

A
Minor Project Report
on
**DEVELOPMENT OF MIDDLEWARE FOR
REMOTE DATABASE**

Submitted in Partial Fulfillment of
the Requirements for the Third Year
of
Bachelor of Engineering
in
Computer Engineering
to
North Maharashtra University, Jalgaon

Submitted by
Puja Anil Naval
Tanaya Gajanan Marathe
Neha Balvant Patil
Ashwini Sudhir Patil

Under the Guidance of
Mr.Sandip S.Patil



DEPARTMENT OF COMPUTER ENGINEERING
SSBT's COLLEGE OF ENGINEERING AND TECHNOLOGY,
BAMBHORI, JALGAON - 425 001 (MS)
2015 - 2016

**SSBT's COLLEGE OF ENGINEERING AND TECHNOLOGY,
BAMBHORI, JALGAON - 425 001 (MS)
DEPARTMENT OF COMPUTER ENGINEERING**

CERTIFICATE

This is to certify that the minor project entitled *Development of Middleware for Remote Database*, submitted by

**Puja Anil Naval
Tanaya Gajanan Marathe
Neha Balvant Patil
Ashwini Sudhir Patil**

in partial fulfillment of the Third Year of *Bachelor of Engineering in Computer Engineering* has been satisfactorily carried out under my guidance as per the requirement of North Maharashtra University, Jalgaon.

Date: April 4, 2016

Place: Jalgaon

Mr.Sandip S.Patil
Guide

Prof. Dr.Girish K. Patnaik
Head

Prof. Dr. K. S. Wani
Principal

Acknowledgements

No work can be accomplished unless it has evolved as a result of co-operating, assistance and understanding of some knowledgeable group of people. We take the opportunity to thank our Principal **Prof.Dr.K.S.Wani** and Head of Department **Prof. Dr.Girish K. Patnaik** for providing all the necessary facilities, which were indispensable in the completion of seminar. We would like to thank my guide **Mr.Sandip S. Patil** for providing to be a great help by giving us guidance through their vast experience and intellectual skills. We are also thankful to all the staff members of the Computer Engineering Department. We would also like to thank the college for providing the required magazines, books and access to the internet for collecting information related to the project. Finally, we would like to thank my parents.

Puja Anil Naval

Tanaya Gajanan Marathe

Neha Balvant Patil

Ashwini Sudhir Patil

Contents

Acknowledgements	ii
Abstract	1
1 Introduction	2
1.1 Background	2
1.2 Motivation	2
1.3 Problem Definition	3
1.4 Scope	3
1.5 Objective	3
1.6 Organization of Report	3
1.7 Summary	4
2 System Analysis	5
2.1 Literature Survey	5
2.2 Proposed System	5
2.3 Feasibility Study	6
2.3.1 Technical Feasibility	6
2.3.2 Operational Feasibility	6
2.3.3 Economical Feasibility	7
2.4 Risk Analysis	7
2.4.1 Risk	7
2.4.2 Need of Risk Analysis	7
2.4.3 Software Risk	7
2.4.4 Project Risk	8
2.4.5 Technical Risks	8
2.4.6 Business Risk	8
2.5 Project Scheduling	9
2.6 Effort Allocation	9
2.7 Summary	10

3	Requirement Analysis	11
3.1	Hardware Requirements	11
3.2	Software Requirements	11
3.3	Functional Requirements	11
3.4	Non-Functional Requirements	12
3.5	Other Requirement and Constraints	12
3.6	Summary	13
4	System Design	14
4.1	System Architecture	14
4.2	E-R diagram	15
4.3	Database Design	15
4.4	Data Flow Diagram	16
4.5	UML Diagrams	17
4.5.1	Use Case Diagram	18
4.5.2	Class Diagram	19
4.5.3	Sequence Diagram	20
4.5.4	Activity Diagram	21
4.5.5	Component Diagram	22
4.5.6	Deployment Diagram	23
4.5.7	State Diagram	24
4.5.8	Collaboration Diagram	25
4.6	Summary	26
5	Implementation	27
5.1	Coding	27
5.2	Implementation Environment	40
5.2.1	Front-End-: Java (programming language)	40
5.2.2	Back End Database-: MySQL Database	41
5.3	Summary	41
6	System Testing	42
6.1	How to Implement Testing	42
6.1.1	Software Testing	42
6.1.2	Steps of Software Testing	42
6.1.3	Software Testing Objectives	43
6.1.4	Testing Principles	43
6.1.5	Manual Testing	43

6.1.6	Steps of Manual Software Testing	43
6.2	Test Cases and Test Results	43
6.3	Summary	45
7	Result and Analysis	46
7.1	Sample Snapshot of Important Processing and Its Explanation.	46
8	Conclusion and Future Scope	50
8.1	Conclusion	50
8.2	Future Scope	50
	Bibliography	51

List of Figures

2.1	Gantt Chart	9
2.2	Chart of Effort Allocation	10
4.1	System Architecture for Farmer System.	14
4.2	E-R Diagram for Farmer System.	15
4.3	Annual Details Schema of Farmer.	15
4.4	Personal Details Schema of Farmer.	16
4.5	Field Details Schema of Farmer.	16
4.6	Level 0 DFD	17
4.7	Level 1 DFD	17
4.8	Use Case Diagram for Farmer System.	18
4.9	Class Diagram for Farmer System.	19
4.10	Sequence Diagram for Farmer System.	20
4.11	Activity Diagram for Farmer System.	21
4.12	Component Diagram for Farmer System.	22
4.13	Deployment Diagram for Farmer System.	23
4.14	state Diagram for Farmer System.	24
4.15	Collaboration Diagram for Farmer System.	25
6.1	Test Case for ID	44
6.2	Test Case for Name	44
6.3	Test Case for Mobile No.	45
7.1	Home Form	46
7.2	Login Form	47
7.3	Accessing Records Form	47
7.4	Personal Information of Farmer	48
7.5	Field Details of Farmer	48
7.6	Annual Details of Farmer	49
7.7	Query Form	49

7.8	Jasper Report	49
-----	-------------------------	----

Abstract

Most of the organizations have a large variety of heterogeneous hardware systems which include personal computers, workstations, minicomputers and mainframes. All these heterogeneous systems use different OS and network architecture. So integration of these systems is difficult. Here comes the middleware in the picture. Middleware deals with providing environments for developing systems that can be distributed effectively over a variety of topologies, computing devices and communication network. It aims to provide developers of networked applications with the required platform. It is the glue that connects diverse computer systems. It is sometimes called plumbing because it connects to applications and passes data between them. It allows data contained in one database to be accessed through another. It is usually utilized within a software application. Middleware services provide a more functional set of application programming interfaces to allow an application to filter data to make them friendly, usable or public via anonymization process for privacy protection. In this project database integration middleware is being developed. Proposed work constructs the data preprocessing, integration and retrieval through the middleware. Data integrating middleware resolves the problems of data inconsistency which leads to proper and concrete data mining for business intelligence.

Chapter 1

Introduction

Introduction chapter will introduce the work, It will focusses exactly on what is the area of project and explains what is actually be done in this work. All ideas about project work are cleared here.

Chapter is of 7 sections. First section describes background of the project, motivations behind the project explains in section 1.2, section 1.3 will give problem defination of our project, scope of the project is defined in section 1.4, section 1.5 gives objectives of the project, organization of the whole project is given in section 1.6 & section 1.7 gives summary.

1.1 Background

Farming is the Prime Occupation in India inspite of this, today the people involved in farming belong to the lower class and is in deep poverty. There is no computerized system in existing system. Currently, the farmer goes to nearest market hand over his product to a particular agent, the agent asks 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 cut 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 of records. No facility is present for the maintaining records.

1.2 Motivation

In the Internet era, where information plays a key role in peoples lives, agriculture is rapidly becoming a very data intensive industry where Government need to collect farmer's information for various purposes. Current system that is manual system for storing of records. So Government does not get the exact information through existing system and many fraud occurs. And hence, will decide to propose such project for which gives better and accurate result related to farmer records.

1.3 Problem Definition

The Advanced techniques and the automated machines which are leading the world to new heights, is being lagging when it is concerned to farming either the lack of awareness of the advanced facilities or the unavailability leads to the poverty in Farming. Even after all the hard work and the production done by the farmers, in todays market the farmers are cheated by the agents, leading to the poverty. This project would do all the things automatic which make easier serving as a better solution to all the problems.

1.4 Scope

Due to this project the manual system which maintains records for farmers for various purposes and an automatic system in computer will use for maintaining records. So that, an authorized agent knows the information regarding to the farmer and farmer's farm, correctly and in accurate manner.

1.5 Objective

The main objective of this project is building a detailed information regarding farmer with the help of middleware techniques for accurateness which will help farmers from Government to sell their products to different city markets and also for authorized agents for getting the information of farmers for various records. It is a computerized approach for better and clear marketing and work. Farmers will get a unique interface where they can avail everything right from learning in the market information they can perform marketing, get the current rates on the market and current status of other farmers and apply as well as get new government schemes. Overall, this system is faster and more secure than the manual work.

1.6 Organization of Report

Chapter 2 describes the overall system analysis. it includes literature survey, proposed system, feasibility study, risk analysis, project scheduling and effort allocation. Chapter 3 gives system requirement specification such as hardware requirements, software requirements, functional requirements, non-functional requirements and other requirements and constraints. Chapter 4 shows system design by drawing system architecture, E-R diagram, Database design, Data flow diagram, user interface design and all important UML diagrams. Chapter 5 is on Implementation. That gives information of implementation details, implementation environment and flow of system development. Chapter 6 describe system testing, in this it shows how

to implement testing, test cases and test results. Chapter 7 is of result and analysis. In this chapter, we give sample snapshot of important processing and its explanation. Chapter 8 is a last chapter of our project report which is based on conclusion and future scope.

1.7 Summary

In this chapter, an overview of the problem statement along with its solution for the work contained in this dissertation is provided. In the next chapter, related work in the area of system analysis is presented.

Chapter 2

System Analysis

System analysis as a preliminary step to the actual design of Farmer has identified the internal as well as external conflicts and problems that the farm manager currently faces. In this chapter, section 2.1 will discuss literature survey, proposed System will discuss in section 2.2, section 2.3 will discuss feasibility study, economical feasibility, operational feasibility, and technical feasibility, risk analysis will discuss in section 2.4, section 2.5 will discuss project scheduling, effort allocation will discuss in section 2.6, section 2.7 gives summary.

2.1 Literature Survey

List of software which required for our project is:-

- Netbeans 7.1.2
- MySQL
- JDK Kit

These all the java supported software. He/She cannot able to install these Netbeans on their own systems for running their applications or programs. And user is bounded on particular machine where he/she regularly work he/she cannot access project of that machine from anywhere and user/client must have Netbeans installed on client machine because of this if any user does not have Netbeans so he/she cannot run source code. So our project provides solution for above drawbacks. One thing also to considered user does not need to set path for executing program user is free from all headache of setting path setting environment variable of Netbeans 7.1.2.

2.2 Proposed System

In this system describe whole structure and design of project. In our project MySQL 5.5 server and Netbeans 7.1.2 are used. Firstly Create the form, the first form is personal details

about the farmer then second form is field details and the third form is annual income details and database are create.Farmer can add the infomation in first form use add function.Farmer also Search information use search function.There are various type of function are used such as display,save,exit,delete,clear etc.Second form shows different types of crops,this crops are used farmer and shows farm sector,irrigation,fertilizers and pesticides are used farmer.Also farmer add details in second form use add function either clear details.Third form shows annual income details and source of selling.Farmer add the income use add function either clear income and source of selling.This system shows overall structure about the project and how the system is work.

2.3 Feasibility Study

The feasibility analysis shows the developers all the aspects of the project and they can know that whether the project is practically possible to develop worth limited resources and time. There are few types of feasibility are exist so developer should take care of these feasibility or developer must aware about these feasibility.

- Economical Feasibility
- Operational Feasibility
- Technical Feasibility

2.3.1 Technical Feasibility

At first it is necessary to check that system (proposed system) is technically feasible or not. Also to determine the technology and skill it is necessary to carry out the project. If they are not available then find out solutions for them. In our project shows about farmer personal details,field details and annual income details.In our project there are view types of tools are used such as Netbeans 7.1.2, MySQL 5.5, Smart Draw tool, JDK kit to develop the system.In our project technically develop Jasper report.Jaspar report gives whole information about farmer to fill from the admin.

2.3.2 Operational Feasibility

Operational feasibility is beneficial if they can be turned into information system will meet the organization operating requirement.The system is user friendly. This feasibility in which only one machine require. This feasibilty is operation feasibilty because no cost required.

2.3.3 Economical Feasibility

Economic feasibility is a cost benefit. In our project we require Netbeans 7.1.2 software and also we require java development tool kit this is also freeware software and we require MySQL database this is also freeware, so our project is feasible for developer and client also on client side generate the farmer registration form fill the admin from user. In our project 2-tier application are used then we say our project is economical. In our project java and MySQL are connected to connector.jar file.

2.4 Risk Analysis

2.4.1 Risk

Risk analysis and management are a series of steps that help a software team to understand and manage uncertainty. Many problems can plague a software project. A risk is a potential problem it might happen, it might not. But, regardless of the outcome, it's a really good idea to identify it, assess its probability of occurrence, estimate its impact, and establish a contingency plan should the problem actually occur. Everyone involved in the software process managers, software engineers, and customers participate in risk analysis and management.

2.4.2 Need of Risk Analysis

Think about the Boy Scout motto: Be prepared. Software is a difficult undertaking. Lots of things can go wrong, and frankly, many often do. It's for this reason that being prepared understanding the risks and taking proactive measures to avoid or manage them is a key element of good software project management.

2.4.3 Software Risk

Although there has been considerable debate about the proper definition for software risk, there is general agreement that risk always involves two characteristics

Uncertainty The risk may or may not happen; that is, there are no 100 percent probable risks.

Loss If the risk becomes a reality, unwanted consequences or losses will occur.

When risks are analyzed, it is important to quantify the level of uncertainty and the degree of loss associated with each risk. To accomplish this, different categories of risks are considered.

2.4.4 Project Risk

Threaten the project plan. That is, if project risks become real, it is likely that project schedule will slip and that costs will increase. Project risks identify potential budgetary, schedule, personnel (staffing and organization), resource, customer, and requirements problems and their impact on a software project. In our project, project risk occurs if our requirement of technical member means technical team is unavailable according to our project plan and estimation and if our project is not completed within time in this situation project risk can occur.

2.4.5 Technical Risks

Threaten the quality and timeliness of the software to be produced. If a technical risk becomes a reality, implementation may become difficult or impossible. Technical risks identify potential design, implementation, interface, verification, and maintenance problems. In addition, specification ambiguity, technical uncertainty, technical obsolescence, and "leading-edge" technology are also risk factors. Technical risks occur because the problem is harder to solve than we thought it would be. In our project if any module of our website is not worked properly according to our expectation then technical risk may occur.

2.4.6 Business Risk

Threaten the viability of the software to be built. Business risks often jeopardize the project or the product. Candidates for the top five business risks are:

1. Building a excellent product or system that no one really wants (market risk).
2. Building a product that no longer fits into the overall business strategy for the company (strategic risk).
3. Building a product that the sales force doesn't understand how to sell.
4. Losing the support of senior management due to a change in focus or a change in people (management risk).
5. Losing budgetary or personnel commitment (budget risks). It is extremely important to note that simple categorization won't always work. Some risks are simply unpredictable in advance.

2.5 Project Scheduling

Software project scheduling is an activity that distributes estimated effort across the planned project duration by allocating the effort to specific software engineering tasks. It is important to note, however, that the schedule evolves over time. During early stages of project planning, a macroscopic schedule is developed. This type of schedule identifies all major software engineering activities and the product functions to which they are applied. As the project gets under way, each entry on the macroscopic schedule is refined into a detailed schedule. Project scheduling can be done by Gantt chart.

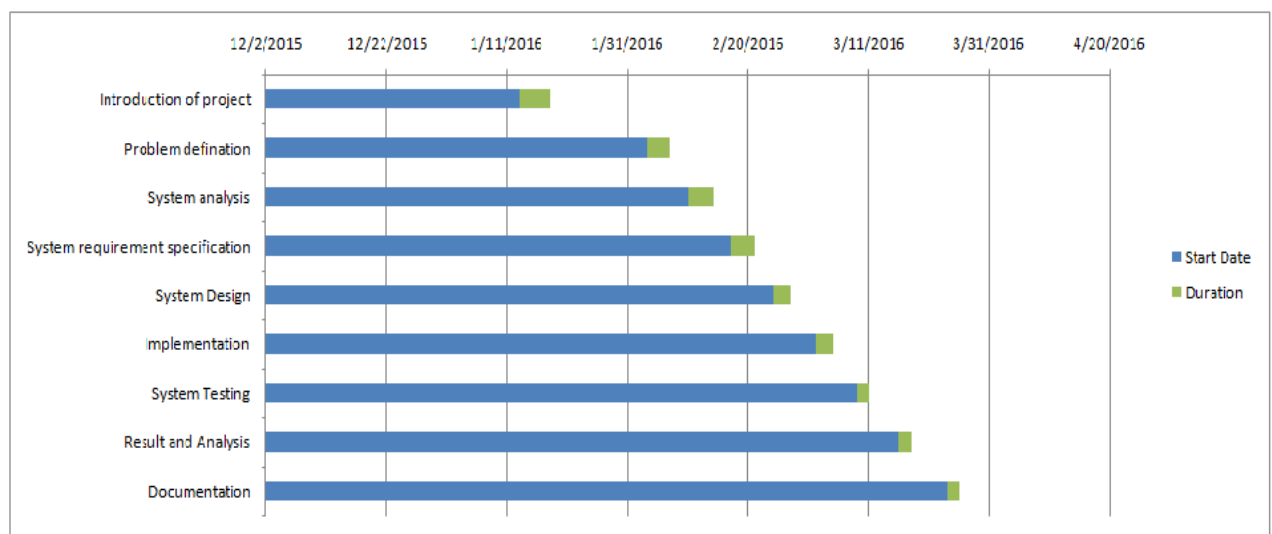


Figure 2.1: Gantt Chart

2.6 Effort Allocation

Software engineer or a team of engineers must incorporate a development strategy that encompasses the process, methods, and tools layers described. This strategy is often referred to as a process model or a software engineering paradigm. A process model for software engineering is chosen based on the nature of the project and application, the methods and tools to be used, and the controls and deliverables that are required. There are many process models in the software engineering, but we have chosen Water Fall Model because the project is totally dependent on previous modules. Another Reason for choosing this model is to provide better user satisfaction.

Allocation of efforts to all project partners- Project means team work; Project is developed by combination of effort of team. So whole project is divided into modules and number of modules is allotted to team members. After completion of each module, it will be link from one module to another module to form a complete project.

Development of middleware in remote database	Puja Naval	Tanaya Marathe	Neha Patil	Ashwini Patil
Introduction of project		✓		
System analysis				✓
System requirement specification			✓	
System design		✓		✓
Implementation	✓			
System testing	✓		✓	
Result and analysis	✓	✓		
Conclusion and future work			✓	✓
Documentation	✓			

Figure 2.2: Chart of Effort Allocation

This effort allocation should be used as a guideline only. The characteristics of each project must dictate the distribution of effort. Work expended on project planning rarely accounts for more than 2-3 percent of effort, unless the plan commits an organization to large expenditures with high risk. Requirements analysis may comprise 10-25 percent of project effort. Effort expended on analysis or prototyping should increase in direct proportion with project size and complexity. A range of 20 to 25 percent of effort is normally applied to software design. Time expended for design review and subsequent iteration must also be considered.

2.7 Summary

In this chapter, section 2.1 discuss literature survey, proposed System explained in section 2.2, section 2.3 discuss feasibility study, economical feasibility operational feasibility, and technical feasibility discuss, risk analysis discuss in section 2.4, section 2.5 discuss project scheduling, Effort Allocation discuss in section 2.6. In the next chapter we will discuss about software testing of our project.

Chapter 3

Requirement Analysis

In this chapter, section 3.1 will explain Hardware requirements, Software requirements will explain in section 3.2, section 3.3 will explain Functional requirements, Non-Functional requirements will explain in section 3.4 and section 3.5 will explain Other requirement and constraints, section 3.6 gives summary.

3.1 Hardware Requirements

Computer system is require in our project, other than that no need of any extra hardware requirement in our project.

3.2 Software Requirements

For these project,few softwares are required to be installed on computer system. As we are used Java language for coding, we need java runtime environment(jre) for execution. Also for storing data we require database server,for our project we are used MySQL.

Software that are required for our project are as follows:

- JDK kit
- NetBeans 7.1.2
- MySQL 5.5

3.3 Functional Requirements

The basic functional requirements are adding records, deleting records, searching records, displaying records from database and display results of various queries which we select from database in one frame.

List of Functional requirements for our project:

- Add records:
In this function, user can fill information form which is on various criteria. This information save in the database.
- Delete records:
In this function, user can delete the records by using Id from the database.
- Display records:
This function gives list of records of farmers who have fill the all forms which is stored in database.
- Search records:
Using this button, user want to see information of particular farmer based on Id.
- Query:
This function facilitates different search queries to retrieve the particular data from database.

3.4 Non-Functional Requirements

In Nonfunctional Requirements of this project implements those functions which does not effect on function and behavior of project for desired goal and objective of project. Non-functional Requirement just provides user friendliness and notifications that are not most necessary for this project. **List of Nonfunctional requirements for our project:**

- Tab Effect:
Tab Effect facility provides auto tab spacing while opening of any function. Tab Effect function provides well structured format of program. List of functional requirements for our project.
- Authorization:
This is one of the non-functional requirement of our project. Through which only authorised person will access our data.
- Jasper Report: It is non-functional requirement of our project. It displays information in printed form by using various queries.

3.5 Other Requirement and Constraints

Middleware is the "glue" that connects diverse computer systems appeared in the late 1980s to represent network connection management software. Types of middleware which is used

in our project:

- RPC: Remote Procedure call is type of middleware which is used for communication based system.
- MOM: Message Oriented Middleware is one of the class of middleware which is used for exchanging the message between two or more application.
- TP Monitor: Transaction Processing Monitor is provide tools and environment for develop and deploying application.

In our project middleware does following functions:

- (1) Data Integration
- (2) Message Passing
- (3) Data Persistence
- (4) Reduce Data Inconsistency
- (5) Reduce Missing Value.

3.6 Summary

In this chapter, section 3.1 explained Hardware requirements, Software requirements explained in section 3.2, section 3.3 explained Functional requirements, Non-Functional requirements explained in section 3.4 and section 3.5 explained Other requirement and constraints. In the next chapter we will discuss about overall system design of our project.

Chapter 4

System Design

4.1 System Architecture

Software architecture is the development work product that gives the highest return on investment with respect to quality, schedule and cost. Software architecture alludes to the overall structure of the software and the ways in which that structure provides conceptual integrity for a system. In its simplest form, architecture is the hierarchical structure of program components. The architectural design description should address how the design architecture achieves requirements for performance, capacity, reliability, security, adaptability, and other system characteristics. The system architectural is shown in given figure 4.1.

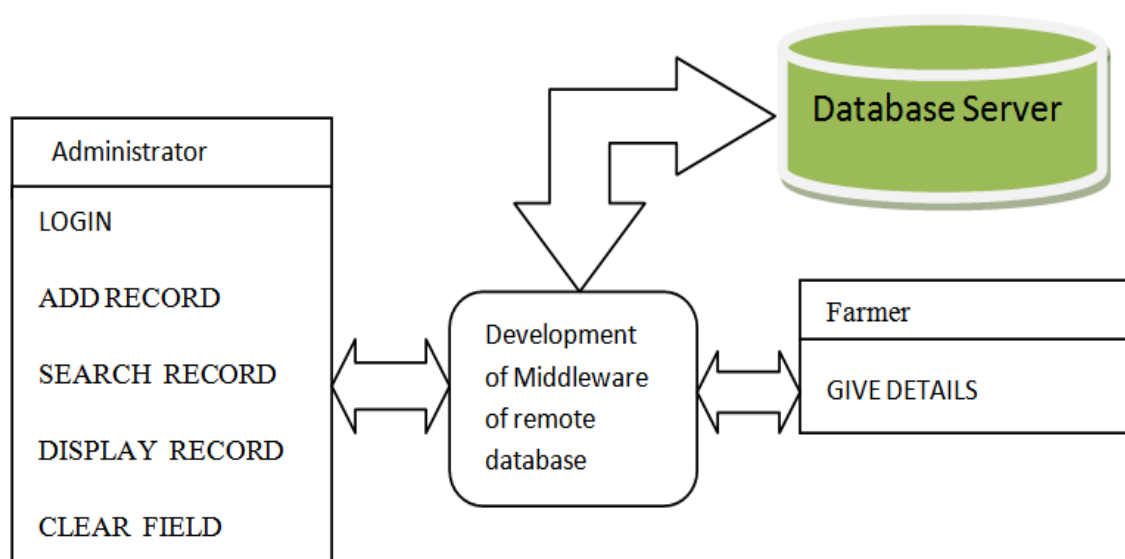


Figure 4.1: System Architecture for Farmer System.

4.2 E-R diagram

The entity relationship data model is based on a perception of a real world that consist of a collection of basic objects called entities, and relation among these objects.E-R diagram is shown in following figure 4.2.

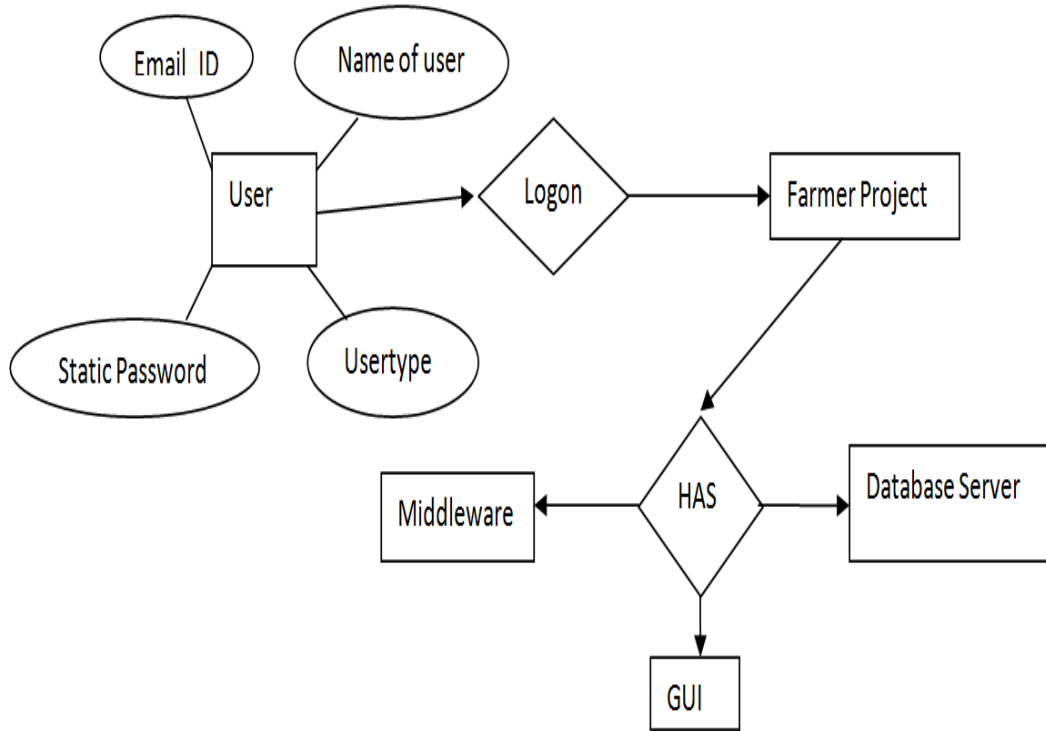


Figure 4.2: E-R Diagram for Farmer System.

4.3 Database Design

The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software. Part of data design may occur in conjunction with the design of software architecture. The data design activity translates these elements of the requirements model into data structures at the software component level and, when necessary, a database architecture at the application level. Schema for different databases is shown in figure 4.4, 4.5, 4.3, ??.

Field	Type	Null	Key	Default	Extra
ID	Int(5)	YES		Null	
Annual_Income	Int(10)	YES		Null	
Source_Of_Sellin g	Varchar(50)	YES		Null	

Figure 4.3: Annual Details Schema of Farmer.

Field	Type	Null	Key	Default	Extra
ID	Int(5)	YES		Null	
Name	Varchar(50)	YES		Null	
Address	Varchar(30)	YES		Null	
Village	Varchar(50)	YES		Null	
Pin_Code	Int(10)	YES		Null	
Country	Varchar(50)	YES		Null	
Gender	Varchar(6)	YES		Null	
Birth_Date	Date	YES		Null	
Mobile_No	Varchar(10)	YES		Null	
Education	Varchar(50)	YES		Null	
Age	Int(2)	YES		Null	

Figure 4.4: Personal Details Schema of Farmer.

Field	Type	Null	Key	Default	Extra
ID	Int(5)	YES		Null	
Crop1	Varchar(50)	YES		Null	
Crop2	Varchar(50)	YES		Null	
Fertilizer_And_Pesticides	Varchar(50)	YES		Null	
Irrigation	Varchar(20)	YES		Null	
Farm_Sector	Varchar(30)	YES		Null	

Figure 4.5: Field Details Schema of Farmer.

4.4 Data Flow Diagram

A DFD is a graphical technique that depicts the information flow and the transformation that we have applied as the data moves from input to output. The data flow diagram also known as data flow graph or bubble chart. A data flow diagram may be used to represent a system or software at any level of abstraction. The data flow diagram can be completed using only four simple notations i.e. special symbols or icons and the annotation that with a specific system. A data flow diagram (DFD) is a graphical technique that depicts information about flow and that are applied as data moves from input to output. The DFD is also called as data flow graph or bubble chart. Named circles show the processes in DFD or named arrows entering or leaving the bubbles represent bubbles and data flow. A rectangle represents a source or sink and is not originate or consumer of data. Data flow diagrams are the basic building blocks that define the flow of data in a system to the particular destination and

difference in the flow when any transformation happens. The data flow diagram serves two purposes: (1) To provide an indication of how data are transform as the moves through the system. (2) To depict the function that transforms the data flow.

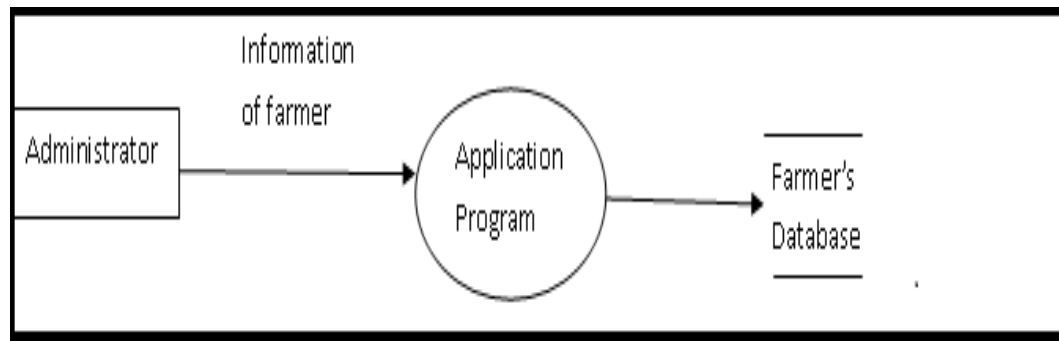


Figure 4.6: Level 0 DFD

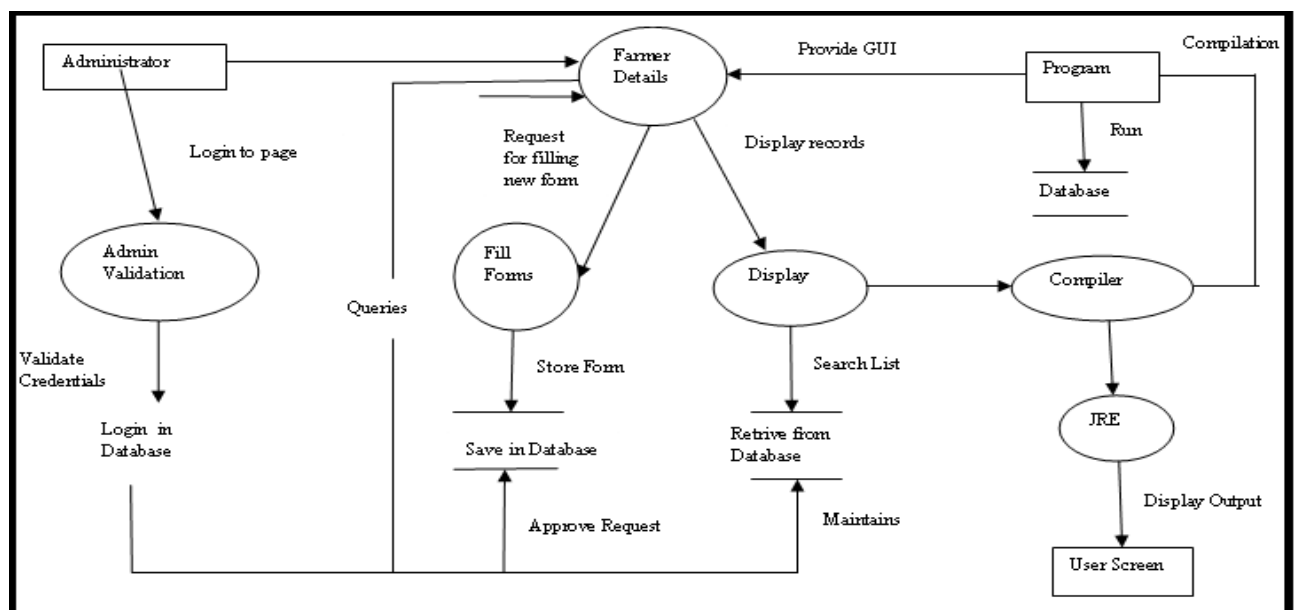


Figure 4.7: Level 1 DFD

4.5 UML Diagrams

The Unified Modeling Language (UML) is a standard language for writing software blueprints. UML may be used to visualize, specify, construct, and document the artifacts of a software-intensive system. Grady Booch, Jim Rumbaugh, and Ivar Jacobson developed UML in the mid 1990s with much feedback from the software development community. UML provides different diagrams for use in software modeling.

4.5.1 Use Case Diagram

UML use-case diagram 4.8 help you determine the functionality and features of the software from the users perspective. A use case describes how a user interacts with the system by defining the steps required to accomplish a specific goal.

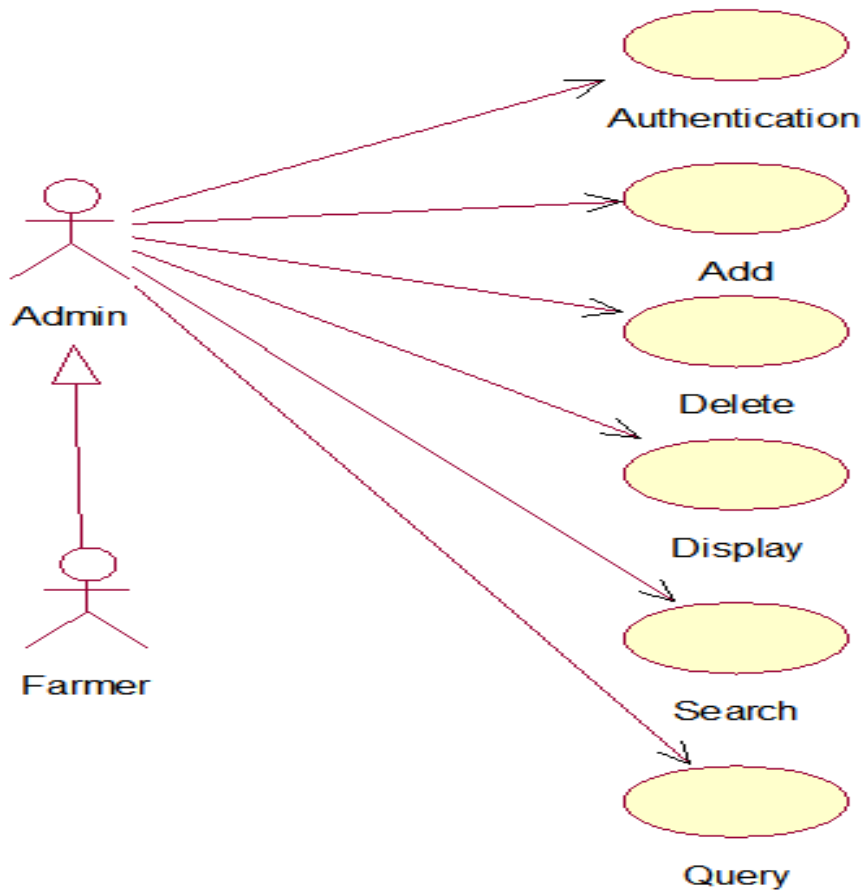


Figure 4.8: Use Case Diagram for Farmer System.

4.5.2 Class Diagram

The main elements of a class diagram 4.9 are boxes, which are the icons used to represent classes and interfaces. Each box is divided into horizontal parts. The top part contains the name of the class. The middle section lists the attributes of the class. An attribute refers to something that an object of that class knows or can provide all the time. The third section of the class diagram contains the operations or behaviors of the class.

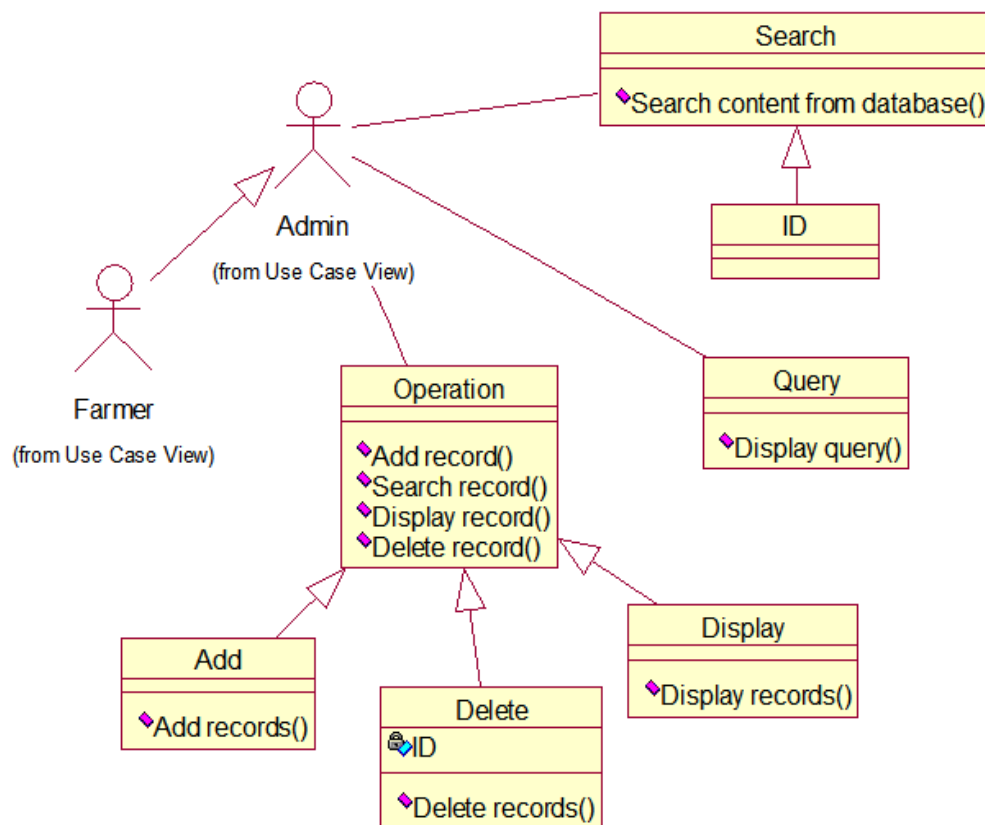


Figure 4.9: Class Diagram for Farmer System.

4.5.3 Sequence Diagram

A sequence diagram 4.10 is used to show the dynamic communications between objects during execution of a task. It shows the temporal order in which messages are sent between the objects to accomplish that task. One might use a sequence diagram to show the interactions in one use case or in one scenario of a software system.

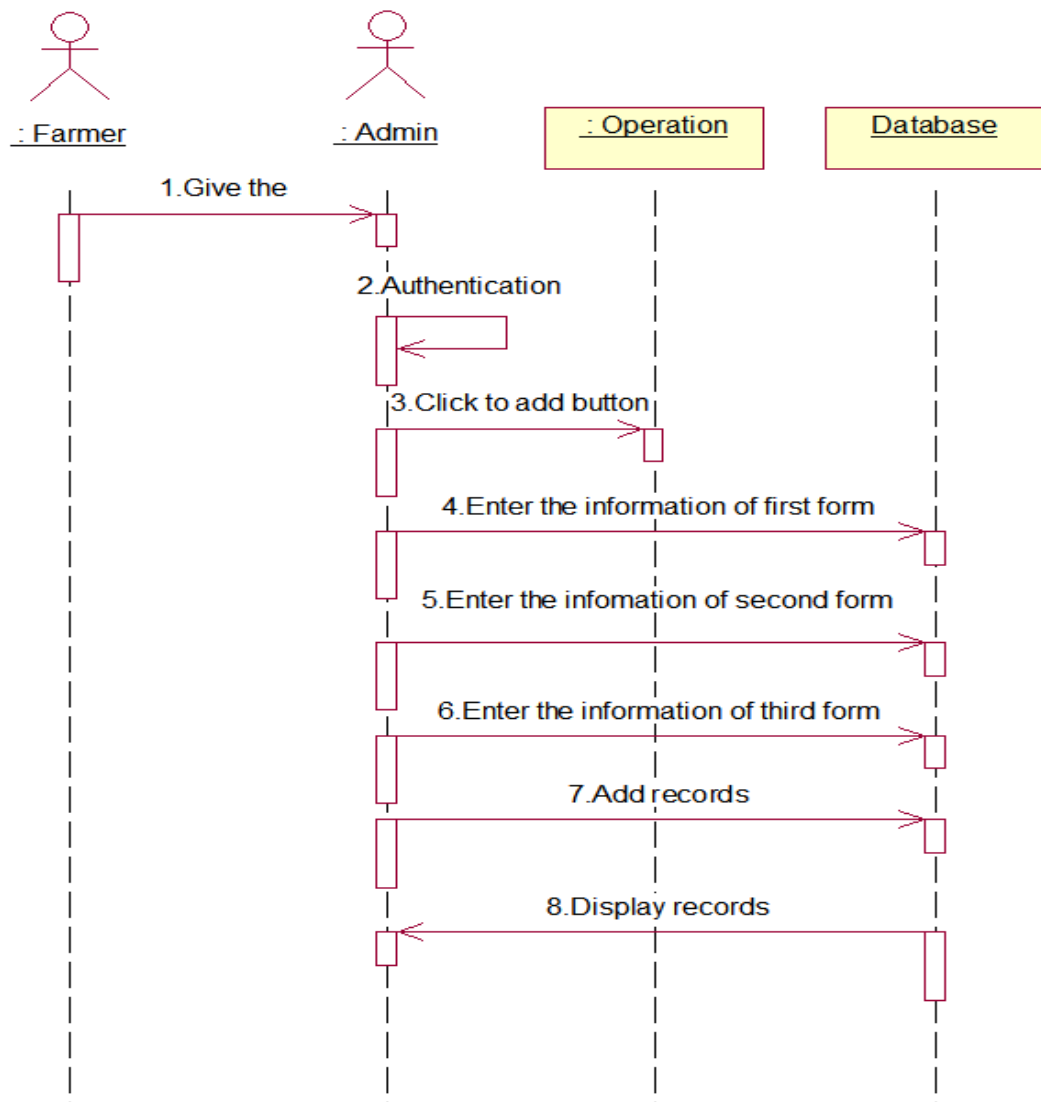


Figure 4.10: Sequence Diagram for Farmer System.

4.5.4 Activity Diagram

A UML activity diagram 4.11 depicts the dynamic behavior of a system or part of a system through the flow of control between actions that the system performs. It is similar to a flowchart except that an activity diagram can show concurrent flows.

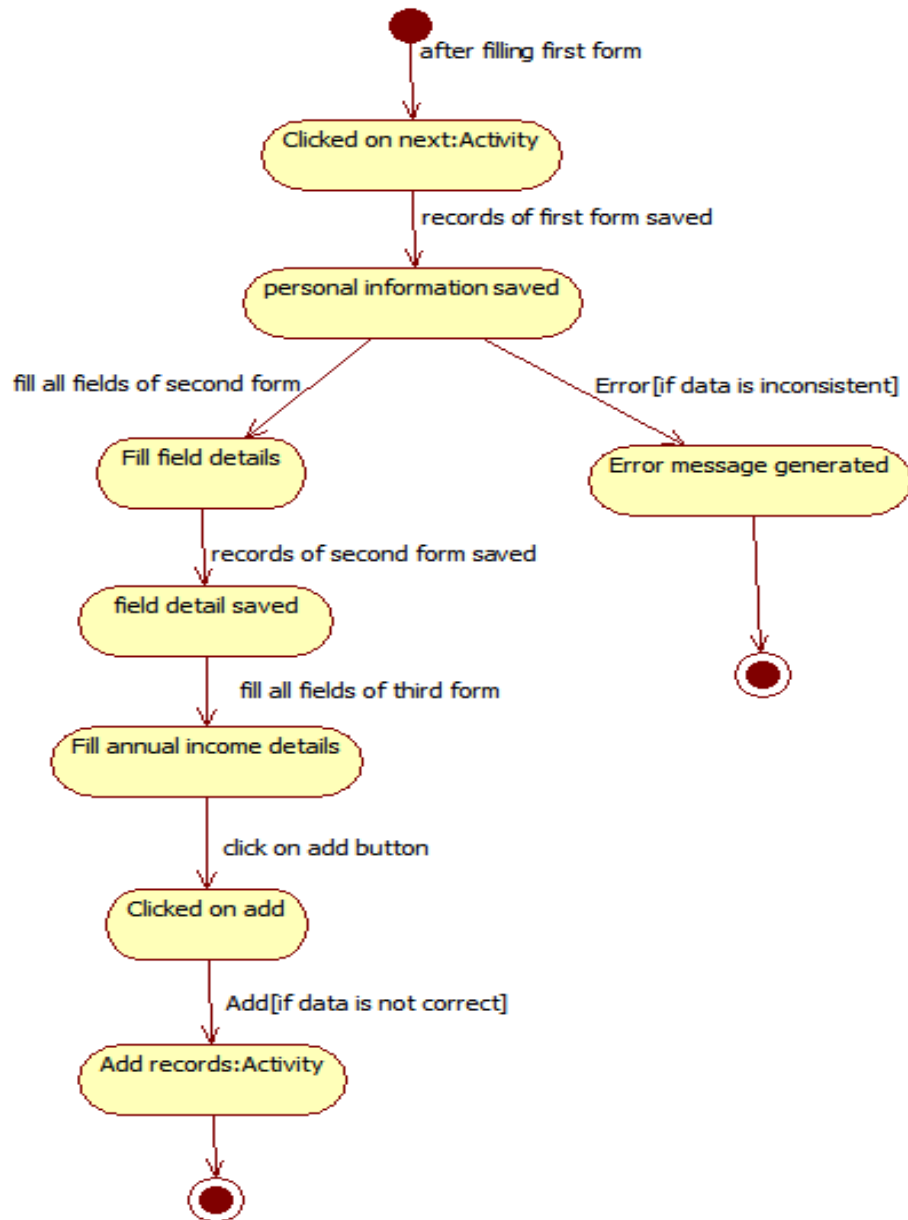


Figure 4.11: Activity Diagram for Farmer System.

4.5.5 Component Diagram

A UML component diagram 4.12 depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

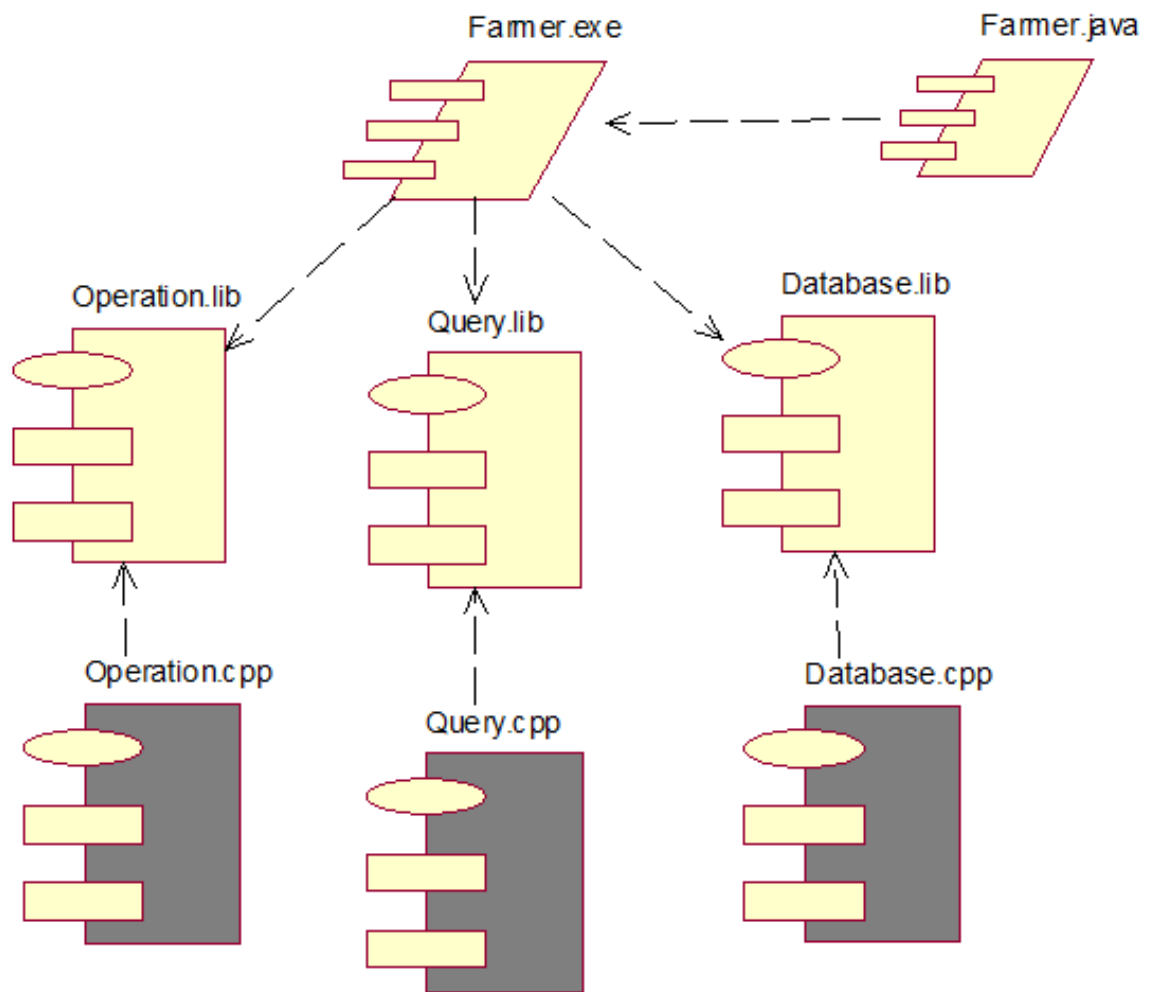


Figure 4.12: Component Diagram for Farmer System.

4.5.6 Deployment Diagram

A UML deployment diagram 4.13 focuses on the structure of a software system and is useful for showing the physical distribution of a software system among hardware platforms and execution environments.

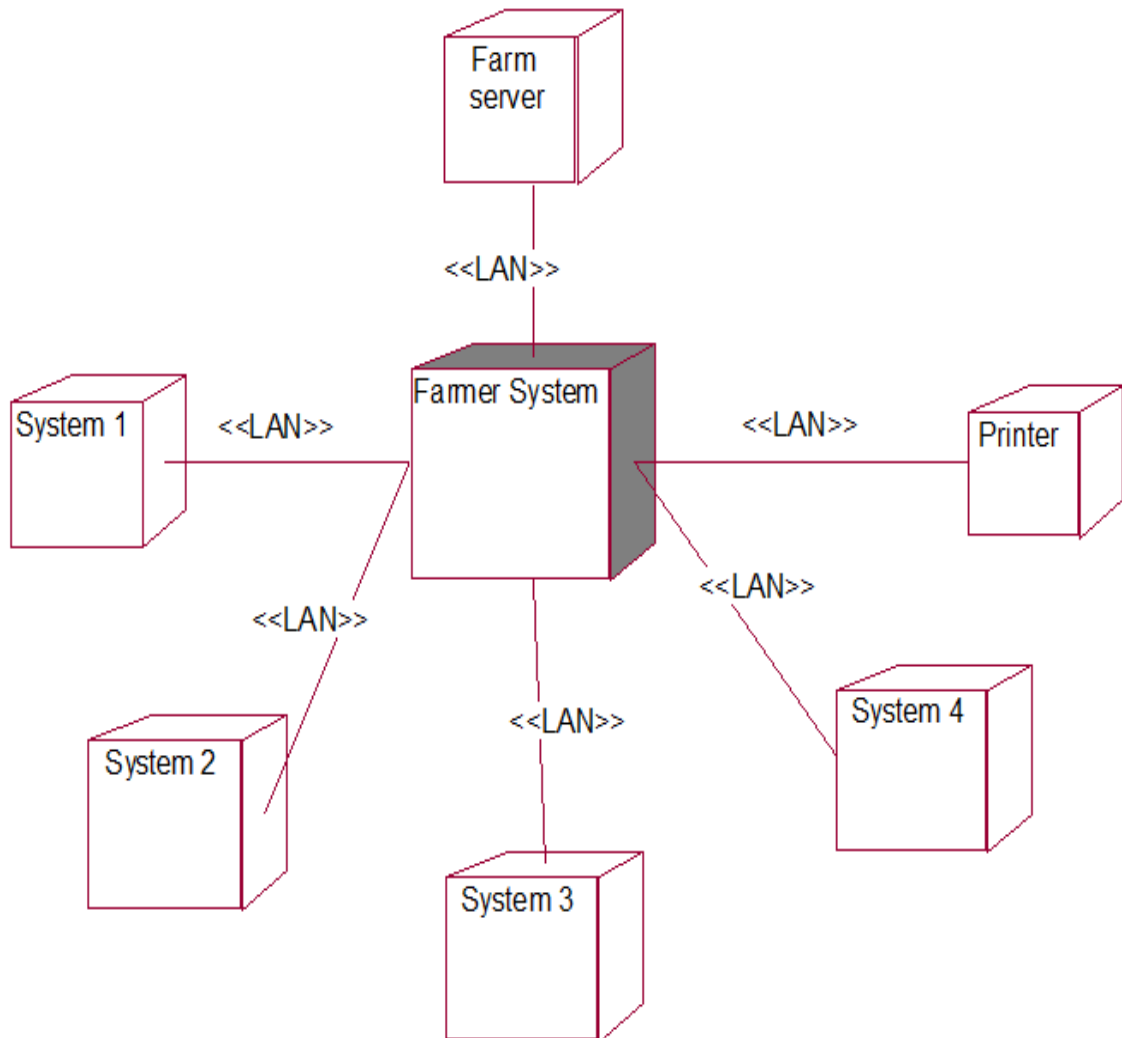


Figure 4.13: Deployment Diagram for Farmer System.

4.5.7 State Diagram

A UML state diagram 4.14 models an objects states, the actions that are performed depending on those states, and the transitions between the states of the object.

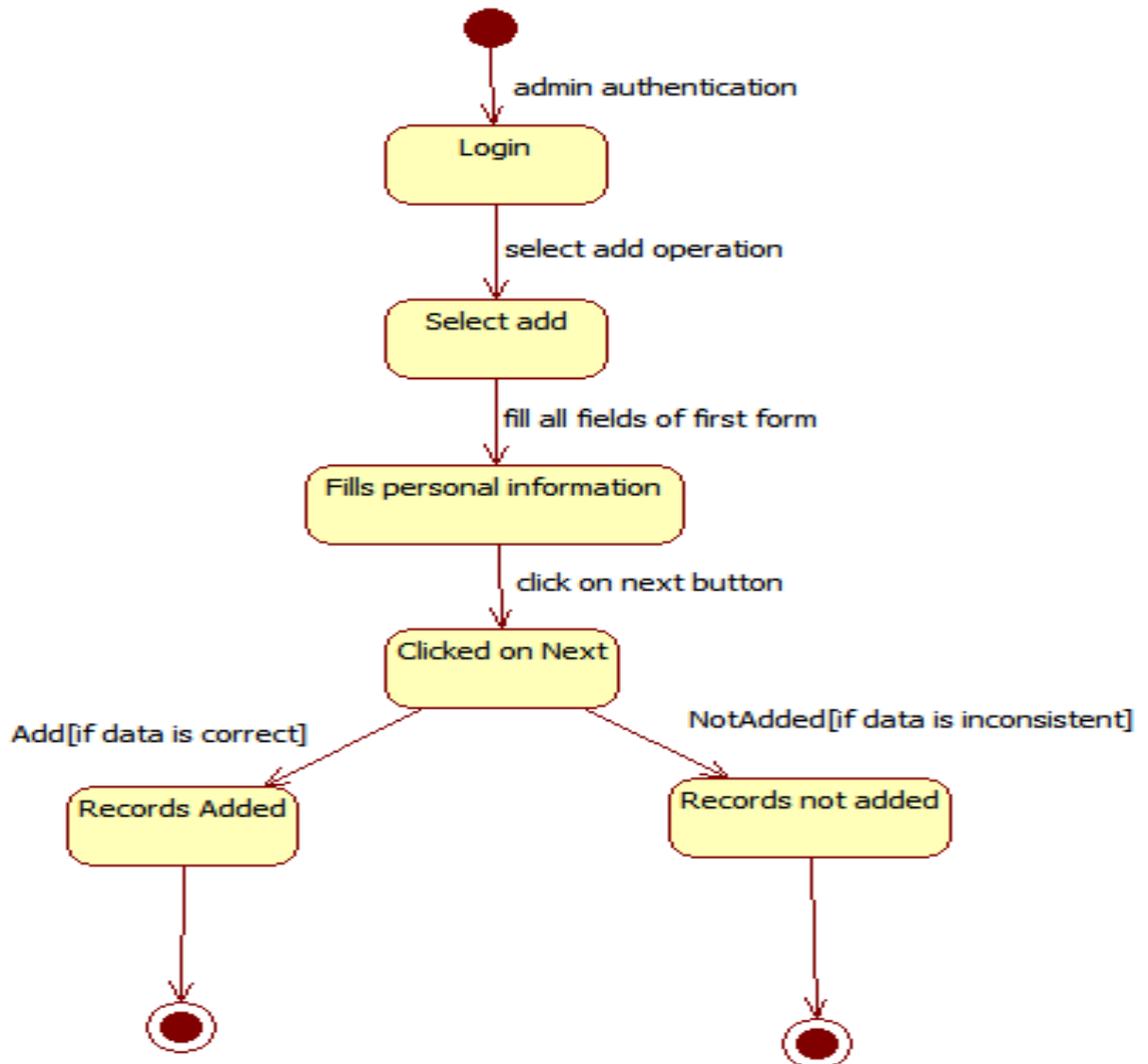


Figure 4.14: state Diagram for Farmer System.

4.5.8 Collaboration Diagram

The UML collaboration diagram (called a communication diagram) 4.15 provides another indication of the temporal order of the communications but emphasizes the relationships among the objects and classes instead of the temporal order.

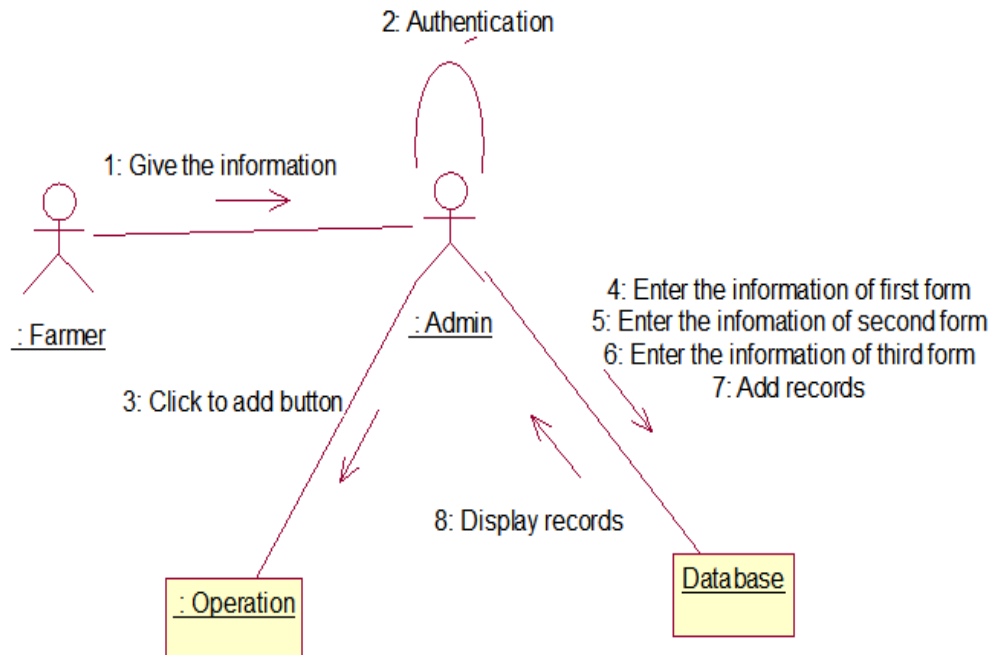


Figure 4.15: Collaboration Diagram for Farmer System.

4.6 Summary

In this chapter, section 4.1 discuss system architecture, E-R diagram explained in section 4.2, section 4.3 discuss database design, section 4.5 discuss data flow diagram, UML discuss in section 4.6. In the next chapter we will discuss implementation of our project.

Chapter 5

Implementation

5.1 Coding

Coding for Button Frame form:- for add records, delete records, search records, display records and exit

```
import java.awt.Color;
import java.sql.*;
import javax.swing.JOptionPane;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
public class ButtonFrame extends javax.swing.JFrame {
    Connection con;
    PreparedStatement ps;
    ResultSet rs;
    String[] columnNames={"ID","NAME","ADDRESS","VILLAGE","PIN_CODE","COUNTRY",
        "GENDER","BIRTH_DATE","MOBILE_NO","EDUCATION","CROP1","CROP2",
        "FERTILIZER_AND_PESTICIDES","IRRIGATION","ANNUAL_INCOME","SOURCE_OF_SELLING"};
    private JTable table;
    private JFrame frame1;
    private String from;
    private JTable table1;
    private JFrame frame2;
    int x;

    public ButtonFrame() {
        initComponents();
        try
        {
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/farmer",
                "info","hello");
            Statement stat=con.createStatement();
        }
    }
}
```

```

        catch(Exception e){
            System.out.println(e);
        }
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt){
        try {
            ps = con.prepareStatement("insert into farm values(?,?,?,?,?,?,?,?,?,?)");
            int s1 = Integer.parseInt(jTextField1.getText());
            String s2 = jTextField2.getText();
            String s3 = jTextField3.getText();
            String s4 = jTextField4.getText();
            int s5 = Integer.parseInt(jTextField5.getText());
            String s6 = jTextField6.getText();
            String s8 = jTextField8.getText();
            String s = jTextField9.getText();
            Long s9 = Long.parseLong(s);
            String s10 = jTextField10.getText();
            int age;
            Date dateOfBirth = new SimpleDateFormat("MM-dd-yyyy").parse(s8);
            java.sql.Date birthdate = new java.sql.Date(dateOfBirth.getTime());
            if (jTextField7.getText().equals("")) {
                Calendar dob = Calendar.getInstance();
                dob.setTime(dateOfBirth);
                Calendar today = Calendar.getInstance();
                age = today.get(Calendar.YEAR) - dob.get(Calendar.YEAR);
                if (today.get(Calendar.DAY_OF_YEAR) <= dob.get(Calendar.DAY_OF_YEAR))
                {
                    age--;
                }
            }
            else {
                age = Integer.parseInt(jTextField7.getText());
            }
            ps.setInt(1, s1);
            ps.setString(2, s2);
            ps.setString(3, s3);
            ps.setString(4, s4);
            ps.setInt(5, s5);
            ps.setString(6, s6);
            ps.setString(7, radioText);
            ps.setDate(8, birthdate);
            ps.setLong(9, s9);
            ps.setString(10, s10);
            ps.setInt(11, age);
            jTextField1.setText("");
            jTextField2.setText("");
            jTextField3.setText("");
            jTextField4.setText("");

```

```

        jTextField5.setText("");
        jTextField6.setText("");
        jTextField8.setText("");
        jTextField9.setText("");
        jTextField10.setText("");
        jTextField7.setText("");
        ps.executeUpdate();
        JOptionPane.showMessageDialog(this, "Record is saved");
        new farm2().setVisible(true);
        this.dispose();
    } catch (Exception e) {
        System.out.println(e);
    }
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt){
    try {
        ps = con.prepareStatement("delete a.*,b.*,c.* from farm as a,farm1 as b,
        farm2 as c where a.ID=b.ID and b.ID=c.ID and a.ID=?");
        ps.setString(1, jTextField1.getText());
        jTextField1.setText("");
        int x = ps.executeUpdate();
        if (x >= 1) {
            JOptionPane.showMessageDialog(null, "Record is deleted");
        } else {
            JOptionPane.showMessageDialog(null, "Record is not found...!");
        }
        new Button().setVisible(true);
        this.dispose();
    }
    catch (Exception e) {
        System.out.println(e);
    }
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt){
    frame2 = new JFrame("Farmer Details");
    frame2.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame2.setLayout(new BorderLayout());
    DefaultTableModel model = new DefaultTableModel();
    model.setColumnIdentifiers(columnNames);
    table1 = new JTable();
    table1.setModel(model);
    table1.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
    table1.setFillViewportHeight(true);
    JScrollPane scroll = new JScrollPane(table1);
    scroll.setHorizontalScrollBarPolicy(
        JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
    scroll.setVerticalScrollBarPolicy(

```

```

        JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
int id ;
String name = "";
String village = "";
String address = "";
int pin_code;
String country = "";
String gender = "";
String birth_date = "";
Long mobile_no;
String education = "";
int age;
String crop1="";
String crop2="";
String fertilizer_and_pesticides = "";
String irrigation="";
String farm_sector="";
int annual_income;
String source_of_selling="";
try
{
ps=con.prepareStatement("select farm.ID,farm.Name,farm.Address,farm.Village,
farm.Gender,farm.Birth_Date,farm.Mobile_No,farm.Education,farm.age,
farm1.Crop1,farm1.Crop2,farm1.Fertilizer_And_Pesticides,farm1.Irrigation,
farm1.Farm_Sector,farm2.Anual_Income,farm2.Source_Of_Selling from farm
join farm1 on farm.ID=farm1.ID join farm2 on farm2.ID=farm1.ID");
    int j=0;
    rs= ps.executeQuery();
    while(rs.next())
    {
        id = rs.getInt("ID");
        name = rs.getString("Name");
        address = rs.getString("Address");
        village = rs.getString("Village");
        pin_code= rs.getInt("Pin_code");
        country = rs.getString("Country");
        gender = rs.getString("Gender");
        birth_date = rs.getString("Birth_date");
        mobile_no = rs.getLong("Mobile_No");
        education = rs.getString("Education");
        age=rs.getInt("Age");
        crop1=rs.getString("Crop1");
        crop2=rs.getString("Crop2");
        fertilizer_and_pesticides = rs.getString("Fertilizer_And_Pesticides");
        irrigation=rs.getString("Irrigation");
        farm_sector=rs.getString("Farm_Sector");
        annual_income = rs.getInt("Annual_Income");
    }
}

```

```

        source_of_selling=rs.getString("Source_Of_Selling");
        model.addRow(new Object[]{id,name,address,village,pin_code,
        country,gender,birth_date,mobile_no,education,age,crop1,crop2,
        fertilizer_and_pesticides,irrigation,farm_sector,annual_income,
        source_of_selling});
        k++;
    }
    if (k < 1) {
        JOptionPane.showMessageDialog(null, "No Record Found",
        "Error", JOptionPane.ERROR_MESSAGE);
    }
    if (k == 1) {
        System.out.println(k + " Record Found");
    } else {
        System.out.println(k + " Records Found");
    }
} catch (Exception ex) {
    JOptionPane.showMessageDialog(null, ex.getMessage(),
    "Error", JOptionPane.ERROR_MESSAGE);
}
frame2.add(scroll);
frame2.setVisible(true);
frame2.setSize(1000,800);
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt){
    frame1 = new JFrame("Farmer Details");
    frame1.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame1.setLayout(new BorderLayout());
    DefaultTableModel model = new DefaultTableModel();
    model.setColumnIdentifiers(columnNames);
    table = new JTable();
    table.setModel(model);
    table.setAutoResizeMode(JTable.AUTO_RESIZE_ALL_COLUMNS);
    table.setFillsViewportHeight(true);
    JScrollPane scroll = new JScrollPane(table);
    scroll.setHorizontalScrollBarPolicy(
        JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);
    scroll.setVerticalScrollBarPolicy(
        JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED);
    int id ;
    String name = "";
    String village = "";
    String address = "";
    int pin_code;
    String country = "";
    String gender = "";
    String birth_date = "";

```

```

Long mobile_no;
String education = "";
int age;
String crop1="";
String crop2="";
String fertilizer_and_pesticides = "";
String irrigation="";
String farm_sector="";
int annual_income;
String source_of_selling="";
try
{
ps=con.prepareStatement("select farm.ID,farm.Name,farm.Address,farm.Village,
farm.Gender,farm.Birth_Date,farm.Mobile_No,farm.Education,farm.age,
farm1.Crop1,farm1.Crop2,farm1.Fertilizer_And_Pesticides,farm1.Irrigation,
farm1.Farm_Sector,farm2.Anual_Income,farm2.Source_Of_Selling from farm
join farm1 on farm.ID=farm1.ID join farm2 on farm2.ID=farm1.ID");
int j=0;
rs= ps.executeQuery();
while(rs.next())
{
    id = rs.getInt("ID");
    name = rs.getString("Name");
    address = rs.getString("Address");
    village = rs.getString("Village");
    pin_code= rs.getInt("Pin_code");
    country = rs.getString("Country");
    gender = rs.getString("Gender");
    birth_date = rs.getString("Birth_date");
    mobile_no = rs.getLong("Mobile_No");
    education = rs.getString("Education");
    age=rs.getInt("Age");
    crop1=rs.getString("Crop1");
    crop2=rs.getString("Crop2");
    fertilizer_and_pesticides = rs.getString("Fertilizer_And_Pesticides");
    irrigation=rs.getString("Irrigation");
    farm_sector=rs.getString("Farm_Sector");
    annual_income = rs.getInt("Annual_Income");
    source_of_selling=rs.getString("Source_Of_Selling");
    model.addRow(new Object[]{id,name,address,village,pin_code,
    country,gender,birth_date,mobile_no,education,age,crop1,crop2,
    fertilizer_and_pesticides,irrigation,farm_sector,annual_income,
    source_of_selling});
    j++;
}
if (j < 1) {
JOptionPane.showMessageDialog(null, "No Record Found", "Error",

```



```

        JOptionPane.ERROR_MESSAGE);
    }
    if (j == 1) {
        System.out.println(j + " Record Found");
    } else {
        System.out.println(j + " Records Found");
    }
} catch (Exception ex) {
    JOptionPane.showMessageDialog(null, ex.getMessage(),
        "Error", JOptionPane.ERROR_MESSAGE);
}
frame1.add(scroll);
frame1.setVisible(true);
frame1.setSize(1000,800);
}

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt){
    try {
        con.close();
        ps.close();
    } catch (Exception e) { }
    System.exit(0);
}

private void jTextField1KeyPressed(java.awt.event.KeyEvent evt) {
    int key = evt.getKeyCode();
    if ((key >= evt.VK_0 && key <= evt.VK_9) || (key >= evt.VK_NUMPAD0 &&
        key <= evt.VK_NUMPAD9) || (key == evt.VK_BACK_SPACE)){
        jTextField1.setEditable(true);
        jTextField1.setBackground(Color.WHITE);
    } else {
        jTextField1.setEditable(false);
        jTextField1.setBackground(Color.YELLOW);
        JOptionPane.showMessageDialog(null, "ID must be numeric value");
    }
}

private void jTextField2KeyPressed(java.awt.event.KeyEvent evt) {
    int key = evt.getKeyCode();
    if ((key >= evt.VK_A && key <= evt.VK_Z) || (key == evt.VK_BACK_SPACE) ||
        (key == evt.VK_SPACE) || (key == evt.VK_SHIFT)){
        jTextField2.setEditable(true);
        jTextField2.setBackground(Color.WHITE);
    } else {
        jTextField2.setEditable(false);
        jTextField2.setBackground(Color.YELLOW);
        JOptionPane.showMessageDialog(null, "Name must be in character form");
    }
}

private void jTextField5KeyPressed(java.awt.event.KeyEvent evt) {

```

```

        int key = evt.getKeyCode();
        if ((key >= evt.VK_0 && key <= evt.VK_9) || (key >= evt.VK_NUMPAD0 &&
        key <= evt.VK_NUMPAD9) || (key == evt.VK_BACK_SPACE)){
            jTextField5.setEditable(true);
            jTextField5.setBackground(Color.WHITE);
        } else {
            jTextField5.setEditable(false);
            jTextField5.setBackground(Color.YELLOW);
            JOptionPane.showMessageDialog(null, "Pin code must be numeric value");
        }
    }

    private void jTextField7KeyPressed(java.awt.event.KeyEvent evt) {
        int key = evt.getKeyCode();
        if ((key >= evt.VK_0 && key <= evt.VK_9) || (key >= evt.VK_NUMPAD0 &&
        key <= evt.VK_NUMPAD9) || (key == evt.VK_BACK_SPACE)){
            jTextField7.setEditable(true);
            jTextField7.setBackground(Color.WHITE);
        } else {
            jTextField7.setEditable(false);
            jTextField7.setBackground(Color.YELLOW);
            JOptionPane.showMessageDialog(null, "Age must be numeric value");
        }
    }

    private void jTextField9KeyPressed(java.awt.event.KeyEvent evt) {
        int key = evt.getKeyCode();
        if ((key >= evt.VK_0 && key <= evt.VK_9) || (key >= evt.VK_NUMPAD0 &&
        key <= evt.VK_NUMPAD9) || (key == evt.VK_BACK_SPACE)){
            jTextField9.setEditable(true);
            jTextField9.setBackground(Color.WHITE);
        } else {
            jTextField9.setEditable(false);
            jTextField9.setBackground(Color.YELLOW);
            JOptionPane.showMessageDialog(null, "Mobile Number must be numeric value");
        }
    }

    String radioText="";
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt){
        if (jRadioButton1.isSelected()) {
            radioText = jRadioButton1.getText();
        }
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt){
        if (jRadioButton2.isSelected()) {
            radioText = jRadioButton2.getText();
        }
    }

    private void jTextField2FocusGained(java.awt.event.FocusEvent evt){

```

```

        jTextField2.setForeground(Color.black);
        jTextField2.setText("");
    }
    private void jTextField3FocusGained(java.awt.event.FocusEvent evt){
        jTextField3.setForeground(Color.black);
        jTextField3.setText("");
    }
    private void jTextField8FocusGained(java.awt.event.FocusEvent evt){
        jTextField8.setForeground(Color.black);
        jTextField8.setText("");
    }

    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new farm3().setVisible(true);
            }
        });
    }

    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton4;
    private javax.swing.JButton jButton6;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JRadioButton jRadioButton1;
    private javax.swing.JRadioButton jRadioButton2;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField10;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    private javax.swing.JTextField jTextField4;
    private javax.swing.JTextField jTextField5;
    private javax.swing.JTextField jTextField6;
    private javax.swing.JTextField jTextField7;
    private javax.swing.JTextField jTextField8;
    private javax.swing.JTextField jTextField9;
}

```

Coding for Farm 2:- Field Details

```
import java.awt.BorderLayout;
import java.sql.*;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;

public class farm2 extends javax.swing.JFrame {
    Connection con;
    PreparedStatement ps;
    ResultSet rs;

    public farm2() {
        initComponents();
        try
        {
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/farmer",
            "info","hello");
            Statement stat=con.createStatement();
        }

        catch(Exception e){
            System.out.println(e);
        }
    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            ps = con.prepareStatement("insert into farm1 values(?,?,?,?,?,?)");
            int s1 = Integer.parseInt(jTextField1.getText());
            String s2 = jComboBox2.getToolTipText();
            String s3 = jComboBox3.getToolTipText();
            String s4 = jComboBox4.getToolTipText();
            String s5 = jComboBox5.getToolTipText();
            String s6 = jTextField6.getText();
            ps.setInt(1, s1);
            ps.setString(2, s2);
            ps.setString(3, s3);
            ps.setString(4, s4);
            ps.setString(5, s5);
            ps.setString(6, s6);
            jTextField1.setText("");
            jComboBox2.removeAllItems();
            jComboBox3.removeAllItems();
            jComboBox4.removeAllItems();
            jComboBox5.removeAllItems();
            jTextField6.setText("");
        }
    }
}
```

```

        int i=ps.executeUpdate();
        System.out.println(i+"Records Affected");
        JOptionPane.showMessageDialog(null, "Record is saved");
        new farm3().setVisible(true);
        this.dispose();
    }
    catch (Exception e) {
        JOptionPane.showMessageDialog(null, e.getMessage(),"ERROR",
        JOptionPane.ERROR_MESSAGE); }
}
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        ps = con.prepareStatement("insert into farm1 values(?,?,?,?,?,?)");
        jTextField1.setText("");
        jComboBox2.removeAllItems();
        jComboBox3.removeAllItems();
        jComboBox4.removeAllItems();
        jComboBox5.removeAllItems();
        jTextField6.setText("");
    }
    catch (Exception e) { }
}
private void jTextField1KeyPressed(java.awt.event.KeyEvent evt) {
    int key = evt.getKeyCode();
    if ((key >= evt.VK_0 && key <= evt.VK_9) || (key >= evt.VK_NUMPAD0 &&
    key <= evt.VK_NUMPAD9) || (key == evt.VK_BACK_SPACE)){
        jTextField1.setEditable(true);
        jTextField1.setBackground(Color.WHITE);}
    else{
        jTextField1.setEditable(false);
        jTextField1.setBackground(Color.YELLOW);
        JOptionPane.showMessageDialog(null, "ID must be numeric value ");
    }
}
private void jComboBox2ActionPerformed(java.awt.event.ActionEvent evt) {
    String s2 = (String) jComboBox2.getSelectedItem();
    jComboBox2.setToolTipText(s2);
}
private void jComboBox3ActionPerformed(java.awt.event.ActionEvent evt) {
    String s3 = (String) jComboBox3.getSelectedItem();
    jComboBox3.setToolTipText(s3);
}
private void jComboBox4ActionPerformed(java.awt.event.ActionEvent evt) {
    String s3 = (String) jComboBox4.getSelectedItem();
    jComboBox4.setToolTipText(s3);
}
private void jComboBox5ActionPerformed(java.awt.event.ActionEvent evt) {

```

```

        String s3 = (String) jComboBox5.getSelectedItem();
        jComboBox5.setToolTipText(s3);
    }
    public static void main(String args[]){
        java.awt.EventQueue.invokeLater(new Runnable(){
            public void run() {
                new farm2().setVisible(true);
            }
        });
    }
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JComboBox jComboBox2;
    private javax.swing.JComboBox jComboBox3;
    private javax.swing.JComboBox jComboBox4;
    private javax.swing.JComboBox jComboBox5;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField6;
}

```

Coding for Farm 3:- Annual Details

```

import java.awt.BorderLayout;
import java.sql.*;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.table.DefaultTableModel;

public class farm2 extends javax.swing.JFrame {
    Connection con;
    PreparedStatement ps;
    ResultSet rs;

    public farm2() {
        initComponents();
        try
        {
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost:3306/farmer",

```

```

        "info","hello");
        Statement stat=con.createStatement();
    }

    catch(Exception e){
        System.out.println(e);
    }
}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        ps = con.prepareStatement("insert into farm2 values(?,?,?)");
        int s1 = Integer.parseInt(jTextField1.getText());
        int s2 = Integer.parseInt(jTextField2.getText());
        String s3 = jComboBox1.getToolTipText();
        ps.setInt(1, s1);
        ps.setInt(2, s2);
        ps.setString(3, s3);
        jTextField1.setText("");
        jTextField2.setText("");
        jComboBox1.removeAllItems();
        ps.executeUpdate();
        JOptionPane.showMessageDialog(null, "Record is saved");
        new NewJFrame().setVisible(true);
        this.dispose();
    } catch (Exception e){ }
}

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        ps = con.prepareStatement("insert into farm2 values(?,?,?)");
        jTextField1.setText("");
        jTextField2.setText("");
        jComboBox1.removeAllItems();
    } catch (Exception e){}
}

private void jTextField1KeyPressed(java.awt.event.KeyEvent evt){
    int key = evt.getKeyCode();
    if ((key >= evt.VK_0 && key <= evt.VK_9) || (key >= evt.VK_NUMPAD0 &&
        key <= evt.VK_NUMPAD9) || (key == evt.VK_BACK_SPACE)){
        jTextField1.setEditable(true);
        jTextField1.setBackground(Color.WHITE);
    } else {
        jTextField1.setEditable(false);
        jTextField1.setBackground(Color.YELLOW);
        JOptionPane.showMessageDialog(null, "ID must be numeric value");
    }
}

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt){
    String s3 = (String) jComboBox1.getSelectedItem();

```

```

        jComboBox1.setToolTipText(s3);
    }
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new farm3().setVisible(true);
            }
        });
    }
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JComboBox jComboBox1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
}

```

5.2 Implementation Environment

It includes description of implemented environment such as which platform and language and technology that will be used, Back end, Front-end database that will be used.

5.2.1 Front-End-: Java (programming language)

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers.[citation needed] Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

NetBeans NetBeans is an open-source project dedicated to providing rock solid software development products (the NetBeans IDE and the NetBeans Platform) that address the needs of developers, users and the businesses who rely on NetBeans as a basis for their

products; particularly, to enable them to develop these products quickly, efficiently and easily by leveraging the strengths of the Java platform and other relevant industry standards.

The NetBeans Platform

The NetBeans Platform is a generic framework for Swing applications. It provides the "plumbing" that, before, every developer had to write themselves saving state, connecting actions to menu items, toolbar items and keyboard shortcuts; window management, and so on. The NetBeans Platform provides a reliable and flexible application architecture. Your application does not have to look anything like an IDE. It can save you years of development time. The NetBeans Platform architecture is modular, it's easy to create applications that are robust and extensible.

5.2.2 Back End Database:- MySQL Database

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons:

- MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL uses a standard form of the well-known SQL data language.
- MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
- MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
- MySQL is very friendly to PHP, the most appreciated language for web development.
- MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).

5.3 Summary

In this chapter, section 5.1 described design details of modules and interaction between them. Implementation environment described in section 5.2 in implementation environment included description of implemented environment such as which platform and language and technology that will be used, Back end, Front-end database that will be used.

Chapter 6

System Testing

In this chapter, we will discuss about software testing, steps for software testing, objectives and principles of software testing in section 6.1. In the section 6.2 we will discuss about various test cases and test results, section 6.3 gives summary.

6.1 How to Implement Testing

6.1.1 Software Testing

Once source code has been generated, software must be tested to uncover (and correct) as many errors as possible before delivery to your customer. Your goal is to design a series of test cases that have a high likelihood of finding errors but how? That's where software testing techniques enter the picture. These techniques provide systematic guidance for designing tests that,

1. Exercise the internal logic of software components, and
2. Exercise the input and output domains of the program to uncover errors in program function, behavior and performance.

During early stages of testing, a software engineer performs all tests. However, as the testing process progresses, testing specialists may become involved.

6.1.2 Steps of Software Testing

Software is tested from two different perspectives:

- Internal program logic is exercised using white box test case design techniques.
- Software requirements are exercised using black box test case design techniques.

In both cases, the intent is to find the maximum number of errors with the minimum amount of effort and time.

6.1.3 Software Testing Objectives

In an excellent book on software testing, Glen Myers [MYE79] states a number of Rules that can serve well as testing objectives:

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

6.1.4 Testing Principles

- All tests should be traceable to customer requirements.
- Tests should be planned long before testing begins.
- The Pareto principle applies to software testing.
- Testing should begin in the small and progress toward testing in the large.
- Exhaustive testing is not possible.
- To be most effective, testing should be conducted by an independent third party.

6.1.5 Manual Testing

It is the oldest and most rigorous types of testing it is performed by human sitting in front of a computer carefully going through application screens, trying various usage and input combinations, comparing the results to the expected behavior and recording their observations about project

6.1.6 Steps of Manual Software Testing

There are certain ways of manual testing first of all we write test cases then we execute that test cases then we generate report according to test case result

6.2 Test Cases and Test Results

Test case is the set of inputs along with the output and some additional information like-

- Test Case ID number : Number identifying the test case

- Test Case Name : Name of the test case.
- Test Case Description : Details/ purpose of the test case
- Steps : Sequence of steps that the tester must follow to perform test case
- Test Data : Input given to the function.
- Expected Results : What the tester should see when test case is executed
- Actual Result : What the tester actually witnesses when test case is executed
- Test Result (P/F): Indicates whether the test case passed, failed, etc.

Test Cases and Test Results:-

Test Case ID	Test Case Name	Test Case Description	Step	Test Data	Expected Result	Actual Result	Test Result (P/F)
ID_01	Validation ID for numeric value	Test whether ID is valid or not	1)Enter ID	Numeric	Valid	Valid	P
			2)Enter ID	Char(abcd)	Invalid	Invalid	F
			3)Enter ID	Numeric+Special symbol	Invalid	Invalid	F

Figure 6.1: Test Case for ID

Test Case ID	Test Case Name	Test Case Description	Step	Test Data	Expected Result	Actual Result	Test Result (P/F)
Name_01	Validation name for string	Test whether Name is valid or not	1)Enter Name	Char(abcd)	Valid	Valid	P
			2)Enter Name	Numeric	Invalid	Invalid	F
			3)Enter Name	Char+Numeric	Invalid	Invalid	F

Figure 6.2: Test Case for Name

Test Case ID	Test Case Name	Test Case Description	Step	Test Data	Expected Result	Actual Result	Test Result (P/F)
Mobile_No	Validation of Mobile No.	Length of Mobile No is ten digit or not	1)Enter Mobile_No	Numeric(10)	Yes	Yes	P
			2)Enter Mobile_No	Char(abcd)	No	No	F
			1)Enter Mobile_No	Numeric(less than 10 digit)	No	No	F

Figure 6.3: Test Case for Mobile No.

6.3 Summary

In this chapter, we discuss about software testing, steps for software testing, objectives and principles of software testing in section 6.1. In the section 6.2 we discussed about various test cases and test results. In the next chapter, results that illustrate how the system designed by you works in practice, and how it is intended to be use and The analysis of results is also done in next chapter.

Chapter 7

Result and Analysis

7.1 Sample Snapshot of Important Processing and Its Explanation.

Home page: In the snapshot 7.1 we have showed the home page of our project first time when authorized person login to our project.



Figure 7.1: Home Form

Login: In the snapshot 7.2 authorized person can simply login into our system by typing username and password.

Accessing Records: In the snapshot 7.3 after login, authorized person accessing records and perform add, delete, search, display operation on records.

Farm1: In the snapshot 7.4 will represent personal details about farmer. This snapshot shows various functionalities such as add, clear, and exit.

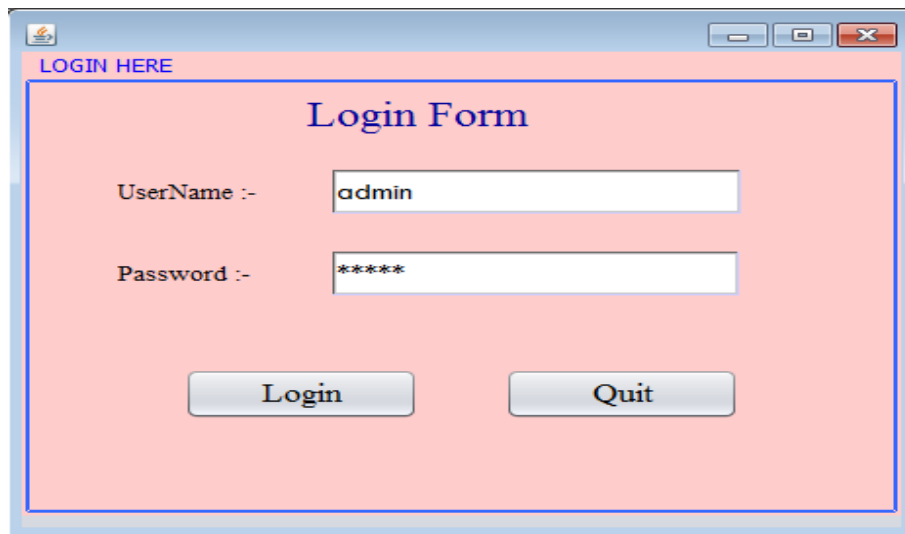


Figure 7.2: Login Form

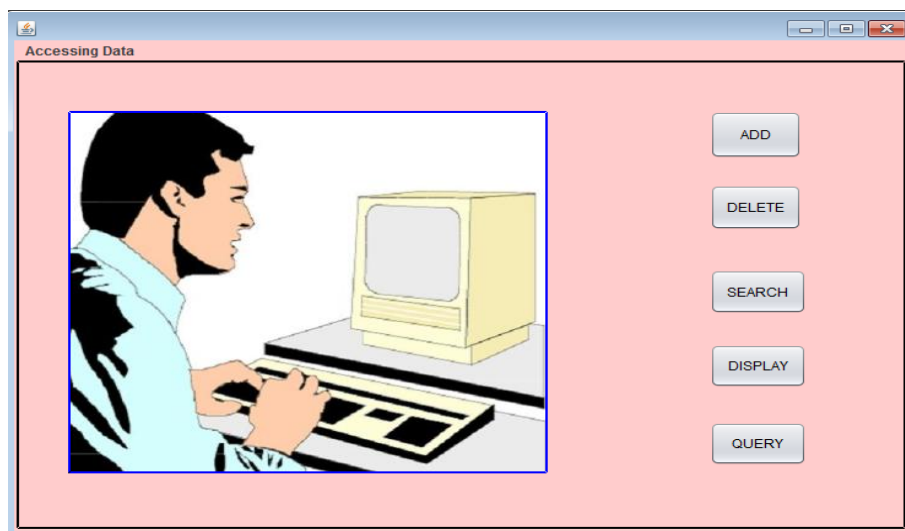


Figure 7.3: Accessing Records Form

Farm2: In the snapshot 7.5 will show field details of farmer and functionalities like add, clear, and exit.

Farm3: In the snapshot 7.6 will show field details of farmer and functionalities like add, clear, and exit.

Query: In the snapshot 7.7 will show the various query related to farmer.

Jasper Report: In the snapshot 7.8 will show information of farmer in the form of jasper report which we select.

Personal Information of Farmer

ID	1
Name	Sudhir Kashinath Patil
Address	Plot No 52 ,Nandra
Village	Nandra
Pin code	425002
Country	India
Gender	<input checked="" type="radio"/> Male <input type="radio"/> Female
Birth Date	11-09-1976
Mobile Number	7878870956
Education	HSC
Age	40

Figure 7.4: Personal Information of Farmer

Field Details

ID	1
Crop1	Cotton
Crop2	Banana
Fertilizer And Pesticides	15:15:15
Irrigation	Sprinkler
Farm Sector	15 Acre

Figure 7.5: Field Details of Farmer



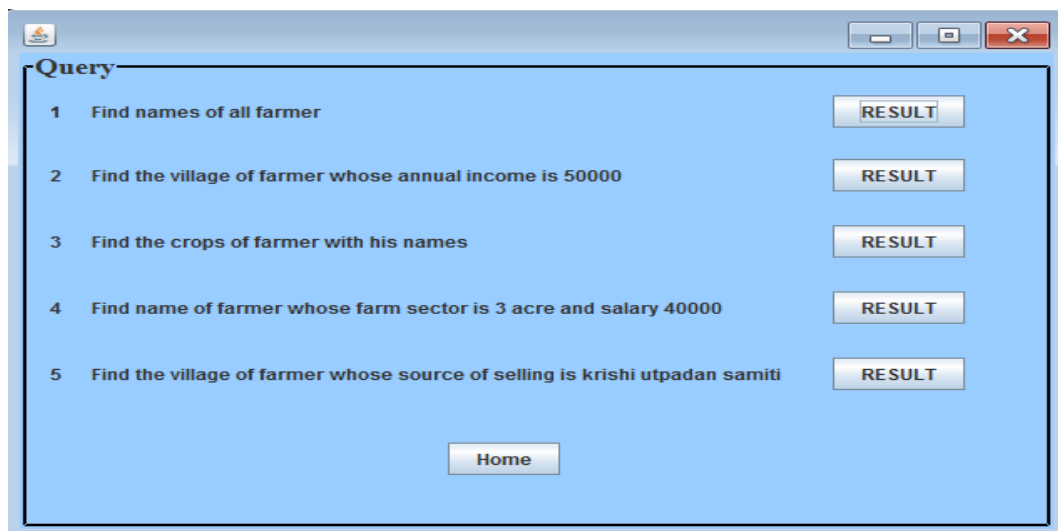
Annual Details

ID:

Annual Income:

Source Of Selling:

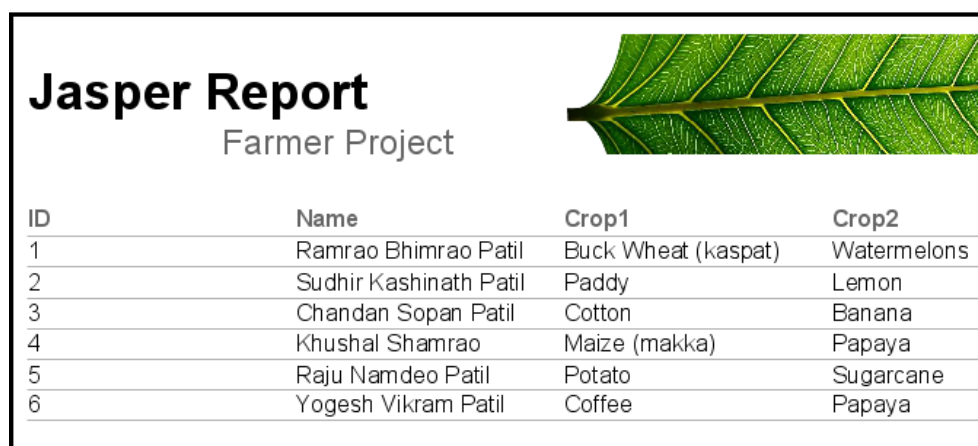
Figure 7.6: Annual Details of Farmer




Query

- Find names of all farmer
- Find the village of farmer whose annual income is 50000
- Find the crops of farmer with his names
- Find name of farmer whose farm sector is 3 acre and salary 40000
- Find the village of farmer whose source of selling is krishi utpadan samiti

Figure 7.7: Query Form



Jasper Report
Farmer Project



ID	Name	Crop1	Crop2
1	Ramrao Bhimrao Patil	Buck Wheat (kaspas)	Watermelons
2	Sudhir Kashinath Patil	Paddy	Lemon
3	Chandan Sopan Patil	Cotton	Banana
4	Khushal Shamrao	Maize (makka)	Papaya
5	Raju Namdeo Patil	Potato	Sugarcane
6	Yogesh Vikram Patil	Coffee	Papaya

Figure 7.8: Jasper Report

Chapter 8

Conclusion and Future Scope

8.1 Conclusion

- Our project reduces fraud which happened in manual system.
- Our project overcomes the problems of data inconsistency and data redundancy which oftenly occurred in manual system.
- Avoid missing of data in computerized system which happened in manual system.
- Government will get correct and accurate information of farmer using proposed system.

8.2 Future Scope

- Up till now, this project developed using two tier application for user, but in future, this project can be extended upto three tier architecture for user and it can be developed using web page design in client-server architecture using servlet and JSP code.
- In future, record will be also accessible to farmer for viewing information of other farmer having more income for improving its' own annual income.

Bibliography

- [1] P.A.Berstein.Middleware:a model for distributed system services.Communication of the ACM ,pages 86-98,1996.
- [2] J.M.Myerson.The complete book of middleware.Auerbach Publications,2002.
- [3] S.Mahendra :Dev Small farmers in India opportunities and challenges.
- [4] NetBeans.org, Welcome to the NetBeans Community, NetBeans IDE, May 2007.