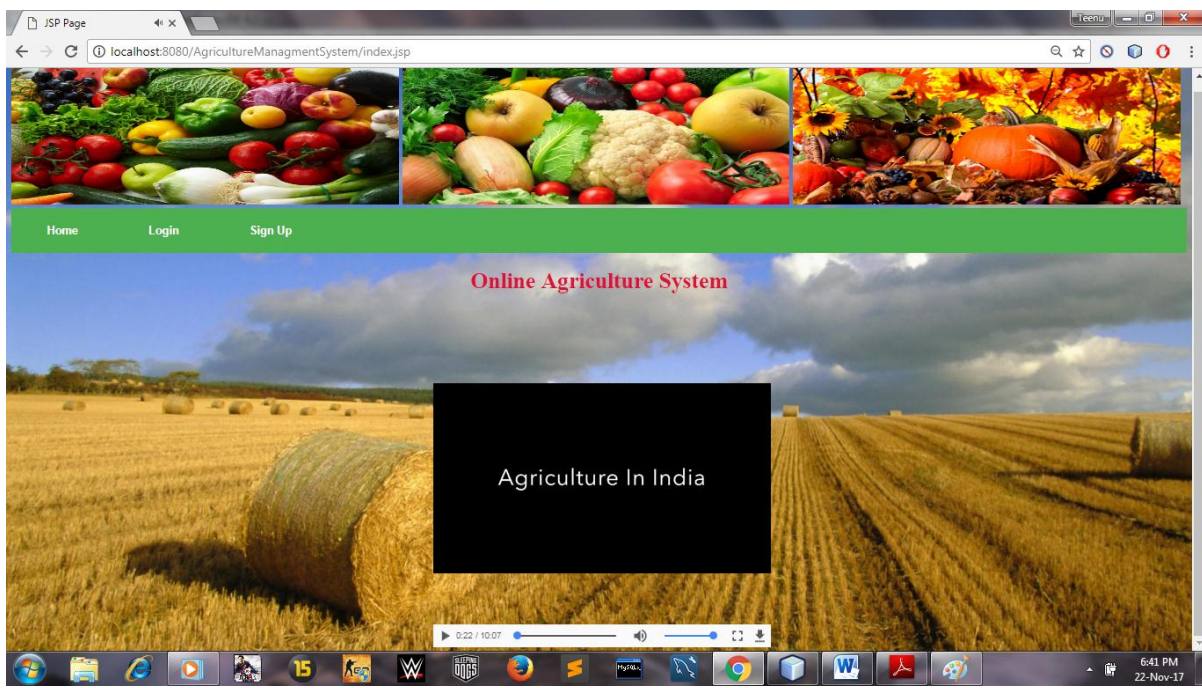
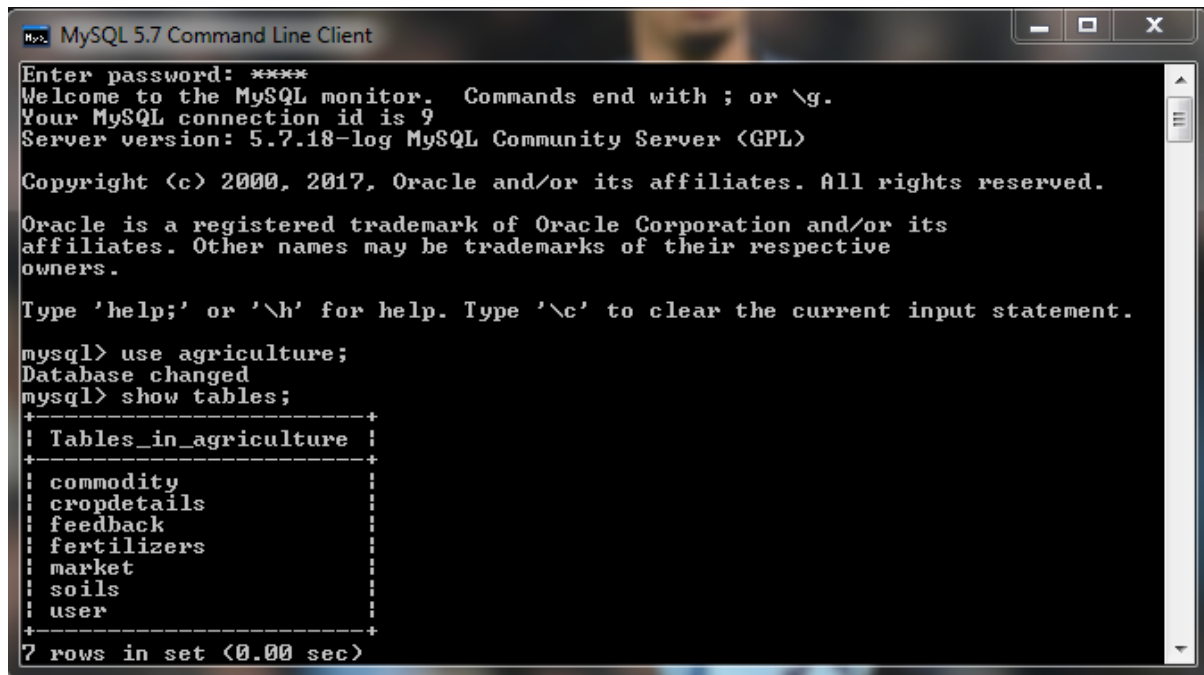


Fig 5.3.5: The Home Page

After logout the user is redirected to the home page.

### 5.3.6 Database Snapshots

Fig 5.3.6(a) shows tables in agriculture database.



```
MySQL 5.7 Command Line Client
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 5.7.18-log MySQL Community Server (GPL)

Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.

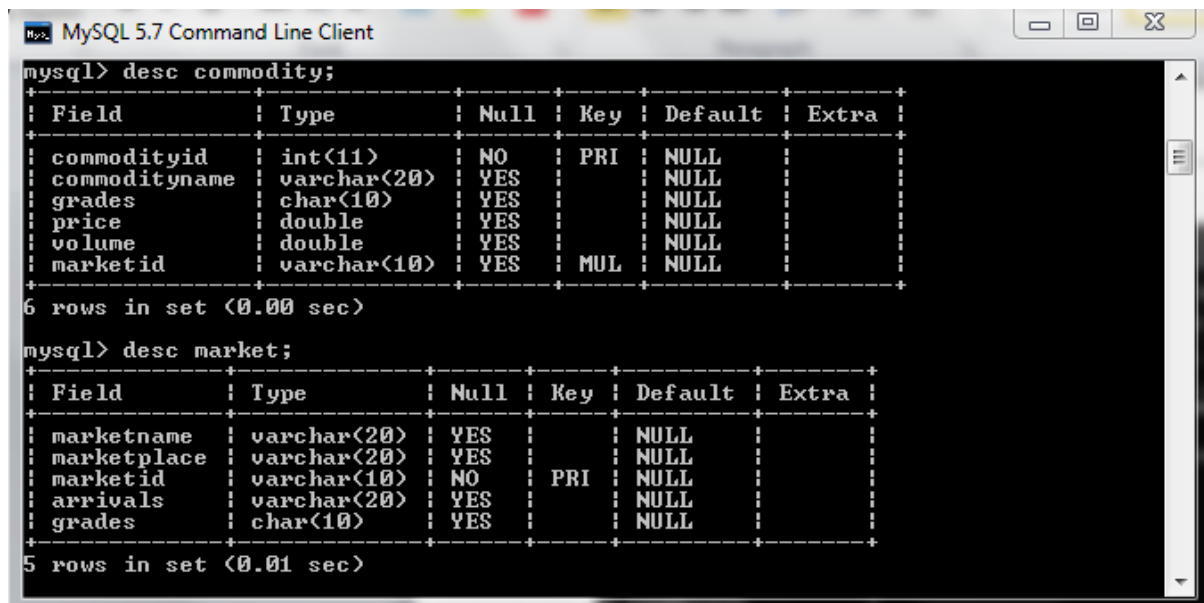
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use agriculture;
Database changed
mysql> show tables;
+-----+
| Tables_in_agriculture |
+-----+
| commodity              |
| cropdetails            |
| feedback               |
| fertilizers            |
| market                 |
| soils                  |
| user                   |
+-----+
7 rows in set (0.00 sec)
```

Fig 5.3.6(a)

Fig 5.3.6(b) shows description of commodity and market tables.



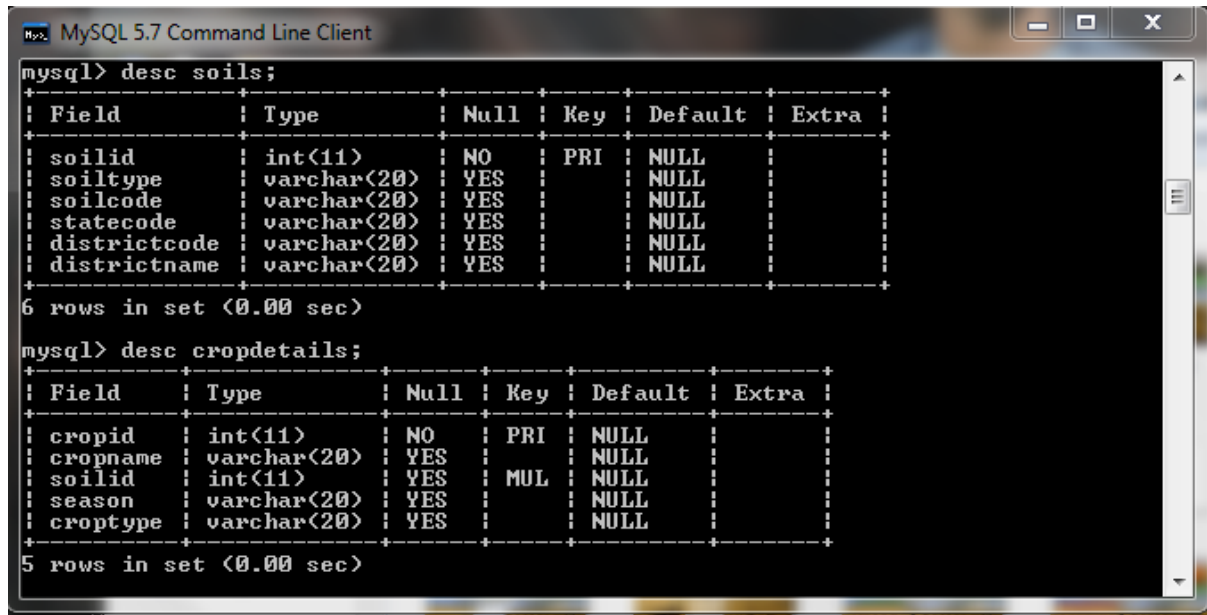
```
MySQL 5.7 Command Line Client

mysql> desc commodity;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| commodityid    | int(11)       | NO   | PRI | NULL    |       |
| commodityname  | varchar(20)   | YES  |     | NULL    |       |
| grades         | char(10)      | YES  |     | NULL    |       |
| price          | double        | YES  |     | NULL    |       |
| volume         | double        | YES  |     | NULL    |       |
| marketid      | varchar(10)   | YES  | MUL | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)

mysql> desc market;
+-----+-----+-----+-----+-----+-----+
| Field          | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| marketname     | varchar(20)   | YES  |     | NULL    |       |
| marketplace    | varchar(20)   | YES  |     | NULL    |       |
| marketid      | varchar(10)   | NO   | PRI | NULL    |       |
| arrivals       | varchar(20)   | YES  |     | NULL    |       |
| grades         | char(10)      | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)
```

Fig 5.3.6(b)

Fig 5.3.6(c) shows description of soils and cropdetails tables.



```
mysql> desc soils;
```

Field	Type	Null	Key	Default	Extra
soilid	int(11)	NO	PRI	NULL	
soiltype	varchar(20)	YES		NULL	
soilcode	varchar(20)	YES		NULL	
statecode	varchar(20)	YES		NULL	
districtcode	varchar(20)	YES		NULL	
districtname	varchar(20)	YES		NULL	

6 rows in set (0.00 sec)

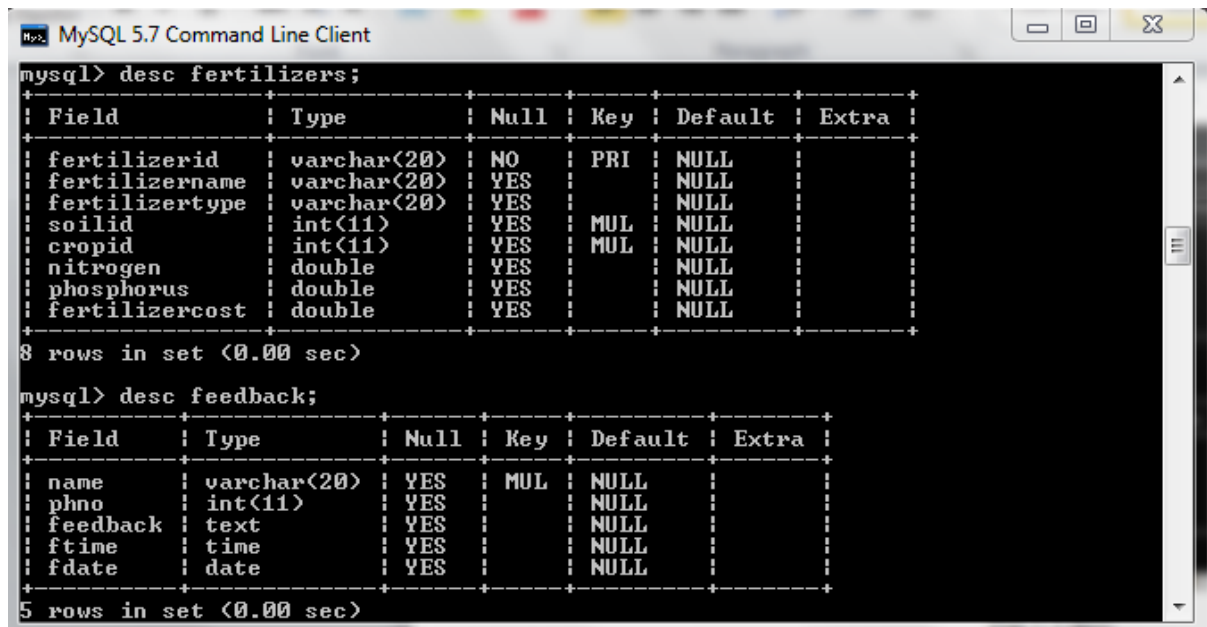
```
mysql> desc cropdetails;
```

Field	Type	Null	Key	Default	Extra
cropid	int(11)	NO	PRI	NULL	
cropname	varchar(20)	YES		NULL	
soilid	int(11)	YES	MUL	NULL	
season	varchar(20)	YES		NULL	
croptype	varchar(20)	YES		NULL	

5 rows in set (0.00 sec)

Fig 5.3.6(c)

Fig 5.3.6(d) shows description of fertilizers and feedback tables.



```
mysql> desc fertilizers;
```

Field	Type	Null	Key	Default	Extra
fertilizerid	varchar(20)	NO	PRI	NULL	
fertilizername	varchar(20)	YES		NULL	
fertilizertype	varchar(20)	YES		NULL	
soilid	int(11)	YES	MUL	NULL	
cropid	int(11)	YES	MUL	NULL	
nitrogen	double	YES		NULL	
phosphorus	double	YES		NULL	
fertilizercost	double	YES		NULL	

8 rows in set (0.00 sec)

```
mysql> desc feedback;
```

Field	Type	Null	Key	Default	Extra
name	varchar(20)	YES	MUL	NULL	
phno	int(11)	YES		NULL	
feedback	text	YES		NULL	
fetime	time	YES		NULL	
fdate	date	YES		NULL	

5 rows in set (0.00 sec)

Fig 5.3.6(d)

## CONCLUSION

A conclusion can be drawn from the project that database management system along with client server interactions are simple to design and has wide range of applications. This kind of application can be used by any company which can afford to buy a web hosting service which will be really helpful for the company in establishing a standard, advertising, company portfolio, deals online which will improve its market.

The combination of client-server along with DBMS is a powerful yet simple which will save lot of time and expenses by reducing work on book keeping and employees required to maintain them as most of the database management systems are open source and reliable which won't be needing much maintenance and has many utilities that accompany the software which will ease the functionalities.

It also is a good exposure to learners who are designers as they get exposure to both scripting and programming side of the project, which also will help them improve approaches to the next project.

## **FUTURE ENHANCEMENTS**

The currently developed project can improve in many areas. Time constraint and deficient knowledge has been a major factor in limiting the features offered by this mini-project.

The following features can be implemented to enhance the project.

- Adding authentication to the login session.
- Report and spamming System for users.
- Social interactions between the users.
- Profile picture to the users.
- Personal description of users.
- Uploading multimedia feature for users
- Opening discussion threads for users.

# REFERENCES

The following books have been referred to complete this project.

Herbert Schildt: JAVA the Complete Reference, 7th/9th Edition, Tata McGraw Hill, 2007.

Jim Keogh: J2EE-TheCompleteReference, McGraw Hill, 2007.

Database systems Models, Languages, Design and Application Programming, RamezElmasri and Shamkant B. Navathe, 7th Edition, 2017, Pearson.

Database management systems, Ramakrishnan, and Gehrke, 3rd Edition, 2014, McGraw Hill.

The following Websites were referred to complete the project.

[www.w3schools.com](http://www.w3schools.com)

[www.wikipedia.org](http://www.wikipedia.org)

[www.javatpoint.com](http://www.javatpoint.com)

[www.creately.com](http://www.creately.com)