VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JNANA SANGAMA", BELAGAVI - 590 018

A MINI PROJECT REPORT

on

"NURSERY MANAGEMENT SYSTEM"

Submitted by

Pushpa Manjunath Naik	**4SF18IS070**
Kshema Kishore	**4SF18IS045**

In partial fulfillment of the requirements for the V semester

DBMS LABORATORY WITH MINI PROJECT

of

BACHELOR OF ENGINEERING

in

INFORMATION SCIENCE & ENGINEERING

Under the Guidance of

Mr. Rithesh Pakkala P.
Assistant Professor,
Department of ISE

At

SAHYADRI

College of Engineering & Management Adyar,

Mangaluru - 575 007

2018 - 19

Department of Information Science & Engineering

CERTIFICATE

This is to certify that the Mini **Project** entitled **"Nursery Management System"** has been carried out by *Pushpa* Manjunath Naik **(4SF18IS070)** and Kshema Kishor **(4SF18IS045)**, the bonafide students of Sahyadri College of Engineering & Management in partial fulfillment of the requirements for the V semester **DBMS Laboratory with Mini Project (18CSL58)** of **Bachelor of Engineering in Information Science & Engineering** of Visvesvaraya Technological University, Belagavi during the year 2020 -21. 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.

———————————— ————————————

Mr. Rithesh Pakkala P.	**Dr. Shamanth Rai**
Assistant Professor	HOD & Associate Professor
Dept. of ISE, SCEM	Dept. of ISE, SCEM

External Practical Examination:

Examiner's Name Signature with Date

1. ....

2. ....

Department of Information Science & Engineering

DECLARATION

We hereby declare that the entire work embodied in this Mini Project Report titled **"Nursery Management System"** has been carried out by us at Sahyadri College of Engineering and Management, Mangaluru under the supervision of **Mr. Rithesh Pakkala P.** and **Mrs. Deepti Rai** as the part of the V semester **DBMS Laboratory with Mini Project (18CSL58)** of **Bachelor of Engineering** in **Information Science & Engineering**. This report has not been submitted to this or any other University.

Pushpa Manjunath Naik **(4SF18IS070)**
Kshema Kishore **(4SF18IS045)**
SCEM, Mangaluru

Abstract

A database management system (DBMS) is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data. A DBMS makes it possible for end users to create, read, update and delete data in a database. DBMS include change management, performance monitoring/tuning and backup and recovery.

A nursery is a place where plants are propagated and grown to a demand age. Plant nursery record management is generally through notebooks which is laborious and consume more time for data entry and retrieval. To conquer this difficulty and to attract youth towards nursery entrepreneurship this unique nursery management system has been developed.

This project is based on the application of database where we are creating a database for green house nursery admin where he can store the details such as supplier, available plants, order, customer and billing. If the customer wants to buy a desirable plant, then instead of searching in the nursery the admin can search in the stored database for the availability so that we can reduce consuming time and can access data based on species wise. This is developed to facilitate efficient nursery management while executing task in speedy and eco-friendly manner. We developed our Plant Nursery Management System using Oracle 10g database and Java.

Acknowledgement

The satisfaction that accompanies the successful completion of any work would be incomplete without thinking of the people who made it perfect with their constant guidance and encouragement.

We are profoundly indebted to our guides, **Mr. Rithesh Pakkala P.** Assistant Professors, Department of Information Science & Engineering for innumerable acts of timely advice, encouragement and We sincerely express our gratitude.

We express our sincere gratitude to **Dr. Shamanth Rai**, Head & Associate Professor, Department of Information Science & Engineering for his invaluable support and guidance.

We sincerely thank **Dr. Rajesh S**, Principal, Sahyadri College of Engineering & Management and **Dr. D. L. Prabhakara**, Director, Sahyadri Educational Institutions,who have always been a great source of inspiration.

Finally, yet importantly, We express our heartfelt thanks to our family & friends for their wishes and encouragement throughout the work.

Pushpa Manjunath Naik
4SF18IS070
VSem, B.E., ISE
SCEM, Mangaluru

Kshema Kishore
4SF18IS045
V Sem, B.E., ISE
SCEM, Mangaluru

Table of Contents

List of Figures

Chapter 1

Introduction

The Technology nowadays plays a very important role in our society. It lessens human error by using machines. It makes us work easier and faster and reduces materials and labour costs. Manual processes and transactions can be computerized by using software applications or computer systems to make work easier. Nowadays, computer based system is commonly used by every company and institution and one of this is the Nursery System. The Nursery System is the heart of the Plant Nursery Management of an Organization,

Nursery System is one of the many transactions that can be processed on a computer. The proponents worked on a custom-made Nursery system done for the specific need of an organization to provide an option to generate the salary automatically every day .

1.1 Database Management System

With the widespread use of computer technology and network technology, the development of database technology has become an important part of advanced information technology. The core of the payroll management system is how to use and operate database, so the database design is critical. This system uses the Oracle10g database which is a relational database management system of Oracle. It is a product that always has been a leading position in the field of database. The Oracle database system is the world popular relational database management system which is easy to use, strong function and suitable for all kinds of large, medium and small, microcomputer environment. It can realize data sharing and the facilities don’t need to have the powerful data storage and processing capabilities so that to reduce the hardware cost of company. Many companies

have their own database, and store a large number of key data in it, which shows the importance of the database.

Every table in the database is broken up into smaller entities called fields. A field is a column in a table that is designed to maintain specific information about every record in the table. A record, also called a row, is each individual entry that exists in a table. A record is a horizontal entity in a table. A column is a vertical entity in a table that contains all information associated with a specific field in a table. In addition to tables, a database can also contain other objects including views, stored procedures, indexes and constraints,triggers along with a transaction log.

1.2 Structured Query Language

SQL is a domain specific language used in programming and designed for managing data held in a database management system. SQL consists of a data definition language, data manipulation language and data control language. The scope of SQL includes data insert, query update and delete, schema creation and modification and data access control. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications which may run either on the same computer or on another computer across a network.

The main mode of retrieving data from a SQL Server database is querying for it. The query declaratively specifies what is to be retrieved. It is processed by the query processor, which figures out the sequence of steps that will be necessary to retrieve the requested data. The sequence of actions necessary to execute a query is called a query plan. There might be multiple ways to process the same query.

SQL Server also allows stored procedures to be defined. Stored procedures are parameterized T-SQL queries, that are stored in the server. Stored procedures can accept values sent by the client as input parameters, and send back results as output parameters. They can call defined functions, and other stored procedures, including the same stored procedure. Unlike other queries, stored procedures have an associated name,which is used at runtime to resolve into the actual queries. Also because the code need not be sent from the client every time it reduces network traffic and improves performance.

1.3 Normalization

Normalization is a process of organizing the data in database to avoid data redundancy, insertion anomaly, update anomaly and deletion anomaly.

Here are the most commonly used normal forms:

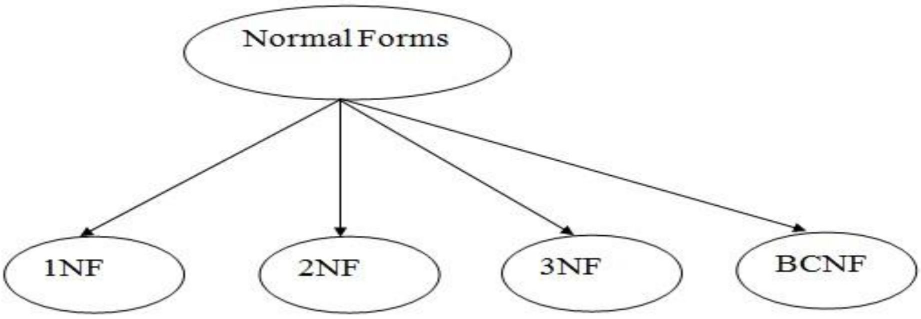


Figure 1.1: Types of Normalization

First Normal Form (1NF) As per the rule of first normal form, an attribute (column) of a table cannot hold multiple values. It should hold only atomic values.

Second Normal Form (2NF)

A table is said to be in 2NF if both the following conditions hold:

1.Table is in 1NF (First normal form)

2.No non-prime attribute is dependent on the proper subset of any candidate key of table.

An attribute that is not part of any candidate key is known as non-prime attribute.

Third Normal Form (3NF)

A table design is said to be in 3NF if both the following conditions hold:

1.Table must be in 2NF

2.Transitive functional dependency of non-prime attribute on any super key should be removed.

An attribute that is not part of any candidate key is known as non-prime attribute.

Boyce Codd Normal Form (BCNF)

It is an advance version of 3NF that's why it is also referred as 3.5NF. BCNF is stricter than 3NF. A table complies with BCNF if it is in 3NF and for every functional dependency
X-¿Y, X should be the super key of the table.

1.4 Trigger

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database. The trigger is mostly used for maintaining the integrity of the information on the database.Triggers can be defined to run instead of or after DML (Data Manipulation Language) actions such as INSERT, UPDATE, and DELETE.

1.5 Stored Procedure

A stored procedure is a set of Structured Query Language (SQL) statements with an assigned name, which are stored in a relational database management system as a group, so it can be reused and shared by multiple programs.Stored procedures can access or modify data in a database, but it is not tied to a specific database or object, which offers a number of advantages.

1.6 Application

This software can be used in any plant nursery to keep track of plant details, customer, supplier and the billing details for the admin.

The main advantage of this software is that it reduces the use of paper and also reduces the time of recording the data, where we can access data based on species wise, data wise search etc. If the customer wants to buy a desirable plant, then instead of searching in the nursery the admin can search in the stored database for the availability so that we can reduce consuming time. This is developed to facilitate efficient nursery management while executing task in speedy and eco-friendly manner..

Chapter 2

Requirements Specification

2.1 Hardware Requirements

- Processor : Any Processor above 500 MHz
- RAM : 2GB
- Hard Disk : 500GB
- Input Device : Standard Keyboard and Mouse
- Output Device : Monitor

2.2 Software Requirements

- Database : Oracle 10g Express Edition.
- Programming Language :Java.
- IDE :NetBeans 8.2
- Operating System:Windows 10

Chapter 3

System Design

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering.

3.1 ER Diagram

An entity relationship diagram(ERD),also known as entity relationship model,is a graphical representation of an information system that depicts the relationship among people,object,places,concepts or events within that system.An ERD is a data modeling technique that can help to define business processes and be used as a foundation for a relational database.

There are three basic elements in an ER Diagram: entity, attribute, relationship. There are more elements which are based on the main elements. They are weak entity, multi valued attribute, derived attribute, weak relationship, and recursive relationship. Cardinality and ordinality are two other notations used in ER diagrams to further define relationships.

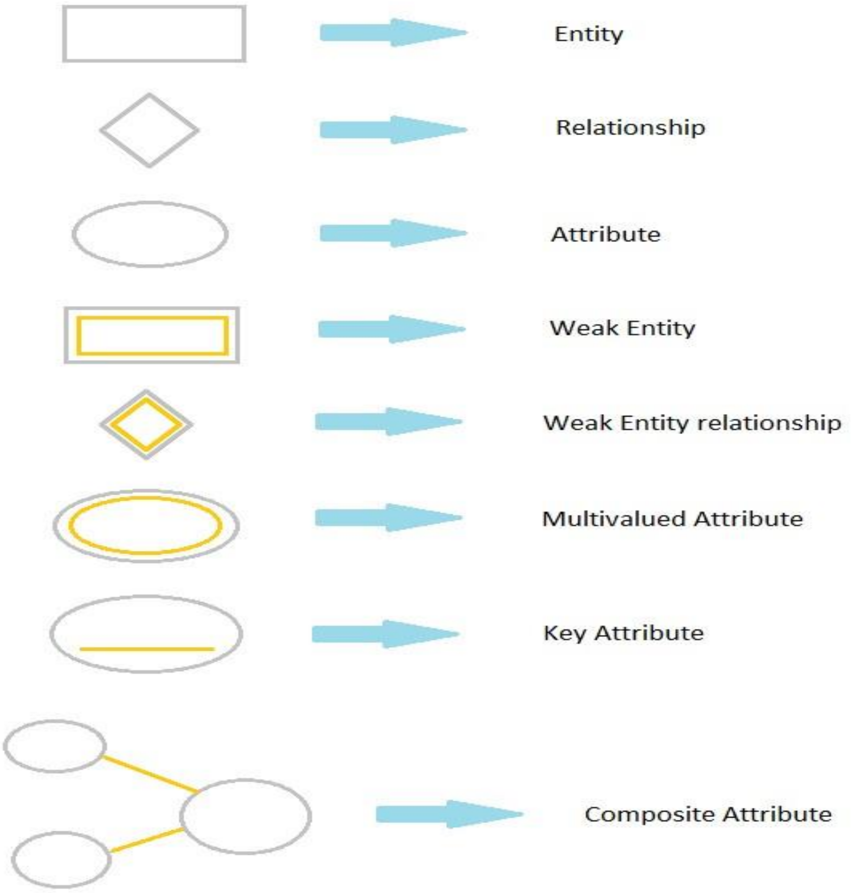


Figure 3.1: Components of ER Diagram

Entity

An entity can be a person, place, event, or object that is relevant to a given system. For example, a school system may include students, teachers, major courses, subjects, fees, and other items. Entities are represented in ER diagrams by a rectangle and named using singular nouns.

Weak Entity

A weak entity is an entity that depends on the existence of another entity. In more technical terms it can be defined as an entity that cannot be identified by its own attributes. It uses a foreign key combined with its attributed to form the primary key. An entity like order item is a good example for this. The order item will be meaningless without an order so it depends on the existence of the order.

Attribute

An attribute is a property, trait, or characteristic of an entity, relationship, or another attribute. For example, the attribute Inventory Item Name is an attribute of the entity Inventory Item. An entity can have as many attributes as necessary. Meanwhile, attributes can also have their own specific attributes. For example, the attribute "customer address" can have the attributes number, street, city, and state. These are called composite attributes. Note that some top level ER diagrams do not show attributes for the sake of simplicity. In those that do, however, attributes are represented by oval shapes.

Multivalued Attribute

If an attribute can have more than one value it is called a multi-valued attribute. It is important to note that this is different from an attribute having its own attributes. For example, a teacher entity can have multiple subject values.

Derived Attribute

An attribute based on another attribute. This is found rarely in ER diagrams. For example, for a circle, the area can be derived from the radius.

Relationship

A relationship describes how entities interact. Relationships are represented by diamond shapes and are labelled using verbs.

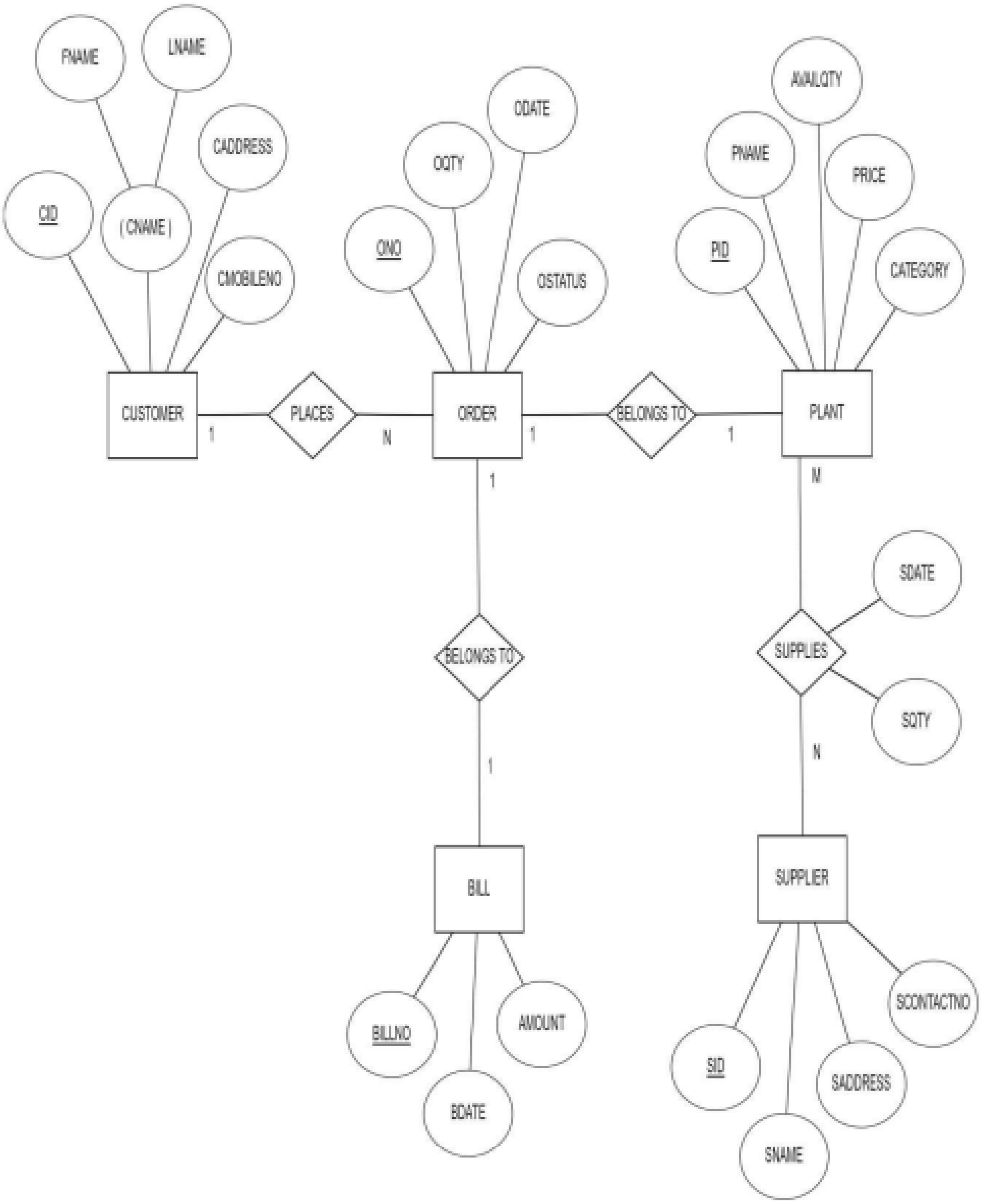


Figure 3.2: ER Diagram of Payroll Management System

3.2 Mapping From ER Diagram to Schema Diagram

STEP 1:

For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E. Include only the simple component attributes of a composite attribute. Choose one of the key attributes of E as primary key for R. If the chosen key of

E is composite, the set of simple attributes that form it will together form the primary key of R.

STEP 2:

For each weak entity type W in the ER schema with owner entity type E, create a relation R, and include all simple attributes (or simple components of composite attributes) of W as attributes of R. In addition, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s); this takes care of the identifying relationship type of W. The primary key of R is the combination of the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any.

STEP 3:

For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. Choose one of the relations — S, say — and include as foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S. Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type R as attributes of S.

STEP 4:

For each regular binary 1:N relationship type R, identify the relation S that represents the participating entity type at the N-side of the relationship type. Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R; this is because each entity instance on the N-side is related to at most one entity instance on the 1-side of the relationship type. Include any simple attributes (or simple components of composite attributes) of the 1:N relationship type as attributes of S.

STEP 5:

For each binary M:N relationship type R, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S. Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S. Notice that we cannot represent an M:N relationship type by a single foreign key attribute in one of the participating relations — as we did for 1:1 or 1:N relationship types — because of the M:N cardinality ratio.

STEP 6:

For each multivalued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K — as a foreign key in R —

of the relation that represents the entity type or relationship type that has A as an attribute. The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components (Note 4).

STEP 7:

For each n-ary relationship type R, where n>2, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types. Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S. The primary key of S is usually a combination of all the foreign keys that reference the relations representing the participating entity types. However, if the cardinality constraints on any of the entity types E participating in R is 1, then the primary key of S should not include the foreign key attribute that references the relation E' corresponding to E.

RELATIONAL SCHEMA (E-R TO RELATIONAL SCHEMA)

STEP 1: MAPPING OF REGULAR ENTITY TYPES:

CUSTOMER

CID	FNAME	LNAME	CADDRESS	CMOBILENO

ORDER

OID	OQTY	ODATE	OSTATUS

PLANT

PID	PNAME	PQTY	PPRICE	CATEGORY

SUPPLIER

SID	SNAME	SADDRESS	SCONTACTNO

BILL

BILLNO	BADTE	AMOUNT

STEP 2: MAPPING OF WEAK ENTITY TYPES:

The ERD of our project does not contain any weak entities.

STEP 3: MAPPING OF BINARY 1:1 RELATION TYPES:

There are two 1:1 relation in our table.

ORDER

OID	OQTY	ODATE	OSTATUS	PID

BILL

BILLNO	BDATE	AMOUNT	OID	PID

STEP 4: MAPPING OF 1:N RELATION TYPES:

There are two 1:N relation in our table.

ORDER

OID	OQTY	ODATE	OSTATUS	CID	PID

STEP 5: MAPPING OF M:N RELATION TYPES:

There is only one M:N relation in our table.

SUPPLIES

PID	SID	SDATE	SQTY

STEP 6: MAPPING OF MULTIVALUED ATTRIBUTES:

The ERD of our project does not contain any multivalued attributes.

STEP 7: MAPPING OF N-ARY RELATION TYPES:

The ERD of our project does not contain any N-ary relation types.

3.3 Schema Diagram

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.It defines tables, views, and integrity constraints.

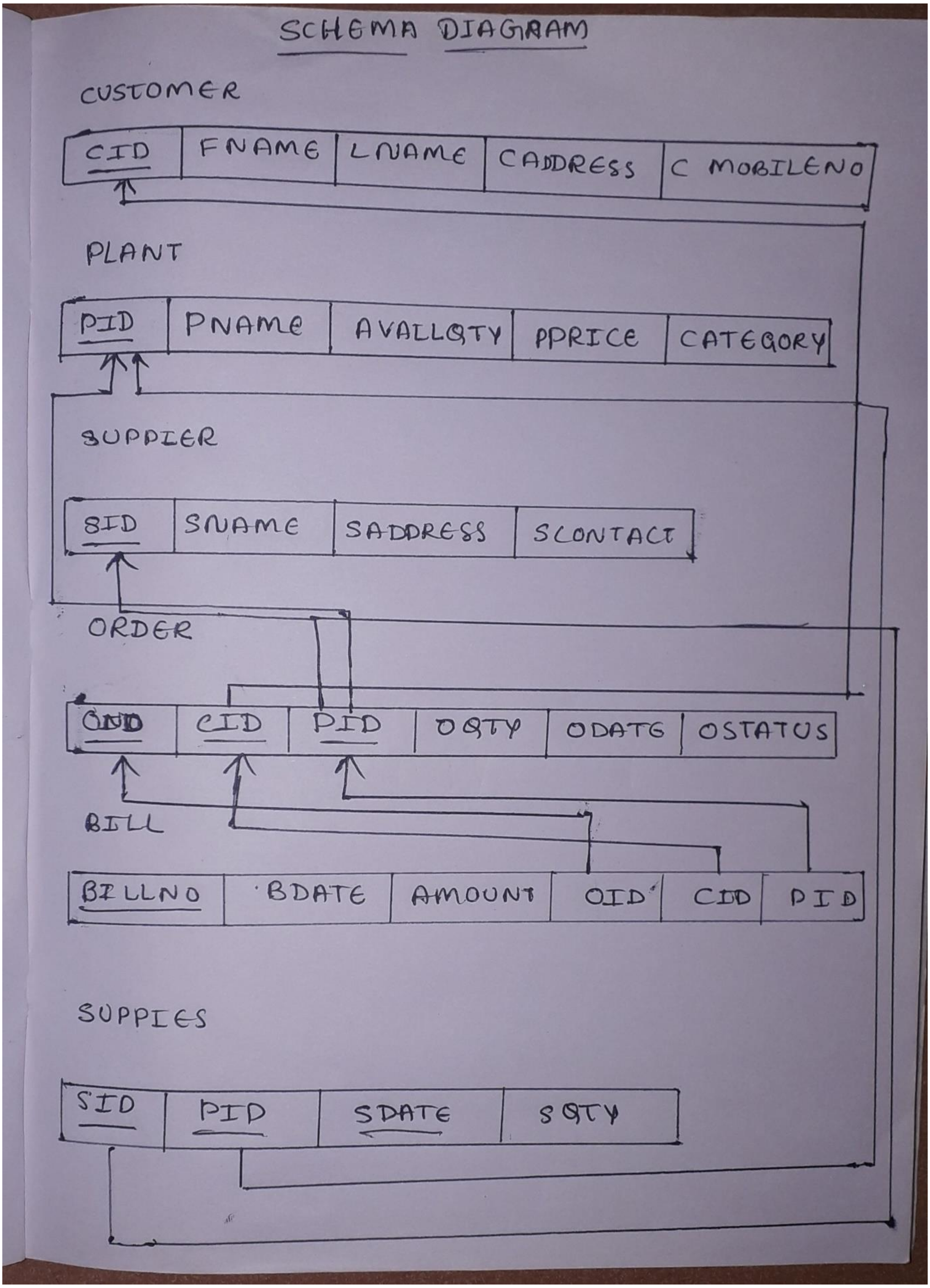


Figure 3.6: Schema Diagram of Nursery Management System

Chapter 4

Implementation

4.1 Table Structure

CUSTOMER TABLE

CREATE TABLE CUSTOMER(CID NUMBER(4),FNAME VARCHAR(15),LNAME VARCHAR(15),CADDRESS VARCHAR(20),CMOBILENO NUMBER(10),CONSTRAINT CONPK_CID PRIMARY KEY(CID));

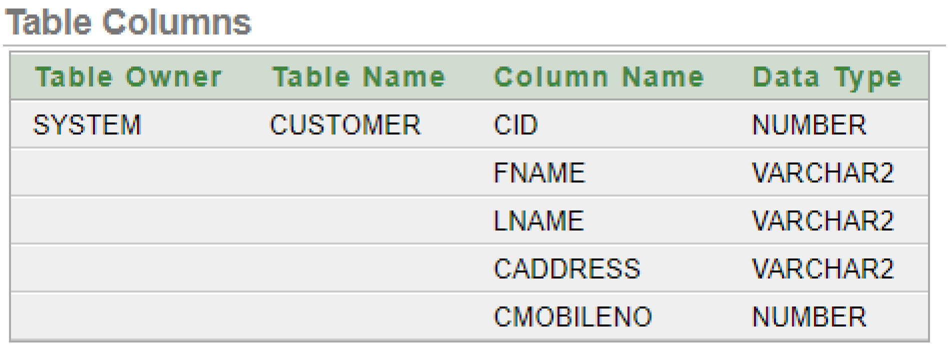


Table Columns

Table Owner	Table Name	Column Name	Data Type
SYSTEM	CUSTOMER	CID	NUMBER
		FNAME	VARCHAR2
		LNAME	VARCHAR2
		CADDRESS	VARCHAR2
		CMOBILENO	NUMBER

Figure 4.1: CUSTOMER Table

PLANT TABLE

CREATE TABLE PLANT (PID NUMBER(4),PNAME VARCHAR(20),PRICE NUMBER(7,2),CATEGORY

```
VARCHAR(20),AVAILQTY NUMBER(5),  CONSTRAINT
CON_PK PRIMARY KEY(PID));
```

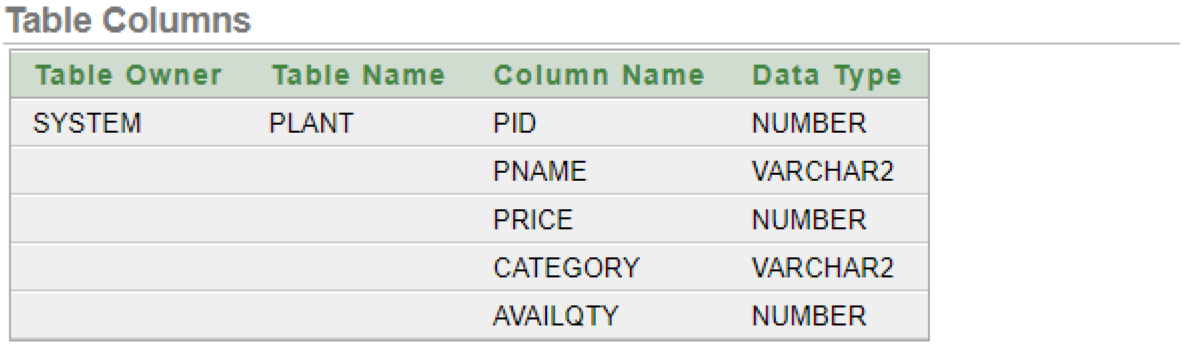


Table Columns

Table Owner	Table Name	Column Name	Data Type
SYSTEM	PLANT	PID	NUMBER
		PNAME	VARCHAR2
		PRICE	NUMBER
		CATEGORY	VARCHAR2
		AVAILQTY	NUMBER

Figure 4.2: PLANT Table

ORDER TABLE

```
CREATE TABLE ORDERS(OID NUMBER(4),PID NUMBER(4),CID
NUMBER(4),ODATE  DATE,OSTATUS VARCHAR(15),OQTY NUMBER(4),
CONSTRAINT CONPK_ORDER PRIMARY
KEY(OID,PID,CID),CONSTRAINT  CONFK_PID FOREIGN
KEY(PID) REFERENCES PLANT(PID),
CONSTRAINT CONFK_CID FOREIGN KEY(CID) REFERENCES
CUSTOMER(CID));
```

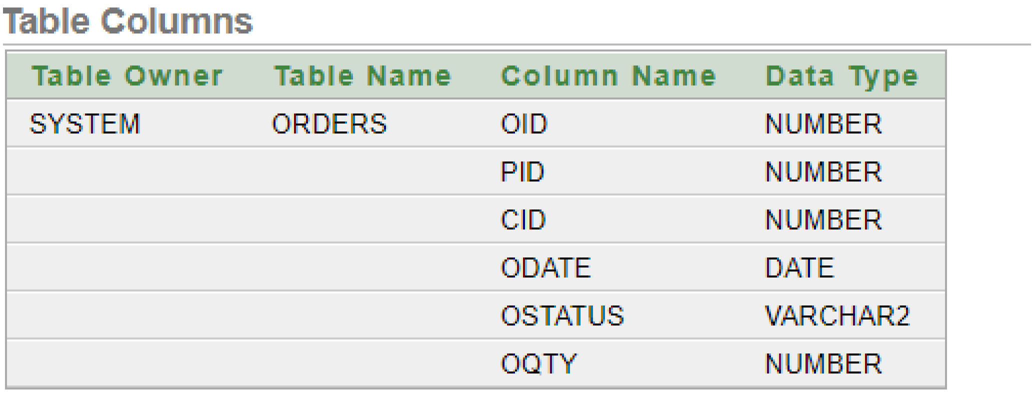


Table Columns

Table Owner	Table Name	Column Name	Data Type
SYSTEM	ORDERS	OID	NUMBER
		PID	NUMBER
		CID	NUMBER
		ODATE	DATE
		OSTATUS	VARCHAR2
		OQTY	NUMBER

Figure 4.3: ORDER Table

SUPPLIER TABLE

```
CREATE TABLE SUPPLIER(SID NUMBER(4),SNAME
VARCHAR(20),SADDRESS  VARCHAR(20),SMOBILENO
NUMBER(10),
CONSTRAINT CONPK_SID PRIMARY KEY(SID));
```

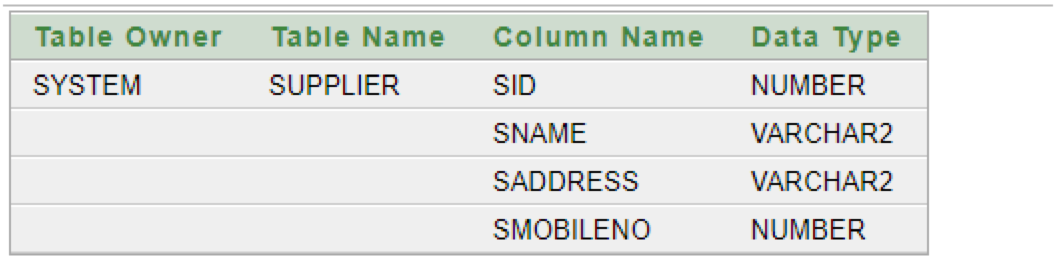
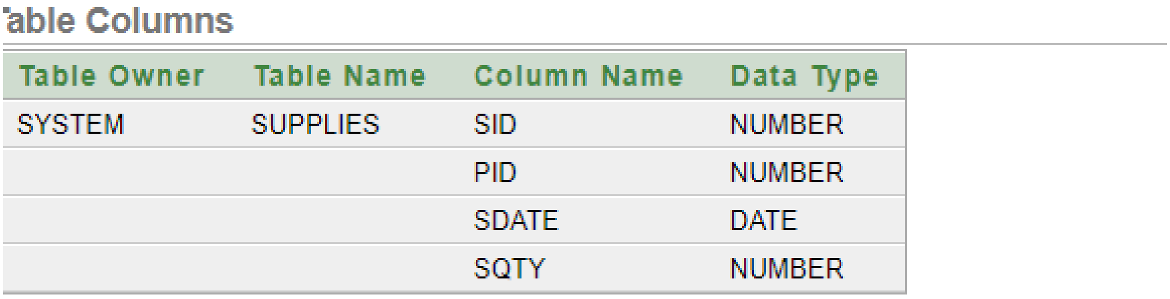


Table Owner	Table Name	Column Name	Data Type
SYSTEM	SUPPLIER	SID	NUMBER
		SNAME	VARCHAR2
		SADDRESS	VARCHAR2
		SMOBILENO	NUMBER

Figure 4.4:SUPPLIER Table

SUPPLIES TABLE

CREATE TABLE SUPPLIES(SID NUMBER(4),PID NUMBER(4),SDATE DATE,SQTY NUMBER(5),CONSTRAINT CONPK_S PRIMARY KEY(SID,PID,SDATE), CONSTRAINT CONFK_P FOREIGN KEY(PID) REFERENCES PLANT(PID), CONSTRAINT CONFK_S FOREIGN KEY(SID) REFERENCES SUPPLIER(SID));



able Columns

Table Owner	Table Name	Column Name	Data Type
SYSTEM	SUPPLIES	SID	NUMBER
		PID	NUMBER
		SDATE	DATE
		SQTY	NUMBER

Figure 4.5: SUPPLIES Table

BILL TABLE

CREATE TABLE BILL(BILLNO NUMBER(4),BILLDATE DATE,OID NUMBER(4),PID NUMBER(4),CID NUMBER(4),AMOUNT NUMBER(10,2),
CONSTRAINT CONPK_BID PRIMARY KEY(BILLNO),CONSTRAINT CONFK_OID FOREIGN KEY(OID,PID,CID) REFERENCES ORDERS(OID,PID,CID));

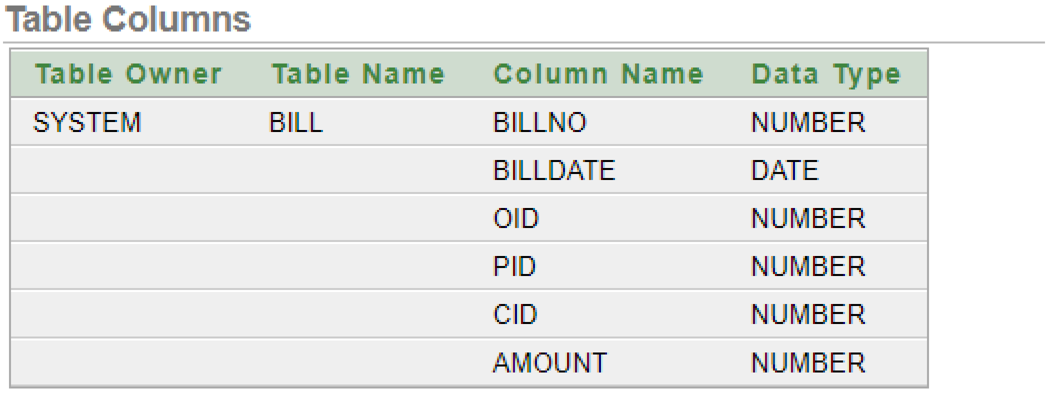
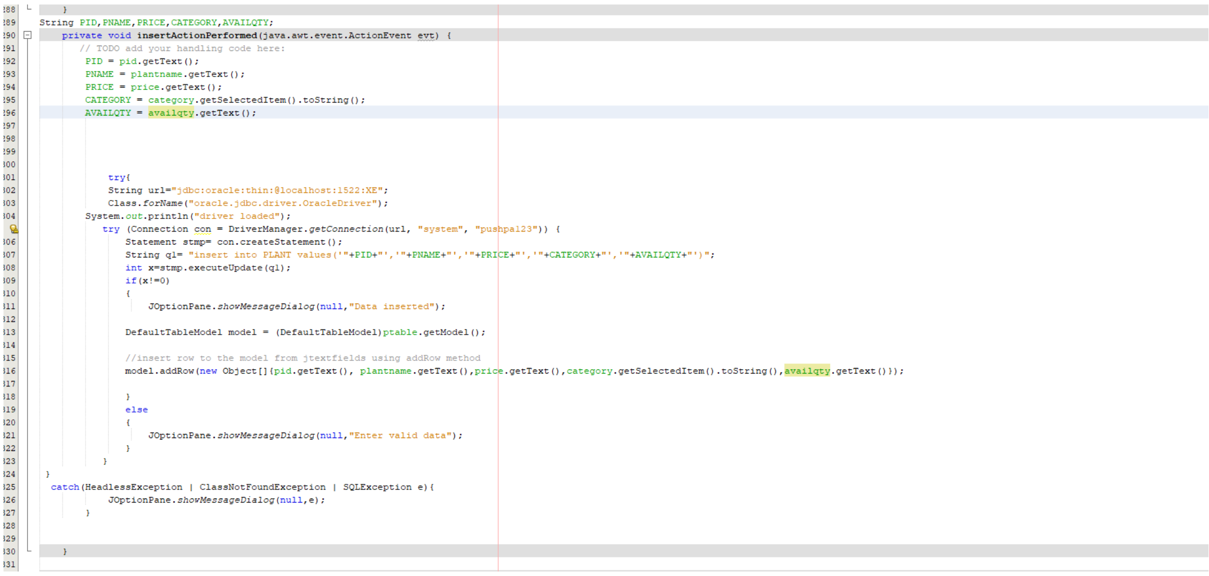


Table Columns

Table Owner	Table Name	Column Name	Data Type
SYSTEM	BILL	BILLNO	NUMBER
		BILLDATE	DATE
		OID	NUMBER
		PID	NUMBER
		CID	NUMBER
		AMOUNT	NUMBER

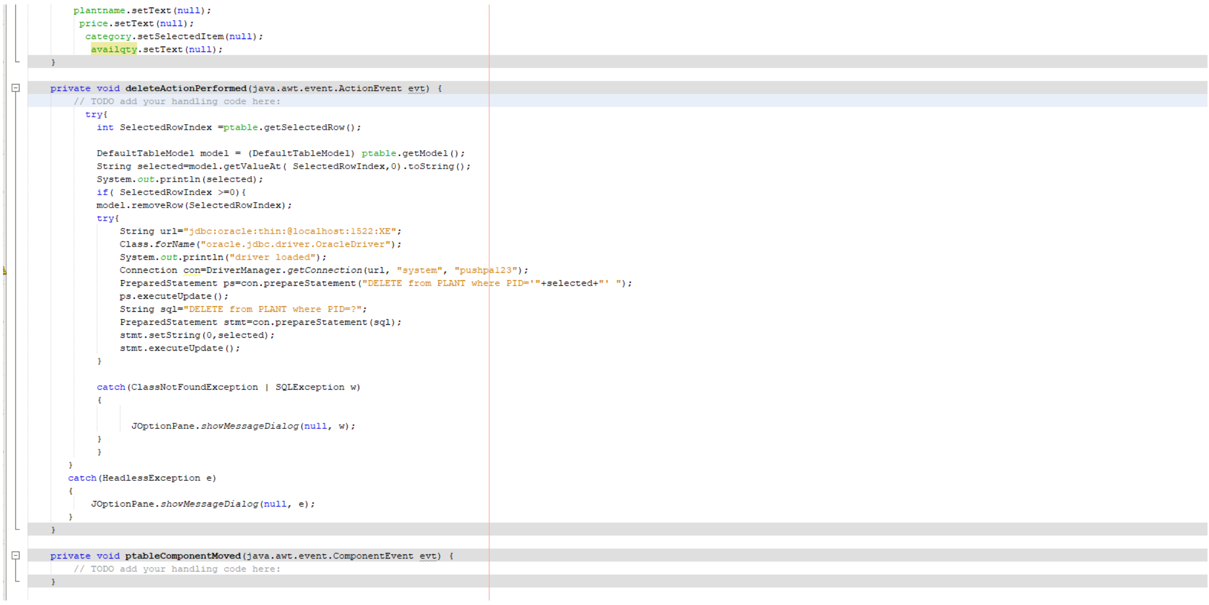
Figure 4.6:BILL Table

4.2 Codes for the Modules



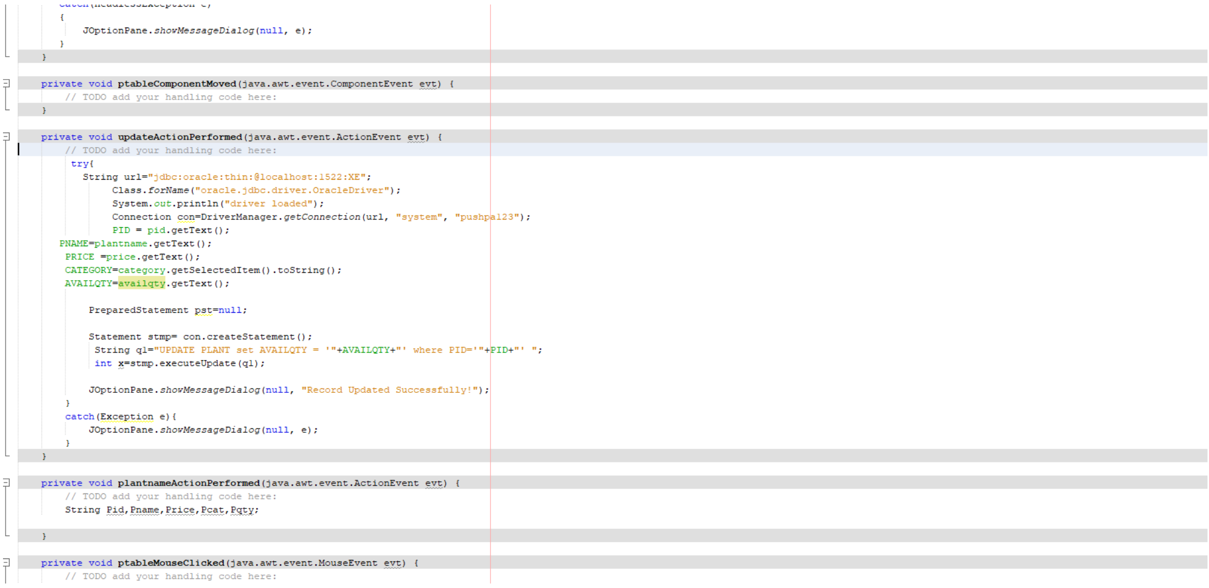
```
    }
String PID,PNAME,PRICE,CATEGORY,AVAILQTY;
    private void insertActionPerformed(java.awt.event.ActionEvent evt) {
       // TODO add your handling code here:
        PID = pid.getText();
        PNAME = plantname.getText();
        PRICE = price.getText();
        CATEGORY = category.getSelectedItem().toString();
        AVAILQTY = availqty.getText();




            try{
            String url="jdbc:oracle:thin:@localhost:1522:XE";
            Class.forName("oracle.jdbc.driver.OracleDriver");
        System.out.println("driver loaded");
           try (Connection con = DriverManager.getConnection(url, "system", "pushpal23")) {
               Statement stmp= con.createStatement();
               String ql= "insert into PLANT values('"+PID+"','"+PNAME+"','"+PRICE+"','"+CATEGORY+"','"+AVAILQTY+"')";
               int x=stmp.executeUpdate(ql);
               if(x!=0)
               {
                   JOptionPane.showMessageDialog(null,"Data inserted");

               DefaultTableModel model = (DefaultTableModel)ptable.getModel();

               //insert row to the model from jtextfields using addRow method
               model.addRow(new Object[]{pid.getText(), plantname.getText(),price.getText(),category.getSelectedItem().toString(),availqty.getText()});

               }
               else
               {
                   JOptionPane.showMessageDialog(null,"Enter valid data");
               }
           }
}
 catch(HeadlessException | ClassNotFoundException | SQLException e){
            JOptionPane.showMessageDialog(null,e);
        }


    }
```

Figure 4.6: Insert Table



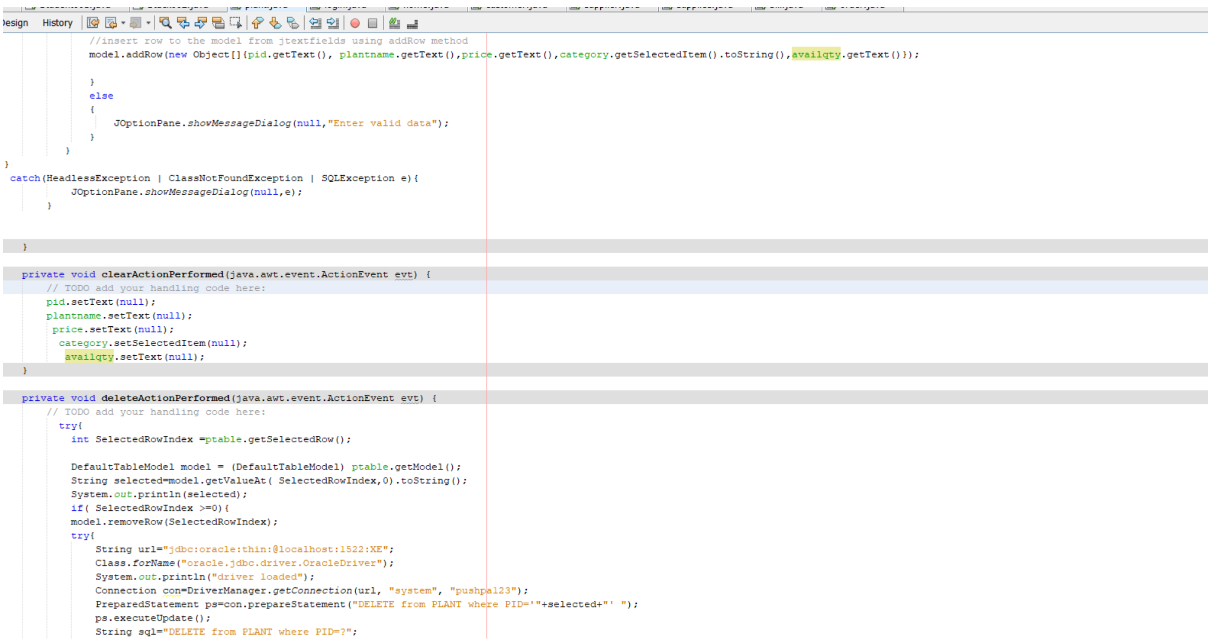
```
            plantname.setText(null);
             price.setText(null);
              category.setSelectedItem(null);
               availqty.setText(null);
    }

    private void deleteActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
          try{
            int SelectedRowIndex =ptable.getSelectedRow();

            DefaultTableModel model = (DefaultTableModel) ptable.getModel();
            String selected=model.getValueAt( SelectedRowIndex,0).toString();
            System.out.println(selected);
            if( SelectedRowIndex >=0){
            model.removeRow(SelectedRowIndex);
            try{
                String url="jdbc:oracle:thin:@localhost:1522:XE";
                Class.forName("oracle.jdbc.driver.OracleDriver");
                System.out.println("driver loaded");
                Connection con=DriverManager.getConnection(url, "system", "pushpal23");
                PreparedStatement ps=con.prepareStatement("DELETE from PLANT where PID='"+selected+"' ");
                ps.executeUpdate();
                String sql="DELETE from PLANT where PID=?";
                PreparedStatement stmt=con.prepareStatement(sql);
                stmt.setString(0,selected);
                stmt.executeUpdate();
            }

            catch(ClassNotFoundException | SQLException w)
            {

                  JOptionPane.showMessageDialog(null, w);
            }
            }
       }
       catch(HeadlessException e)
       {
           JOptionPane.showMessageDialog(null, e);
       }
    }

    private void ptableComponentMoved(java.awt.event.ComponentEvent evt) {
        // TODO add your handling code here:
    }
```

Figure 4.7: Delete Table



```
       catch(HeadlessException e)
       {
           JOptionPane.showMessageDialog(null, e);
       }
    }

    private void ptableComponentMoved(java.awt.event.ComponentEvent evt) {
        // TODO add your handling code here:
    }

    private void updateActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
         try{
           String url="jdbc:oracle:thin:@localhost:1522:XE";
                Class.forName("oracle.jdbc.driver.OracleDriver");
                System.out.println("driver loaded");
                Connection con=DriverManager.getConnection(url, "system", "pushpal23");
                PID = pid.getText();
       PNAME=plantname.getText();
        PRICE =price.getText();
        CATEGORY=category.getSelectedItem().toString();
        AVAILQTY=availqty.getText();

            PreparedStatement pst=null;

            Statement stmp= con.createStatement();
             String ql="UPDATE PLANT set AVAILQTY = '"+AVAILQTY+"' where PID='"+PID+"' ";
             int x=stmp.executeUpdate(ql);

            JOptionPane.showMessageDialog(null, "Record Updated Successfully!");
        }
        catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
        }
    }

    private void plantnameActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String Pid,Pname,Price,Pcat,Pqty;

    }

    private void ptableMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
```

Figure 4.8: Update Table



```
                //insert row to the model from jtextfields using addRow method
                model.addRow(new Object[]{pid.getText(), plantname.getText(),price.getText(),category.getSelectedItem().toString(),availqty.getText()});

                }
                else
                {
                    JOptionPane.showMessageDialog(null,"Enter valid data");
                }
            }
}
 catch(HeadlessException | ClassNotFoundException | SQLException e){
            JOptionPane.showMessageDialog(null,e);
        }


    }

    private void clearActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        pid.setText(null);
        plantname.setText(null);
         price.setText(null);
          category.setSelectedItem(null);
           availqty.setText(null);
    }

    private void deleteActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
          try{
            int SelectedRowIndex =ptable.getSelectedRow();

            DefaultTableModel model = (DefaultTableModel) ptable.getModel();
            String selected=model.getValueAt( SelectedRowIndex,0).toString();
            System.out.println(selected);
            if( SelectedRowIndex >=0){
            model.removeRow(SelectedRowIndex);
            try{
                String url="jdbc:oracle:thin:@localhost:1522:XE";
                Class.forName("oracle.jdbc.driver.OracleDriver");
                System.out.println("driver loaded");
                Connection con=DriverManager.getConnection(url, "system", "pushpal23");
                PreparedStatement ps=con.prepareStatement("DELETE from PLANT where PID='"+selected+"' ");
                ps.executeUpdate();
                String sql="DELETE from PLANT where PID=?";
```

Figure 4.9: Clear Table

TRIGGER

Trigger to increase the availability of plant quantity when supplier supplies particular amount of plant.

CREATE OR REPLACE TRIGGER PLANTORDER1

AFTER INSERT ON SUPPLIES

FOR EACH ROW

BEGIN

UPDATE PLANT SET AVAILQTY=AVAILQTY+:NEW.SQTY;

END;

Trigger to decrease the availability of plant quantity when customer orders particular amount of plant.

CREATE OR REPLACE TRIGGER PLANTORDER

AFTER INSERT ON ORDERS

FOR EACH ROW

BEGIN

UPDATE PLANT SET AVAILQTY=AVAILQTY-:NEW.OQTY;

END;

Trigger to update the order status when admin generates a bill for that particular order.

```
CREATE OR REPLACE TRIGGER ORDBILL2

AFTER INSERT ON BILL

FOR EACH ROW

BEGIN

UPDATE ORDERS SET OSTATUS='ORDERED' WHERE
OID=:NEW.OID AND  PID=:NEW.PID AND CID=:NEW.CID;
END;
```

Chapter 5

Results

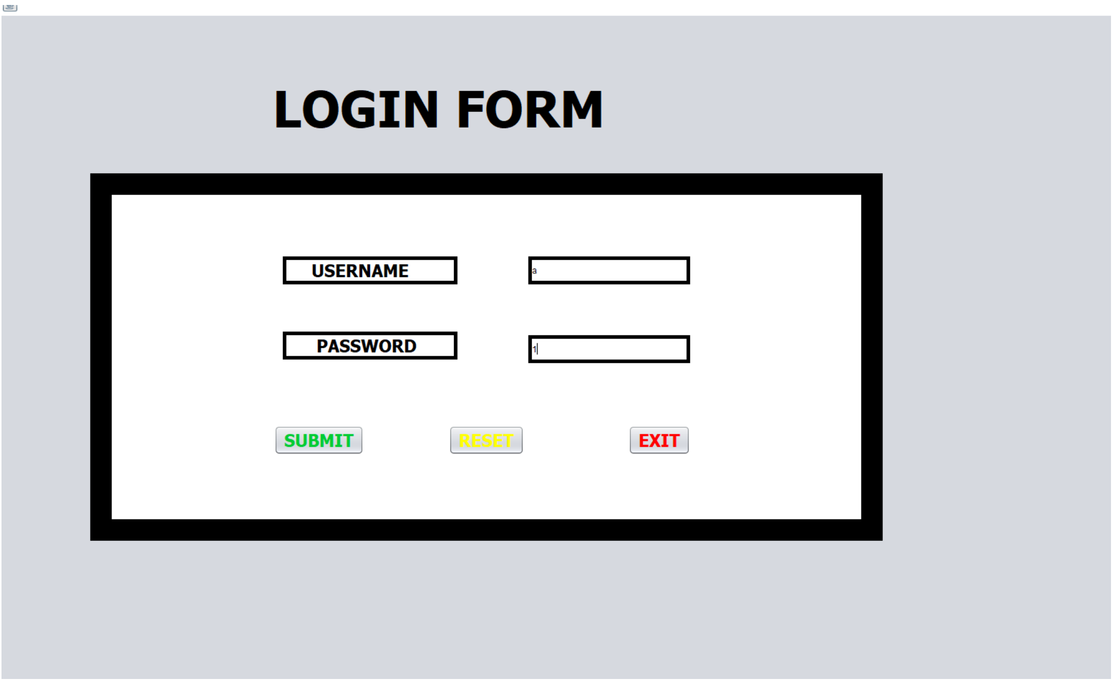


Figure 5.1: Login Page

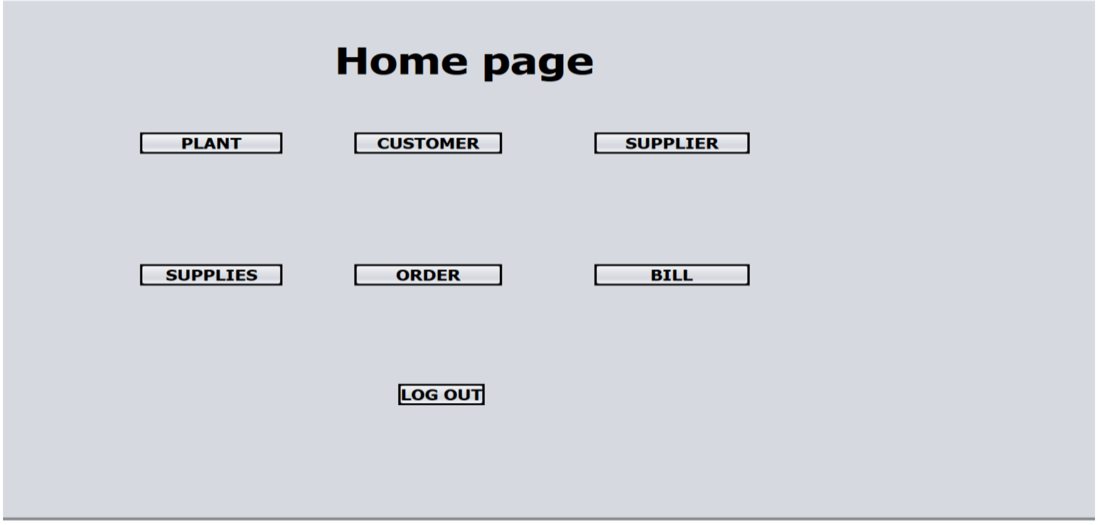
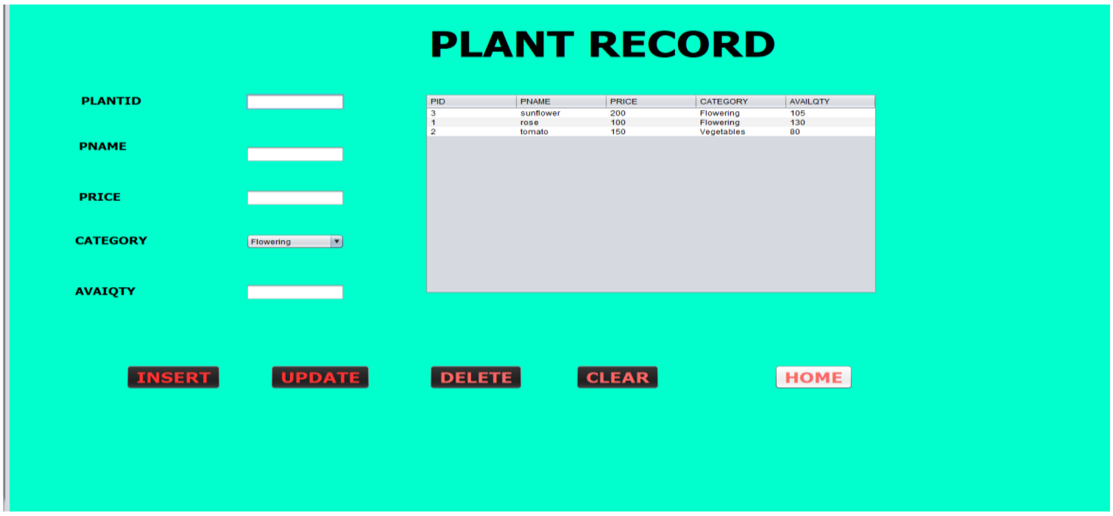


Figure 5.2: Home Page



PLANT RECORD

PLANTID

PNAME

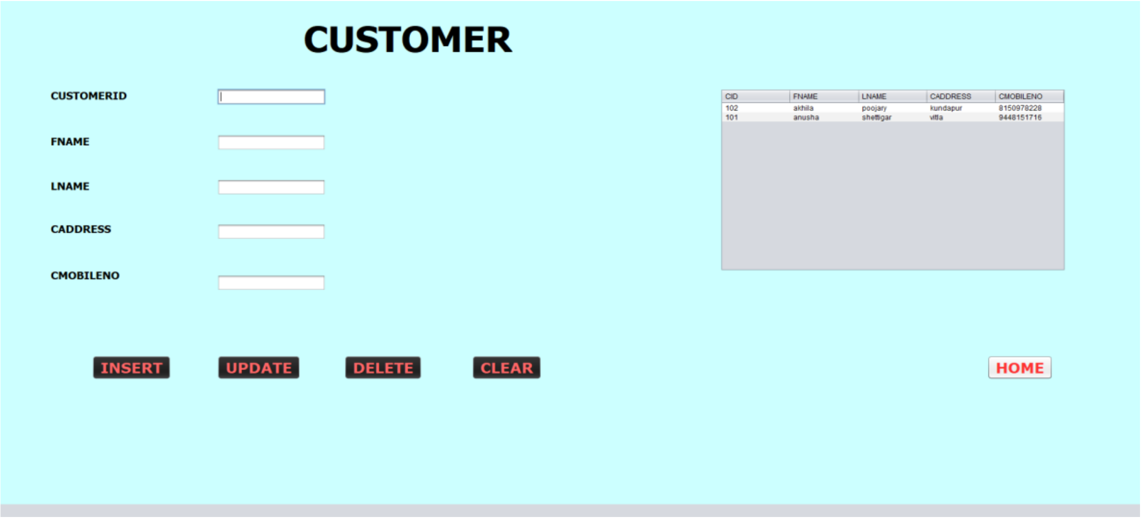
PRICE

CATEGORY Flowering

AVAIQTY

PID	PNAME	PRICE	CATEGORY	AVAILQTY
3	sunflower	200	Flowering	105
1	rose	100	Flowering	130
2	tomato	150	Vegetables	80

Figure 5.3: Plant Record Page



CUSTOMER

CUSTOMERID

FNAME

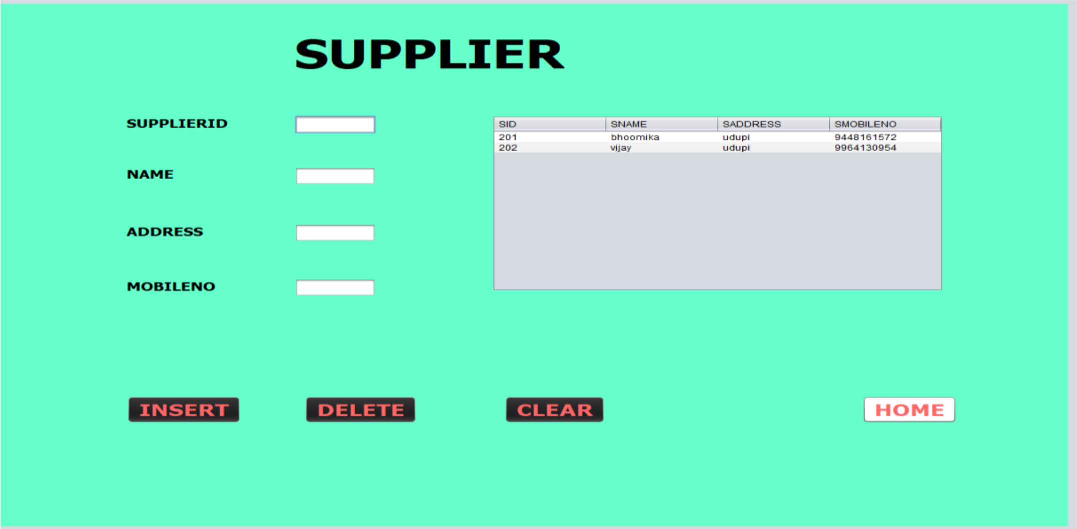
LNAME

CADDRESS

CMOBILENO

CID	FNAME	LNAME	CADDRESS	CMOBILENO
102	akhila	poojary	kundapur	8150978228
101	anusha	shettigar	vitla	9448151716

Figure 5.4: Customer Page



SUPPLIER

SUPPLIERID

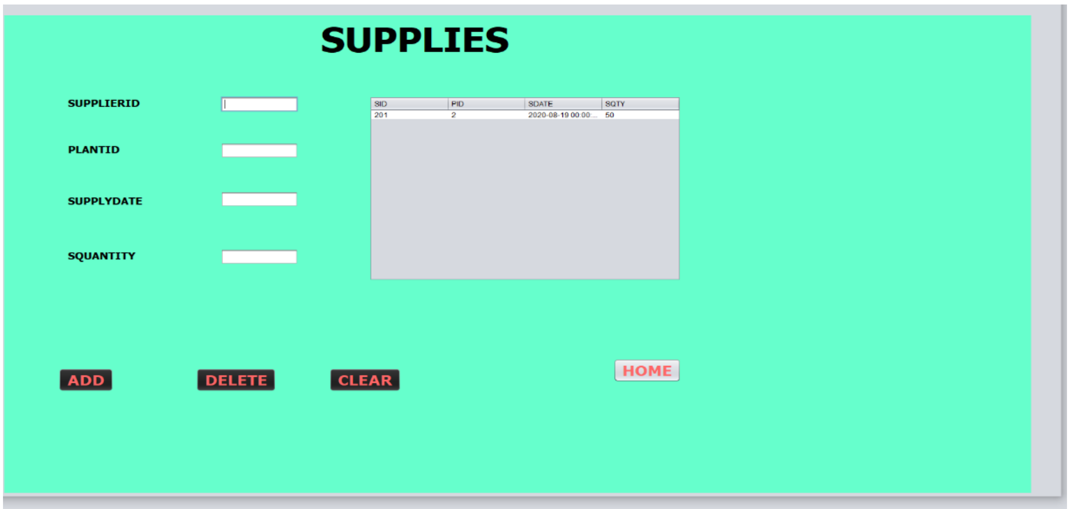
NAME

ADDRESS

MOBILENO

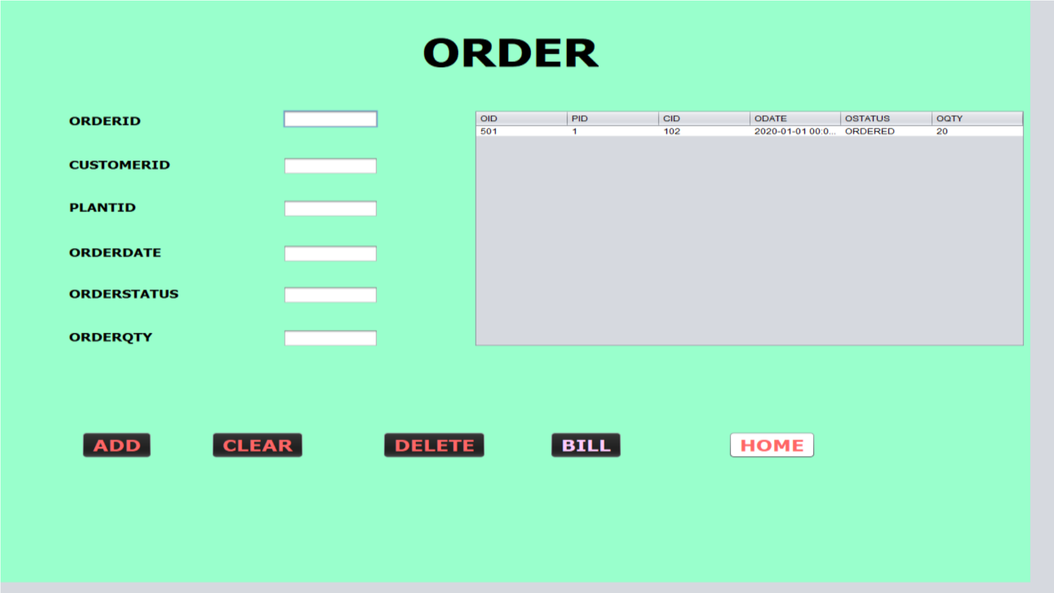
SID	SNAME	SADDRESS	SMOBILENO
201	bhoomika	udupi	9448161572
202	vijay	udupi	9964130954

Figure 5.5: Supplier Page



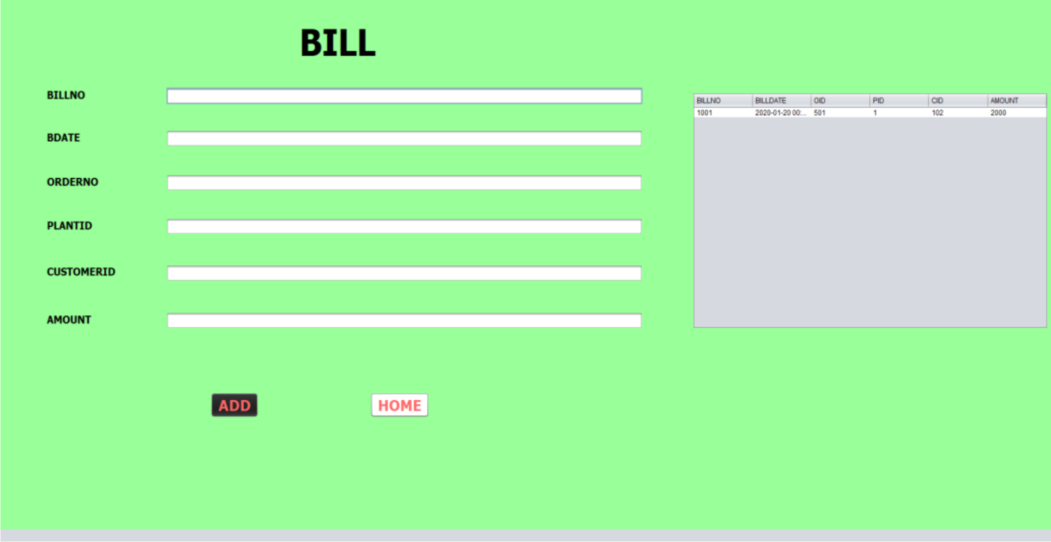
SID	PID	SDATE	SQTY
201	2	2020-08-19 00:00:...	50

Figure 5.6: Supplies Page



OID	PID	CID	ODATE	OSTATUS	OQTY
501	1	102	2020-01-01 00:0...	ORDERED	20

Figure 5.7: Order Page



BILLNO	BILLDATE	OID	PID	CID	AMOUNT
1001	2020-01-20 00:...	501	1	102	2000

Figure 5.8: Bill Page

Chapter 6

Conclusion and Future work

It has been a matter of immense pleasure, honor and challenge to have this opportunity to take up this project and complete it successfully. While developing this project we have learnt a lot about Plant Nursery Management and the hardships customers go through to have a little salon time. We have also learnt how to make the system user friendly (easy to use and handle) by hiding the complicated parts of it from users. The project has a simple clean UI for the admin to use and to make changes according to the requirements. During the development process we studied carefully and understood criteria for making Software more demanding. We have tried to implement the project making it as user friendly and error free as possible.

References

[1] Database systems Models, Languages, Design and Application Programming, Ramez Elmasri and Shamkant B. Navathe, 6th Edition, Pearson.

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

[3] Silberschatz Korth and Sudharshan: Database System Concepts, 6th Edition, McGraw Hill, 2013.

[4] Coronel, Morris, and Rob, Database Principles Fundamentals of Design, Implementation and Management, Cengage Learning 2012.