**JAVA SWING BASED – CAREERDENDROGRAM – SQL CONNECTIVITY USING JDBC**

*A Report*

*Submitted in partial fulfillment of the Requirements*

*for the COURSE*

**DATABASE MANAGEMENT SYSTEMS**

**By**

### BHANUKIRAN <1602-21-737-009>

**Under the guidance of Ms B. Leelavathy**



### Department of Information Technology Vasavi College of Engineering (Autonomous) (Affiliated to Osmania University) Ibrahimbagh, Hyderabad-31

**2022-2023**

BONAFIDE CERTIFICATE

This is to certify that this project report titled

***‘TWITTER DATABASE MANAGEMENT’***

is a project work of **Anurag Sai.Y** bearing roll no. 1602-21-737-009who carried out this project under my supervision in the IV semester for the academic year 2022- 2023

3

Signature Signature

External Examiner Internal Examiner

# ABSTRACT

Career Dendrogram is a console-based project that connects students looking for career choices with great skills and requirements. Every user, must enter their details and information required to generate the right career choice. The students can look out for jobs they are interested in. It allows the student to know what can be the best choice of career.

# Requirement Analysis

## List of Tables:

* users
* authentication
* tweet
* retweets

## List of Attributes with their Domain Types:

**USERS**

USERNAME NOT NULL VARCHAR2(40)

FIRSTNAME NOT NULL VARCHAR2(50)

LASTNAME VARCHAR2(50)

GENDER NOT NULL CHAR(1)

BIRTHDATE NOT NULL DATE

FOLLOWERS NUMBER(10)

FOLLOWING NUMBER(10)

**Authentication**

EMAIL NOT NULL VARCHAR2(100)

USERNAME VARCHAR2(40)

PASSWORD NOT NULL VARCHAR2(32)

SEC\_CODE NOT NULL NUMBER(20)

**Tweet**

* SID VARCHAR2(15)
* SS1 NOT NULL VARCHAR2(30)
* SS2 NOT NULL VARCHAR2(30)
* AOI NOT NULL VARCHAR2(30)

**Retweets**

TWEETID NOT NULL NUMBER(20)

USERNAME VARCHAR2(40)

# AIM AND PRIORITY OF THE PROJECT

To create a **Java GUI-based** desktop application that connects students looking for career choices with skills and Interest. It takes values like student name, username, Age, Skills, etc through forms which are then updated in the database using JDBC connectivity.

# ARCHITECTURE AND TECHNOLOGY

### Software used:

Java, Oracle 11g Database, Java SE version 14, Run SQL.

### Java SWING:

**Java SWING** is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) - an API for providing a graphical user interface (GUI) for Java programs.

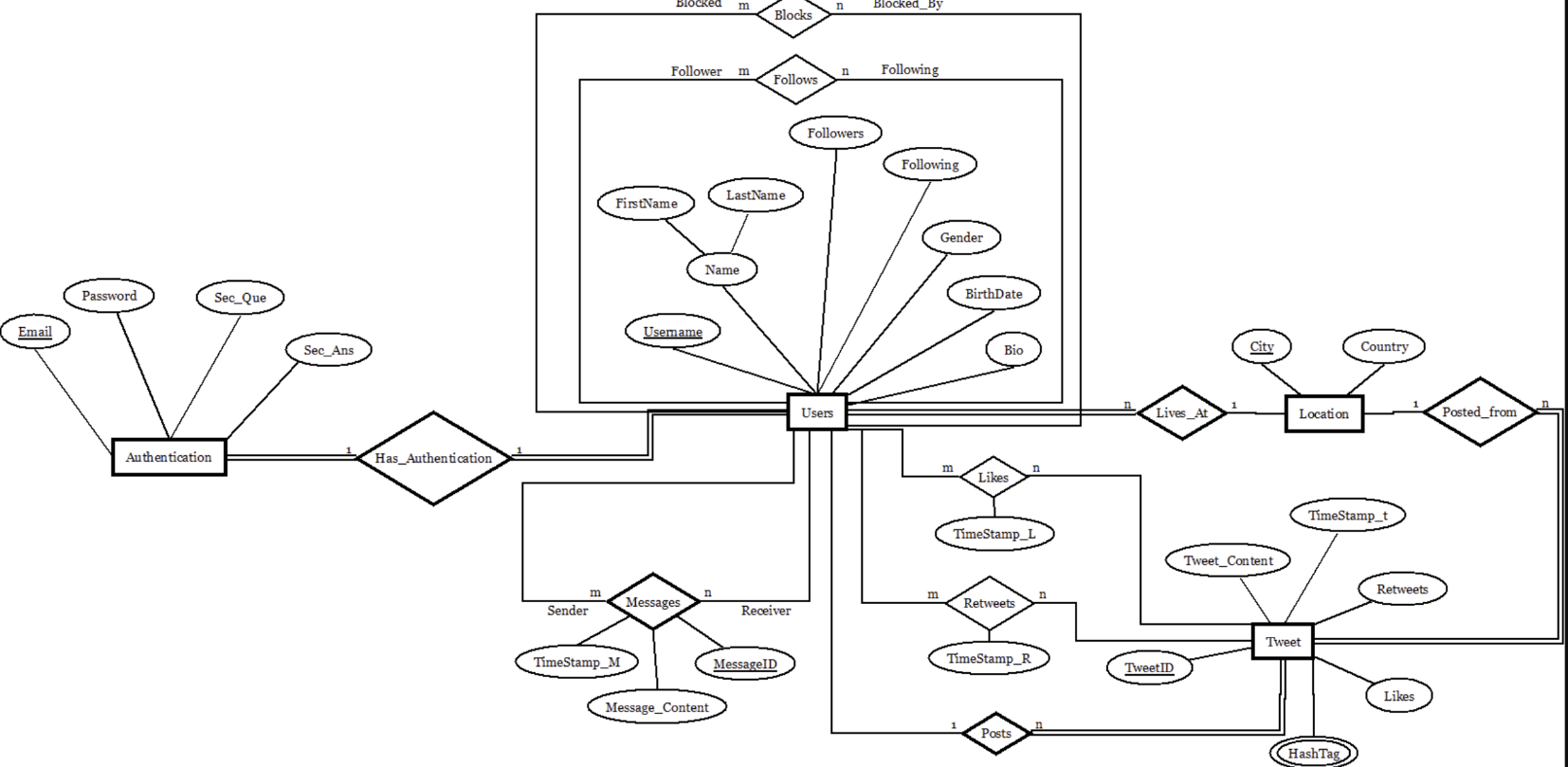
Swing was developed to provide a more sophisticated set of GUI components than the earlier AWT. Swing provides a look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

### SQL:

Structure Query Language(SQL) is a database query language used for storing and managing data in **Relational** DBMS. SQL was the first commercial language introduced for E.F Codd's Relational model of database. Today almost all RDBMS (MySql, Oracle, Infomix, Sybase, MS Access) use **SQL** as the standard database query language. SQL is used to perform all types of data operations in RDBMS.

**DESIGN**

### Entity Relationship Diagram

****

**TABLE CREATED IN SQL:**

1. Users Table

A screenshot of a computer program

Description automatically generated

1. Authentication Table

A black screen with white text

Description automatically generated

1. retweets Table

A black screen with white text

Description automatically generated

1. tweet table

A screen shot of a computer

Description automatically generated

### DATABASE DESIGN:

SQL> desc authentication;

Name Null? Type

----------------------------------------- -------- ----------------------------

EMAIL NOT NULL VARCHAR2(100)

USERNAME VARCHAR2(40)

PASSWORD NOT NULL VARCHAR2(32)

SEC\_CODE NOT NULL NUMBER(20)

SQL> desc users;

Name Null? Type

----------------------------------------- -------- ----------------------------

USERNAME NOT NULL VARCHAR2(40)

FIRSTNAME NOT NULL VARCHAR2(50)

LASTNAME VARCHAR2(50)

GENDER NOT NULL CHAR(1)

BIRTHDATE NOT NULL DATE

FOLLOWERS NUMBER(10)

FOLLOWING NUMBER(10)

SQL> desc tweet;

Name Null? Type

----------------------------------------- -------- ----------------------------

TWEETID NOT NULL NUMBER(20)

TWEET\_CONTENT NOT NULL VARCHAR2(280)

USERNAME NOT NULL VARCHAR2(40)

LIKES NUMBER(20)

RETWEETS NUMBER(20)

# IMPLEMENTATION

### JAVA-SQL Connectivity using JDBC:

**Java Database Connectivity (JDBC)** is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database and is oriented towards relational databases.

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

import javax.lang.model.util.ElementScanner14;

import javax.swing.\*;

import java.sql.\*;

import java.util.Properties;

import java.awt.event.\*;

import java.awt.\*;

public class mainframe extends JFrame implements ActionListener

{

//private JFrame mainframe;

private JScrollPane scrollPane;

private JPanel mainPanel;

private JRadioButton ins;

private JRadioButton upd;

private JRadioButton del;

private JRadioButton view;

private JTextField[] jtf;

private JComboBox<String> tables;

private JTextArea textArea;

private JPanel updatPanel;

private JPanel insertPanel;

private JPanel deletPanel;

private JPanel selectPanel;

private JScrollPane scrol;

private int i,size;

private String tablename;

public mainframe()

{

setTitle("Twitter Database");

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(null);

mainPanel = new JPanel();

mainPanel.setBounds(0,0,600,400);

mainPanel.setLayout(null);

add(mainPanel);

createMenuBar();

selectPanel = new JPanel();

selectPanel.setBounds(0,0,600,400);

selectPanel.setLayout(null);

// System.out.println("hello1");

JLabel msgop = new JLabel("Select the Table which you want ");

msgop.setBounds(50, 50, 300, 25);

tables = new JComboBox<String>();

tables.setBounds(50, 80, 200, 25);

tables.addItem("users");

tables.addItem("authentication");

tables.addItem("tweet");

tables.addItem("retweets");

JLabel oprop = new JLabel("Select the operation you want to perform ");

oprop.setBounds(50, 120, 300, 25);

ins = new JRadioButton("INSERT", true);

ins.setBounds(50, 150, 100, 25);

upd = new JRadioButton("UPDATE", false);

upd.setBounds(150, 150, 100, 25);

del = new JRadioButton("DELETE", false);

del.setBounds(250, 150, 100, 25);

view = new JRadioButton("VIEW",false);

view.setBounds(350,150, 100, 25);

ButtonGroup bg = new ButtonGroup();

bg.add(ins);

bg.add(upd);

bg.add(del);

bg.add(view);

selectPanel.add(msgop);

selectPanel.add(tables);

selectPanel.add(oprop);

selectPanel.add(ins);

selectPanel.add(upd);

selectPanel.add(del);

selectPanel.add(view);

JButton submit = new JButton("SUBMIT");

submit.setBounds(50, 200, 100, 25);

submit.addActionListener(this);

selectPanel.add(submit);

insertPanel = new JPanel();

insertPanel.setBounds(0,0,700,400);

insertPanel.setLayout(null);

insertPanel.setVisible(false);

deletPanel = new JPanel();

deletPanel.setBounds(0,0,700,400);

deletPanel.setLayout(null);

deletPanel.setVisible(false);

updatPanel = new JPanel();

updatPanel.setBounds(0,0,700,400);

updatPanel.setLayout(null);

updatPanel.setVisible(false);

textArea = new JTextArea();

scrollPane = new JScrollPane(textArea);

scrollPane.setBounds(50, 250, 500, 100);

mainPanel.add(scrollPane);

mainPanel.add(selectPanel);

mainPanel.add(insertPanel);

mainPanel.add(deletPanel);

mainPanel.add(updatPanel);

//selectPanel.setVisible(true);

//setBounds(100, 100, 1000, 1000);

//scrol = new JScrollPane(mainPanel);

//add(scrol);

setPreferredSize(new Dimension(700,500));

pack();

setVisible(true);

}

public void actionPerformed(ActionEvent ae)

{

tablename =(String)tables.getSelectedItem();

if(ins.isSelected())

{

insertvalues(tablename);

}else if(del.isSelected())

{

delete(tablename);

}else if(upd.isSelected())

{

update(tablename);

}else if(view.isSelected())

{

viewtable(tablename);

}

}

public void viewtable(String tablename)

{

textArea.setText("");

try{

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","twitterdb","root");

Statement stmt = con.createStatement();

String qry = "Select \* from "+tablename;

ResultSet rs = stmt.executeQuery(qry);

ResultSetMetaData rsm = rs.getMetaData();

String s = "";

for(int j = 1;j<= rsm.getColumnCount();j++)

{

s+= rsm.getColumnName(j);

s+= " ";

}

textArea.append(s+"\n");

s= "";

while(rs.next())

{

for(int j = 1;j<=rsm.getColumnCount();j++)

{

s+= rs.getString(rsm.getColumnName(j));

s+= " ";

}

textArea.append(s+"\n");

s = "";

}

}catch(Exception ex)

{

textArea.append("couldnt display the table");

}

}

private void insertvalues(String tablename)

{

deletPanel.removeAll();

deletPanel.revalidate();

deletPanel.repaint();

updatPanel.removeAll();

updatPanel.revalidate();

updatPanel.repaint();

insertPanel.removeAll();

insertPanel.revalidate();

insertPanel.repaint();

try{

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","twitterdb","root");

Statement stmt = con.createStatement();

String qry = "Select \* from "+tablename;

ResultSet rs = stmt.executeQuery(qry);

ResultSetMetaData rsm = rs.getMetaData();

jtf = new JTextField[rsm.getColumnCount()];

JLabel[] jl = new JLabel[rsm.getColumnCount()];

JLabel title = new JLabel(" INSERT ");

deletPanel.add(title);

size = rsm.getColumnCount();

int x = 50; // Initial x-coordinate

int y = 50; // Initial y-coordinate

int labelWidth = 200; // Width of the label

int textFieldWidth = 150; // Width of the text field

int height = 25; // Height of each component

int spacing = 30; // Vertical spacing between components

for(int i = 0;i<rsm.getColumnCount();i++)

{

jl[i] = new JLabel(rsm.getColumnName(i+1));

jtf[i] = new JTextField();

jl[i].setBounds(x,y,labelWidth,height);

jtf[i].setBounds(x + labelWidth + spacing, y, textFieldWidth, height);

insertPanel.add(jl[i]);

insertPanel.add(jtf[i]);

y += height+spacing;

}

JButton sub = new JButton("SUBMIT", null);

sub.addActionListener(new ActionListener()

{

@Override

public void actionPerformed(ActionEvent e)

{

try{

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","twitterdb","root");

Statement stmt = con.createStatement();

String qry = "insert into "+tablename+" values(";

//System.out.println(qry);

if(size == 2)

{

qry += jtf[0].getText()+",";

qry += "'"+jtf[1].getText()+"'";

}else if(size == 4)

{

qry += "'"+jtf[0].getText()+"'";

qry += "'"+jtf[1].getText()+"'";

qry += "'"+jtf[2].getText()+"'";

qry += jtf[3].getText()+",";

}

else if(size == 7)

{

qry += "'"+jtf[0].getText()+"'";

qry += "'"+jtf[1].getText()+"'";

qry += "'"+jtf[2].getText()+"'";

qry += "'"+jtf[3].getText()+"'";

qry += "'"+jtf[4].getText()+"'";

qry += jtf[5].getText()+",";

qry += jtf[6].getText()+",";

}

else if(size == 5)

{

qry += jtf[0].getText()+",";

qry += "'"+jtf[1].getText()+"'";

qry += "'"+jtf[2].getText()+"'";

qry += jtf[3].getText()+",";

qry += jtf[4].getText()+",";

}

qry+=")";

System.out.println(qry);

stmt.executeQuery(qry);

textArea.setText("");

textArea.append("a row is inserted into "+tablename);

}

catch(Exception ex)

{

textArea.setText("");

textArea.append("couldnt perform insert");

}

}

}

);

sub.setBounds(x, y, textFieldWidth, height);

insertPanel.add(sub);

//f.setLayout(new GridLayout(size+1, 2, 3, 1));

insertPanel.setSize(700,500);

//System.out.println("hello");

deletPanel.setVisible(false);

selectPanel.setVisible(false);

insertPanel.setVisible(true);

}catch(Exception ex)

{

System.out.println(ex);

}

}

private void update(String tablename)

{

deletPanel.removeAll();

deletPanel.revalidate();

deletPanel.repaint();

insertPanel.removeAll();

insertPanel.revalidate();

insertPanel.repaint();

updatPanel.removeAll();

updatPanel.revalidate();

updatPanel.repaint();

try{

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","twitterdb","root");

Statement stmt = con.createStatement();

String qry = "Select \* from "+tablename;

ResultSet rs = stmt.executeQuery(qry);

ResultSetMetaData rsm = rs.getMetaData();

jtf = new JTextField[rsm.getColumnCount()];

JLabel title = new JLabel(" DELETE ");

updatPanel.add(title);

JLabel[] jl = new JLabel[rsm.getColumnCount()];

size = rsm.getColumnCount();

int x = 50; // Initial x-coordinate

int y = 50; // Initial y-coordinate

int labelWidth = 200; // Width of the label

int textFieldWidth = 150; // Width of the text field

int height = 25; // Height of each component

int spacing = 30; // Vertical spacing between components

for(int i = 0;i<rsm.getColumnCount();i++)

{

jl[i] = new JLabel(rsm.getColumnName(i+1));

jtf[i] = new JTextField();

jl[i].setBounds(x,y,labelWidth,height);

jtf[i].setBounds(x + labelWidth + spacing, y, textFieldWidth, height);

updatPanel.add(jl[i]);

updatPanel.add(jtf[i]);

y += height+spacing;

}

JButton sub = new JButton("SUBMIT", null);

sub.addActionListener(new ActionListener()

{

@Override

public void actionPerformed(ActionEvent e)

{

try{

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","twitterdb","root");

Statement stmt = con.createStatement();

String qry = "update "+tablename+" set ";

if(size == 2)

{

if(jtf[0].getText().length() == 0)

{

System.out.println("Cannot append");

}else

{

qry+= rsm.getColumnName(2)+" = '"+jtf[1].getText()+"' where "+rsm.getColumnName(1)+" = "+jtf[0].getText();

}

}else if(size == 3)

{

if(jtf[0].getText().length() == 0 || jtf[2].getText().length() == 0)

{

System.out.println("Cannot append");

}else{

qry += rsm.getColumnName(2)+" = "+jtf[1].getText()+" where "+rsm.getColumnName(1)+" = "+jtf[0].getText()+" and "+ rsm.getColumnName(3)+" = '"+jtf[2].getText()+" ' ";

}

}

else if(size == 6)

{

if(jtf[0].getText().length() == 0)

{

System.out.println("Cannot append");

}else{

int flag = 0;

int[] arr = new int[5];

for(int j = 1;j<size;j++)

{

arr[j]= jtf[j].getText().length();

}

for(int j = 1;j<size;j++)

{

if(flag == 0)

{

qry += rsm.getColumnName(j+1) + " = "+ jtf[j].getText();

flag = 1;

}else{

qry += " , "+rsm.getColumnName(j+1) + " = "+ jtf[j].getText();

}

}

qry += "where"+rsm.getColumnName(1)+" = "+jtf[0].getText();

}

}

System.out.println(qry);

stmt.executeQuery(qry);

textArea.setText("");

textArea.append(tablename+" is updated ");

}

catch(Exception ex)

{

System.out.println(ex);

}

}

});

sub.setBounds(x, y, textFieldWidth, height);

updatPanel.add(sub);

//f.setLayout(new GridLayout(size+1, 2, 3, 1));

updatPanel.setSize(700,500);

//System.out.println("hello");

insertPanel.setVisible(false);

selectPanel.setVisible(false);

updatPanel.setVisible(true);

deletPanel.setVisible(false);

}catch(Exception e)

{

System.out.println(e);

}

}

private void delete(String tablename)

{

insertPanel.removeAll();

insertPanel.revalidate();

insertPanel.repaint();

updatPanel.removeAll();

updatPanel.revalidate();

updatPanel.repaint();

deletPanel.removeAll();

deletPanel.revalidate();

deletPanel.repaint();

try{

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","twitterdb","root");

Statement stmt = con.createStatement();

String qry = "Select \* from "+tablename;

ResultSet rs = stmt.executeQuery(qry);

ResultSetMetaData rsm = rs.getMetaData();

jtf = new JTextField[rsm.getColumnCount()];

JLabel title = new JLabel(" DELETE ");

deletPanel.add(title);

JLabel[] jl = new JLabel[rsm.getColumnCount()];

size = rsm.getColumnCount();

int x = 50; // Initial x-coordinate

int y = 50; // Initial y-coordinate

int labelWidth = 200; // Width of the label

int textFieldWidth = 150; // Width of the text field

int height = 25; // Height of each component

int spacing = 30; // Vertical spacing between components

for(int i = 0;i<rsm.getColumnCount();i++)

{

jl[i] = new JLabel(rsm.getColumnName(i+1));

jtf[i] = new JTextField();

jl[i].setBounds(x,y,labelWidth,height);

jtf[i].setBounds(x + labelWidth + spacing, y, textFieldWidth, height);

deletPanel.add(jl[i]);

deletPanel.add(jtf[i]);

y += height+spacing;

}

JButton sub = new JButton("SUBMIT", null);

sub.addActionListener(new ActionListener()

{

@Override

public void actionPerformed(ActionEvent e)

{

try{

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","twitterdb","root");

Statement stmt = con.createStatement();

String qry = "Delete from "+tablename+" where ";

if(size == 2)

{

if(jtf[1].getText().length() == 0)

{

qry += rsm.getColumnName(1)+" = ";

qry += jtf[0].getText();

}else if(jtf[0].getText().length() == 0)

{

qry += rsm.getColumnName(2)+" = ";

qry += "'"+jtf[1].getText()+"'";

}else if(jtf[0].getText().length() != 0 && jtf[1].getText().length() != 0)

{

qry += rsm.getColumnName(1)+" = ";

qry += jtf[0].getText()+" and ";

qry += rsm.getColumnName(2)+" = ";

qry += "'"+jtf[1].getText()+"'";

}else

{

System.out.println("no");

}

}else if(size == 3)

{

int[] arr = new int[3];

for(int j = 0;j<3;j++)

{

arr[j] = jtf[j].getText().length();

}

int flag = 0;

for(int j = 0;j<size;j++)

{

if(arr[j] != 0)

{

if(flag == 0)

{

if(j == size-1)

{

qry += rsm.getColumnName(j+1)+" = ";

qry += "'"+jtf[j].getText()+"'";

}else{

qry += rsm.getColumnName(j+1)+" = ";

qry += jtf[j].getText();

flag = 1;

}

}else

{

if(j == size-1)

{

qry += " and "+rsm.getColumnName(j+1)+" = ";

qry += "'"+jtf[j].getText()+"'";

}else{

qry += " and "+ rsm.getColumnName(j+1)+" = ";

qry += jtf[j].getText();

}

}

}

}

}

else if(size == 6)

{

for(int i = 0;i<size;i++)

{

int[] arr = new int[3];

for(int j = 0;j<3;j++)

{

arr[j] = jtf[j].getText().length();

}

int flag = 0;

for(int j = 0;j<size;j++)

{

if(arr[j] != 0)

{

if(flag == 0)

{

qry += rsm.getColumnName(j+1)+" = ";

qry += jtf[j].getText();

flag = 1;

}else

{

qry += " and "+ rsm.getColumnName(j+1)+" = ";

qry += jtf[j].getText();

}

}

}

}

}

System.out.println(qry);

stmt.executeQuery(qry);

textArea.setText("");

textArea.append("1 row deleted from "+tablename);

}

catch(Exception ex)

{

System.out.println(ex);

}

}

});

sub.setBounds(x, y, textFieldWidth, height);

deletPanel.add(sub);

//f.setLayout(new GridLayout(size+1, 2, 3, 1));

deletPanel.setSize(500,400);

//System.out.println("hello");

insertPanel.setVisible(false);

updatPanel.setVisible(false);

selectPanel.setVisible(false);

deletPanel.setVisible(true);

}catch(Exception e)

{

System.out.println(e);

}

}

private void createMenuBar()

{

JMenuBar menuBar = new JMenuBar();

JMenu updateTables = new JMenu("UPDATE");

JMenu insertTables = new JMenu("INSERT");

JMenu deleteTables = new JMenu("DELETE");

JMenu menuop = new JMenu("MENU");

JMenu view = new JMenu("VIEW");

JMenuItem[] updateItems = new JMenuItem[4];

JMenuItem[] insertItems = new JMenuItem[4];

JMenuItem[] deleteItems = new JMenuItem[4];

JMenuItem[] viewItems = new JMenuItem[4];

String[] tableNames = {"users", "authentication", "tweet", "retweets"};

JMenuItem menu = new JMenuItem("MENU");

JMenuItem exit = new JMenuItem("EXIT");

menu.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e){

updatPanel.setVisible(false);

deletPanel.setVisible(false);

insertPanel.setVisible(false);

selectPanel.setVisible(true);

}

}

);

exit.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e){

System.exit(0);

}

});

menuop.add(menu);

menuop.add(exit);

for (int i = 0; i < 4; i++) {

String tableName = tableNames[i];

updateItems[i] = new JMenuItem(tableName);

updateItems[i].addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

update(tableName);

}

});

updateTables.add(updateItems[i]);

insertItems[i] = new JMenuItem(tableName);

insertItems[i].addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

System.out.println(tableName);

insertvalues(tableName);

}

});

insertTables.add(insertItems[i]);

deleteItems[i] = new JMenuItem(tableName);

deleteItems[i].addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

delete(tableName);

}

});

deleteTables.add(deleteItems[i]);

viewItems[i] = new JMenuItem(tableName);

viewItems[i].addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

viewtable(tableName);

}

});

view.add(viewItems[i]);

}

menuBar.add(menuop);

menuBar.add(updateTables);

menuBar.add(insertTables);

menuBar.add(deleteTables);

menuBar.add(view);

setJMenuBar(menuBar);

}

public static void main(String[] args) {

new mainframe();

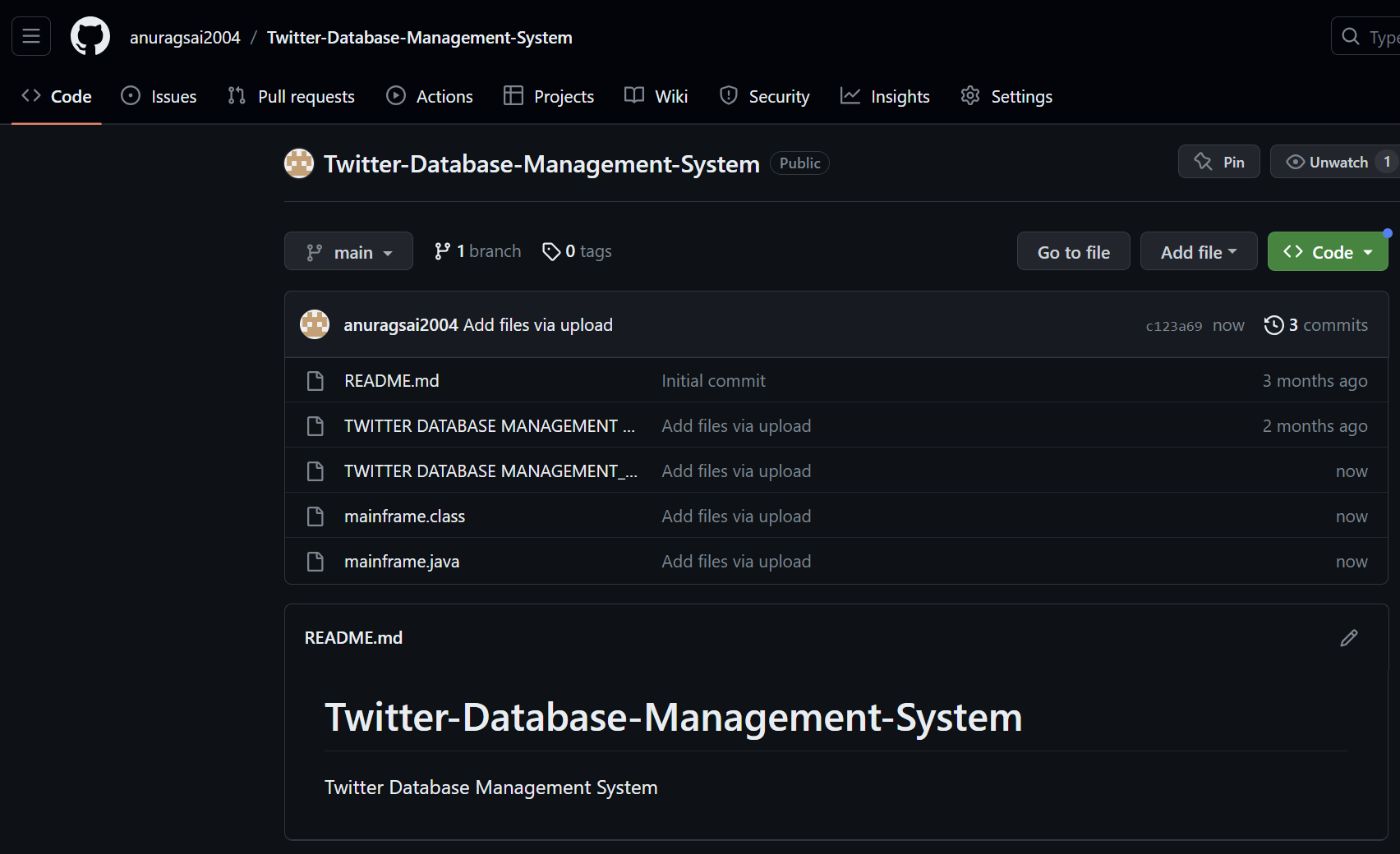
}

}

**GitHub Links and Folder Structure**

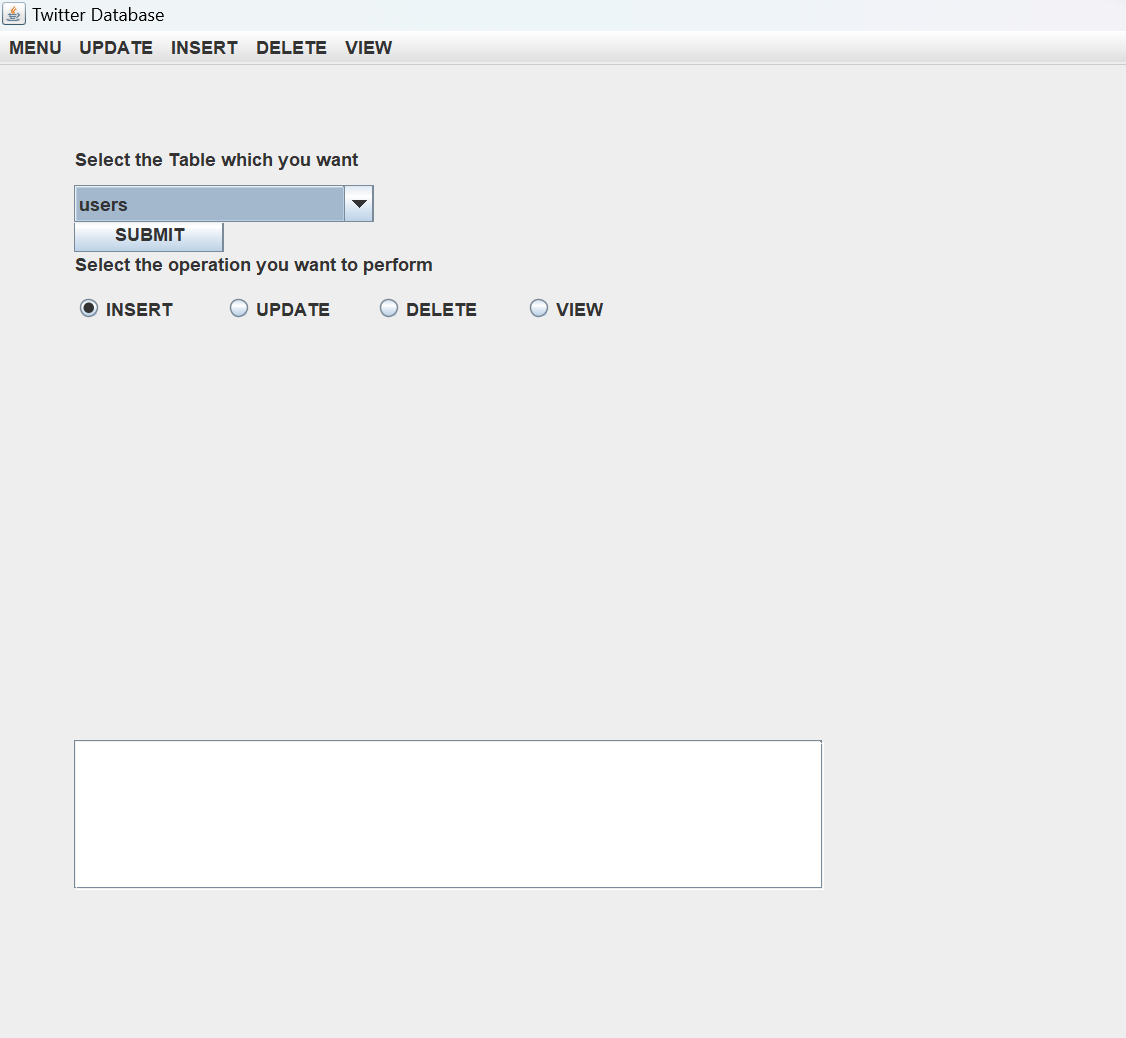
**Link:** [**https://github.com/anuragsai2004/Twitter-Database-Management-System.git**](https://github.com/anuragsai2004/Twitter-Database-Management-System.git)

**Folder Structure:**

****

# TESTING

**TWITTER DATBASE MANAGEMENT:**

****

**j**

**I.UPDATE PAGE:**

**A screenshot of a computer

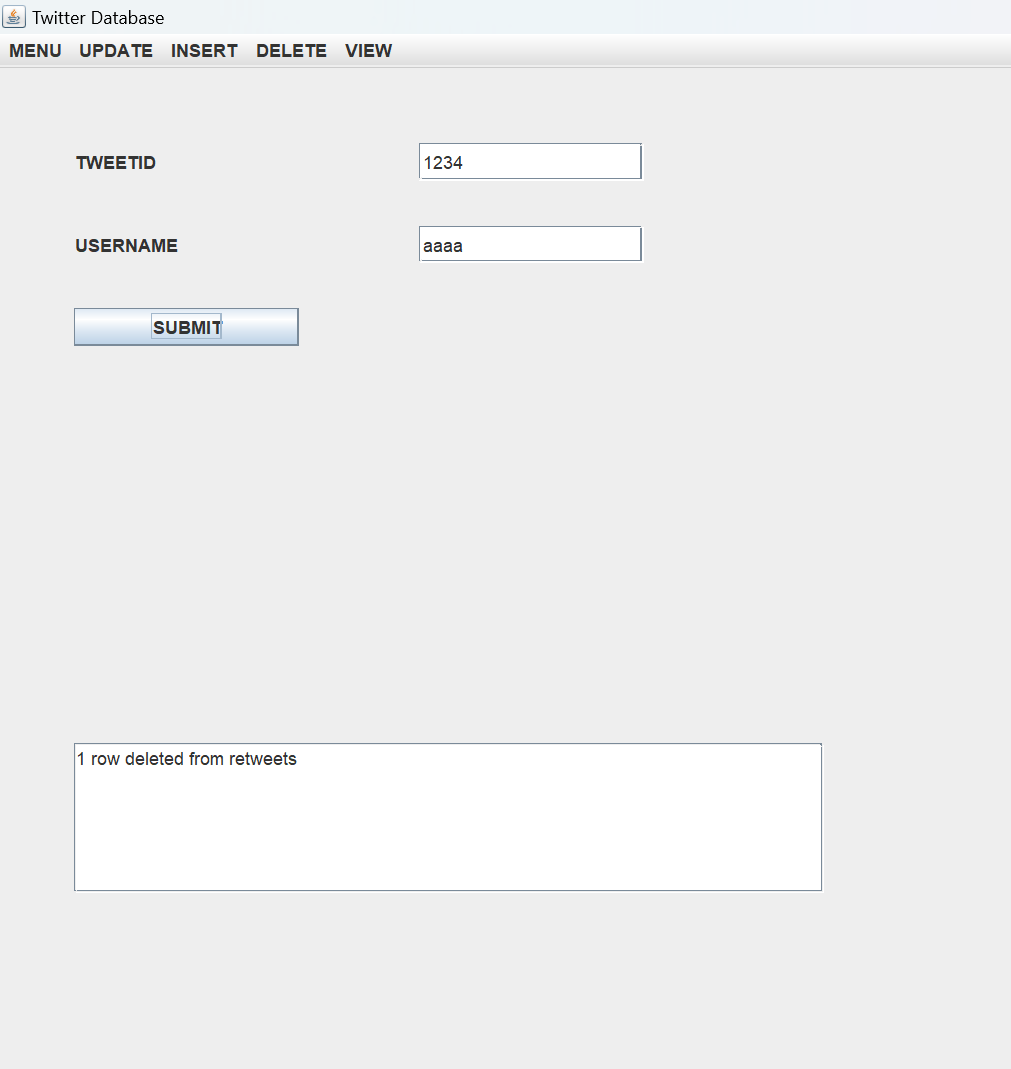
Description automatically generated**

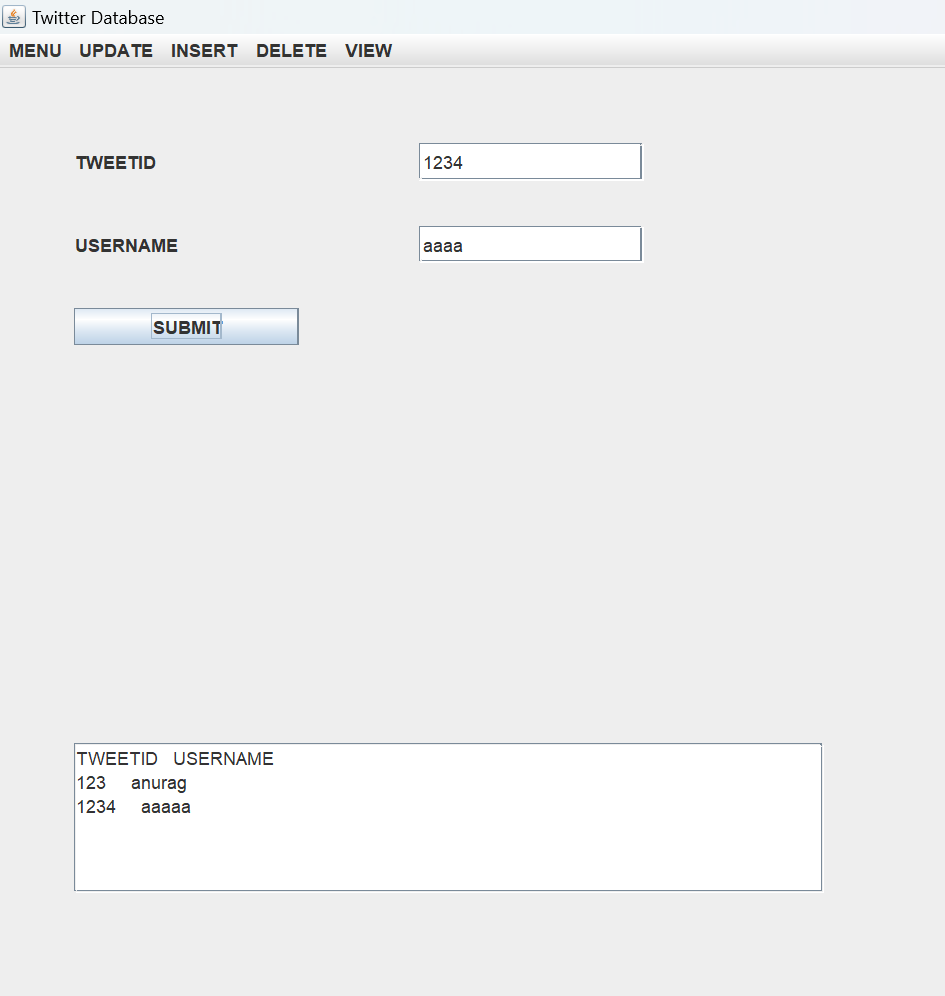
**II.INSERT PAGE**

A screenshot of a computer

Description automatically generated

1. **DELETE Page**



1. **VIEW PAGE**

**RESULTS**

I have successfully completed the mini-project ***“TWITTER DATABASE MANAGEMENT SYSTEM”*** .

## DISCUSSION AND FUTURE WORK

This project contains the basic interaction of giving information by students for suggesting the correct career choice. It has a very basic user interface.

Future scope would be to make the UI more appealing by using graphics. more feature would be to allow student-users to upload their resumes and official One documents required so that we can suggest more accurate career choices. We can also think of including a feedback system to allow the users to leave their valuable feedback after using this app. Making this feedback to be publicly viewable, would attract many more users to use this app.

## REFERENCES

* https://docs.oracle.com/javase/7/docs/api/
* [https://www.javatpoint.com/java-swing](http://www.javatpoint.com/java-swing)
* https://stackoverflow.com/