

A
PROJECT REPORT
ON
“SQL Automation”

Submitted By

Mr. Shubham Kailas Deshmukh

Guided By

Mr.Kawade S.N.

In partial fulfillment of the award of the degree

of

T.Y.B.Sc. (Computer Science)

of

**S.N. Arts, D.J.Malpani Commerce & B.N Sarda Science College
Sangamner,**

DURING ACADEMIC YEAR

2021-2022



SHIKSHANPRASARAKSANSTHA'S

**SANGAMNER NAGARPALIKA ARTS,
D.J. MALPANI COMMERCE &
225893**

**B.N. SARDA SCIENCE COLLEGE
SANGAMNER-422605 Dist –A.NAGAR**

Grade)

“Spread knowledge unto the last”

E-mail: – sangamnercollege@rediffmail.com

www.sangamnercollege.edu.in

Ph. (02425) 225893 (0)

Resi- 225164 (R)

Fax – (02425)

(Id No. PU/AN/ASC/03/1961)

•NAAC Re-accredited College (A+

• Pune University Best College

Award 2002-03

• DBT Star College, DST-FIST

Date: / /

CERTIFICATE

This certificate is awarded to **Mr. Shubham Kailas Deshmukh** of **T.Y.B.Sc. (Computer Science)** in appreciation of his meritorious performance as practical requirement of B.Sc. Computer Science by Savitribai Phule Pune University during academic Year 2021-2022.

Project Guide

Head of Department

Internal Examiner

External Examiner

INDEX

Sr.No.	Title
1.	Acknowledgement.
2.	Introduction.
3.	Objectives of System
4.	Preliminary Investigation
5.	Feasibility Study: 1)Operational Feasibility 2)Technical Feasibility 3)Economical Feasibility
6.	Requirement Analysis.
7.	Scope of Project
8.	Data Dictionary
9.	Entity Relationship Diagram
10.	Data flow Diagram
11.	UML Diagrams: 1)Class diagram 2)Object diagram 3)Use case diagram 4)Activity diagram
12.	Coding
13.	Input/output Screens
14.	Future Enhancement
15.	Advantages
16.	Conclusion
17.	Bibliography

ACKNOWLEDGEMENT

A few words of gratitude to be inserted with project “**SQL Automation**”. It is our earnest duty to express our thanks to all those who contributed directly or indirectly to our project.

Firstly, we would like to thank **S. N. Arts, D. J. Malpani Commerce & B. N. Sarda Science College, Sangamner and Department of Computer Science** for giving us an opportunity. Thanks to **Dr. Laddha R. S. Vice Principal, Head of Department** for his encouragement and valuable guidance.

We would like to thank **Mr. Kawade Sir, Mrs. Gite Madam, Mrs. Talnikar Madam and Mrs. Vikhe Madam** who initiated us to complete this project and guided us timely. It's our privilege to express our gratitude to **all staff members & non-Teaching Staff members** for their excellent suggestions and active co-ordination.

And last but not least, we would like to thank all **our friends** for their support and the timely help.

Yours Sincerely,

Mr. Shubham Kailas Deshmukh

T.Y.B.Sc.(Computer Science)

INTRODUCTION

- The given system is SQL Automation system.
- The System contains information about PostgreSQL that we can creating table, giving relationship to that tables, dropping tables, updating tables, viewing tables, and executing queries. The system having 3 major entities that are table relationship and queries.
- Table is a real-world entity and has attributes that are properties, relationship means how one table is related to other and queries means operations performed on tables that are view, update insert, delete etc.

OBJECTIVES

- The system provides proper security and reduces the manual work.
- The proposed system will help the user to reduce the workload and mental conflict.
- The proposed system helps the user to work user friendly and he can easily do his jobs.
- The application has login feature enabling data security.
- To facilitate easy retrieval of data and information.
- It enhances speed of work and accuracy.
- Reduction of paper work.
- To pass information more quickly.
- The main aim of this system is to create the tables in the database in short time period and help to student or benefiter to complete database work in short time.

PRELIMINARY INVESTIGATION

Preliminary Investigation implies that which methods are applied to make research of company & by which means information is collected regarding product or about us. Generally, I can use following methods for collecting the information.

These are: -

1) Interviews: -

1. we collect information from many professionals where they actually work on PostgreSQL database
2. we plan in advance type of questions that we are going to ask employees of Atos Syntel pvt ltd company and some students as well

In an interview we ask some questions like:

- 1) How they maintain their database?
- 2) which type of database they can prefer?
- 3) Which type of problems they are facing while maintaining database?

The whole system investigation part by taking interview by the concerned people, by taking valuable suggestion, guidance regarding system.

2) Observation: -

We observe some software's gives best way to maintain database.

Example: PHP MyAdmin, Pg Admin, etc.

FEASIBILITY STUDY

1) TECHNICAL FEASIBILITY STUDY: -

- This developed system has technically feasible because the technology needed for the proposed system is available. i.e., java software, java editor, PostgreSQL are easily available.
- The proposed system is integrated within the organization is easy.
- The proposed system needs no new equipment's installed at the user end.
- Technically this system is very much safe and sound.
- The system does not need extra hardware to run the given project.

2) ECONOMIC FEASIBILITY STUDY: -

- The proposed system is economically feasible because
- the cost of the project is not more at organization level. If the organization has to be identified the financial benefits and cost associated with the development of project is big difference then the project will more feasible at organization level.
- The benefits are high and cost is less. I.e., economically feasible.

3) OPERATIONAL FEASIBILITY STUDY: -

- As system is very much user friendly so user of the system can easily work and interact with the system. I.e., no need of expert person to handle this system.
- This system will fit with current operation if reconstruction is needed it is easy using technical person.

REQUIREMENT ANALYSIS

Technical Requirements

The minimum requirements are java, jdk.16 or above, PostgreSQL, java jdbc driver.

Development Environments

This desktop application can smoothly run on any operation system like Windows, Linux, etc.

Environments:

Windows
JDK 18
Visual Studio Code
JDBC driver
POSTGRESQL

SCOPE OF SYSTEM

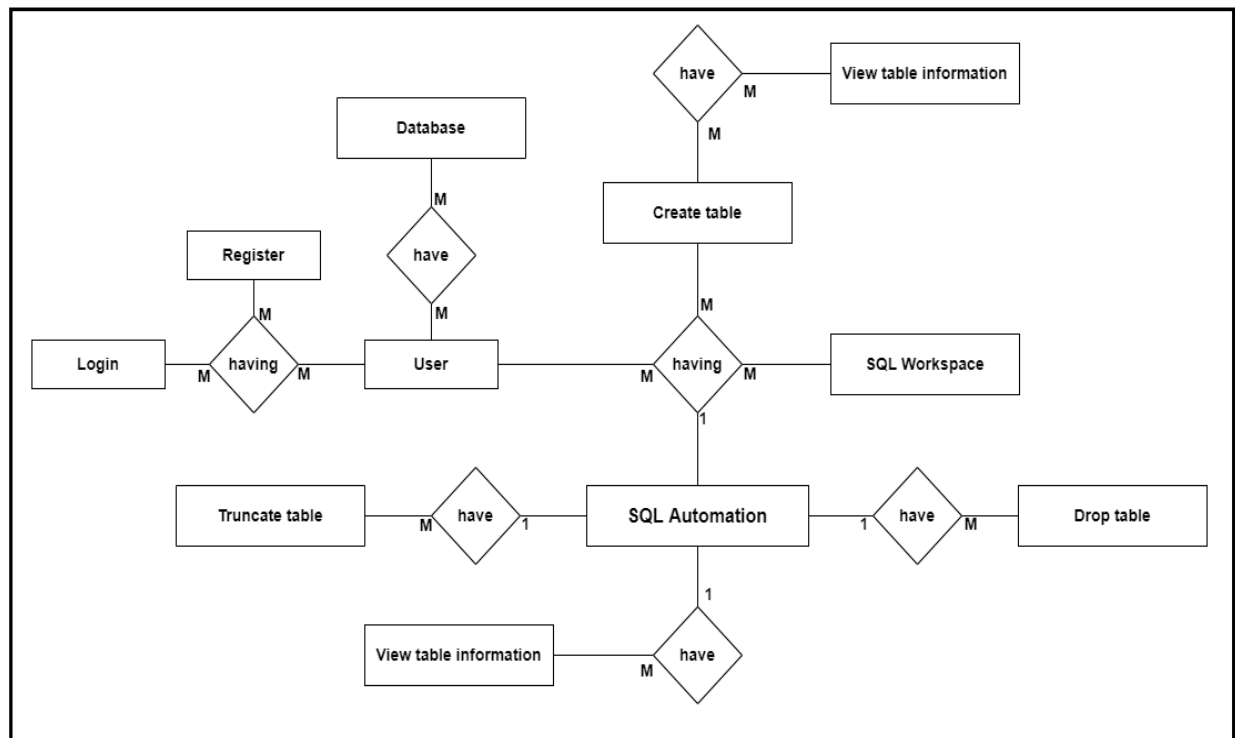
- We can use this for creating, modifying tables.
- Handling relationship with tables is too much easy.
- Executing queries with SQL workspace is much easy.
- This software run on windows platform easily.
- This software is easy to handle and understandable.
- This application is device independent with sample files.
- This application is very simple to understand.

Data Dictionary:

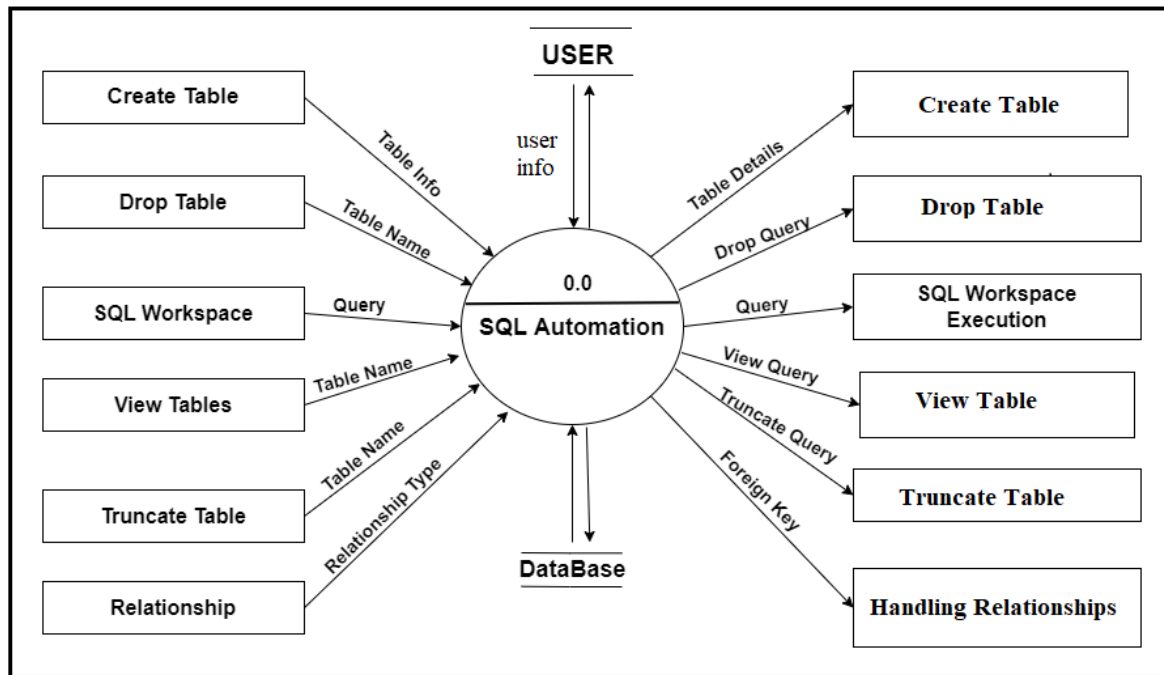
Login:

fieldname	datatype	size	constraint	description
Name	Char	50	Not null	Full name
email	varchar	50	Not null	Email id
Username	varchar	25	Pk	Username
Password	varchar	15	Not null	User password

Entity Relationship Diagram:

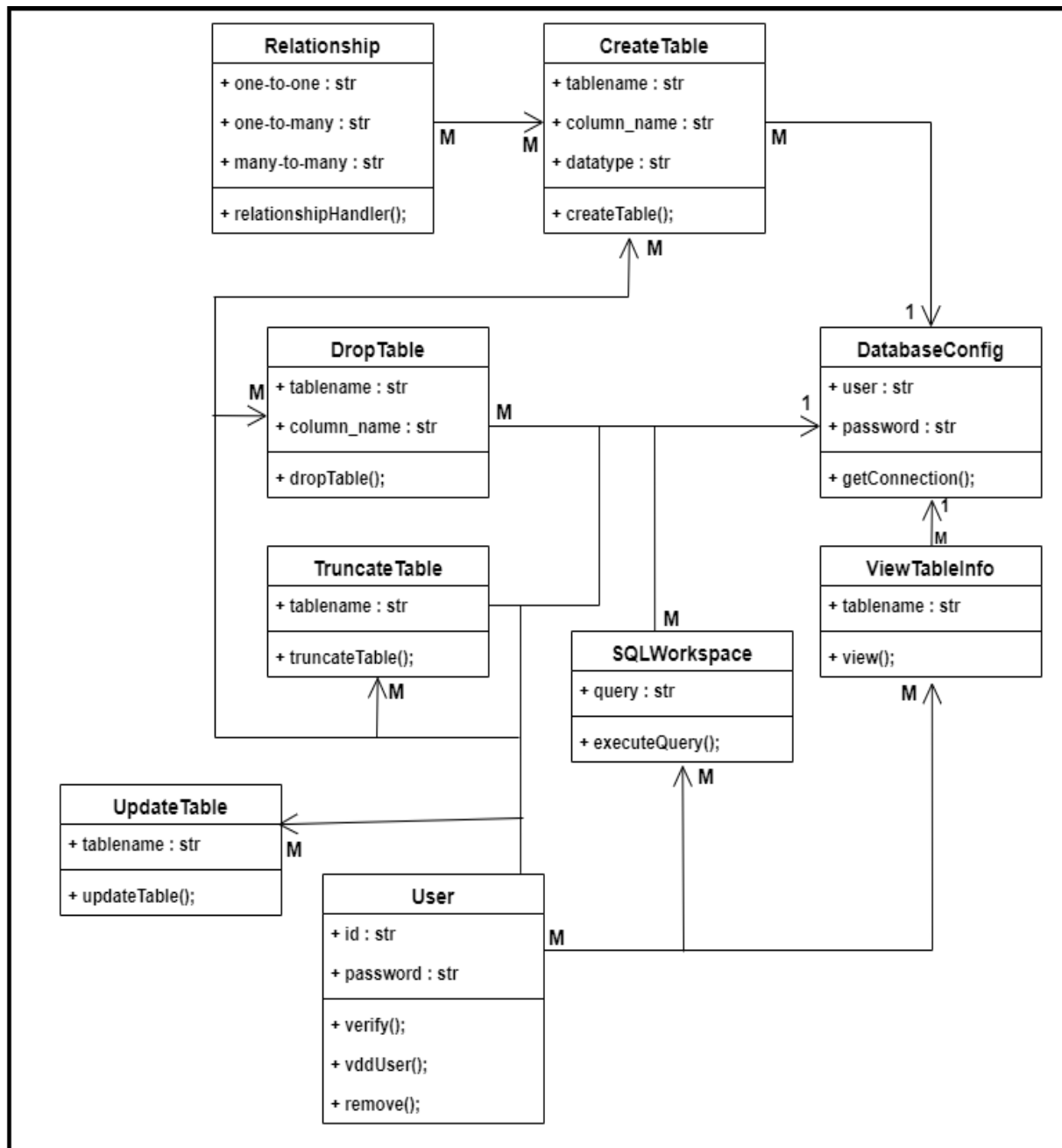


Data Flow Diagram:

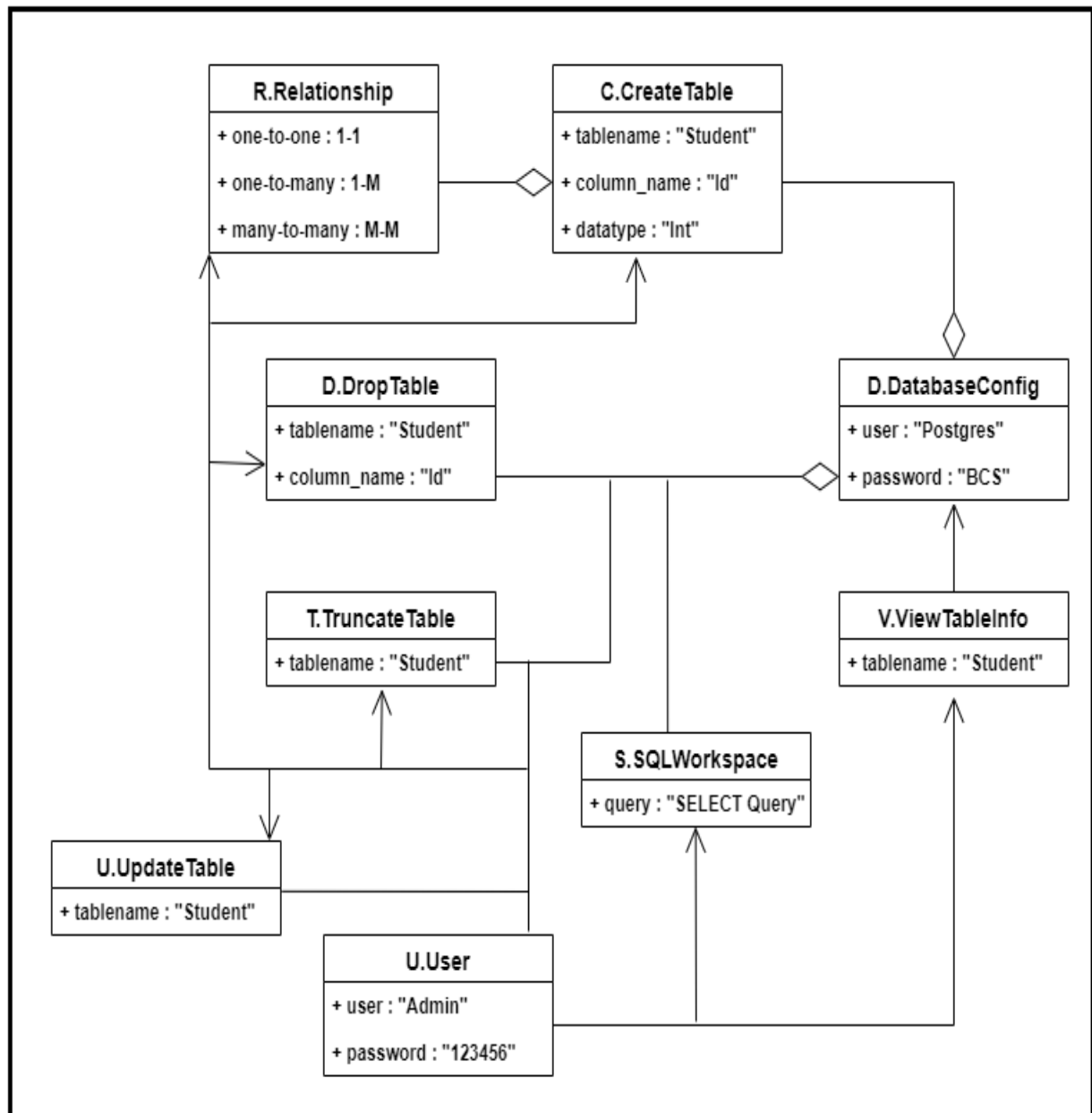


UML Diagrams:

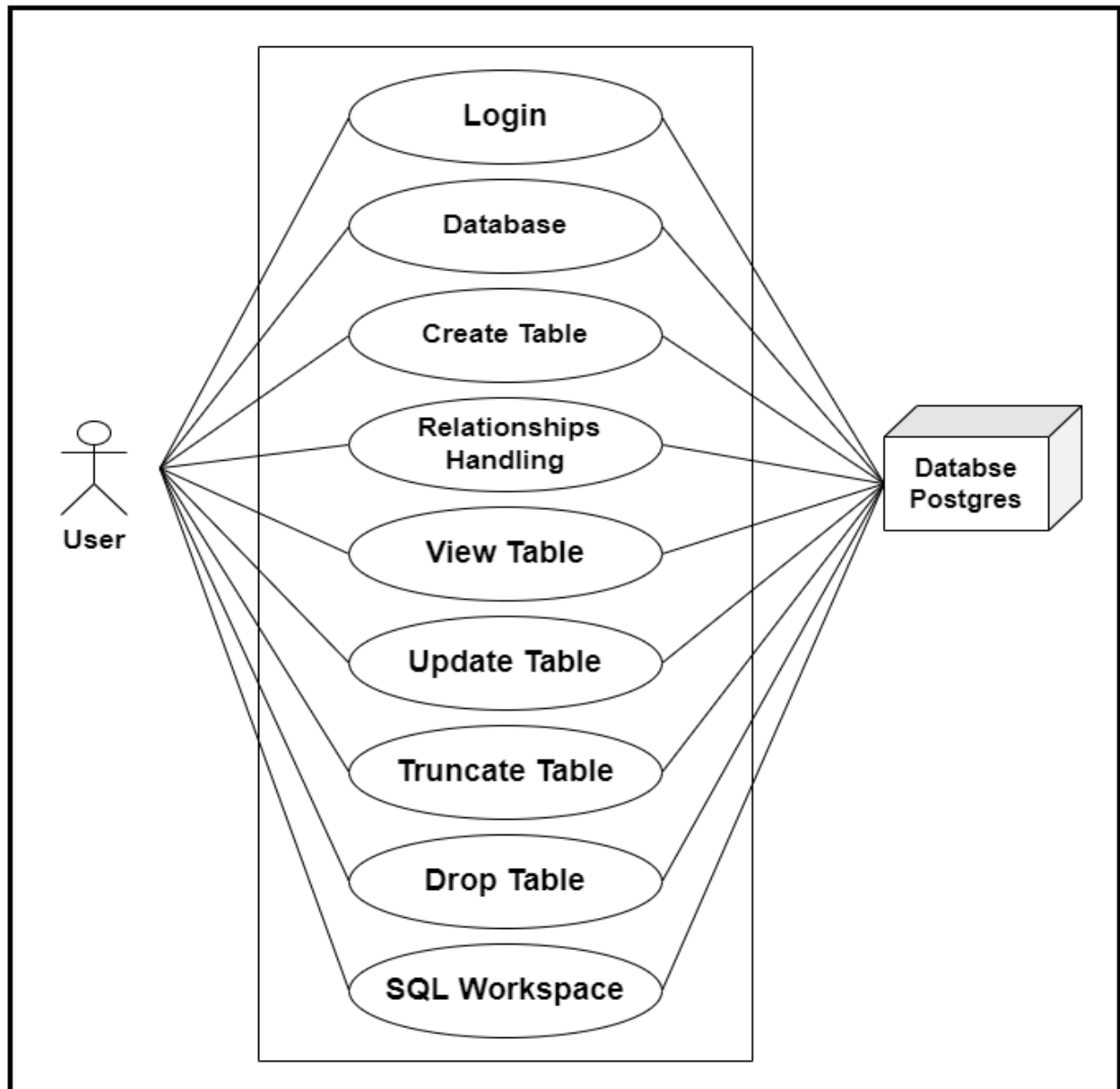
1. Class Diagram



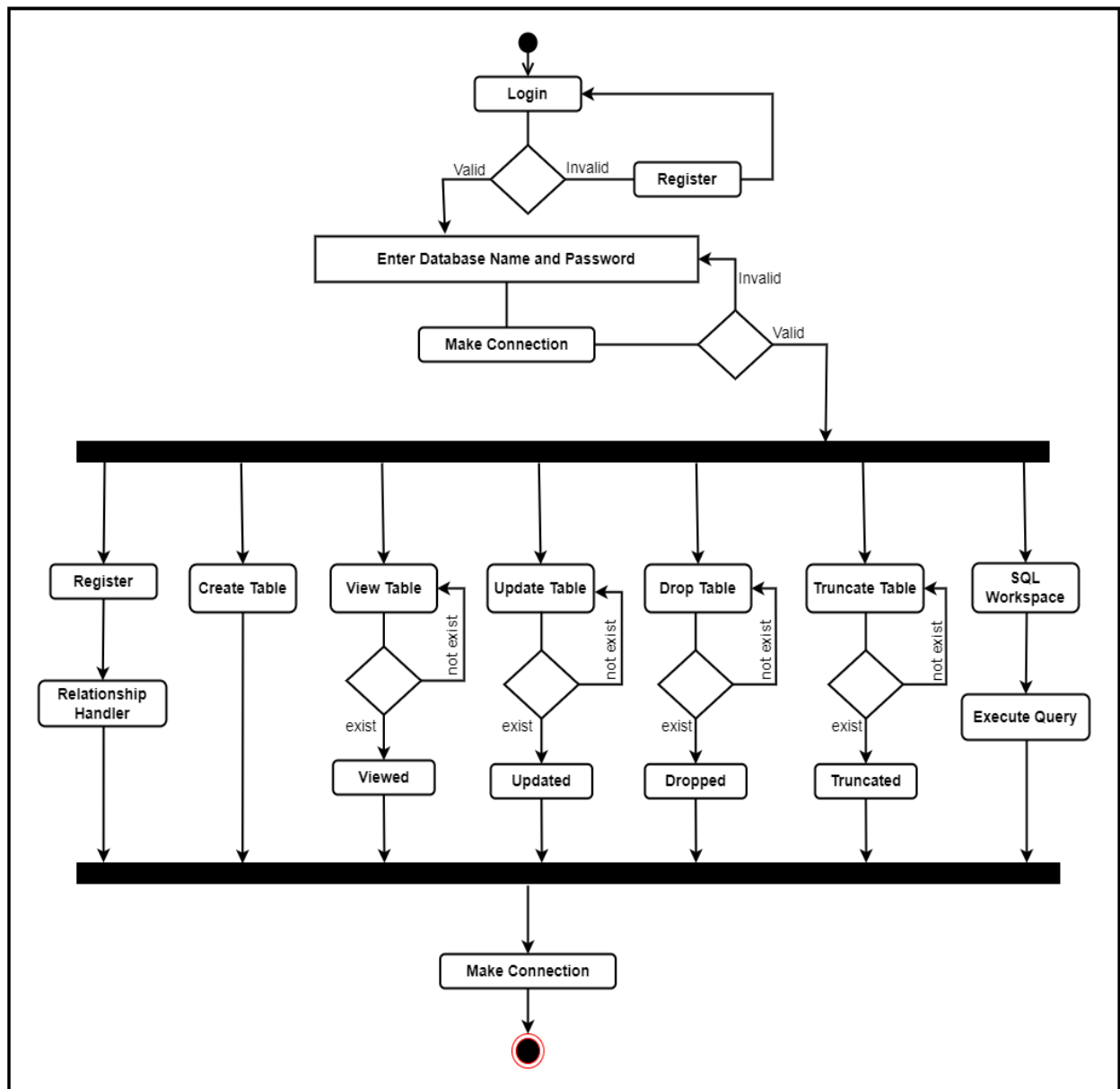
2. Object Diagram



3. Use Case Diagram



4. Activity Diagram



CODING

Stock list coding: -

Class Main:

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

```
import java.sql.*;
```

```
import javax.swing.table.*;
```

```
import java.awt.*;
```

```
import javax.swing.*;
```

```
import java.awt.Toolkit;
```

```
import java.awt.Dimension;
```

```
public class Main extends JFrame implements ActionListener {
```

```
    JButton createTable, viewTable, dropTable, UpdateTableData, Relationships,  
    Truncate, SQL, Menu, name, done;
```

```
    CreateTable cobj;
```

```
    RelationshipHandler robj;
```

```
    DatabaseConfig db;
```

```
    UpdateTable uobj;
```

```
    ViewTableInfo vobj;
```

```
    login lobj;
```

```
    DropTableData dobj;
```

```
    SQL_Executer seobj;
```

```
    public void callLogin() {
```

```
        lobj = new login();
```

```
}
```

```
void createMain() {
```

```
    setTitle("SQL Automation");
```

```
    setSize(1100, 700);
```

```
    setVisible(true);
```

```
    setLayout(null);
```

```
    setResizable(false);
```

```
    getContentPane().setBackground(Color.darkGray);
```

```
    // For Placing Frame in the Middle
```

```
    Toolkit toolkit = getToolkit();
```

```
    Dimension size = toolkit.getScreenSize();
```

```
    setLocation(size.width / 2 - getWidth() / 2, size.height / 2 - getHeight() / 2);
```

```
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
    JLabel head = new JLabel("Let's Make It Easy !");
```

```
    add(head);
```

```
    head.setBounds(610, 80, 460, 60);
```

```
    head.setForeground(Color.WHITE);
```

```
    head.setFont(new Font("serif", Font.BOLD, 25));
```

```
    name = new JButton("SQL AUTOMATION");
```

```
    add(name);
```

```
    name.setBounds(410, 20, 650, 60);
```

```
name.setBackground(Color.BLACK);
name.setForeground(Color.WHITE);
name.setFont(new Font("serif", Font.BOLD, 25));

Menu = new JButton("Menu");
add(Menu);

Menu.setBounds(10, 20, 400, 60);
Menu.setBackground(Color.BLACK);
Menu.setForeground(Color.WHITE);
Menu.setFont(new Font("serif", Font.BOLD, 25));

ImageIcon i1 = new ImageIcon(ClassLoader.getResource("plogo.jpg"));
Image i2 = i1.getImage().getScaledInstance(460, 360,
Image.SCALE_DEFAULT);
ImageIcon i3 = new ImageIcon(i2);
JLabel l3 = new JLabel(i3);
l3.setBounds(530, 140, 420, 420);
add(l3);

SQL = new JButton("SQL WorkSpace");
add(SQL);

SQL.setBounds(10, 80, 400, 60);
SQL.setBackground(Color.BLACK);
SQL.setForeground(Color.WHITE);
SQL.setFont(new Font("serif", Font.BOLD, 20));
```

```
createTable = new JButton("Create Table");
add(createTable);

createTable.setBounds(10, 140, 400, 60);
createTable.setBackground(Color.BLACK);
createTable.setForeground(Color.WHITE);
createTable.setFont(new Font("serif", Font.BOLD, 20));


Relationships = new JButton("Relationships");
add(Relationships);

Relationships.setBounds(10, 200, 400, 60);
Relationships.setBackground(Color.BLACK);
Relationships.setForeground(Color.WHITE);
Relationships.setFont(new Font("serif", Font.BOLD, 20));


viewTable = new JButton("View Table ");
add(viewTable);

viewTable.setBounds(10, 260, 400, 60);
viewTable.setBackground(Color.BLACK);
viewTable.setForeground(Color.WHITE);
viewTable.setFont(new Font("serif", Font.BOLD, 20));


UpdateTableData = new JButton("Update Table ");
add(UpdateTableData);

UpdateTableData.setBounds(10, 320, 400, 60);
UpdateTableData.setBackground(Color.BLACK);
UpdateTableData.setForeground(Color.WHITE);
```

```
UpdateTableData.setFont(new Font("serif", Font.BOLD, 20));
```

```
dropTable = new JButton("Drop Table");
```

```
add(dropTable);
```

```
dropTable.setBounds(10, 380, 400, 60);
```

```
dropTable.setBackground(Color.BLACK);
```

```
dropTable.setForeground(Color.WHITE);
```

```
dropTable.setFont(new Font("serif", Font.BOLD, 20));
```

```
Truncate = new JButton("Truncate Table");
```

```
add(Truncate);
```

```
Truncate.setBounds(10, 440, 400, 60);
```

```
Truncate.setBackground(Color.BLACK);
```

```
Truncate.setForeground(Color.WHITE);
```

```
Truncate.setFont(new Font("serif", Font.BOLD, 20));
```

```
done = new JButton("Done");
```

```
add(done);
```

```
done.setBounds(10, 500, 400, 60);
```

```
done.setBackground(Color.BLACK);
```

```
done.setForeground(Color.WHITE);
```

```
done.setFont(new Font("serif", Font.BOLD, 20));
```

```
// closing frame
```

```
done.addActionListener(e -> {
```

```
    dispose();
```

```
});
```

```
// adding action listeners

SQL.addActionListener(this);

createTable.addActionListener(this);

Relationships.addActionListener(this);

viewTable.addActionListener(this);

UpdateTableData.addActionListener(this);

dropTable.addActionListener(this);

Truncate.addActionListener(this);
}

public void actionPerformed(ActionEvent ae) {

    if (ae.getSource() == createTable) {

        cobj = new CreateTable();

        cobj.createTableMethod();

    }

    if (ae.getSource() == SQL) {

        seobj = new SQL_Executer();

    }

    if (ae.getSource() == viewTable) {

        vobj = new ViewTableInfo();

    }

    if (ae.getSource() == dropTable) {
```

```
        dobj = new DropTableData();

    }

    if (ae.getSource() == Truncate) {
        try {
            db = new DatabaseConfig();

            String tname = JOptionPane.showInputDialog(null, "Enter Table Name");

            String truncateSQL = "truncate table " + tname;

            db.ps = db.con.prepareStatement(truncateSQL);

            db.ps.execute();

            JOptionPane.showMessageDialog(null, "Table Truncated Sucessfully",
            "Sucess",
                JOptionPane.WARNING_MESSAGE);

        } catch (Exception e) {
        }

    }

    if (ae.getSource() == UpdateTableData) {

        uobj = new UpdateTable();

    }
```

```

        if (ae.getSource() == Relationships) {
            robj = new RelationshipHandler();
        }

    }

    public static void main(String[] args) {
        Main m1 = new Main();
        m1.callLogin();
    }
}

```

Create Table Class:

```

class CreateTable extends JFrame implements ActionListener {
    JComboBox<Object> Datatype;
    JTextField fieldname, dsize;
    JRadioButton r1, r2;
    JButton add_field, view_all, search_table, truncate, done;
    static JTable table;
    String datatype[] = { "char", "int", "varchar", "bigint", "text", "numeric" };

    DatabaseConfig db;
    String tname, one_tname;
    ButtonGroup bg;
    int col_cnt = 1;
    int pk_cnt = 1, t_cnt = 1;
}

```



```
public void createTableMethod() {  
    setTitle("Table");  
    setSize(1100, 700);  
    setVisible(true);  
    setLayout(null);  
    setResizable(false);  
    // For Placing Frame in the Middle  
    Toolkit toolkit = getToolkit();  
    Dimension size = toolkit.getScreenSize();  
    setLocation(size.width / 2 - getWidth() / 2, size.height / 2 - getHeight() / 2);  
    getContentPane().setBackground(Color.gray);  
    setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);  
  
    tname = JOptionPane.showInputDialog(null, "Enter Table Name");  
  
    JLabel table = new JLabel("Create Table " + tname + "");  
    add(table);  
    table.setForeground(Color.BLACK);  
    table.setFont(new Font("serif", Font.BOLD, 35));  
    table.setBounds(300, 10, 400, 100);  
  
    JLabel name = new JLabel("FieldName");  
    add(name);  
    name.setBounds(150, 130, 200, 60);  
    name.setBackground(Color.BLACK);
```

```
name.setForeground(Color.BLACK);
name.setFont(new Font("serif", Font.BOLD, 30));

fieldname = new JTextField(10);
add(fieldname);
fieldname.setBounds(400, 130, 200, 60);
fieldname.setBackground(Color.BLACK);
fieldname.setForeground(Color.WHITE);
fieldname.setFont(new Font("serif", Font.BOLD, 25));
JLabel data = new JLabel("DataType ");
add(data);
data.setBounds(150, 230, 200, 60);
data.setBackground(Color.BLACK);
data.setForeground(Color.BLACK);
data.setFont(new Font("serif", Font.BOLD, 30));

Datatype = new JComboBox<Object>(datatype);
add(Datatype);
Datatype.setBounds(400, 230, 200, 60);
Datatype.setBackground(Color.BLACK);
Datatype.setForeground(Color.WHITE);
Datatype.setFont(new Font("serif", Font.BOLD, 25));

JLabel pkey = new JLabel("Primary Key");
add(pkey);
pkey.setBounds(150, 330, 200, 60);
```

```
pkey.setBackground(Color.BLACK);  
pkey.setForeground(Color.BLACK);  
pkey.setFont(new Font("serif", Font.BOLD, 30));
```

```
r1 = new JRadioButton("yes");  
add(r1);  
r1.setBounds(400, 330, 100, 60);  
r1.setActionCommand("yes");  
r1.setBackground(Color.BLACK);  
r1.setForeground(Color.WHITE);  
r1.setFont(new Font("serif", Font.BOLD, 30));
```

```
r2 = new JRadioButton("no");  
add(r2);  
r2.setBounds(510, 330, 100, 60);  
r2.setActionCommand("no");  
r2.setBackground(Color.BLACK);  
r2.setForeground(Color.WHITE);  
r2.setFont(new Font("serif", Font.BOLD, 30));
```

```
bg = new ButtonGroup();  
bg.add(r1);  
bg.add(r2);
```

```
JLabel size1 = new JLabel("Size");  
add(size1);
```

```
size1.setBounds(150, 430, 100, 60);  
size1.setBackground(Color.BLACK);  
size1.setForeground(Color.BLACK);  
size1.setFont(new Font("serif", Font.BOLD, 30));
```

```
dsize = new JTextField(10);  
add(dsize);  
dsize.setBounds(400, 430, 100, 60);  
dsize.setText("0");  
dsize.setBackground(Color.BLACK);  
dsize.setForeground(Color.WHITE);  
dsize.setFont(new Font("serif", Font.BOLD, 30));
```

```
add_field = new JButton("Add Column");  
add(add_field);  
add_field.setBounds(700, 450, 220, 60);  
add_field.setBackground(Color.BLACK);  
add_field.setForeground(Color.WHITE);  
add_field.setFont(new Font("serif", Font.BOLD, 30));
```

```
done = new JButton("Done");  
add(done);  
done.setBounds(730, 550, 170, 60);  
done.setBackground(Color.BLACK);  
done.setForeground(Color.WHITE);  
done.setFont(new Font("serif", Font.BOLD, 30));
```

```

// closing frame

done.addActionListener(e -> {

    dispose();

});

// action listeners add_field

add_field.addActionListener(this);

}

public void actionPerformed(ActionEvent ae) {

    db = new DatabaseConfig();

    if (ae.getSource() == add_field) {

        String cname = (fieldname.getText());

        String datatype = (String) (Datatype.getSelectedItem());

        String initSQL = "create table " + tname + "()";

        int sizeOfDatatype = Integer.parseInt(dsize.getText());

        String primary_key = bg.getSelection().getActionCommand();

        try {

            // creating only table structure(empty table)

            if (col_cnt == 1) {

                db.ps = db.con.prepareStatement(initSQL);

                db.ps.execute();

                col_cnt++;

                t_cnt++;

            }

```

```

// adding column of primary key only one per table

if (primary_key == "yes" && pk_cnt == 1) {

    // checking of integer datatypes since it does not required size

    if (datatype == "int" || datatype == "bigint") {

        String SQL = "alter table " + tname + " add " + cname + " " + datatype
            + ";";

        db.ps = db.con.prepareStatement(SQL);

        db.ps.execute();

        // to add primary key to above table

        String pk = "alter table " + tname + " add primary key(" + cname +
");";

        db.ps = db.con.prepareStatement(pk);

        db.ps.execute();

        pk_cnt++;

    } // if of int & bigint

    else {

        // for other datatypes since it required size

        String SQL = "alter table " + tname + " add " + cname + " " + datatype
+ "("

            + sizeOfDatatype + ")" + ";";

        db.ps = db.con.prepareStatement(SQL);

```

```

        db.ps.execute();

        String pk = "alter table " + tname + " add primary key(" + cname +
");";

        db.ps = db.con.prepareStatement(pk);
        db.ps.execute();
        pk_cnt++;

    }
} // if of primary key
else {
    // adding columns of non primary key
    // integer datatypes
    if (datatype == "int" || datatype == "bigint") {
        String SQL = "alter table " + tname + " add " + cname + " " + datatype
            + ";";
        db.ps = db.con.prepareStatement(SQL);
        db.ps.execute();
    } else { // non int datatypes
        String SQL = "alter table " + tname + " add " + cname + " " + datatype
+ "("
            + sizeOfDatatype + ");";
        db.ps = db.con.prepareStatement(SQL);
        db.ps.execute();
    }
}

```

```

    }

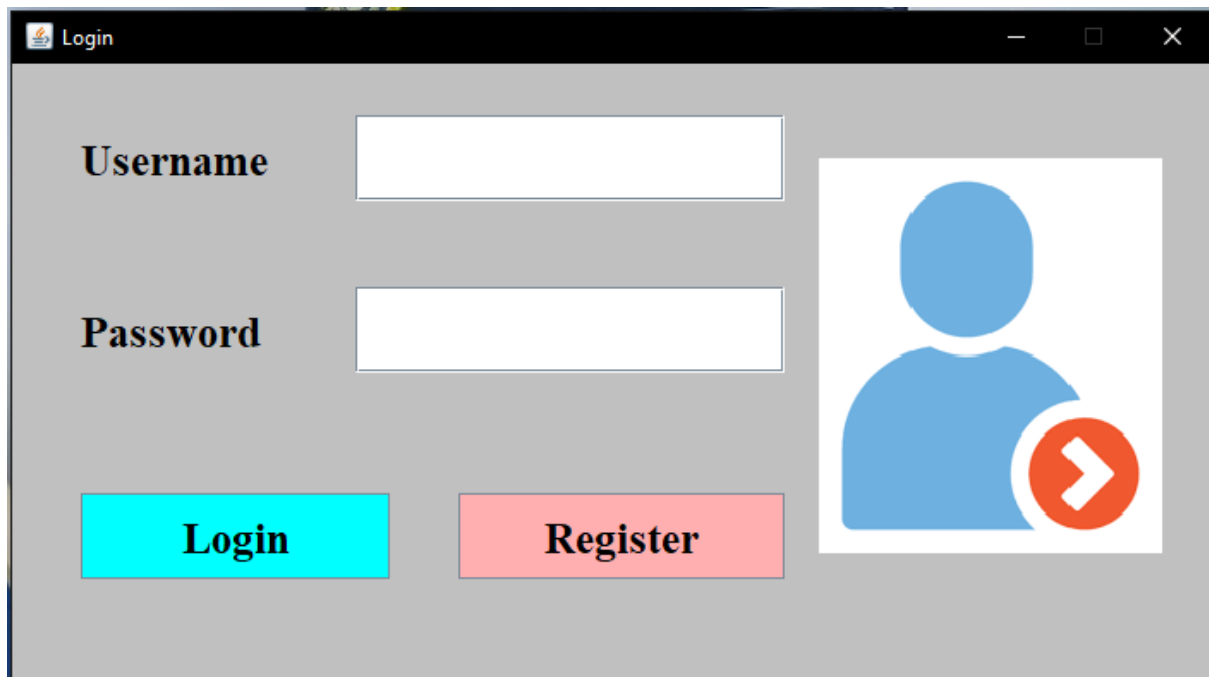
    JOptionPane.showMessageDialog(null, "Column Added Sucessfully",
    "Sucess",
        JOptionPane.INFORMATION_MESSAGE);

    fieldname.setText("");
    dsize.setText("0");
    db.con.close();
} catch (Exception e) {
    JOptionPane.showMessageDialog(null, e, "Error",
    JOptionPane.ERROR_MESSAGE);
    System.out.println(e);
}
} // add_field
} // actionPerformed
}

```

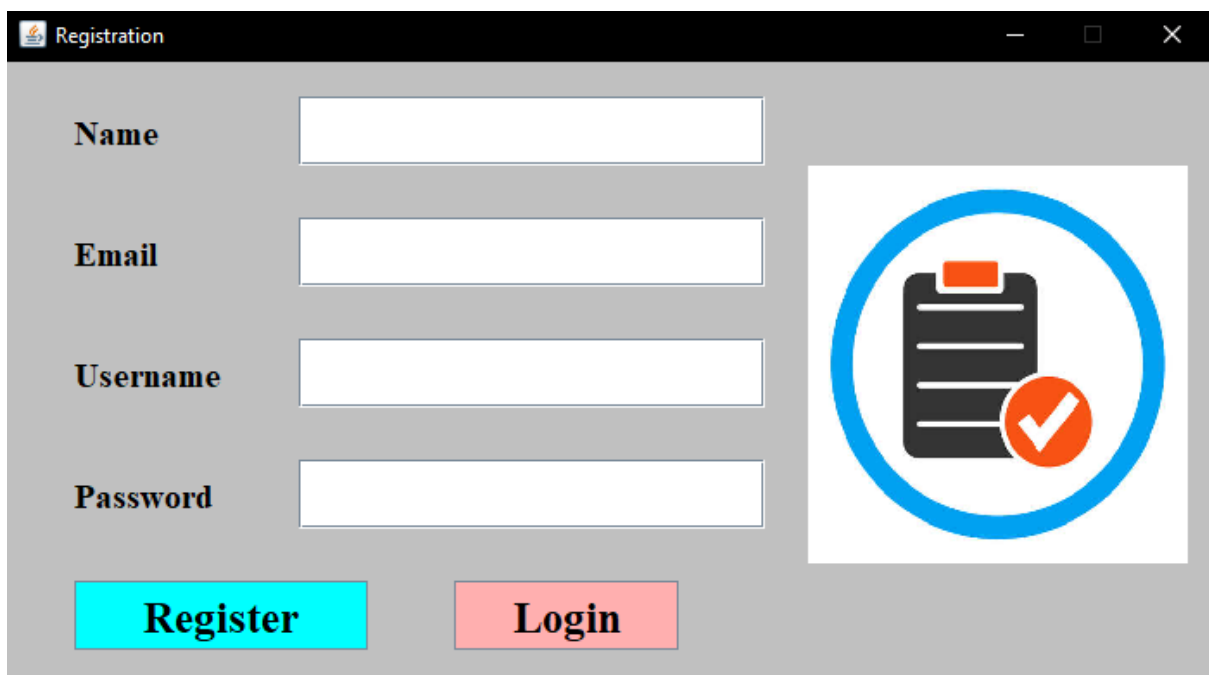

Input/Output screens:

Login Frame:



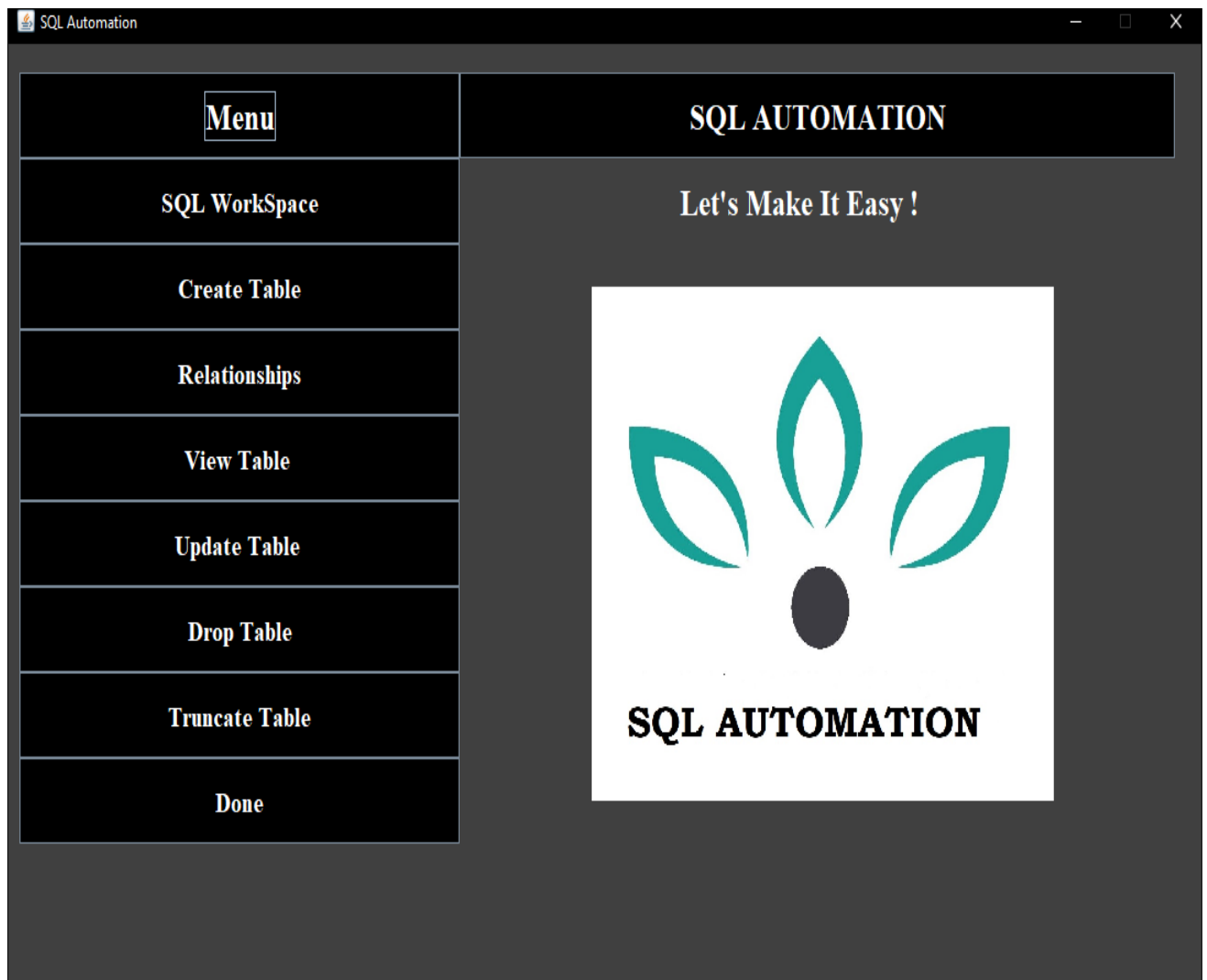
The Login Frame is a window titled "Login" with a standard macOS-style title bar. It features a light gray background. On the left, there are two text input fields: the first is labeled "Username" and the second is labeled "Password". Below these fields are two buttons: a cyan button labeled "Login" and a pink button labeled "Register". On the right side of the frame, there is a white square containing a blue silhouette of a person's head and shoulders, with a red circular icon containing a white right-pointing arrow overlaid on the bottom right corner of the square.

Registration Frame:



The Registration Frame is a window titled "Registration" with a standard macOS-style title bar. It features a light gray background. On the left, there are four text input fields: the first is labeled "Name", the second is labeled "Email", the third is labeled "Username", and the fourth is labeled "Password". Below these fields are two buttons: a cyan button labeled "Register" and a pink button labeled "Login". On the right side of the frame, there is a white square containing a blue circular icon with a white checkmark inside, overlaid on a black clipboard icon with a red checkmark.

Main Frame:



SQL Executor Frame:



The image shows a web browser window with the title "Sql Query Execution". The main content area has a light gray background. At the top, the text "Execute Queries" is displayed in a large, bold, black serif font. Below this, the text "Type SQL Query Here" is shown in a smaller, bold, black serif font. Underneath the text is a large, empty white rectangular box with a thin gray border, intended for entering a SQL query. At the bottom right of the gray area is a black rectangular button with the text "Go :)" in a white, bold, sans-serif font.

Sql Query Execution

Execute Queries

Type SQL Query Here

Go :)

Create Table Frame:

Table

Create Table demotable

FieldName

DataType

char

Primary Key

☒ yes

☐ no

Size

0

Add Column

Done

FUTURER ENHANCEMENT

1. The proposed system has some limitations but after some research and development system become much advance and simpler.
2. In the future we can create dynamic websites.
3. We try to provide database choosing and creating options.
4. Also, we are trying other advanced techniques.

ADVANTAGES

1. The system is user friendly.
2. Which is handle by those people who know some knowledge about computer and PostgreSQL.
3. Using this system manual work will be minimizing.
4. Creating tables & handling complex relationships become so much easy.
5. It is time saving technology.

CONCLUSION

In any organization where the database & its maintenance is most important for their perspective & doing database related work manually is quite difficult for anyone & it is very lengthy.

At that time application is very useful because you don't have to type any queries.

Also, where the professionals need to do this work very quickly at that time our system is impactful.

To avoid all these problems this project is very helpful. Our project helps them in reducing their precious time.

BIBLIOGRAPHY

On-line resources -

<https://docs.oracle.com>

<https://www.w3resource.com>

<https://www.w3schools.com/>

<https://www.tutorialspoint.com>

Books:

Java Complete References

Balguru Swami.

Black Book