Electricity Billing System

MINOR PROJECT REPORT

By

Archisman Hes [Reg. No.: RA2211003010273]
B.Tech. CSE - Core

Saloni Bhardwaj [Reg. No.: RA2211003010268]
B.Tech. CSE – Core

Shovik Banerjee [Reg. No.: RA2211003010270]
B.Tech. CSE - Core

Under the guidance of

Dr. Ajanthaa Lakkshmanan

In partial fulfilment for the Course

of

21CSC203P – ADVANCED PROGRAMMING PRACTICE

in Computational intelligence

FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR

NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this minor project report for the course **21CSC203P** ADVANCED **PROGRAMMING PRACTICE** entitled in **electricity billing system** is the bonafide work of **ARCHISMAN HES [RA2211003010273], SALONI BHARDWAJ [RA2211003010268] And SHOVIK BANERJEE [RA2211003010270]** who carried out the work under my supervision.

SIGNATURE

Dr. Ajanthaa Lakkshmanan

Assistant Professor
Computing Technology
SRM Institute of Science and Technology
Kattankulathur

SIGNATURE

Dr. M. Pushpalatha
Head of department
Computing Technology
Srm institute of science and Technology
Kattankulathur

ABSTRACT

The Electricity Billing System Project in Java is a comprehensive software solution designed to streamline and enhance the management of electricity billing processes. This project aims to provide utility companies and their customers with an efficient and user-friendly platform for billing, payment, and consumption tracking.

Key features of the system include customer registration, meter reading input, automated bill generation, and secure online payment options. Customers can access their billing history and consumption patterns through a user-friendly web interface, promoting transparency and accountability. Utility companies can automate meter reading collection, reduce human errors, and improve overall efficiency in managing their billing operations.

The Java-based system employs object-oriented programming principles to ensure scalability and maintainability. With this project, the electricity billing process becomes more accurate, efficient, and accessible, benefitting both utility companies and their customers.

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy,** for his encouragement.

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal,** for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman,** for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professors Dr. Vadivu. G , Professor, Department of Data Science and Business Systems and Dr. Sasikala. E Professor, Department of Data Science and Business Systems** and **Course Coordinators** for their constant encouragement and support.

We are highly thankful to my Course project **Dr. Ajanthaa Lakkshmanan, Ast. Professor, Department of Computing Technologies,** for his/her assistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our **HOD Dr. M. Pushpalatha, Professor, C. Tech Department** and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

TABLE OF CONTENTS

CHAPTER -1

INTRODUCTION

Electricity Billing System is a software-based application.

i. This project aims at serving the department of electricity by computerizing the billing system.

ii. It mainly focuses on the calculation of units consumed during the specified time and the money to be charged by the electricity offices.

iii. This computerized system will make the overall billing system easy, accessible, comfortable, and effective for consumers.

To design the billing system more service oriented and simple, the following features have been implemented in the project. The application has high speed of performance with accuracy and efficiency.

The software provides facility of data sharing, it does not require any staff as in the conventional system. Once it is installed on the system only the meter readings are to be given by the admin where customer can view all details, it has the provision of security restriction.

The electricity billing software calculates the units consumed by the customer and makes bills, it requires small storage for installation and functioning. There is provision for debugging if any problem is encountered in the system.

The system excludes the need of maintaining paper electricity bill, administrator does not have to keep a manual track of the users, users can pay the amount without visiting the office. Thus, it saves human efforts and resources.

PURPOSE-

We, the owners of our project, respect all customers and make them happy with our service.

The main aim of our project is to satisfy customer by saving their time by payment process, maintaining records, and allowing the customer to view his/her records and permitting them to update their details . The firm handles all the work manually, which is very tedious and mismatched

SCOPE-

The scope of our project are as follows:

• To keep the information of consuming unit energy of current month

- To keep the information of Customer.
- To keep the information of consuming unit energy of previous month.
- To calculate the units consumed every month regularly.
- To generate the bills adding penalty and rent. To save the time by implementing payment process online.

APPLICABILITY-

The manual system is suffering from a series of drawbacks. Since whole of the bills is to be maintained with hands the process of keeping and maintaining the information is very tedious and lengthy to customer. It is very time consuming and laborious process because, staff need to be visited the customers place every month to give the bills and to receive the payments. For this reason, we have provided features Present system is partially automated (computerized), existing system is quite laborious as one must enter same information at different places.

CHAPTER -2

ANALYSIS AND SYSTEM REQUIREMENT

Existing and Proposed System :-

The conventional system of electricity billing is not so effective; one staff must visit each customer's house to note the meter readings and collect the data. Then, another staff must compute the consumed units and calculate the money to be paid. Again, the bills prepared are to be delivered to customers. Finally, individual customer must go to electricity office to pay their dues.

Hence, the conventional electricity billing system is uneconomical, requires many staffs to do simple jobs and is a lengthy process overall. In order to solve this lengthy process of billing, a web based computerized system is essential. This proposed electricity billing system project overcomes all these drawbacks with the features. It is beneficial to both consumers and the company which provides electricity.

With the new system, there is reduction in the number of staffs to be employed by the company. The working speed and performance of the software is faster with high performance which saves time. Furthermore, there is very little chance of miscalculation and being corrupted by the staffs.

Software & Hardware Requirements :-

Hardware Requirements:

- ➢ Hardware Specification: -Processor Intel Pentium V or higher
- ➢ Clock Speed: -1.7 GHz or more
- ➢ System Bus: -64 bits
- ➢ RAM: -16GB
- ➢ HDD: -2TB
- ➢ Monitor: -LCD Monitor
- ➢ Keyboard: -Standard keyboard
- ➢ Mouse: -Compatible mouse Software Requirements:-
- ➢ Operating System: -Windows 10
- ➢ Software: -Microsoft SQL Server
- ➢ Front End: -Java core/swings (NetBeans)
- ➢ Back End: -My SQL

CHAPTER 3

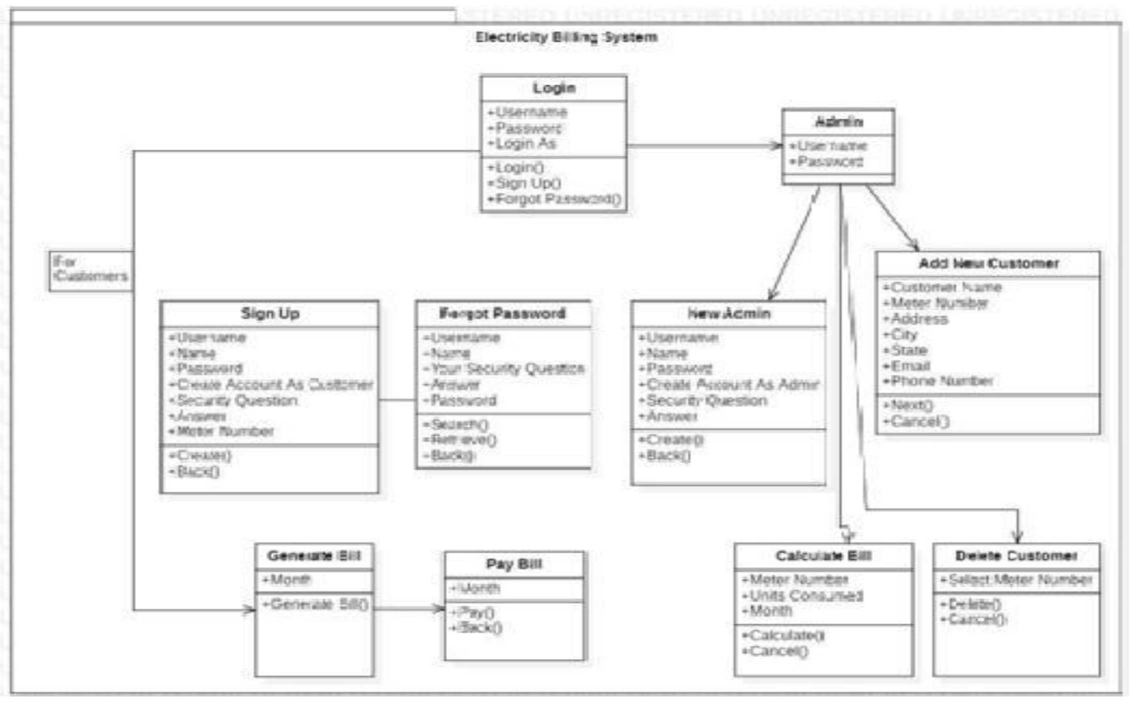
SYSTEM DESIGN AND MODELING

Preliminary Design:-

System design is an abstract representation of a system component and their relationship and which describe the aggregated functionally and performance of the system. It is also the plan or blueprint for how to obtain answer to the question being asked. The design specifies various type of approach.

Database design is one of the most important factors to keep in mind if you are concerned with application performance management. By designing your database to be efficient in each call it makes and to effectively create rows of data in the database, you can reduce the amount of CPU needed by the server to complete your request, thereby ensuring a faster application.

UML Diagram: -



Use Case Diagrams:

Schema Diagram: -

Database schema is described as database connections and constraints. It contains attributes. Every database has a state instance represent current set of databases with values. There are different types of keys in a database schema.

A primary key is a table column that can be used to uniquely identify every row of the table. Any column that has this property, these columns are called candidate key.

A composite primary key is a primary key consisting of more than one column. A foreign is a column or combination of columns that contains values that are found in the primary key of some table.

All the attributes of each table are interconnected by foreign key which is primary key in another column and composite key. Primary key cannot be null. The fact that many foreign key values repeat simply reflects the fact that its one- to-many relationship. In one-to-many relationship, the primary key has the one value and foreign key has many values.

Figure is a Schema diagram of Electricity Billing System which has six tables i.e., login, customer, tax, rent, bill, and meter_info where each table contain attributes some with primary key, foreign key. In the login table there are 6 attributes "meter_no", "username", "password", "user", "question", "answer".

The customer table has 7 attributes "name", "meter_no"(primary key), "address", "city", "state", "email", "phone". The rent table has 3 attributes "cost_per_unit"(primary key), " meter_rent", "service_charge". The tax table has 3 attributes " service_tax", "swacch_bharat_cess", "gst". The bill table has 5 attributes "meter_no"(foreign key that references the primary key of the customer table meter_no), "month", "units","total_bill", "status". The meter_info table has 6 attributes "meter_no"(foreign key that references the primary key of the customer table meter_no), "meter_location", "meter_type", "phase_code", " bill_type", "days ".

Schema Diagram:-

Login

Meter No	Username	Password	User	Question	Answer

Normalization:-

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

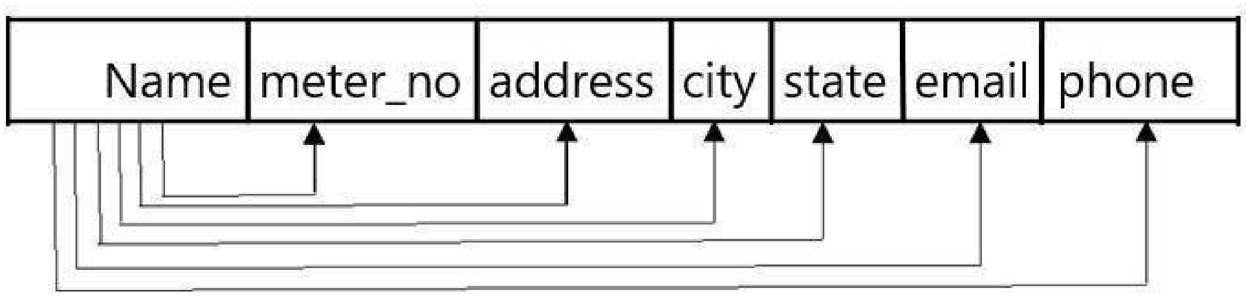
Let’s discuss about anomalies first then we will discuss normal forms with examples. Anomalies in DBMS There are three types of anomalies that occur when the database is not normalized. These are –Insertion, update and deletion anomaly.

First normal form(1NF) :-

As per the rule of first normal form,

- All rows must be unique (no duplicate rows).
- Each Cell must only contain a single value (not a list).
- Each value should be non-divisible (can’t be split down further).

Customer:-



Second normal form(2NF): -

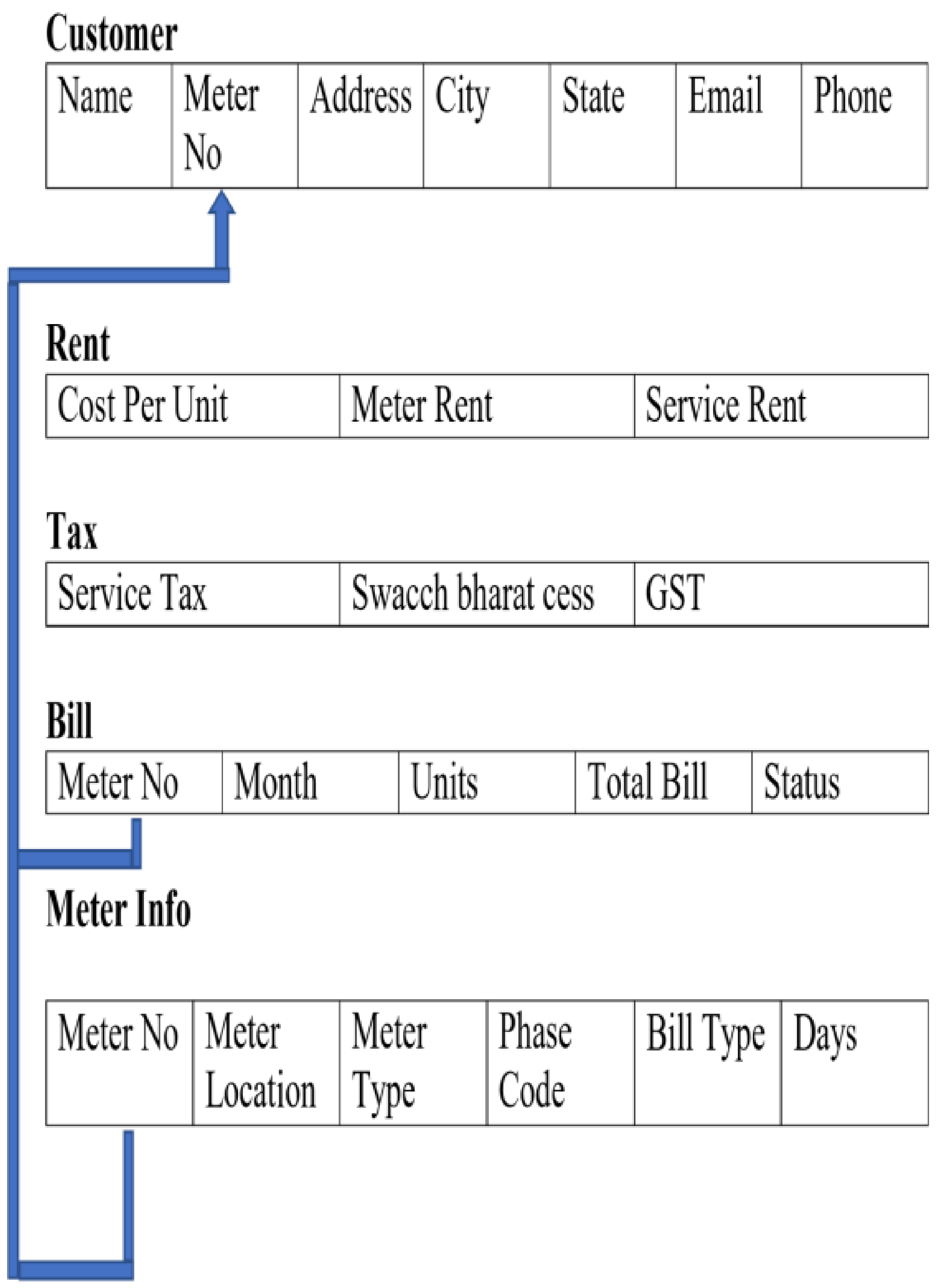
As per the rule of second normal form,

- Database must be in First Normal Form.
- Non partial dependency-All non-prime attributes should be fully functionally dependent on the candidate key.

Third normal form(3NF) :-

As per the rule of third normal form,

- Database must be in First and Second Normal Form.
- Non transitive dependency-All fields must only be determinable by the primary/composite key, not by other keys.



Customer

Name	Meter No	Address	City	State	Email	Phone

Rent

Cost Per Unit	Meter Rent	Service Rent

Tax

Service Tax	Swacch bharat cess	GST

Bill

Meter No	Month	Units	Total Bill	Status

Meter Info

Meter No	Meter Location	Meter Type	Phase Code	Bill Type	Days

CHAPTER 4

IMPLEMENTATION

Implementation of operations:-

• **Adding Customer:** Here admin can add new customer to the customer list who started using electricity bill system.

• **Searching Deposit Details:** Here admin can search according to meter number and month to view deposit details.

• **Viewing Details:** Here admin and user can view customer details and about details.

• **Adding Tax**: Here admin can add tax details.

• **Updating Customer:** Here customer can update his/her details by using meter_no of the customer.

• **Delete Customer:** Here admin can delete details based on meter number.

Implementation of SQL statements :-

Insert statement:

• The INSERT INTO statement is used to insert new records in a table.

• The INSERT INTO syntax would be as follows: INSERT INTO table_name VALUES (value1, value2, value3, ...).

• The following SQL statement insert's a new record in the “customer” table: Insert into customer VALUES (“sai”,”12345”,” btm”,” Bangalore”, “Karnataka”, “sai@gmail.com”, “9876543333”).

Update statement:

• An SQL UPDATE statement changes the data of one or more records in a table. Either all the rows can be updated, or a subset may be chosen using a condition.

• The UPDATE syntax would be as follows: UPDATE table_name SET column_name =value, column_name=value... [WHERE condition].

The following SQL statement update's a new record in the “customer” table: UPDATE TABLE customer SET email= su@gmail.com WHERE meter_no =”12345”.

Delete statement:

• The DELETE statement is used to delete existing records in a table.

• The DELETE syntax would be as follows: DELETE FROM table_name WHERE condition.

• The following SQL statement delete's a record in the “customer” table: delete from customer where meter_no=12345.

Create statement:

- The CREATE TABLE Statement is used to create tables to store data. Integrity Constraints like primary key, unique key, foreign key can be defined for the columns while creating the table.

- The syntax would be as follows: CREATETABLE table_name (column1datatype, column2datatype, column3 datatype, column datatype, PRIMARY KEY (one or more columns)).

- The following SQL statement creates a table “customer” table: create table customer (name varchar (30), meter_no varchar (20) primary key, address varchar (50), city varchar (20), state varchar (30), email varchar (30), phone varchar (30));

- The following SQL statement creates a table “login” table: create table login (meter_no varchar (30), username varchar (30), password varchar (30), user varchar (30), question varchar (40), answer varchar (30));

- The following SQL statement creates a table “tax” table: create table tax (cost_per_unit int (20) primary key, meter_rent int(20), service_charge int (20), service_tax int (20), swacch_bharat_cess int (20), gst int (20));

- The following SQL statement creates a table “bill” table: create table bill (meter_no varchar (20), foreign key(meter_no) references customer(meter_no) on delete cascade, month varchar (20), units int (20), total_bill int (20), status varchar (40));The following SQL statement creates a table “meter_info” table: create table varchar (10), meter_type varchar (15), phase_code int (5), bill_type varchar (10), days int (5));

Algorithm or pseudocode of implementation: -

Explanation of Algorithm or pseudocode of system:

✓ Start system

✓ Enter login name and password

✓ On clicking the login button

✓ Connect to database

✓ Query database to know whether user credentials are correct

✓ If not, deny access and return login page with an error message

✓ If correct, check if credentials for administrator

✓ If yes, allow login

✓ Set admin session, re-direct administrator to admin login page

✓ If no, allow login set user session

✓ Re-direct user to user home page

Login: -

This program will allow the admin to enter the username and password.

- If the entered credentials are correct, then the login will be successful otherwise need to be signup.
- If admin forgets password, it can be retrieved by giving username and answer for security question
- After successful login the admin will be redirected to admin portal page where he/she can do following activities

Code: - package electricity.billing.system;

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;

public class Login extends JFrame implements ActionListener{

    JButton login, cancel, signup;
    JTextField username, password;
    Choice logginin;
    Login() {
        super("Login Page");
        getContentPane().setBackground(Color.WHITE);
        setLayout(null);

        JLabel lblusername = new JLabel("Username");
        lblusername.setBounds(300, 20, 100, 20);
        add(lblusername);

        username = new JTextField();
        username.setBounds(400, 20, 150, 20);
        add(username);

        JLabel lblpassword = new JLabel("Password");
        lblpassword.setBounds(300, 60, 100, 20);
        add(lblpassword);

        password = new JTextField();
        password.setBounds(400, 60, 150, 20);
        add(password);

        JLabel loggininas = new JLabel("Loggin in as");
        loggininas.setBounds(300, 100, 100, 20);
        add(loggininas);

        logginin = new Choice();
        logginin.add("Admin");
        logginin.add("Customer");
        logginin.setBounds(400, 100, 150, 20);
        add(logginin);
        ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icon/login.png"));
        Image i2 = i1.getImage().getScaledInstance(16, 16, Image.SCALE_DEFAULT);
        login = new JButton("Login", new ImageIcon(i2));
        login.setBounds(330, 160, 100, 20);
        login.addActionListener(this);
        add(login);

        ImageIcon i3 = new ImageIcon(ClassLoader.getSystemResource("icon/cancel.jpg"));
        Image i4 = i3.getImage().getScaledInstance(16, 16, Image.SCALE_DEFAULT);
        cancel = new JButton("Cancel", new ImageIcon(i4));
        cancel.setBounds(450, 160, 100, 20);
        cancel.addActionListener(this);
        add(cancel);

        ImageIcon i5 = new ImageIcon(ClassLoader.getSystemResource("icon/signup.png"));
        Image i6 = i5.getImage().getScaledInstance(16, 16, Image.SCALE_DEFAULT);
        signup = new JButton("Signup", new ImageIcon(i6));
        signup.setBounds(380, 200, 100, 20);
        signup.addActionListener(this);
        add(signup);

        ImageIcon i7 = new ImageIcon(ClassLoader.getSystemResource("icon/second.jpg"));
        Image i8 = i7.getImage().getScaledInstance(250, 250, Image.SCALE_DEFAULT);
        ImageIcon i9 = new ImageIcon(i8);
        JLabel image = new JLabel(i9);
        image.setBounds(0, 0, 250, 250);
        add(image);

        setSize(640, 300);
        setLocation(400, 200);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent ae) {
        if (ae.getSource() == login) {
            String susername = username.getText();
            String spassword = password.getText();
            String user = logginin.getSelectedItem();

            try {
                Conn c = new Conn();
                String query = "select * from login where username = '"+susername+"' and password
= '"+spassword+"' and user = '"+user+"'";

                ResultSet rs = c.s.executeQuery(query);

                if (rs.next()) {
                    String meter = rs.getString("meter_no");
                    setVisible(false);
                    new Project(user, meter);
                } else {
                    JOptionPane.showMessageDialog(null, "Invalid Login");
                    username.setText("");
                    password.setText("");
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
        } else if (ae.getSource() == cancel) {
            setVisible(false);
        } else if (ae.getSource() == signup) {
            setVisible(false);

            new Signup();
        }
    }

    public static void main(String[] args) {
        new Login();
    }
}
```

New Customer: -

- This program will allow the admin to enter the customer details and automatically generates unique meter number.
- If customer name, address, city, state, email and phone number is entered, insert the values into customer

```
else print
error
while
next=true
enter the meter_info
details else print
meter_info error
```

- Submit the details of customer that has been entered by clicking onto next button.
- If we need to cancel the particulars that has been entered click onto cancel option.
- If we need to submit the particulars that has been entered click onto submit option.

Code: - package electricity.billing.system;

```
import javax.swing.*;
import java.awt.*;
import java.util.*;
import java.awt.event.*;

public class NewCustomer extends JFrame implements
ActionListener{

    JTextField tfname, tfaddress, tfstate, tfcity, tfemail, tfphone;
    JButton next, cancel;
    JLabel lblmeter;
    NewCustomer() {
        setSize(700, 500);
        setLocation(400, 200);

        JPanel p = new JPanel();
        p.setLayout(null);
        p.setBackground(new Color(173, 216, 230));
```

```
add(p);

JLabel heading = new JLabel("New Customer");
heading.setBounds(180, 10, 200, 25);
heading.setFont(new Font("Tahoma", Font.PLAIN, 24));
p.add(heading);

JLabel lblname = new JLabel("Customer Name");
lblname.setBounds(100, 80, 100, 20);
p.add(lblname);

tfname = new JTextField();
tfname.setBounds(240, 80, 200, 20);
p.add(tfname);

JLabel lblmeterno = new JLabel("Meter Number");
lblmeterno.setBounds(100, 120, 100, 20);
p.add(lblmeterno);

lblmeter = new JLabel("");
lblmeter.setBounds(240, 120, 100, 20);
p.add(lblmeter);

Random ran = new Random();
long number = ran.nextLong() % 1000000;
lblmeter.setText("" + Math.abs(number));

JLabel lbladdress = new JLabel("Address");
lbladdress.setBounds(100, 160, 100, 20);
p.add(lbladdress);

tfaddress = new JTextField();
tfaddress.setBounds(240, 160, 200, 20);
p.add(tfaddress);

JLabel lblcity = new JLabel("City");
lblcity.setBounds(100, 200, 100, 20);
p.add(lblcity);

tfcity = new JTextField();
tfcity.setBounds(240, 200, 200, 20);
```

```
p.add(tfcity);

JLabel lblstate = new JLabel("State");
lblstate.setBounds(100, 240, 100, 20);
p.add(lblstate);

tfstate = new JTextField();
tfstate.setBounds(240, 240, 200, 20);
p.add(tfstate);

JLabel lblemail = new JLabel("Email");
lblemail.setBounds(100, 280, 100, 20);
p.add(lblemail);

tfemail = new JTextField();
tfemail.setBounds(240, 280, 200, 20);
p.add(tfemail);

JLabel lblphone = new JLabel("Phone Number");
lblphone.setBounds(100, 320, 100, 20);
p.add(lblphone);

tfphone = new JTextField();
tfphone.setBounds(240, 320, 200, 20);
p.add(tfphone);

next = new JButton("Next");
next.setBounds(120, 390, 100,25);
next.setBackground(Color.BLACK);
next.setForeground(Color.WHITE);
next.addActionListener(this);
p.add(next);

cancel = new JButton("Cancel");
cancel.setBounds(250, 390, 100,25);
cancel.setBackground(Color.BLACK);
cancel.setForeground(Color.WHITE);
cancel.addActionListener(this);
p.add(cancel);

setLayout(new BorderLayout());
```

```
            add(p, "Center");

            ImageIcon i1 = new
ImageIcon(ClassLoader.getSystemResource("icon/hicon1.jpg"));
            Image i2 = i1.getImage().getScaledInstance(150, 300,
Image.SCALE_DEFAULT);
            ImageIcon i3 = new ImageIcon(i2);
            JLabel image = new JLabel(i3);
            add(image, "West");

            getContentPane().setBackground(Color.WHITE);

            setVisible(true);
        }

        public void actionPerformed(ActionEvent ae) {
            if (ae.getSource() == next) {
                String name = tfname.getText();
                String meter = lblmeter.getText();
                String address = tfaddress.getText();
                String city = tfcity.getText();
                String state = tfstate.getText();
                String email = tfemail.getText();
                String phone = tfphone.getText();

                String query1 = "insert into customer values('"+name+"',
'"+meter+"', '"+address+"', '"+city+"', '"+state+"', '"+email+"',
'"+phone+"')";
                String query2 = "insert into login values('"+meter+"', '',
'"+name+"', '', '')";

                try {
                    Conn c = new Conn();
                    c.s.executeUpdate(query1);
                    c.s.executeUpdate(query2);

                    JOptionPane.showMessageDialog(null, "Customer Details
Added Successfully");
                    setVisible(false);
                // new frame
                new MeterInfo(meter);
            } catch (Exception e) {
                e.printStackTrace();
            }
        } else {
            setVisible(false);
        }
    }

    public static void main(String[] args) {
        new NewCustomer();
    }
}
```

Customer Details: -

 This program will allow the admin to view customer details.
 If we need to print the particulars that has been viewed click onto print option.

Code: -
```
package electricity.billing.system;

import java.awt.*;
import javax.swing.*;
import java.sql.*;
import net.proteanit.sql.DbUtils;
import java.awt.event.*;

public class CustomerDetails extends JFrame implements
ActionListener{

    Choice meternumber, cmonth;
    JTable table;
    JButton search, print;

    CustomerDetails(){
        super("Customer Details");

        setSize(1200, 650);
        setLocation(200, 150);

        table = new JTable();

        try {
            Conn c = new Conn();
            ResultSet rs = c.s.executeQuery("select * from customer");

            table.setModel(DbUtils.resultSetToTableModel(rs));
        } catch (Exception e) {
            e.printStackTrace();
        }

        JScrollPane sp = new JScrollPane(table);
        add(sp);

        print = new JButton("Print");
        print.addActionListener(this);
        add(print, "South");


        setVisible(true);
    }

    public void actionPerformed(ActionEvent ae) {
        try {
            table.print();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        new CustomerDetails();
    }
}
```

Deposit Details: -

□ This program will allow the admin to view bill details. If we need to sort the particulars based on meter_no and month.

□ If we need to search the particulars that has been viewed click onto search option.

□ If we need to print the particulars that has been viewed click onto print option.

```
   Code: - package electricity.billing.system;

   import java.awt.*;
   import javax.swing.*;
   import java.sql.*;
   import net.proteanit.sql.DbUtils;
   import java.awt.event.*;

   public class DepositDetails extends JFrame implements
ActionListener{

       Choice meternumber, cmonth;
       JTable table;
       JButton search, print;

       DepositDetails(){
           super("Deposit Details");

           setSize(700, 700);
           setLocation(400, 100);

           setLayout(null);
           getContentPane().setBackground(Color.WHITE);

           JLabel lblmeternumber = new JLabel("Search By Meter
Number");
           lblmeternumber.setBounds(20, 20, 150, 20);
           add(lblmeternumber);

           meternumber = new Choice();
           meternumber.setBounds(180, 20, 150, 20);
```

```
add(meternumber);

try {
    Conn c  = new Conn();
    ResultSet rs = c.s.executeQuery("select * from customer");
    while(rs.next()) {
        meternumber.add(rs.getString("meter_no"));
    }
} catch (Exception e) {
    e.printStackTrace();
}

JLabel lblmonth = new JLabel("Search By Month");
lblmonth.setBounds(400, 20, 100, 20);
add(lblmonth);

cmonth = new Choice();
cmonth.setBounds(520, 20, 150, 20);
cmonth.add("January");
cmonth.add("February");
cmonth.add("March");
cmonth.add("April");
cmonth.add("May");
cmonth.add("June");
cmonth.add("July");
cmonth.add("August");
cmonth.add("September");
cmonth.add("October");
cmonth.add("November");
cmonth.add("December");
add(cmonth);

table = new JTable();

try {
    Conn c = new Conn();
    ResultSet rs = c.s.executeQuery("select * from bill");

    table.setModel(DbUtils.resultSetToTableModel(rs));
} catch (Exception e) {
    e.printStackTrace();
```

```
        }

        JScrollPane sp = new JScrollPane(table);
        sp.setBounds(0, 100, 700, 600);
        add(sp);

        search = new JButton("Search");
        search.setBounds(20, 70, 80, 20);
        search.addActionListener(this);
        add(search);

        print = new JButton("Print");
        print.setBounds(120, 70, 80, 20);
        print.addActionListener(this);
        add(print);


        setVisible(true);
    }

    public void actionPerformed(ActionEvent ae) {
        if (ae.getSource() == search) {
            String query = "select * from bill where meter_no =
'"+meternumber.getSelectedItem()+"' and month =
'"+cmonth.getSelectedItem()+"'";

            try {
                Conn c = new Conn();
                ResultSet rs = c.s.executeQuery(query);
                table.setModel(DbUtils.resultSetToTableModel(rs));
            } catch (Exception e) {

            }
        } else  {
            try {
                table.print();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
    public static void main(String[] args) {
        new DepositDetails();
    }
}
```

Tax Details: -

 This program will allow the admin to add tax details. insert the values into tax
 else print error

 Submit the details of tax that has been entered by clicking onto submit button.
 If we need to cancel the particulars that has been entered click onto cancel option.

Calculate Bill: -

 This program will allow the admin to calculate total_bill when units consumed are inserted where meter_no and month is selected.
 Insert the
values into bill
else print error
 Submit the details of tax that has been entered by clicking onto submit button.
 If we need to cancel the particulars that has been entered click onto cancel option.

```
Code: - package electricity.billing.system;

import javax.swing.*;
import java.awt.*;
import java.util.*;
import java.awt.event.*;
import java.sql.*;

public class CalculateBill extends JFrame implements ActionListener{

    JTextField tfname, tfaddress, tfstate, tfunits, tfemail, tfphone;
    JButton next, cancel;
    JLabel lblname, labeladdress;
    Choice meternumber, cmonth;
    CalculateBill() {
        setSize(700, 500);
        setLocation(400, 150);

        JPanel p = new JPanel();
        p.setLayout(null);
        p.setBackground(new Color(173, 216, 230));
        add(p);

        JLabel heading = new JLabel("Calculate Electricity Bill");
        heading.setBounds(100, 10, 400, 25);
        heading.setFont(new Font("Tahoma", Font.PLAIN, 24));
        p.add(heading);

        JLabel lblmeternumber = new JLabel("Meter Number");
        lblmeternumber.setBounds(100, 80, 100, 20);
        p.add(lblmeternumber);

        meternumber = new Choice();

        try {
            Conn c  = new Conn();
            ResultSet rs = c.s.executeQuery("select * from customer");
            while(rs.next()) {
                meternumber.add(rs.getString("meter_no"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        meternumber.setBounds(240, 80, 200, 20);
        p.add(meternumber);

        JLabel lblmeterno = new JLabel("Name");
        lblmeterno.setBounds(100, 120, 100, 20);
        p.add(lblmeterno);

        lblname = new JLabel("");
        lblname.setBounds(240, 120, 100, 20);
        p.add(lblname);

        JLabel lbladdress = new JLabel("Address");
        lbladdress.setBounds(100, 160, 100, 20);
        p.add(lbladdress);

        labeladdress = new JLabel();
        labeladdress.setBounds(240, 160, 200, 20);
        p.add(labeladdress);

        try {
            Conn c = new Conn();
            ResultSet rs = c.s.executeQuery("select * from customer where
    meter_no = '"+meternumber.getSelectedItem()+"'");
            while(rs.next()) {
                lblname.setText(rs.getString("name"));
                labeladdress.setText(rs.getString("address"));
```

```
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }

                meternumber.addItemListener(new ItemListener() {
                    public void itemStateChanged(ItemEvent ie) {
                        try {
                            Conn c = new Conn();
                            ResultSet rs = c.s.executeQuery("select * from customer where
        meter_no = '"+meternumber.getSelectedItem()+"'");
                            while(rs.next()) {
                                lblname.setText(rs.getString("name"));
                                labeladdress.setText(rs.getString("address"));
                            }
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    }
                });

                JLabel lblcity = new JLabel("Units Consumed");
                lblcity.setBounds(100, 200, 100, 20);
                p.add(lblcity);

                tfunits = new JTextField();
                tfunits.setBounds(240, 200, 200, 20);
                p.add(tfunits);

                JLabel lblstate = new JLabel("Month");
                lblstate.setBounds(100, 240, 100, 20);
                p.add(lblstate);

                cmonth = new Choice();
                cmonth.setBounds(240, 240, 200, 20);
                cmonth.add("January");
                cmonth.add("February");
                cmonth.add("March");
                cmonth.add("April");
                cmonth.add("May");
                cmonth.add("June");
                cmonth.add("July");
                cmonth.add("August");
                cmonth.add("September");
                cmonth.add("October");
                cmonth.add("November");
                cmonth.add("December");
                p.add(cmonth);

                next = new JButton("Submit");
                next.setBounds(120, 350, 100,25);
                next.setBackground(Color.BLACK);
                next.setForeground(Color.WHITE);
                next.addActionListener(this);
                p.add(next);

                cancel = new JButton("Cancel");
                cancel.setBounds(250, 350, 100,25);
                cancel.setBackground(Color.BLACK);
                cancel.setForeground(Color.WHITE);
                cancel.addActionListener(this);
                p.add(cancel);

                setLayout(new BorderLayout());

                add(p, "Center");

                ImageIcon i1 = new
        ImageIcon(ClassLoader.getSystemResource("icon/hicon2.jpg"));
                Image i2 = i1.getImage().getScaledInstance(150, 300,
        Image.SCALE_DEFAULT);
                ImageIcon i3 = new ImageIcon(i2);
                JLabel image = new JLabel(i3);
                add(image, "West");

                getContentPane().setBackground(Color.WHITE);

                setVisible(true);
            }

            public void actionPerformed(ActionEvent ae) {
                if (ae.getSource() == next) {
                        String meter = meternumber.getSelectedItem();
                        String units = tfunits.getText();
                        String month = cmonth.getSelectedItem();

                        int totalbill = 0;
                        int unit_consumed = Integer.parseInt(units);

                        String query = "select * from tax";

                        try {
                            Conn c = new Conn();
                            ResultSet rs = c.s.executeQuery(query);

                            while(rs.next()) {
                                totalbill += unit_consumed *
        Integer.parseInt(rs.getString("cost_per_unit"));
                                totalbill += Integer.parseInt(rs.getString("meter_rent"));
                                totalbill += Integer.parseInt(rs.getString("service_charge"));
                                totalbill += Integer.parseInt(rs.getString("service_tax"));
                                totalbill += Integer.parseInt(rs.getString("swacch_bharat_cess"));
                                totalbill += Integer.parseInt(rs.getString("fixed_tax"));
                            }
                        } catch (Exception e) {
                            e.printStackTrace();
                        }

                        String query2 = "insert into bill values('"+meter+"', '"+month+"',
        '"+units+"', '"+totalbill+"', 'Not Paid')";

                        try {
                            Conn c  =  new Conn();
                            c.s.executeUpdate(query2);

                            JOptionPane.showMessageDialog(null, "Customer Bill Updated
        Successfully");
                            setVisible(false);
                        } catch (Exception e) {
                            e.printStackTrace();
                        }
                    } else {
                        setVisible(false);
```

```
        }
    }

    public static void main(String[] args) {
        new CalculateBill();
    }
}
```

Delete Customer: -

 This Program will allow the admin to delete the customer info when meter_no is selected.
 If we need to delete the particulars that has been saved click onto delete option.
 If we need to cancel the particulars that has been entered click onto back option.

Algorithm or pseudocode of Customer:-

Login:-

 This program will allow the customer to enter the username and password. If the entered credentials are correct, then the login will be successful otherwise need to be signup with the meter_no which is given by admin.
 If customer forgets password, it can be retrieved by giving username and answer for security question. After successful login the customer will be redirected to customer portal page where he/she can do following activities.

```
Code:- package electricity.billing.system;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;

public class Login extends JFrame implements ActionListener{

    JButton login, cancel, signup;
    JTextField username, password;
    Choice logginin;
    Login() {
        super("Login Page");
        getContentPane().setBackground(Color.WHITE);
        setLayout(null);

        JLabel lblusername = new JLabel("Username");
        lblusername.setBounds(300, 20, 100, 20);
        add(lblusername);
        username = new JTextField();
        username.setBounds(400, 20, 150, 20);
        add(username);

        JLabel lblpassword = new JLabel("Password");
        lblpassword.setBounds(300, 60, 100, 20);
        add(lblpassword);

        password = new JTextField();
        password.setBounds(400, 60, 150, 20);
        add(password);

        JLabel loggininas = new JLabel("Loggin in as");
        loggininas.setBounds(300, 100, 100, 20);
        add(loggininas);

        logginin = new Choice();
        logginin.add("Admin");
        logginin.add("Customer");
        logginin.setBounds(400, 100, 150, 20);
        add(logginin);

        ImageIcon i1 = new
ImageIcon(ClassLoader.getSystemResource("icon/login.png"));
        Image i2 = i1.getImage().getScaledInstance(16, 16,
Image.SCALE_DEFAULT);
        login = new JButton("Login", new ImageIcon(i2));
        login.setBounds(330, 160, 100, 20);
        login.addActionListener(this);
        add(login);

        ImageIcon i3 = new
ImageIcon(ClassLoader.getSystemResource("icon/cancel.jpg"));
        Image i4 = i3.getImage().getScaledInstance(16, 16,
Image.SCALE_DEFAULT);
        cancel = new JButton("Cancel", new ImageIcon(i4));
        cancel.setBounds(450, 160, 100, 20);
        cancel.addActionListener(this);
        add(cancel);
```

```
            ImageIcon i5 = new
ImageIcon(ClassLoader.getSystemResource("icon/signup.png"));
            Image i6 = i5.getImage().getScaledInstance(16, 16,
Image.SCALE_DEFAULT);
            signup = new JButton("Signup", new ImageIcon(i6));
            signup.setBounds(380, 200, 100, 20);
            signup.addActionListener(this);
            add(signup);

            ImageIcon i7 = new
ImageIcon(ClassLoader.getSystemResource("icon/second.jpg"));
            Image i8 = i7.getImage().getScaledInstance(250, 250,
Image.SCALE_DEFAULT);
            ImageIcon i9 = new ImageIcon(i8);
            JLabel image = new JLabel(i9);
            image.setBounds(0, 0, 250, 250);
            add(image);

            setSize(640, 300);
            setLocation(400, 200);
            setVisible(true);
        }

        public void actionPerformed(ActionEvent ae) {
            if (ae.getSource() == login) {
                String susername = username.getText();
                String spassword = password.getText();
                String user = logginin.getSelectedItem();

                try {
                    Conn c = new Conn();
                    String query = "select * from login where username =
'"+susername+"' and password = '"+spassword+"' and user = '"+user+"'";

                    ResultSet rs = c.s.executeQuery(query);

                    if (rs.next()) {
                        String meter = rs.getString("meter_no");
                        setVisible(false);
                        new Project(user, meter);
                    } else {
                        JOptionPane.showMessageDialog(null, "Invalid Login");
                        username.setText("");
                        password.setText("");
                    }

                } catch (Exception e) {
                    e.printStackTrace();
                }
            } else if (ae.getSource() == cancel) {
                setVisible(false);
            } else if (ae.getSource() == signup) {
                setVisible(false);

                new Signup();
            }
        }

        public static void main(String[] args) {
            new Login();
        }
        }
```

UpdateInfo: -

 This program will allow the customer to update the customer details. If customer address, city, state, email and phone number is updated.
 update the values into
customer else print error
 update the details of customer that has been updated by clicking onto
update button.
 If we need to cancel the particulars that has been updated, click onto back option.
View Info:

 This program will allow the customer to view his/her own details.
 If we need to go back from the particulars that has been viewed
click onto back option.

Code: - package electricity.billing.system;

```
import javax.swing.*;
import java.awt.*;
import java.sql.*;
import java.awt.event.*;

public class UpdateInformation extends JFrame implements ActionListener{

    JTextField tfaddress, tfstate, tfcity, tfemail, tfphone;
    JButton update, cancel;
    String meter;
    JLabel name;
    UpdateInformation(String meter) {
        this.meter = meter;
        setBounds(300, 150, 1050, 450);
        getContentPane().setBackground(Color.WHITE);
        setLayout(null);

        JLabel heading = new JLabel("UPDATE CUSTOMER INFORMATION");
        heading.setBounds(110, 0, 400, 30);
        heading.setFont(new Font("Tahoma", Font.PLAIN, 20));
        add(heading);

        JLabel lblname = new JLabel("Name");
        lblname.setBounds(30, 70, 100, 20);
        add(lblname);

        name = new JLabel("");
        name.setBounds(230, 70, 200, 20);
        add(name);
```

```
JLabel lblmeternumber = new JLabel("Meter Number");
lblmeternumber.setBounds(30, 110, 100, 20);
add(lblmeternumber);

JLabel meternumber = new JLabel("");
meternumber.setBounds(230, 110, 200, 20);
add(meternumber);

JLabel lbladdress = new JLabel("Address");
lbladdress.setBounds(30, 150, 100, 20);
add(lbladdress);

tfaddress = new JTextField();
tfaddress.setBounds(230, 150, 200, 20);
add(tfaddress);

JLabel lblcity = new JLabel("City");
lblcity.setBounds(30, 190, 100, 20);
add(lblcity);

tfcity = new JTextField();
tfcity.setBounds(230, 190, 200, 20);
add(tfcity);

JLabel lblstate = new JLabel("State");
lblstate.setBounds(30, 230, 100, 20);
add(lblstate);

tfstate = new JTextField();
tfstate.setBounds(230, 230, 200, 20);
add(tfstate);

JLabel lblemail = new JLabel("Email");
lblemail.setBounds(30, 270, 100, 20);
add(lblemail);

tfemail = new JTextField();
tfemail.setBounds(230, 270, 200, 20);
add(tfemail);

JLabel lblphone = new JLabel("Phone");
lblphone.setBounds(30, 310, 100, 20);
add(lblphone);

tfphone = new JTextField();
tfphone.setBounds(230, 310, 200, 20);
add(tfphone);

try {
   Conn c = new Conn();
   ResultSet rs = c.s.executeQuery("select * from customer where meter_no = '"+meter+"'");
   while(rs.next()) {
```

```
                name.setText(rs.getString("name"));
                tfaddress.setText(rs.getString("address"));
                tfcity.setText(rs.getString("city"));
                tfstate.setText(rs.getString("state"));
                tfemail.setText(rs.getString("email"));
                tfphone.setText(rs.getString("phone"));
                meternumber.setText(rs.getString("meter_no"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        update = new JButton("Update");
        update.setBackground(Color.BLACK);
        update.setForeground(Color.WHITE);
        update.setBounds(70, 360, 100, 25);
        add(update);
        update.addActionListener(this);

        cancel = new JButton("Cancel");
        cancel.setBackground(Color.BLACK);
        cancel.setForeground(Color.WHITE);
        cancel.setBounds(230, 360, 100, 25);
        add(cancel);
        cancel.addActionListener(this);

        ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icon/update.jpg"));
        Image i2 = i1.getImage().getScaledInstance(400, 300, Image.SCALE_DEFAULT);
        ImageIcon i3 = new ImageIcon(i2);
        JLabel image = new JLabel(i3);
        image.setBounds(550, 50, 400, 300);
        add(image);


        setVisible(true);
    }

    public void actionPerformed(ActionEvent ae) {
        if (ae.getSource() == update) {
            String address = tfaddress.getText();
            String city = tfcity.getText();
            String state = tfstate.getText();
            String email = tfemail.getText();
            String phone = tfphone.getText();

            try {
                Conn c = new Conn();
                c.s.executeUpdate("update customer set address = '"+address+"', city = '"+city+"', state
= '"+state+"', email = '"+email+"', phone = '"+phone+"' where meter_no = '"+meter+"'");

                JOptionPane.showMessageDialog(null, "User Information Updated Successfully");
                setVisible(false);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            } else {
                setVisible(false);
            }
        }

        public static void main(String[] args) {
            new UpdateInformation("");
        }
    }
```

Pay Bill: -

 This program will allow the customer to view bill details and redirects to pay.
 the bill where status will be updated.
 If we need to cancel the particulars that has been viewed click onto back option.
 If we need to pay the bill amount that has been viewed click onto pay option.

Code: - package electricity.billing.system;

```
import javax.swing.*;
import java.awt.*;
import java.sql.*;
import java.awt.event.*;

public class PayBill extends JFrame implements ActionListener{

    Choice cmonth;
    JButton pay, back;
    String meter;
    PayBill(String meter) {
        this.meter = meter;
        setLayout(null);
        setBounds(300, 150, 900, 600);

        JLabel heading = new JLabel("Electicity Bill");
        heading.setFont(new Font("Tahoma", Font.BOLD, 24));
        heading.setBounds(120, 5, 400, 30);
        add(heading);

        JLabel lblmeternumber = new JLabel("Meter Number");
        lblmeternumber.setBounds(35, 80, 200, 20);
        add(lblmeternumber);

        JLabel meternumber = new JLabel("");
        meternumber.setBounds(300, 80, 200, 20);
        add(meternumber);

        JLabel lblname = new JLabel("Name");
        lblname.setBounds(35, 140, 200, 20);
```

```
add(lblname);

JLabel labelname = new JLabel("");
labelname.setBounds(300, 140, 200, 20);
add(labelname);

JLabel lblmonth = new JLabel("Month");
lblmonth.setBounds(35, 200, 200, 20);
add(lblmonth);

cmonth = new Choice();
cmonth.setBounds(300, 200, 200, 20);
cmonth.add("January");
cmonth.add("February");
cmonth.add("March");
cmonth.add("April");
cmonth.add("May");
cmonth.add("June");
cmonth.add("July");
cmonth.add("August");
cmonth.add("September");
cmonth.add("October");
cmonth.add("November");
cmonth.add("December");
add(cmonth);

JLabel lblunits = new JLabel("Units");
lblunits.setBounds(35, 260, 200, 20);
add(lblunits);

JLabel labelunits = new JLabel("");
labelunits.setBounds(300, 260, 200, 20);
add(labelunits);

JLabel lbltotalbill = new JLabel("Total Bill");
lbltotalbill.setBounds(35, 320, 200, 20);
add(lbltotalbill);

JLabel labeltotalbill = new JLabel("");
labeltotalbill.setBounds(300, 320, 200, 20);
add(labeltotalbill);
```

```
        JLabel lblstatus = new JLabel("Status");
        lblstatus.setBounds(35, 380, 200, 20);
        add(lblstatus);

        JLabel labelstatus = new JLabel("");
        labelstatus.setBounds(300, 380, 200, 20);
        labelstatus.setForeground(Color.RED);
        add(labelstatus);

        try {
            Conn c = new Conn();
            ResultSet rs = c.s.executeQuery("select * from customer where
meter_no = '"+meter+"'");
            while(rs.next()) {
                meternumber.setText(meter);
                labelname.setText(rs.getString("name"));
            }

            rs = c.s.executeQuery("select * from bill where meter_no =
'"+meter+"' AND month = 'January'");
            while(rs.next()) {
                labelunits.setText(rs.getString("units"));
                labeltotalbill.setText(rs.getString("totalbill"));
                labelstatus.setText(rs.getString("status"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        cmonth.addItemListener(new ItemListener(){
            @Override
            public void itemStateChanged(ItemEvent ae) {
                try {
                    Conn c = new Conn();
                    ResultSet rs = c.s.executeQuery("select * from bill where
meter_no = '"+meter+"' AND month = '"+cmonth.getSelectedItem()+"'");
                    while(rs.next()) {
                        labelunits.setText(rs.getString("units"));
                        labeltotalbill.setText(rs.getString("totalbill"));
                        labelstatus.setText(rs.getString("status"));
                    }
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });



        pay = new JButton("Pay");
        pay.setBackground(Color.BLACK);
        pay.setForeground(Color.WHITE);
        pay.setBounds(100, 460, 100, 25);
        pay.addActionListener(this);
        add(pay);

        back = new JButton("Back");
        back.setBackground(Color.BLACK);
        back.setForeground(Color.WHITE);
        back.setBounds(230, 460, 100, 25);
        back.addActionListener(this);
        add(back);

        getContentPane().setBackground(Color.WHITE);

        ImageIcon i1 = new
ImageIcon(ClassLoader.getSystemResource("icon/bill.png"));
        Image i2 = i1.getImage().getScaledInstance(600, 300,
Image.SCALE_DEFAULT);
        ImageIcon i3 = new ImageIcon(i2);
        JLabel image = new JLabel(i3);
        image.setBounds(400, 120, 600, 300);
        add(image);

        setVisible(true);
}

public void actionPerformed(ActionEvent ae) {
        if (ae.getSource() == pay) {
            try {
                Conn c = new Conn();
                c.s.executeUpdate("update bill set status = 'Paid' where
meter_no = '"+meter+"' AND month='"+cmonth.getSelectedItem()+"'");
            } catch (Exception e) {
               e.printStackTrace();
            }
            setVisible(false);
            new Paytm(meter);
         } else {
            setVisible(false);
         }
      }

      public static void main(String[] args){
         new PayBill("");
      }
   }
```

Bill Details:

 This program will allow the customer to view bill details.

 If we need to print the particulars that has been viewed click onto print option.

```
Code: - package electricity.billing.system;

import javax.swing.*;
import java.awt.*;
import java.sql.*;
import net.proteanit.sql.DbUtils;

public class BillDetails extends JFrame{

    BillDetails(String meter) {

        setSize(700, 650);
        setLocation(400, 150);

        getContentPane().setBackground(Color.WHITE);

        JTable table = new JTable();

        try {
            Conn c = new Conn();
            String query = "select * from bill where meter_no = '"+meter+"'";
            ResultSet rs = c.s.executeQuery(query);

            table.setModel(DbUtils.resultSetToTableModel(rs));
        } catch (Exception e) {
            e.printStackTrace();
        }

        JScrollPane sp = new JScrollPane(table);
        sp.setBounds(0, 0, 700, 650);
        add(sp);

        setVisible(true);
    }

    public static void main(String[] args) {
        new BillDetails("");
    }
}
```

Generate Bill: -

- This program will allow the customer to generate bill when meter_no and month is selected.
- Generate the details by clicking on generate bill button.

Code: -
```
package electricity.billing.system;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;

public class GenerateBill extends JFrame implements ActionListener{

    String meter;
    JButton bill;
    Choice cmonth;
    JTextArea area;
    GenerateBill(String meter) {
        this.meter = meter;

        setSize(500, 800);
        setLocation(550, 30);

        setLayout(new BorderLayout());

        JPanel panel = new JPanel();

        JLabel heading = new JLabel("Generate Bill");
        JLabel meternumber = new JLabel(meter);

        cmonth = new Choice();

        cmonth.add("January");
        cmonth.add("February");
        cmonth.add("March");
        cmonth.add("April");
        cmonth.add("May");
        cmonth.add("June");
        cmonth.add("July");
        cmonth.add("August");
        cmonth.add("September");
        cmonth.add("October");
        cmonth.add("November");
        cmonth.add("December");

        area = new JTextArea(50, 15);
        area.setText("\n\n\t--------Click on the---------\n\t Generate Bill Button to get\n\tthe bill of
the Selected Month");
        area.setFont(new Font("Senserif", Font.ITALIC, 18));

        JScrollPane pane = new JScrollPane(area);
        bill = new JButton("Generate Bill");
        bill.addActionListener(this);

        panel.add(heading);
        panel.add(meternumber);
        panel.add(cmonth);
        add(panel, "North");

        add(pane, "Center");
        add(bill, "South");

        setVisible(true);
    }

    public void actionPerformed(ActionEvent ae) {
        try {
            Conn c = new Conn();

            String month = cmonth.getSelectedItem();

            area.setText("\tReliance Power Limited\nELECTRICITY BILL GENERATED FOR
THE MONTH\n\tOF "+month+", 2022\n\n\n");

            ResultSet rs = c.s.executeQuery("select * from customer where meter_no =
'"+meter+"'");

            if(rs.next()) {
                area.append("\n    Customer Name: " + rs.getString("name"));
                area.append("\n    Meter Number   : " + rs.getString("meter_no"));
                area.append("\n    Address            : " + rs.getString("address"));
                area.append("\n    City                : " + rs.getString("city"));
                area.append("\n    State               : " + rs.getString("state"));
                area.append("\n    Email               : " + rs.getString("email"));
                area.append("\n    Phone               : " + rs.getString("phone"));
                area.append("\n---------------------------------------------------");
                area.append("\n");
            }

            rs = c.s.executeQuery("select * from meter_info where meter_no = '"+meter+"'");

            if(rs.next()) {
                area.append("\n    Meter Location: " + rs.getString("meter_location"));
                area.append("\n    Meter Type:     " + rs.getString("meter_type"));
                area.append("\n    Phase Code:        " + rs.getString("phase_code"));
                area.append("\n    Bill Type:          " + rs.getString("bill_type"));
                area.append("\n    Days:               " + rs.getString("days"));
                area.append("\n---------------------------------------------------");
                area.append("\n");
            }

            rs = c.s.executeQuery("select * from tax");
            if(rs.next()) {
                area.append("\n");
                area.append("\n    Cost Per Unit: " + rs.getString("cost_per_unit"));
                area.append("\n    Meter Rent:     " + rs.getString("cost_per_unit"));
                area.append("\n    Service Charge:        " + rs.getString("service_charge"));
                area.append("\n    Service Tax:          " + rs.getString("service_charge"));
                area.append("\n    Swacch Bharat Cess:                " +
rs.getString("swacch_bharat_cess"));
                area.append("\n    Fixed Tax: " + rs.getString("fixed_tax"));
                area.append("\n");
            }

            rs = c.s.executeQuery("select * from bill where meter_no = '"+meter+"' and
month='"+month+"'");

            if(rs.next()) {
                area.append("\n");
                area.append("\n    Current Month: " + rs.getString("month"));
                area.append("\n    Units Consumed:     " + rs.getString("units"));
                area.append("\n    Total Charges:        " + rs.getString("totalbill"));
                area.append("\n-------------------------------------------------------");
                area.append("\n    Total Payable: " + rs.getString("totalbill"));
                area.append("\n");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) {
        new GenerateBill("");
    }
}
```

NOTE: Utility (notepad, browser, calculator), query and logout is given to both customer and admin portals.

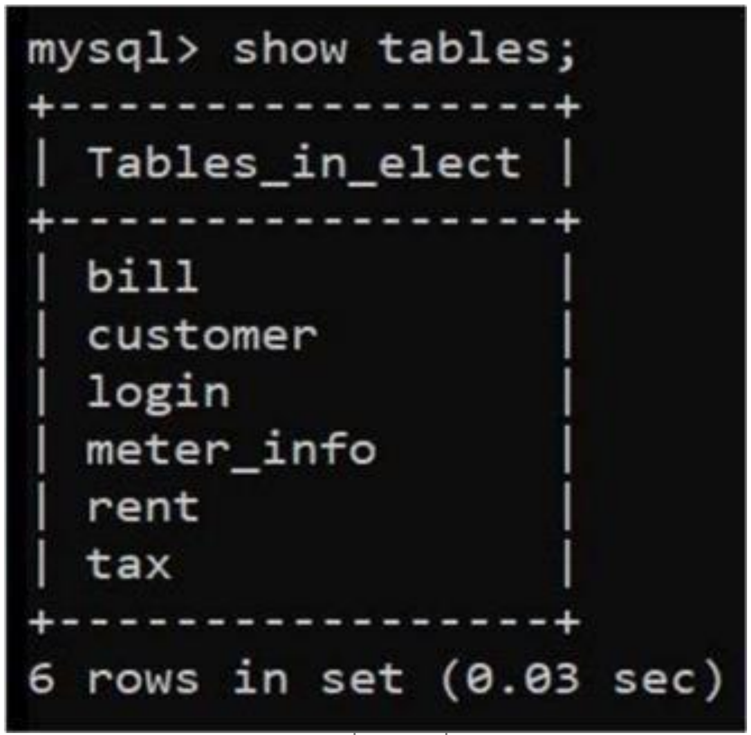
CHAPTER-5

DISCUSSION AND SNAPSHOTS

TABLES:-

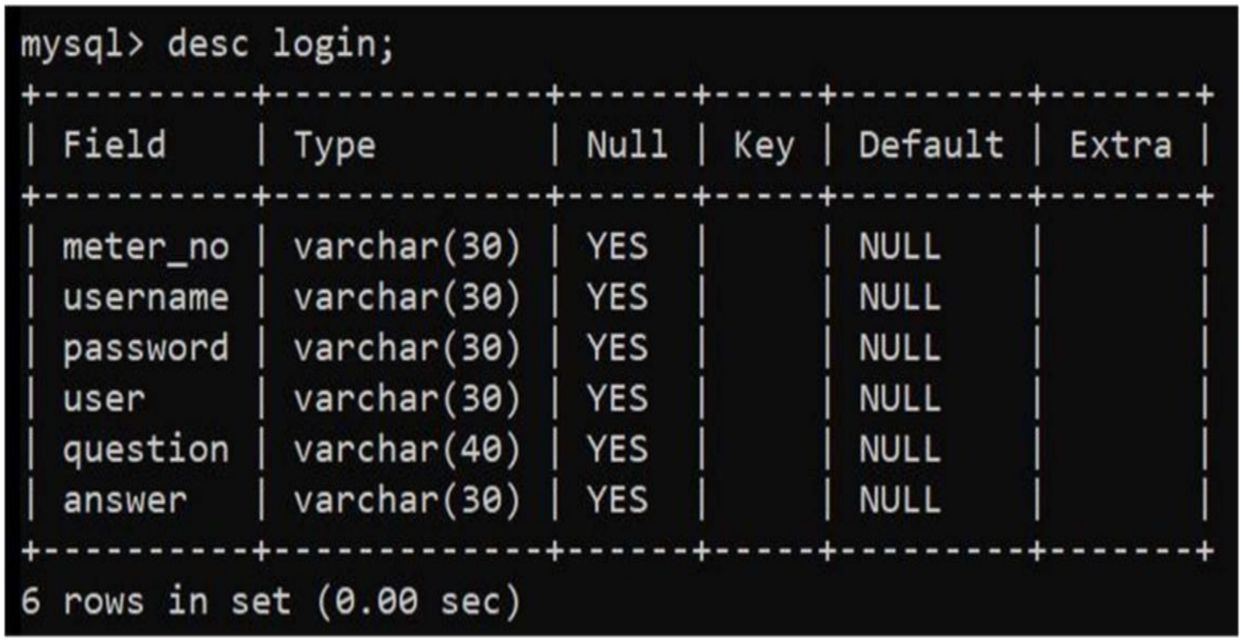
The given below table is a snapshot of backend view of the localhost and the structures of the tables present in Electricity Billing System. The tables present are login, customer, tax, bill, meter_info.

- ✓ The login is used to store the details of login's admin and customer with meter_no.
- ✓ The customer is used to store details of customer.
- ✓ The tax is used to store tax values.
- ✓ The rent is used to store rent values.
- ✓ The bill is used to store details of bill of meter.
- ✓ The meter_info is used to store information of meter placed.



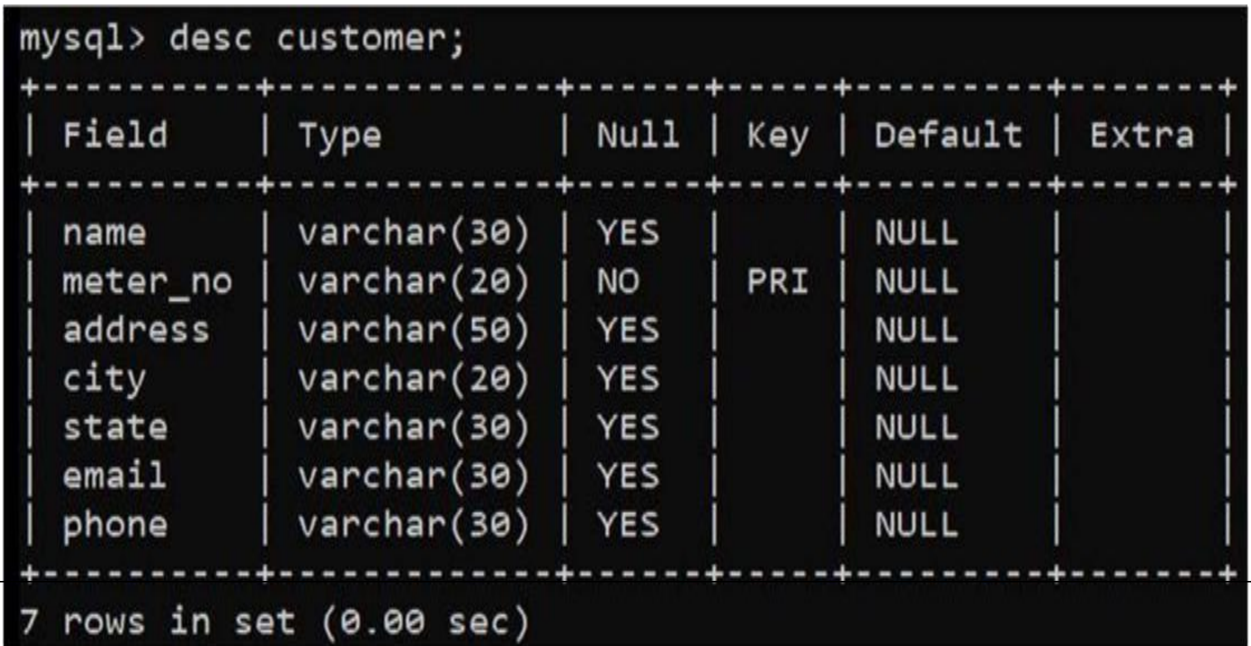
```
mysql> show tables;
+-----------------+
| Tables_in_elect |
+-----------------+
| bill            |
| customer        |
| login           |
| meter_info      |
| rent            |
| tax             |
+-----------------+
6 rows in set (0.03 sec)
```

Login Table:-



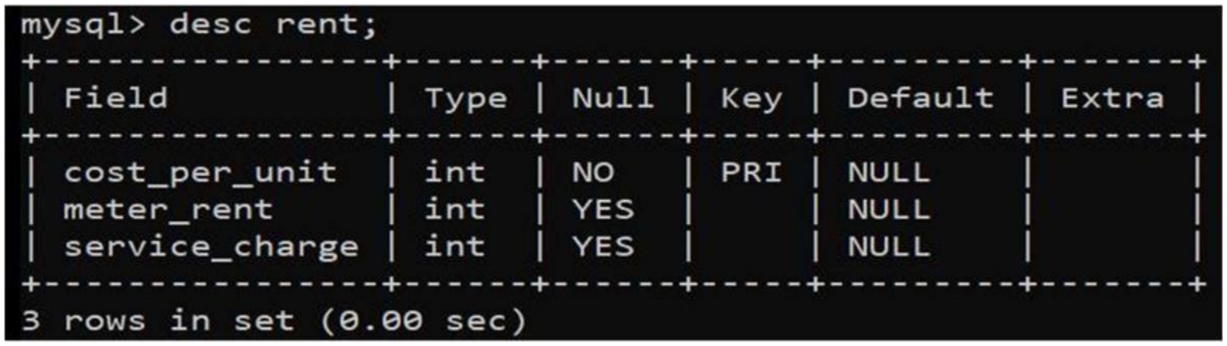
```
mysql> desc login;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| meter_no | varchar(30) | YES  |     | NULL    |       |
| username | varchar(30) | YES  |     | NULL    |       |
| password | varchar(30) | YES  |     | NULL    |       |
| user     | varchar(30) | YES  |     | NULL    |       |
| question | varchar(40) | YES  |     | NULL    |       |
| answer   | varchar(30) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Customer Table:-



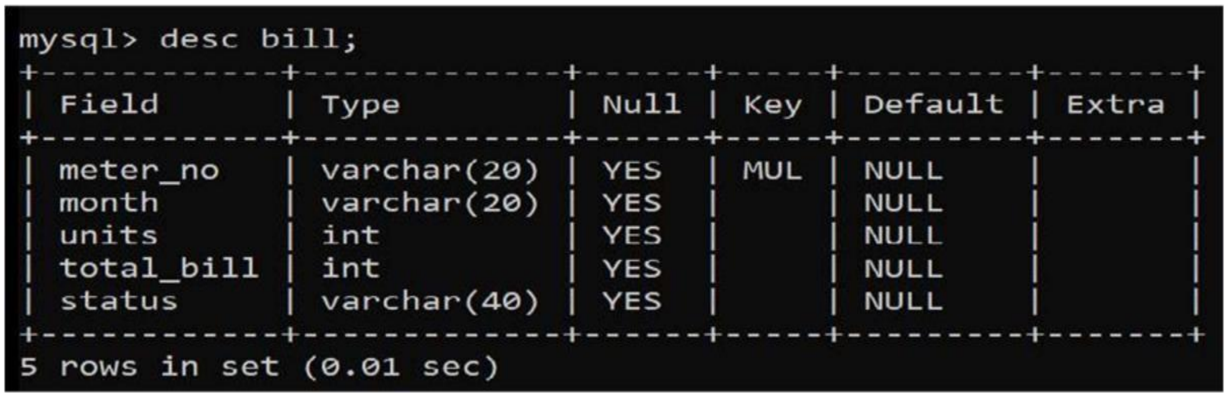
```
mysql> desc customer;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| name     | varchar(30) | YES  |     | NULL    |       |
| meter_no | varchar(20) | NO   | PRI | NULL    |       |
| address  | varchar(50) | YES  |     | NULL    |       |
| city     | varchar(20) | YES  |     | NULL    |       |
| state    | varchar(30) | YES  |     | NULL    |       |
| email    | varchar(30) | YES  |     | NULL    |       |
| phone    | varchar(30) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
7 rows in set (0.00 sec)
```

Rent Table:-



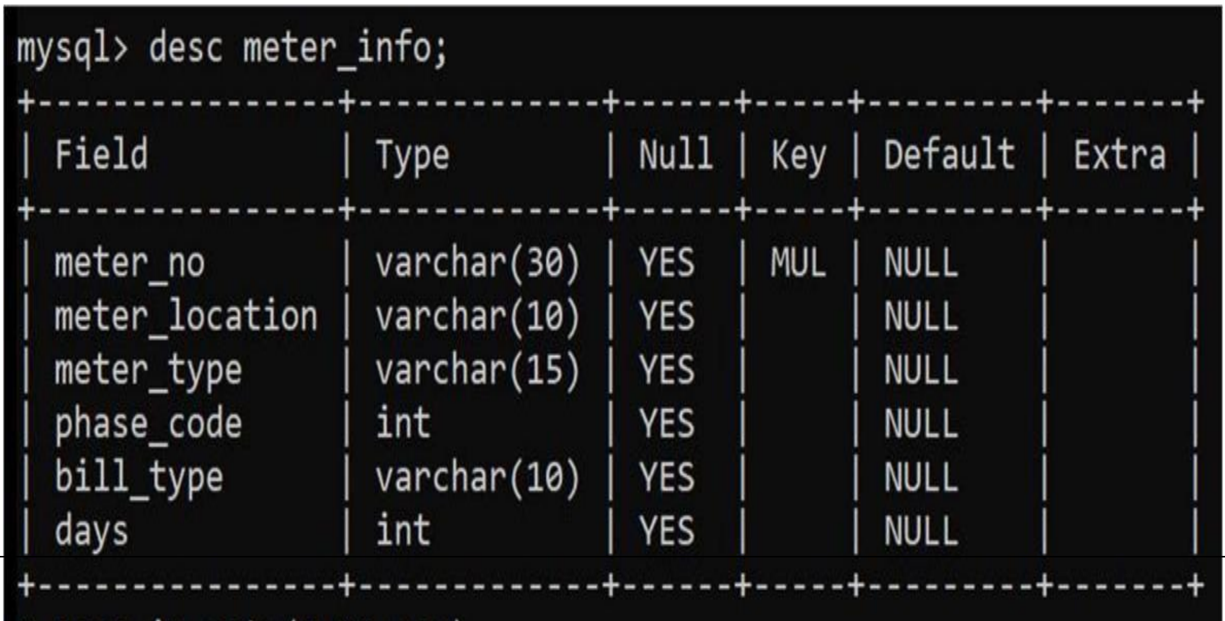
```
mysql> desc rent;
+----------------+------+------+-----+---------+-------+
| Field          | Type | Null | Key | Default | Extra |
+----------------+------+------+-----+---------+-------+
| cost_per_unit  | int  | NO   | PRI | NULL    |       |
| meter_rent     | int  | YES  |     | NULL    |       |
| service_charge | int  | YES  |     | NULL    |       |
+----------------+------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

Bill Table:-



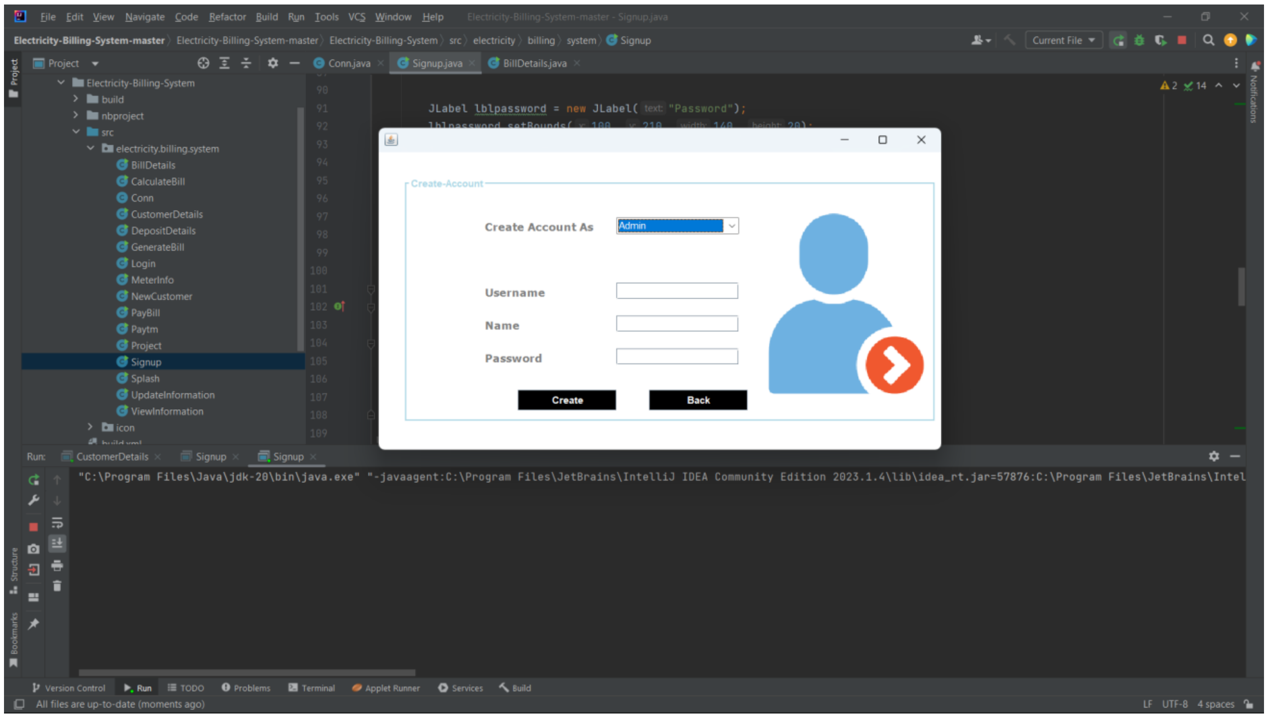
```
mysql> desc bill;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| meter_no   | varchar(20) | YES  | MUL | NULL    |       |
| month      | varchar(20) | YES  |     | NULL    |       |
| units      | int         | YES  |     | NULL    |       |
| total_bill | int         | YES  |     | NULL    |       |
| status     | varchar(40) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

Meter_Info Table:-



```
mysql> desc meter_info;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| meter_no       | varchar(30) | YES  | MUL | NULL    |       |
| meter_location | varchar(10) | YES  |     | NULL    |       |
| meter_type     | varchar(15) | YES  |     | NULL    |       |
| phase_code     | int         | YES  |     | NULL    |       |
| bill_type      | varchar(10) | YES  |     | NULL    |       |
| days           | int         | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
```

SNAPSHOTS:-



```
JLabel lblpassword = new JLabel( text: "Password");
lblpassword.setBounds( x: 100, y: 210, width: 140, height: 20);
```

Create Account As: Admin

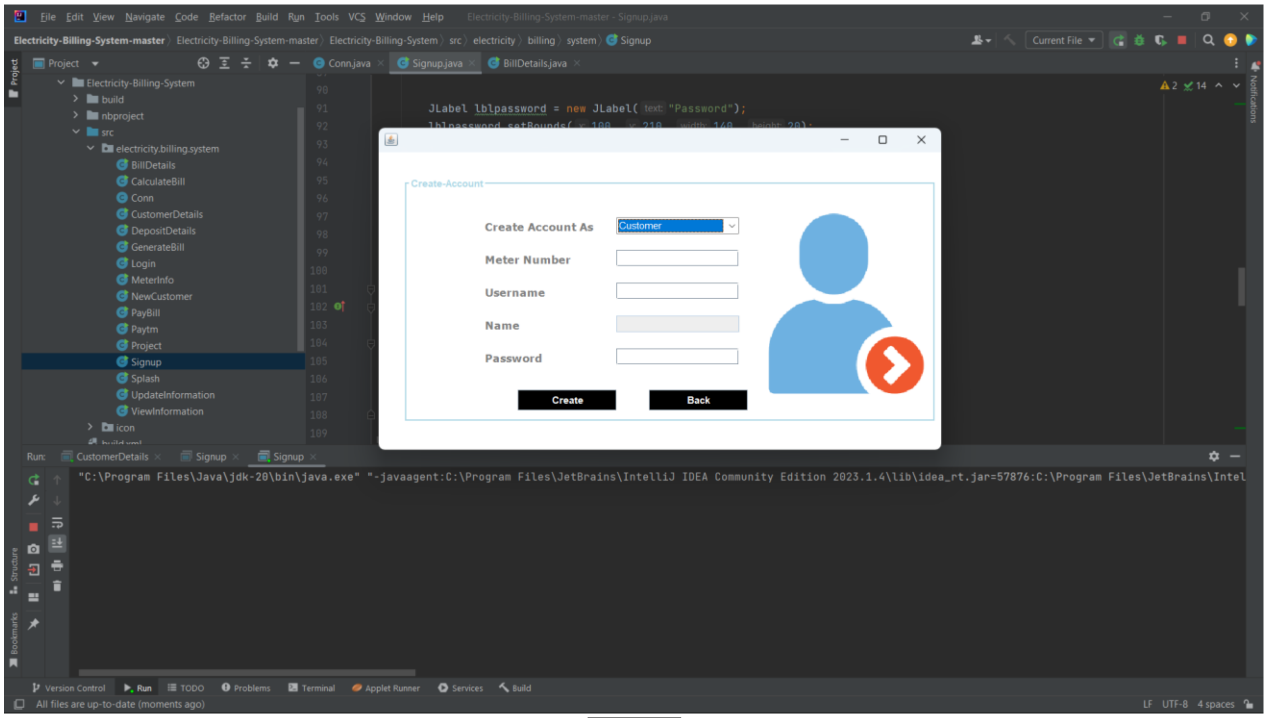
Username:

Name:

Password:

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.1.4\lib\idea_rt.jar=57876:C:\Program Files\JetBrains\Intel
```

All files are up-to-date (moments ago)


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
Electricity-Billing-System-master - Signup.java
Conn.java
Signup.java
BillDetails.java
Create-Account
Create
Back

```
JLabel lblpassword = new JLabel( text: "Password");
lblpassword.setBounds( x: 100, y: 210, width: 140, height: 20);
```

Create Account As: Customer

Meter Number:

Username:

Name:

Password:

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.1.4\lib\idea_rt.jar=57876:C:\Program Files\JetBrains\Intel
```

All files are up-to-date (moments ago)

Sign Up Screen:-

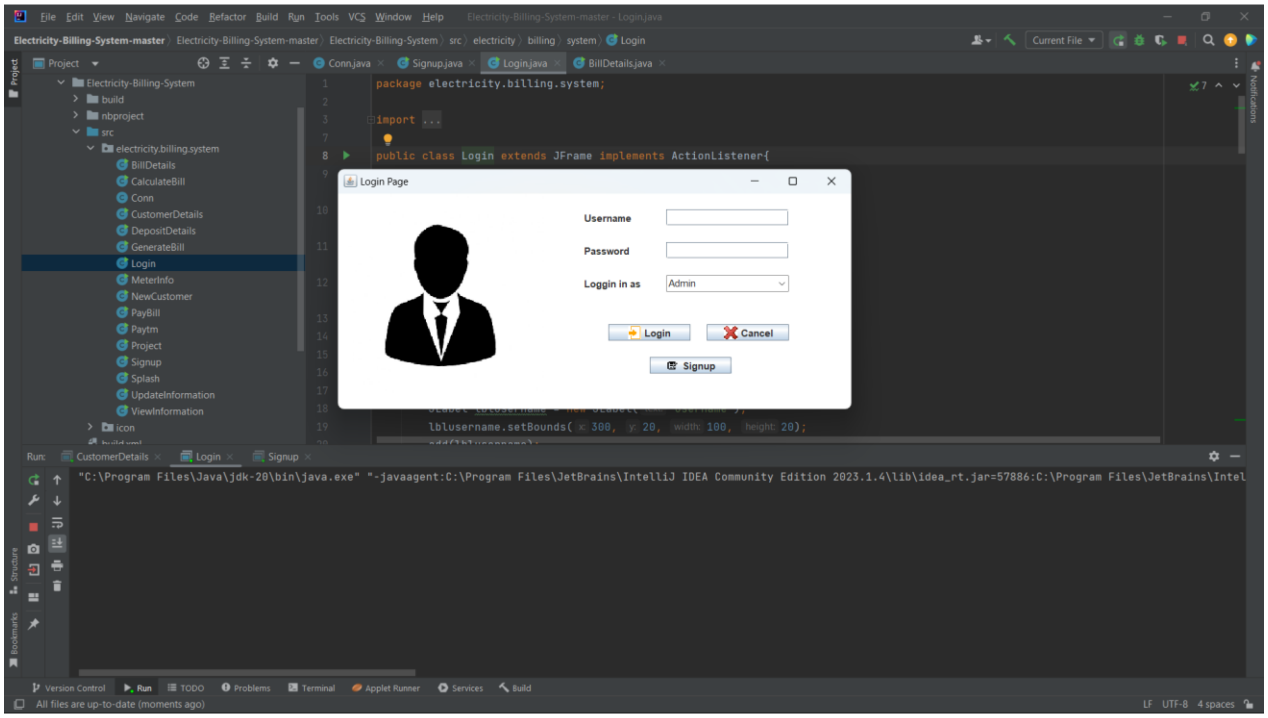
Here New customers will signup to access their accounts.

User have to enter username, name, password, choose security question and answer to that question.

Every user must enter their unique Meter Number to complete their signup process.

Login Screen:-

Here Customer and Admin can login to their respective accounts. The dropdown menu allows to choose whether to login as an admin or as a customer.



```
package electricity.billing.system;

import ...

public class Login extends JFrame implements ActionListener{
```

Username

Password

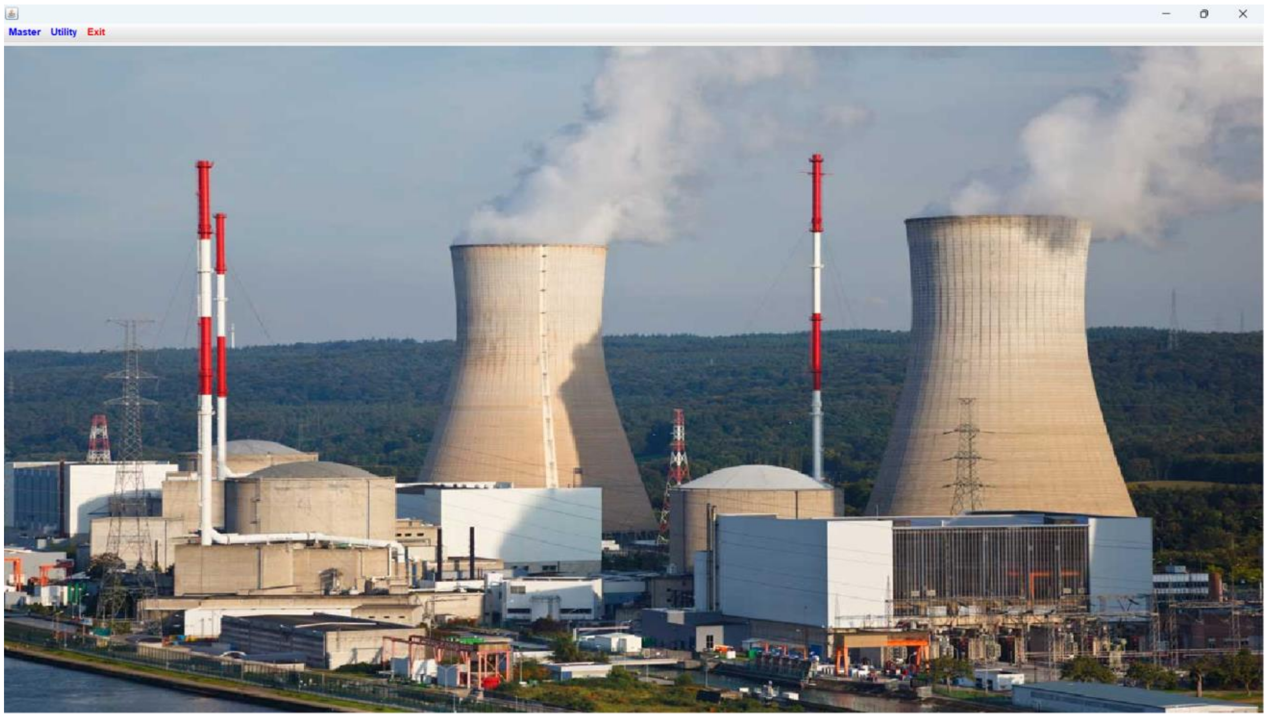
Loggin in as: Admin

```
        lblusername.setBounds( x: 300, y: 20, width: 100, height: 20);
```

```
"C:\Program Files\Java\jdk-20\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2023.1.4\lib\idea_rt.jar=57886:C:\Program Files\JetBrains\Intel
```

Admin lands on this page after successful login:-

Admin's Home Screen:



CHAPTER 6

FUTURE SCOPE AND LIMITATIONS

SOFTWARE developers may not expect. The following principles enhances extensibility like hide data structure, avoid traversing multiple.
Links or methods avoid case statements on object type and distinguish public and private operations.

• Reusability: Reusability is possible as and when require in this application. We can update it next version. Reusable software reduces design, coding and testing cost by amortizing effort
Over several designs. Reducing the amount of code also simplifies understanding, which increases the likelihood that the code is correct.
We follow up both types of reusability:
Sharing of newly written code within a project and reuse of previously written code on new projects.

• Understand ability: A method is understandable if someone other than the creator of the method can understand the code (as well as the creator after a time lapse). We use the method, which small and coherent helps to accomplish this.

• Cost-effectiveness: Its cost is under the budget and make within given time period. It is desirable to aim for a system with a minimum cost subject to the condition that it must satisfy the entire requirement. Scope of this document is to put down the requirements, clearly identifying the information needed by the user, the source of the information and outputs expected from the system.

LIMITATIONS:-

This application cannot be accessed remotely.
 This application requires knowledgeable person to use this application.
 This application does not have journals.

CHAPTER 7

CONCLUSION

After all the hard work is done for electricity bill management system is here. It is a software which helps the user to work with the billing cycles, paying bills, managing different DETAILS under which are working etc.

This software reduces the amount of manual data entry and gives greater efficiency. The User Interface of it is very friendly and can be easily used by anyone.

It also decreases the amount of time taken to write details and other modules.

CHAPTER-8

REFERENCES

Here are some of the references which we followed to complete this project-

- Youtube
- Oracle website
- Java tutorials
- Scribd
- MySql Documentation
- Jdbc connection
- Geeksforgeeks
- JavaTpoint