

# **JAVA AWT BASED -E-VOTING SYSTEM-SQL CONNECTIVITY USING JDBC**

A

Report

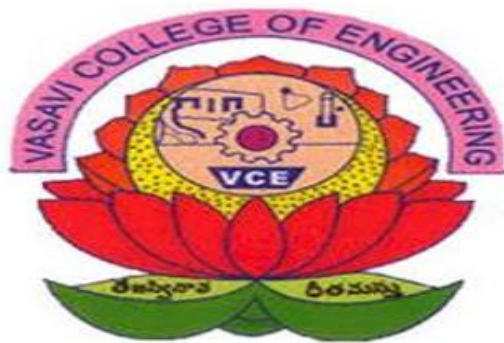
submitted in partial fulfilment of  
BE IV SEMESTER DATABASE MANAGEMENT SYSTEM LAB  
INFORMATION TECHNOLOGY

By

V.Swetha Reddy <1602-18-737-110>

Under the Guidance of

**B.Leelavathy**



**Department of Information Technology**

**Vasavi College of Engineering(Autonomous)**

**(Affiliated to Osmania University)**

**Ibrahimbagh, Hyderabad-500031**

**2019-2020**

# BONAFIDE CERTIFICATE

This is to certify that this project report titled “**E-Voting System**” is the bonafide mini project work of V.SWETHA REDDY bearing hall ticket number 1602-18-737-110 under the guidance of B. Leelavathy during 4th semester B.E for the academic year 2019-2020.

**External Examiner**

**Internal Examiner**

B.LEELAVATHY

Assistant professor

Department of Information Technology

## **AIM AND PRIORITY OF THE PROJECT:**

To create a GUI form for the project of E-VOTING SYSTEM where in the user registers and were given his credentials. The user can cast his vote without going to the polling booth using his credentials(login ID and password).

The values entered (insertion, updation, deletion) by the admin for respective table in GUI should be updated in the database using **JDBC**.

## **ABSTRACT:**

Online voting is a web-based voting system that will help to manage elections easily and securely. In this system the voter do not have to go the polling booth to cast their vote.

They can use their personal computer to cast their vote. There is a database which is maintained in which all the names of voters with their complete information is stored. The system administrator registers the voters and candidates. After registration, the voter is assigned with a user id with which he/she can use to login and cast their vote.

## **INTRODUCTION:**

## **REQUIREMENTS:**

### List of Tables:

- ADMIN
- MANAGES
- LOGIN
- USERS
- VOTERS
- CANDIDATES

### List of attributes with their domain types:

#### Admin:

Admin id: aid-Number()

#### Manages:

Admin id: aid-Number()

User id: user\_id-Number()

Aadhar No.: aadhar\_no-Number()

#### Login:

User id: user\_id-Number()

Password: password-Varchar()

### Users:

User id: user\_id-Number()

Name: name -Varchar()

Date of Birth: dob-varchar()

Gender: gender-Varchar()

Aadhar No.: aadhar\_no-Number()

Voter id: voter\_id-Number()

Address: address-Varchar()

### Voters:

User id: user\_id-Number()

Aadhar No.: aadhar\_no-Varchar()

### Candidates:

Candidate id: cid-Number()

User id: user\_id-Number()

Aadhar No.: aadhar\_no-Number()

Symbol: symbol-Varchar()

## **ARCHITECTURE AND TECHNOLOGY USED:**

Java Eclipse, Oracle 11g Database, java SE version 8, SQL \*plus, java AWT

### **Eclipse:**

It is an integrated development environment (IDE) used in computer programming. It contains a base workspace and an extensible plug in system for customizing the environment. The Eclipse software development kit (SDK), which include java development tools is meant for java developers.

### **SQL \*plus:**

SQL \*plus is a command line tool proprietary to oracle. You can send SQL Queries to the server using the tool. It can also help you format the result of query. SQL is the query language that is used to communicate with the oracle server to access and modify data.

### **JAVA AWT:**

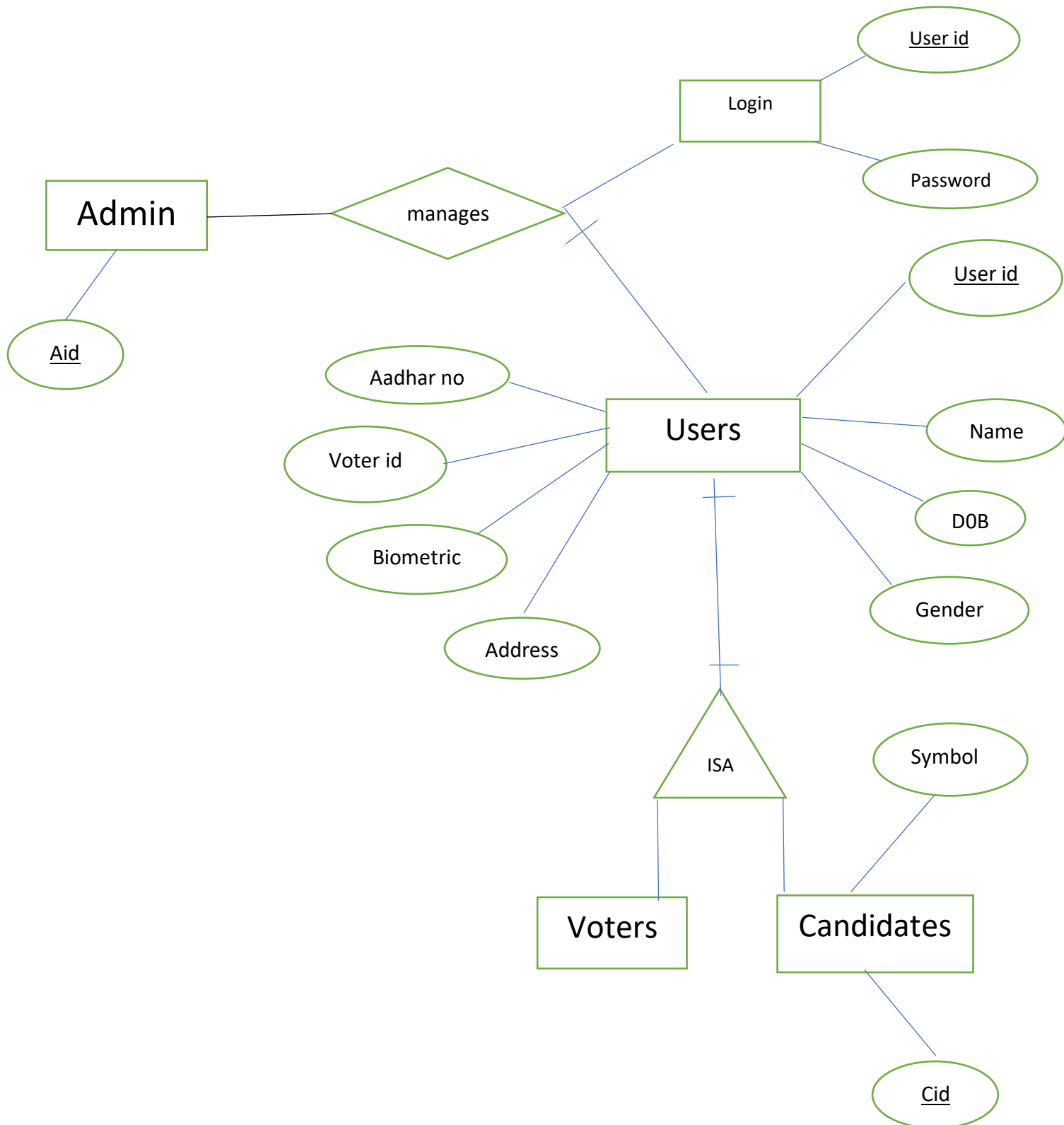
Abstract window tool kit is an API to develop GUI or Window based applications in java. Java AWT components are platform dependent i.e components are displayed according to the view of operating system. AWT is heavy weight that is components are using the resources of O.S.

### **JDBC:**

Java Database Connectivity is an application programming interface (API) for the programming language java , which defines how a client may access database. It is a java based data access technology used

for java database connectivity. It is the part of java Standard Edition Platform, from oracle corporation.

## ER DIAGRAM OF THIS PROJECT:





## **CREATION OF TABLES:**

### **DDL COMMANDS:**

```
SQL> create table admin(aid number(5),primary key(aid));
```

Table created.

```
SQL> create table users(user_id number(5),name  
varchar(15),dob varchar(10),gender varchar(2),aadhar_no  
number(5),voter_id number(5),address varchar(15),primary  
key(user_id,aadhar_no));
```

Table created.

```
SQL> create table login(user_id number(5),password  
varchar(5),primary key(user_id));
```

Table created.

```
SQL> create table manages(aid number(5),user_id  
number(5),aadhar_no number(5),foreign key(aid) references  
admin,foreign key(user_id,aadhar_no) references users);
```

Table created.

```
SQL> create table voters(user_id number(5),aadhar_no
number(5),foreign key(user_id,aadhar_no) references users);
```

Table created.

```
SQL> create table candidates(cid number(5),user_id
number(5),aadhar_no number(5),symbol varchar(20),foreign
key(user_id,aadhar_no) references users);
```

Table created.

### **TABLES DESCRIPTION:**

```
SQL> desc admin;
```

| Name  | Null?    | Type      |
|-------|----------|-----------|
| ----- |          |           |
| AID   | NOT NULL | NUMBER(5) |

```
SQL> desc users;
```

| Name      | Null?    | Type         |
|-----------|----------|--------------|
| -----     |          |              |
| USER_ID   | NOT NULL | NUMBER(5)    |
| NAME      |          | VARCHAR2(15) |
| DOB       |          | VARCHAR2(10) |
| GENDER    |          | VARCHAR2(2)  |
| AADHAR_NO | NOT NULL | NUMBER(5)    |

|          |              |
|----------|--------------|
| VOTER_ID | NUMBER(5)    |
| ADDRESS  | VARCHAR2(15) |

SQL> desc login;

| Name  | Null? | Type |
|-------|-------|------|
| ----- |       |      |

|          |          |             |
|----------|----------|-------------|
| USER_ID  | NOT NULL | NUMBER(5)   |
| PASSWORD |          | VARCHAR2(5) |

SQL> desc manages;

| Name  | Null? | Type |
|-------|-------|------|
| ----- |       |      |

|           |  |           |
|-----------|--|-----------|
| AID       |  | NUMBER(5) |
| USER_ID   |  | NUMBER(5) |
| AADHAR_NO |  | NUMBER(5) |

SQL> desc voters;

| Name  | Null? | Type |
|-------|-------|------|
| ----- |       |      |

|           |  |           |
|-----------|--|-----------|
| USER_ID   |  | NUMBER(5) |
| AADHAR_NO |  | NUMBER(5) |

SQL> desc candidates;

| Name  | Null? | Type |
|-------|-------|------|
| ----- |       |      |

|           |              |
|-----------|--------------|
| CID       | NUMBER(5)    |
| USER_ID   | NUMBER(5)    |
| AADHAR_NO | NUMBER(5)    |
| SYMBOL    | VARCHAR2(20) |

## **IMPLEMENTATION**

### **1.FRONT END PROGRAMS AND CONNECTIVITY**

#### **JAVA-SQL CONNECTIVITY USING JDBC:**

The connection to the database can be performed using java programming (JDBC API) as:

```
public void connectToDB() {
    try {

connection=DriverManager.getConnection("jdbc:oracle:thin:@localhost:
1521:ORCL","swetha","vasavi");
statement = connection.createStatement();
}
catch (SQLException connectException) {
    System.out.println(connectException.getMessage());
    System.out.println(connectException.getSQLState());
    System.out.println(connectException.getErrorCode());
    System.exit(1);
}
catch(Exception e){
    System.out.println("Unable to find and load driver");
    System.exit(1);
}
```

```
}}
```

AS THIS PROJECT CONTAINS 6 TABLES

**ADMIN ,MANAGES ,LOGIN ,USERS ,VOTERS  
,CANDIDATES**

BELOW IS THE CODE FOR ALL DML OPERATIONS ON THE  
TABLE USERS:

### Insert USERS:

```
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class InsertUser extends Frame
{
    Button insertUserButton;
    TextField useridText, nameText, dateofbirthText,
genderText,aadharnoText,voteridText,addressText;
    TextArea errorText;
    Connection connection;
    Statement statement;
    public InsertUser()
    {
        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
        catch (Exception e)
        {
            System.err.println("Unable to find and load driver");
            System.exit(1);
        }
        connectToDB();
    }

    public void connectToDB()
    {
        try
        {
            connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "swetha", "vasavi
");
            statement = connection.createStatement();
        }
        catch (SQLException connectException)
        {
            System.out.println(connectException.getMessage());
            System.out.println(connectException.getSQLState());
        }
    }
}
```

```

        System.out.println(connectException.getErrorCode());
        System.exit(1);
    }
}

public void buildGUI()
{
    insertUserButton = new Button("Insert User");
    insertUserButton.addActionListener(new ActionListener()
    {
        public void actionPerformed(ActionEvent e)
        {
            try
            {
                String query= "INSERT INTO users VALUES(" +
useridText.getText() + "," + "'" + nameText.getText() + "'," + "'" +
dateofbirthText.getText() + "'," + "'" + genderText.getText() + "',"
+aadharnoText.getText() + "," + voteridText.getText() + "," + "'" +
addressText.getText() + "'"");
                int i = statement.executeUpdate(query);
                errorText.append("\nInserted " + i + " rows
successfully");
            }
            catch (SQLException insertException)
            {
                displaySQLErrors(insertException);
            }
        }
    });

    useridText = new TextField(15);
    nameText = new TextField(15);
    dateofbirthText = new TextField(15);
    genderText = new TextField(15);
    aadharnoText = new TextField(15);
    voteridText = new TextField(15);
    addressText = new TextField(15);

    errorText = new TextArea(10, 40);
    errorText.setEditable(false);

    Panel first = new Panel();
    first.setLayout(new GridLayout(4, 2));
    first.add(new Label("User ID:"));
    first.add(useridText);
    first.add(new Label("Name:"));
    first.add(nameText);
    first.add(new Label("Date of Birth:"));
    first.add(dateofbirthText);
    first.add(new Label("Gender:"));
    first.add(genderText);
    first.add(new Label("Aadhar No.:"));
    first.add(aadharnoText);
    first.add(new Label("Voter ID:"));
    first.add(voteridText);
    first.add(new Label("Address:"));
    first.add(addressText);
    first.setBounds(150,90,200,100);

```

```

        Panel second = new Panel(new GridLayout(4, 1));
        second.add(insertUserButton);
        second.setBounds(150,220,150,100);

        Panel third = new Panel();
        third.add(errorText);
        third.setBounds(150,320,300,200);

        setLayout(null);

        add(first);
        add(second);
        add(third);

        setTitle("New User Creation");
        setSize(600,750);
        setVisible(true);
    }

    private void displaySQLExceptions(SQLException e)
    {
        errorText.append("\nSQLException: " + e.getMessage() + "\n");
        errorText.append("SQLState:      " + e.getSQLState() + "\n");
        errorText.append("VendorError:  " + e.getErrorCode() + "\n");
    }

    public static void main(String[] args)
    {
        InsertUser inu = new InsertUser();

        inu.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        inu.buildGUI();
    }
}

```

## UPDATE USERS:

```

import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class UpdateUser extends Frame
{
    Button updateUserButton;
    List userIDList;
    TextField useridText, nameText,dateofbirthText,
genderText,aadharnoText,voteridText,addressText;
    TextArea errorText;
    Connection connection;
}

```

```

Statement statement;
ResultSet rs;

public UpdateUser()
{
    try
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch (Exception e)
    {
        System.err.println("Unable to find and load driver");
        System.exit(1);
    }
    connectToDB();
}

public void connectToDB()
{
    try
    {
        connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","swetha","vasavi
");
        statement = connection.createStatement();
    }
    catch (SQLException connectException)
    {
        System.out.println(connectException.getMessage());
        System.out.println(connectException.getSQLState());
        System.out.println(connectException.getErrorCode());
        System.exit(1);
    }
}

private void loadUsers()
{
    try
    {
        rs = statement.executeQuery("SELECT user_id FROM users");
        while (rs.next())
        {
            userIDList.add(rs.getString("user_id"));
        }
    }
    catch (SQLException e)
    {
        displaySQLErrors(e);
    }
}

public void buildGUI()
{
    userIDList = new List(10);
    loadUsers();
    add(userIDList);

    userIDList.addItemListener(new ItemListener()

```



```

{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {
            rs = statement.executeQuery("SELECT * FROM users
where user_id =" + userIDList.getSelectedItemAt());
            rs.next();
            useridText.setText(rs.getString("user_id"));
            nameText.setText(rs.getString("name"));
            dateofbirthText.setText(rs.getString("dob"));
            genderText.setText(rs.getString("gender"));
            aadharNoText.setText(rs.getString("aadhar_no"));
            voteridText.setText(rs.getString("voter_id"));
            addressText.setText(rs.getString("address"));

        }
        catch (SQLException selectException)
        {
            displaySQLErrors(selectException);
        }
    }
});

updateUserButton = new Button("Update User");
updateUserButton.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        try
        {
            Statement statement =
connection.createStatement();

            int i = statement.executeUpdate("UPDATE users "
+ "SET name=' " + nameText.getText()
+ "dob=' " +
dateofbirthText.getText() + "', "
+ "gender=' " + genderText.getText() +
"', " + "aadhar_no = " + aadharNoText.getText()
+ ", " + "voter_id =
" + voteridText.getText() + ", "
+ "address=' " +
addressText.getText() + "' WHERE user_id = "
+ userIDList.getSelectedItemAt());
            errorText.append("\nUpdated " + i + " rows
successfully");

            userIDList.removeAll();
            loadUsers();
        }
        catch (SQLException insertException)
        {
            displaySQLErrors(insertException);
        }
    }
});

```

```

useridText = new TextField(15);
useridText.setEditable(false);
nameText = new TextField(15);
dateofbirthText = new TextField(15);
genderText = new TextField(15);
aadharNoText = new TextField(15);
voteridText = new TextField(15);
addressText = new TextField(15);

errorText = new TextArea(10, 40);
errorText.setEditable(false);

Panel first = new Panel();
first.setLayout(new GridLayout(4, 2));
first.add(new Label("User ID:"));
first.add(useridText);
first.add(new Label("Name:"));
first.add(nameText);
first.add(new Label("Date of Birth:"));
first.add(dateofbirthText);
first.add(new Label("Gender:"));
first.add(genderText);
first.add(new Label("Aadhar No.:"));
first.add(aadharNoText);
first.add(new Label("Voter ID:"));
first.add(voteridText);
first.add(new Label("Address:"));
first.add(addressText);

Panel second = new Panel(new GridLayout(4, 1));
second.add(updateUserButton);

Panel third = new Panel();
third.add(errorText);

add(first);
add(second);
add(third);

setTitle("Update User");
setSize(600, 800);
setLayout(new FlowLayout());
setVisible(true);
}

private void displaySQLExceptions(SQLException e)
{
    errorText.append("\nSQLException: " + e.getMessage() + "\n");
    errorText.append("SQLState: " + e.getSQLState() + "\n");
    errorText.append("VendorError: " + e.getErrorCode() + "\n");
}

public static void main(String[] args)
{
    UpdateUser upu = new UpdateUser();

    upu.addWindowListener(new WindowAdapter(){
        public void windowClosing(WindowEvent e)

```

```

        {
            System.exit(0);
        }
    });

    upu.buildGUI();
}
}

```

## DELETE USERS:

```

import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class DeleteUser extends Frame
{
    Button deleteUserButton;
    List userIDList;
    TextField useridText, nameText, dateofbirthText,
genderText, aadharnoText, voteridText, addressText;
    TextArea errorText;
    Connection connection;
    Statement statement;
    ResultSet rs;

    public DeleteUser()
    {
        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
        catch (Exception e)
        {
            System.err.println("Unable to find and load driver");
            System.exit(1);
        }
        connectToDB();
    }

    public void connectToDB()
    {
        try
        {
            connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "swetha", "vasavi
");
            statement = connection.createStatement();

        }
        catch (SQLException connectException)
        {
            System.out.println(connectException.getMessage());
            System.out.println(connectException.getSQLState());
            System.out.println(connectException.getErrorCode());
            System.exit(1);
        }
    }

    private void loadUsers()

```

```

{
    try
    {
        rs = statement.executeQuery("SELECT * FROM users");
        while (rs.next())
        {
            userIDList.add(rs.getString("user_id"));
        }
    }
    catch (SQLException e)
    {
        displaySQLErrors(e);
    }
}

public void buildGUI()
{
    userIDList = new List(10);
    loadUsers();
    add(userIDList);

    userIDList.addItemListener(new ItemListener()
    {
        public void itemStateChanged(ItemEvent e)
        {
            try
            {
                rs = statement.executeQuery("SELECT * FROM
users");
                while (rs.next())
                {
                    if
(rs.getString("user_id").equals(userIDList.getSelectedItem()))
                    break;
                }
                if (!rs.isAfterLast())
                {
                    useridText.setText(rs.getString("user_id"));
                    nameText.setText(rs.getString("name"));

                    dateofbirthText.setText(rs.getString("dob"));

                    genderText.setText(rs.getString("gender"));

                    aadharnoText.setText(rs.getString("aadhar_no"));

                    voteridText.setText(rs.getString("voter_id"));

                    addressText.setText(rs.getString("Address"));
                }
            }
            catch (SQLException selectException)
            {
                displaySQLErrors(selectException);
            }
        }
    });
}

```

```

deleteUserButton = new Button("Delete User");
deleteUserButton.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        try
        {
            Statement statement =
connection.createStatement();
            int i = statement.executeUpdate("DELETE FROM
users WHERE user_id = "
+ userIDList.getSelectedItem());
            errorText.append("\nDeleted " + i + " rows
successfully");

            useridText.setText(null);
            nameText.setText(null);
            dateofbirthText.setText(null);
            genderText.setText(null);
            aadharnoText.setText(null);
            voteridText.setText(null);
            addressText.setText(null);
            userIDList.removeAll();
            loadUsers();
        }
        catch (SQLException insertException)
        {
            displaySQLErrors(insertException);
        }
    }
});

useridText = new TextField(20);
nameText = new TextField(20);
dateofbirthText = new TextField(20);
genderText = new TextField(20);
aadharnoText = new TextField(20);
voteridText = new TextField(20);
addressText = new TextField(20);
errorText = new TextArea(10, 40);
errorText.setEditable(false);

Panel first = new Panel();
first.setLayout(new GridLayout(4, 2));
first.add(new Label("User ID:"));
first.add(useridText);
first.add(new Label("Name:"));
first.add(nameText);
first.add(new Label("Date of Birth:"));
first.add(dateofbirthText);
first.add(new Label("Gender:"));
first.add(genderText);
first.add(new Label("Aadhar No:"));
first.add(aadharnoText);
first.add(new Label("Voter Id:"));
first.add(voteridText);
first.add(new Label("Address:"));
first.add(addressText);

```

```

        Panel second = new Panel(new GridLayout(4, 1));
        second.add(deleteUserButton);

        Panel third = new Panel();
        third.add(errorText);

        add(first);
        add(second);
        add(third);

        setTitle("Remove User");
        setSize(750, 800);
        setLayout(new FlowLayout());
        setVisible(true);
    }

    private void displaySQLExceptions(SQLException e)
    {
        errorText.append("\nSQLException: " + e.getMessage() + "\n");
        errorText.append("SQLState:      " + e.getSQLState() + "\n");
        errorText.append("VendorError:  " + e.getErrorCode() + "\n");
    }

    public static void main(String[] args)
    {
        DeleteUser delu = new DeleteUser();

        delu.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                System.exit(0);
            }
        });

        delu.buildGUI();
    }
}

```

## MAIN FRAME GUI:

```

import java.awt.*;
import java.awt.event.*;

class EVoting extends Frame implements ActionListener
{
    String msg = "";
    Label ll;
    InsertUser inu;
    UpdateUser upu;
    DeleteUser delu;
    InsertVoter inv;
    UpdateVoter upv;
    DeleteVoter delv;
}

```

```
InsertCandidate inc;  
UpdateCandidate upc;  
DeleteCandidate delc;
```

```
EVoting()  
{
```

```
    ll = new Label();  
    ll.setAlignment(Label.CENTER);  
    ll.setBounds(90,250,250,100);  
    ll.setText("Welcome to E-VOTING System");  
    add(ll);  
  
    // create menu bar and add it to frame  
    MenuBar mbar = new MenuBar();  
    setMenuBar(mbar);  
  
    // create the menu items and add it to Menu  
    Menu user = new Menu("Users");  
    MenuItem item1, item2, item3;  
    user.add(item1 = new MenuItem("Insert User"));  
    user.add(item2 = new MenuItem("Update User"));  
    user.add(item3 = new MenuItem("Delete User"));  
    mbar.add(user);  
  
    Menu voter = new Menu("Voters");  
    MenuItem item4, item5, item6;  
    voter.add(item4 = new MenuItem("Insert Voter"));  
    voter.add(item5 = new MenuItem("Update Voter"));  
    voter.add(item6 = new MenuItem("Delete Voter"));  
    mbar.add(voter);  
  
    Menu candidate = new Menu("Candidates");  
    MenuItem item7, item8, item9;  
    candidate.add(item7 = new MenuItem("Insert Candidate"));  
    candidate.add(item8 = new MenuItem("Update Candidate"));  
    candidate.add(item9 = new MenuItem("Delete Candidate"));  
    mbar.add(candidate);  
    Menu login = new Menu("logins");  
    MenuItem item10;  
    login.add(item10 = new MenuItem("Register"));  
  
    mbar.add(login);  
  
    // register listeners  
    item1.addActionListener(this);  
    item2.addActionListener(this);  
    item3.addActionListener(this);  
    item4.addActionListener(this);  
    item5.addActionListener(this);  
    item6.addActionListener(this);  
    item7.addActionListener(this);  
    item8.addActionListener(this);  
    item9.addActionListener(this);  
    item10.addActionListener(this);
```

handle the Window event: windowClosing

```
// Anonymous inner class which extends WindowAdaptor to
addWindowListener(new WindowAdapter(){
    public void windowClosing(WindowEvent we)
    {
        System.exit(0);
    }
});

//Frame properties
setTitle("E-Voting System");
setFont(new Font("White" ,Font.BOLD, 14));
setLayout(null);
setSize(500, 600);
setVisible(true);
}

public void actionPerformed(ActionEvent ae)
{
    String arg = ae.getActionCommand();
    if(arg.equals("Insert User"))
    {
        inu = new InsertUser();

        inu.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                inu.dispose();
            }
        });
        inu.buildGUI();
    }

    else if(arg.equals("Update Users"))
    {
        upu = new UpdateUser();
        upu.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                upu.dispose();
            }
        });
        upu.buildGUI();
    }

    else if(arg.equals("Delete User"))
    {
        delu = new DeleteUser();

        delu.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                delu.dispose();
            }
        });
    }
}
```



```

        });
        delu.buildGUI();
    }

    else if(arg.equals("Insert Voter"))
    {
        inv = new InsertVoter();
        inv.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                inv.dispose();
            }
        });
        inv.buildGUI();
    }
    else if(arg.equals("Update Voter"))
    {
        upv = new UpdateVoter();
        setVisible(false);
        upv.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                upv.dispose();
                setVisible(true);
            }
        });
        upv.buildGUI();
    }
    else if(arg.equals("Delete Voter"))
    {
        delv = new DeleteVoter();
        setVisible(false);
        delv.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                delv.dispose();
                setVisible(true);
            }
        });
        delv.buildGUI();
    }
    else if(arg.equals("Insert Candidate"))
    {
        inc = new InsertCandidate();
        setVisible(false);
        inc.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e)
            {
                inc.dispose();
                setVisible(true);
            }
        });
        inc.buildGUI();
    }
    else if(arg.equals("Delete Candidate"))
    {
        delc = new DeleteCandidate();
        setVisible(false);
        delc.addWindowListener(new WindowAdapter(){

```

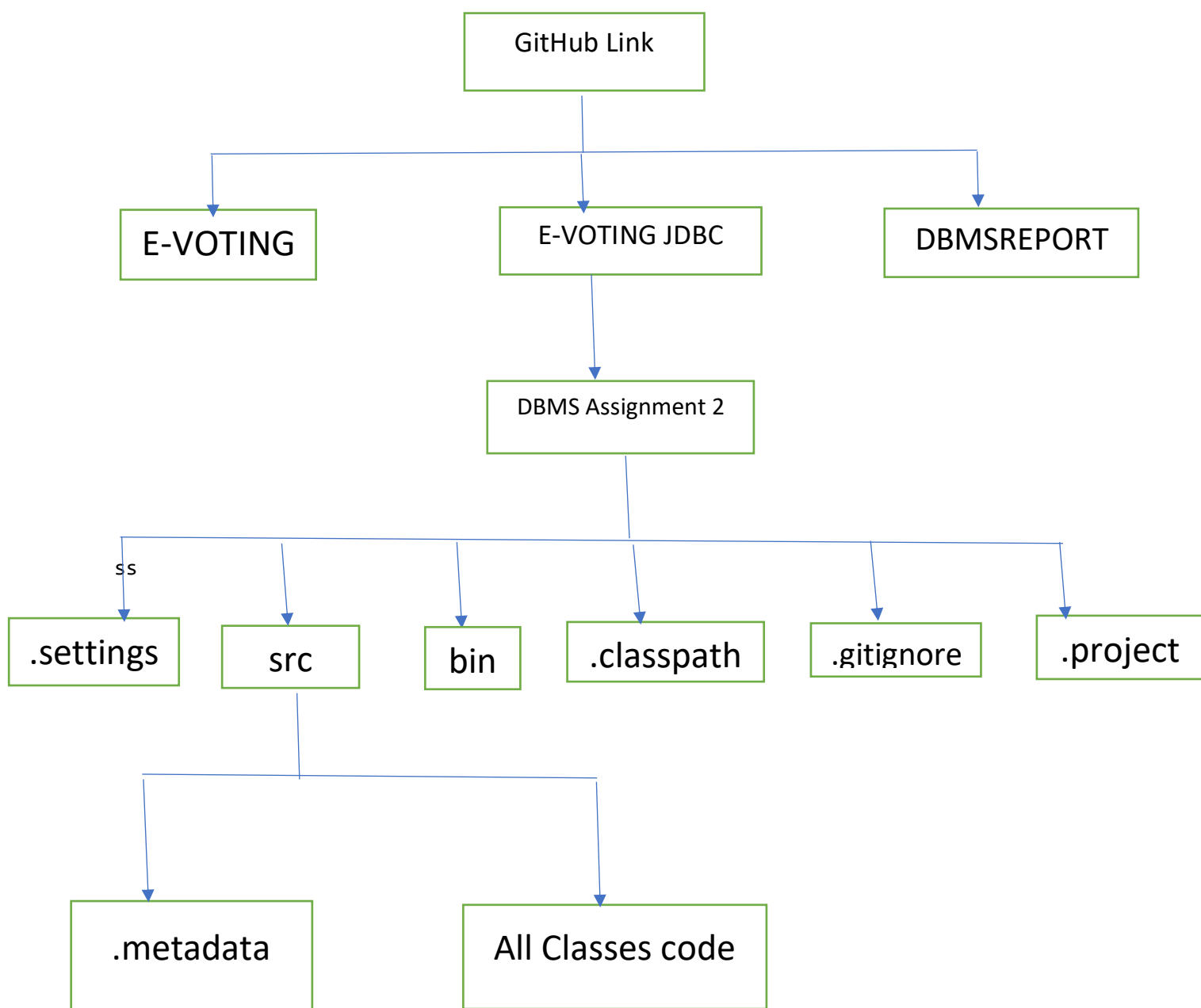
```

        public void windowClosing(WindowEvent e)
        {
            delc.dispose();
            setVisible(true);
        }
    });
    delc.buildGUI();
}
else if(arg.equals("Update Candidate"))
{
    upc = new UpdateCandidate();
    setVisible(false);
    upc.addWindowListener(new WindowAdapter(){
        public void windowClosing(WindowEvent e)
        {
            upc.dispose();
            setVisible(true);
        }
    });
    upc.buildGUI();
}
}
public static void main(String...args)
{
    new EVoting();
}
}

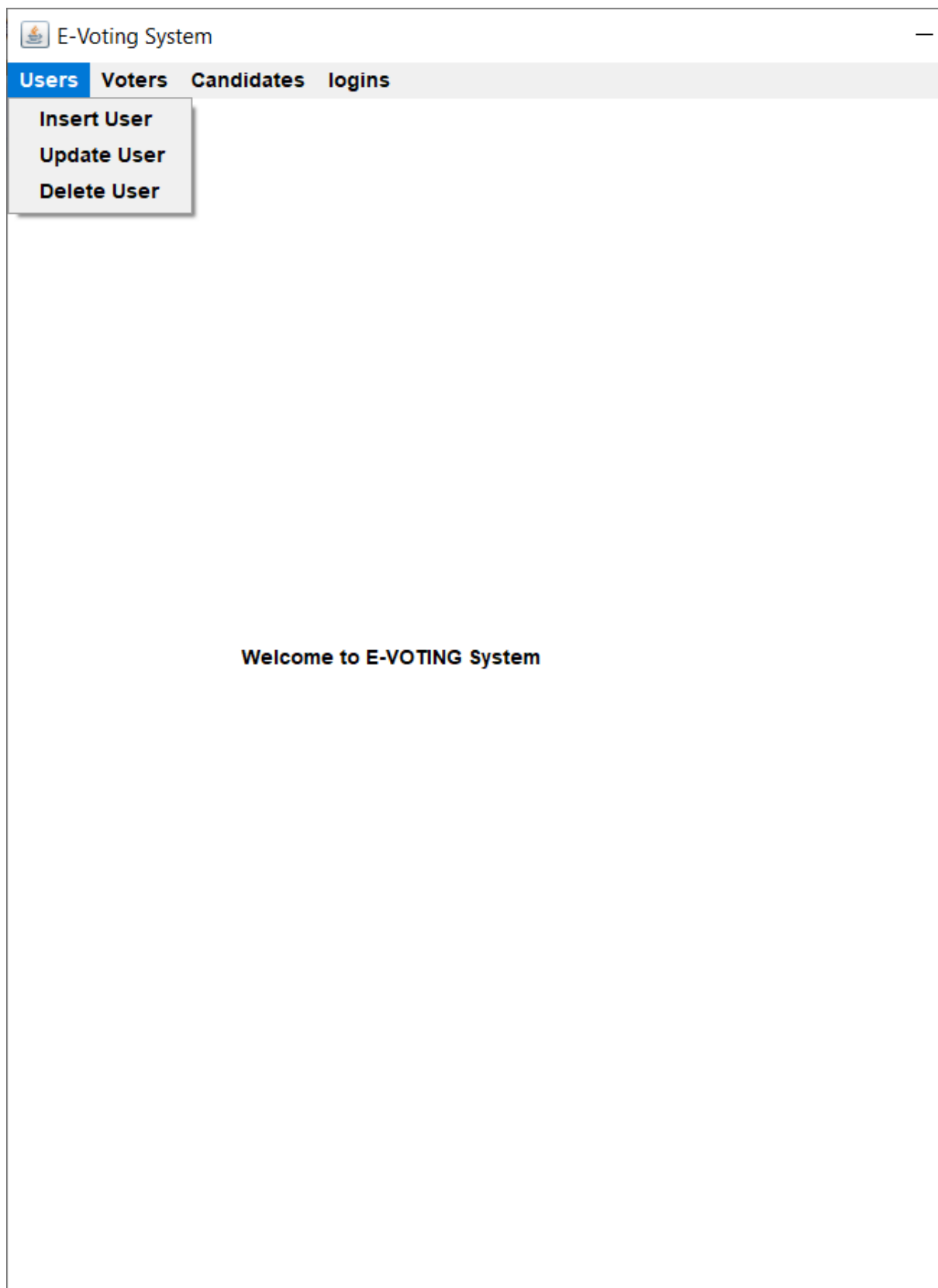
```

## 2.GITHUB LINK AND FOLDER STRUCTURE:

<https://github.com/swethareddy0/DBMSPROJECT>



THE FIRST FRAME THAT WILL BE DISPLAYED WHEN WE EXECUTE THE PROGRAM IS



## SCREENSHOT OF INSERT USER:

 New User Creation

User ID:  Name:   
Date of Birth:  Gender:   
Aadhar No:  Voter ID:   
Address:

Inserted 1 rows successfully



SQL\*Plus: Release 11.2.0.2.0 Production on Mon Apr 27 20:19:52 2020

Copyright (c) 1982, 2010, Oracle. All rights reserved.

SQL> conn swetha/vasavi

Connected.

SQL> select \* from users;

| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS   |
|---------|---------|-----------|----|-----------|----------|-----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda  |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda  |
| 103     | Soumya  | 06-jun-93 | F  | 67892     | 3434     | Hyderabad |
| 104     | Vishnu  | 01-may-95 | M  | 55232     | 7502     | Hyderabad |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda  |

SQL>

## SCREENSHOT OF UPDATE USER:

Update User

101  
102  
103  
104  
105

User ID: 104  
Date of Birth: 01-may-97  
Aadhar No.: 55232  
Address: Nalgonda

Name: Vishnu  
Gender: M  
Voter ID: 7502

Update User

Updated 1 rows successfully

### Run SQL Command Line

SQL\*Plus: Release 11.2.0.2.0 Production on Mon Apr 27 20:19:52 2020

Copyright (c) 1982, 2010, Oracle. All rights reserved.

SQL> conn swetha/vasavi

Connected.

SQL> select \* from users;


| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS   |
|---------|---------|-----------|----|-----------|----------|-----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda  |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda  |
| 103     | Soumya  | 06-jun-93 | F  | 67892     | 3434     | Hyderabad |
| 104     | Vishnu  | 01-may-95 | M  | 55232     | 7502     | Hyderabad |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda  |

SQL> select \* from users;

| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS   |
|---------|---------|-----------|----|-----------|----------|-----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda  |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda  |
| 103     | Soumya  | 06-jun-93 | F  | 67892     | 3434     | Hyderabad |
| 104     | Vishnu  | 01-may-97 | M  | 55232     | 7502     | Nalgonda  |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda  |

SQL> ■

## SCREENSHOT OF DELETE USER:

 Remove User

101  
102  
**104**  
105


User ID:  
Date of Birth:  
Aadhar No:  
Address:

104  
01-may-95  
55232  
Nalgonda

Name:  
Gender:  
Voter Id:

vishnu  
M  
7502

Delete User

 Remove User

101  
102  
105

User ID:  
Date of Birth:  
Aadhar No:  
Address:

Name:  
Gender:  
Voter Id:

Delete Sailor

Deleted 1 rows successfully

SQL\*Plus: Release 11.2.0.2.0 Production on Mon Apr 27 20:19:52 2020

Copyright (c) 1982, 2010, Oracle. All rights reserved.

SQL> conn swetha/vasavi

Connected.

SQL> select \* from users;

| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS   |
|---------|---------|-----------|----|-----------|----------|-----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda  |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda  |
| 103     | Soumya  | 06-jun-93 | F  | 67892     | 3434     | Hyderabad |
| 104     | Vishnu  | 01-may-95 | M  | 55232     | 7502     | Hyderabad |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda  |

SQL> select \* from users;

| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS   |
|---------|---------|-----------|----|-----------|----------|-----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda  |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda  |
| 103     | Soumya  | 06-jun-93 | F  | 67892     | 3434     | Hyderabad |
| 104     | Vishnu  | 01-may-97 | M  | 55232     | 7502     | Nalgonda  |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda  |

SQL> select \* from users;

| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS  |
|---------|---------|-----------|----|-----------|----------|----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda |

SQL> \_



SQL\*Plus: Release 11.2.0.2.0 Production on Mon Apr 27 20:19:52 2020

Copyright (c) 1982, 2010, Oracle. All rights reserved.

SQL> conn swetha/vasavi

Connected.

SQL> select \* from users;

| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS   |
|---------|---------|-----------|----|-----------|----------|-----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda  |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda  |
| 103     | Soumya  | 06-jun-93 | F  | 67892     | 3434     | Hyderabad |
| 104     | Vishnu  | 01-may-95 | M  | 55232     | 7502     | Hyderabad |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda  |

SQL> select \* from users;


| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS   |
|---------|---------|-----------|----|-----------|----------|-----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda  |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda  |
| 103     | Soumya  | 06-jun-93 | F  | 67892     | 3434     | Hyderabad |
| 104     | Vishnu  | 01-may-97 | M  | 55232     | 7502     | Nalgonda  |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda  |

SQL> select \* from users;

| USER_ID | NAME    | DOB       | GE | AADHAR_NO | VOTER_ID | ADDRESS  |
|---------|---------|-----------|----|-----------|----------|----------|
| 101     | Mounika | 03-jan-98 | F  | 23451     | 8904     | Nalgonda |
| 102     | Arjun   | 12-mar-98 | M  | 25671     | 3452     | Nalgonda |
| 105     | Swetha  | 04-aug-98 | F  | 66681     | 9991     | Nalgonda |

SQL> \_

## SCREENSHOT OF INSERT CANDIDATE:

 **New candidate Creation** — □ ×

Candidate ID:

User id:

Aadhar No.:


Symbol:

Inserted 1 rows successfully

^

v

va - DBMS Assignment 2/src/InsertCandidate.java - Eclipse IDE

 **Run SQL Command Line** — □ ×

```
SQL*Plus: Release 11.2.0.2.0 Production on Tue Apr 28 08:23:00 2020
Copyright (c) 1982, 2010, Oracle. All rights reserved.

SQL> conn swetha/vasavi
Connected.
SQL> select * from users;

  USER_ID NAME      DOB      GE  AADHAR_NO  VOTER_ID ADDRESS
-----
    101 Mounika    03-jan-98 F    23451      8904 Nalgonda
    102 Arjun      12-mar-98 M    25671      3452 Nalgonda
    107 Soumya     01-mar-97 M    13456      7777 Nalgonda
    105 Swetha     04-aug-98 F    66681      9991 Nalgonda
    106 Pavani     08-may-91 F    66692      5502 Nalgonda

SQL> select * from candidates;

  CID  USER_ID  AADHAR_NO  SYMBOL
-----
    51     102    25671 Car
    53     105    66681 scissors
    52     107    13456 Lotus

SQL>
```

## SCREENSHOT OF UPDATE CANDIDATE:

51  
53  
52

Update Candidate

Candidate ID:

53

User Id:

105

Aadhar No.:

66681

Symbol:

Ring

Updated 1 rows successfully

Update Candidate

Run SQL Command Line

```
SQL> conn swetha/vasavi
Connected.
SQL> select * from users;

  USER_ID NAME      DOB      GE  AADHAR_NO  VOTER_ID ADDRESS
-----
    101 Mounika    03-jan-98  F    23451      8904 Nalgonda
    102 Arjun      12-mar-98  M    25671      3452 Nalgonda
    107 Soumya     01-mar-97  M    13456      7777 Nalgonda
    105 Swetha     04-aug-98  F    66681      9991 Nalgonda
    106 Pavani     08-may-91  F    66692      5502 Nalgonda

SQL> select * from candidates;

  CID  USER_ID  AADHAR_NO  SYMBOL
-----
    51     102    25671  Car
    53     105    66681  scissors
    52     107    13456  Lotus

SQL> select * from candidates;

  CID  USER_ID  AADHAR_NO  SYMBOL
-----
    51     102    25671  Car
    53     105    66681  Ring
    52     107    13456  Lotus

SQL>
```

## SCREENSHOT OF DELETE CANDIDATE:

Remove Candidate

51

53

52

Candidate ID:

User ID:

Aadhar No.:

Symbol:

53

105

66681

Ring

Delete Candidate

Remove Candidate

51

52

Candidate ID:

User ID:

Aadhar No.:

Symbol:

Delete Candidate

Deleted 1 rows successfully

```
SQL> select * from candidates;
```

| CID | USER_ID | AADHAR_NO | SYMBOL   |
|-----|---------|-----------|----------|
| 51  | 102     | 25671     | Car      |
| 53  | 105     | 66681     | scissors |
| 52  | 107     | 13456     | Lotus    |

```
SQL> select * from candidates;
```

| CID | USER_ID | AADHAR_NO | SYMBOL |
|-----|---------|-----------|--------|
| 51  | 102     | 25671     | Car    |
| 53  | 105     | 66681     | Ring   |
| 52  | 107     | 13456     | Lotus  |

```
SQL> select * from candidates;
```

| CID | USER_ID | AADHAR_NO | SYMBOL |
|-----|---------|-----------|--------|
| 51  | 102     | 25671     | Car    |
| 52  | 107     | 13456     | Lotus  |

```
SQL>
```

## **RESULT:**

The process of entering information into the frame created by java code so that the data is reflected in the database using JDBC connectivity is done successfully.

## **DISCUSSION AND FUTURE WORK:**

The application done upto now is a basic interface where the admin manages users by entering the data which were reflected in the data base through JDBC connectivity. In future the project will be edited in such a manner to add more secured information related to users so that there will be no issues.

## **REFERENCES:**

<https://docs.oracle.com/javase/8/docs/api/>

<https://www.geeksforgeeks.org/establishing-jdbc-connection-in-java/>

<https://www.javatpoint.com/java-awt>