JAVA AWT BASED- NETWORK CONNECTION MANAGEMENT SYSTEM - SQL CONNECTIVITY USING JDBC

A

Report

Submitted in partial fulfilment of the

Requirements for the award of the Degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

By

A.RUCHITA <1602-18-737-064>

Under the guidance of B.Leelavathy

Department of Information Technology

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2019

BONAFIDE CERTIFICATE

This is to certify that the project report titled "**VCE NETWORK CONNECTION MANAGEMENT SYSTEM**" project work of Miss.Ruchita Ananthula bearing Roll.no:1602-18-737-064 who carried out this project under my supervision in the IV Semester for the academic year 2019-2020.

Signature **Signature**

External examine Internal examine

ABSTRACT:-

The Network Connection Management System is a Web-based system that provides a single-stop for registration and maintenance of the network connection database of the campus network. The database contains both data (wired and wireless) and voice network connections in the Main Campus and the Student Residences. Network management system (NMS) is important both in ensuring the correct operation of network devices and in maintaining the services that run on them. This project has total of 11 tables .It describes how the network is being connected in our college across the various blocks. When you enter the data it is stored in the data base and is displayed as of when it is needed.

Roll no:1602-18-737-064
Name:A.Ruchita

AIM:

To create a **Java GUI based NETWORK CONNECTION MANAGEMENT SYSTEM** which takes the values like: computer ID, computer name , manufacturer , type , count, routers speed , username , website , block name , hod , server ipaddress , operating system name , version etc from the user. These values are to be updated in the database using **JDBC connectivity.**

INTRODUCTION

Requirements:

List of tables:

- Internet
- Computers
- Routers
- Block
- Server
- operating_system
- contains
- has
- are_having
- provides_network_to
- connected_to

List of attributes with their domain types:-

Internet:

Mac address : mac_address-varchar2(20)

HTML: html -varchar2(30)

Service provider: serv_provider- varchar2(20));

Computers:

Id of the computer: cid- varchar2(20)

Type of the computer: type - varchar2(20)

Count of computers: count -number

Manufacturer name: manufacturer - varchar2(20)

Type of model: model-varchar2(20)

Routers:

Website name :website -varchar2(50)

Speed of the router:speed- varchar2(10)

Model :model- varchar2 (20)

Username: username -varchar2(20)

Company :company- varchar2 (20)

Block:

Name of the block:bname-varchar2(20)

Name of the HOD:hod -varchar2(20)

Branch of the block:branch varchar2(20)

Server:

Ipaddress : ipaddress varchar2(20)

Operating_system:

Name of the operating system : osname- varchar2(20)

Version: version -varchar2(20)

Vendor: vendor-varchar2(20)

ARCHITECTURE AND TECHNOLOGY :

Software used:

Java Eclipse, Oracle 11g Database, Java SE version 7, SQL*Plus.

Java AWT:

Java AWT (Abstract Window Toolkit) 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 heavyweight i.e. its components are using the resources of OS.

The java.awt package provides classes for AWT API such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.

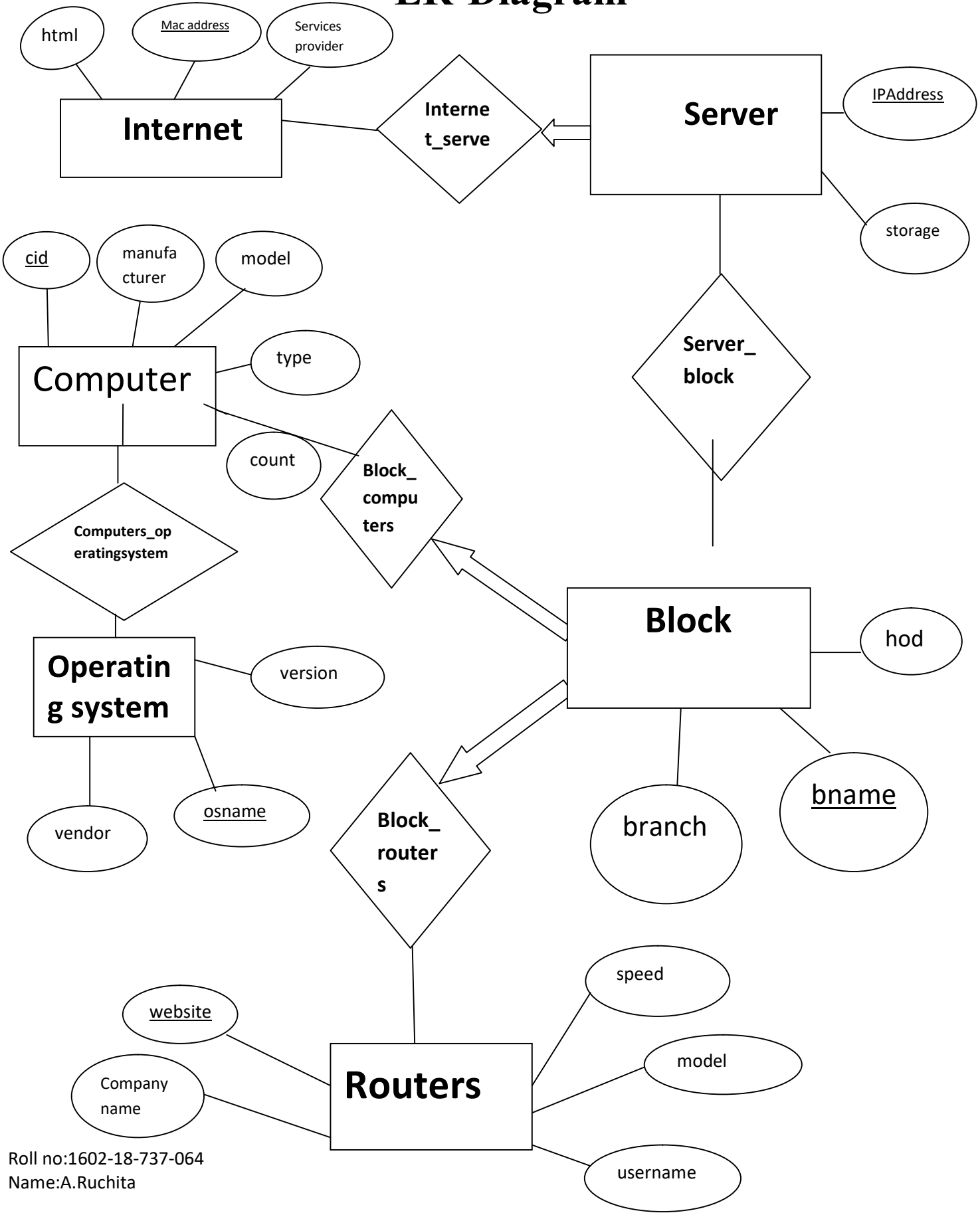
SQL:

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

DBMS MINIPROJECT
TITLE:VCE Network Connection Management System

ER-Diagram



Roll no:1602-18-737-064
Name:A.Ruchita

DDL COMMANDS

create table **internet**(

mac_address varchar2(20) primary key,html varchar2(30),serv_provider varchar2(20));

create table **computers**(

cid varchar2(20) primary key, type varchar2(20), count number, manufacturer varchar2(20),model varchar2(20)) ;

create table **routers** (

website varchar2(50) primary key ,speed varchar2(10),model varchar2 (20), username varchar2(20),company varchar2 (20)) ;

create table **Block**(

bname varchar2(20)primary key ,hod varchar2(20),branch varchar2(20));

create table **server** (ipaddress varchar2(20)primary key);

create table **operating_system**(

osname varchar2(20) , version varchar2(20), vendor varchar2(20), primary key(osname));

create table **Block_computer**(

cid varchar2(20), bname varchar2(20),foreign key(cid) references computers(cid), foreign key (bname) references block(bname),primary key(cid,bname)) ;

create table **Computers_operatingsystem**(

cid varchar2(20), osname varchar2(20),foreign key (cid) references computers, foreign key (osname) reference operating_system) ;

create table **Block_routers**(

website varchar2(50), bname varchar2(20), foreign key(website) references routers, foreign key(bname) references block);

create table **Server_block**(

bname varchar2(20),ipaddress varchar(20),foreign key(ipaddress) references server, foreign key(bname) references block);

create table **Internet_server**(
mac varchar2(20),ipadd varchar2(20),foreign key(mac) references internet, foreign key(ipadd) references server);

```
SQL> desc internet;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 HTML                                               VARCHAR2(30)
 SERV_PROVIDER                                      VARCHAR2(20)
 MAC_ADDRESS                               NOT NULL VARCHAR2(20)

SQL> desc server;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 IPADDRESS                                 NOT NULL VARCHAR2(20)

SQL> desc block;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 BNAME                                     NOT NULL VARCHAR2(20)
 HOD                                                VARCHAR2(20)
 BRANCH                                             VARCHAR2(20)

SQL> desc routers
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SPEED                                              VARCHAR2(10)
 MODEL                                              VARCHAR2(20)
 USERNAME                                           VARCHAR2(20)
 COMPANY                                            VARCHAR2(20)
 WEBSITE                                   NOT NULL VARCHAR2(50)

SQL> desc computers
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 TYPE                                               VARCHAR2(20)
 COUNT                                              NUMBER
 MANUFACTURER                                       VARCHAR2(20)
 MODEL                                              VARCHAR2(20)
 CID                                       NOT NULL VARCHAR2(20)
```

```
RA-04043: object operationg_system does not exist

QL> desc operating_system
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 OSNAME                                    NOT NULL VARCHAR2(20)
 VERSION                                            VARCHAR2(20)
 VENDOR                                             VARCHAR2(20)

QL> desc connected_to
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 MAC                                                VARCHAR2(20)
 IPADD                                              VARCHAR2(20)

QL> desc provides_network_to
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 BNAME                                              VARCHAR2(20)
 IPADDRESS                                          VARCHAR2(20)

QL> desc contains
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CID                                       NOT NULL VARCHAR2(20)
 BNAME                                     NOT NULL VARCHAR2(20)

QL> desc has
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 CID                                                VARCHAR2(20)
 OSNAME                                             VARCHAR2(20)

QL> desc are_having
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 WEBSITE                                            VARCHAR2(50)
 BNAME                                              VARCHAR2(20)
```

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:

```
private void connToDb(){
              try {
              Class.forName("oracle.jdbc.driver.OracleDriver");
              connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1522:xe","rachana","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.err.println("Unable to find and load driver");
                System.exit(1);
           }
       }
```

Thus, the connection from Java to Oracle database is performed and therefore, can be used for updating tables in the database directly.

Table Created in SQL for above mentioned purpose is as:

```
create table computers
(
cid varchar2(20) primary key ,
type varchar2(20) ,
count number ,
manufacturer varchar2(20) ,
model varchar2(20)
) ;
```

Program to insert computers:

```
package java_ass2;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class InsertComputers extends Frame
{
        Button insertComputersButton;
        TextField cidText, typeText, countText, manufacturerText;
        TextArea errorText;
        Connection connection;
        Statement statement;
        public InsertComputers()
        {
                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","ruchi","04032001");
                  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()
        {
                //Handle Insert Account Button
                insertComputersButton = new Button("Insert Computers");
                insertComputersButton.addActionListener(new ActionListener()
                {
```

```
                    public void actionPerformed(ActionEvent e)
                    {
                        try
                        {
                          String query= "INSERT INTO Computers VALUES('" + cidText.getText() + "', " +
"'" + typeText.getText() + "'," + countText.getText() + ",'" + manufacturerText.getText() + "')";
                          int i = statement.executeUpdate(query);
                          errorText.append("\nInserted " + i + " rows successfully");
                        }
                        catch (SQLException insertException)
                        {
                          displaySQLErrors(insertException);
                        }
                    }
                });
                cidText = new TextField(15);
                typeText = new TextField(15);
                countText = new TextField(15);
                manufacturerText = 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("Computer ID:"));
                first.add(cidText);
                first.add(new Label("Type:"));
                first.add(typeText);
                first.add(new Label("Count:"));
```

```
                first.add(countText);
                first.add(new Label("Manufacturer:"));
                first.add(manufacturerText);
                first.setBounds(125,90,200,100);

                Panel second = new Panel(new GridLayout(4, 1));
                second.add(insertComputersButton);
                second.setBounds(125,220,150,100);

                Panel third = new Panel();
                third.add(errorText);
                third.setBounds(125,320,300,200);

                setLayout(null);

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

                setTitle("New Computers Creation");
                setSize(500, 600);
                setVisible(true);
        }
        private void displaySQLErrors(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)
```

```
        {
                InsertComputers cc = new InsertComputers();
                cc.addWindowListener(new WindowAdapter(){
                  public void windowClosing(WindowEvent e)
                  {
                        System.exit(0);
                  }
                });
                cc.buildGUI();
        }
}
```

Program to update computers:

```
package java_ass2;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class ViewComputers extends Frame
{
        Button updateComputersButton;
        List ComputersList;
        TextField cidText, typeText, countText, manufacturerText;
        TextArea errorText;
        Connection connection;
        Statement statement;
        ResultSet rs;

        public ViewComputers()
```

```
        {
                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","ruchi","04032001
");
                  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 loadComputers()
        {
            try
            {
             rs = statement.executeQuery("SELECT * FROM Computers");
             while (rs.next())
             {
                    ComputersList.add(rs.getString("CID"));
             }
            }
            catch (SQLException e)
            {
             displaySQLErrors(e);
            }
        }
        public void buildGUI()
        {
            ComputersList = new List(6);
            loadComputers();
            add(ComputersList);

            //When a list item is selected populate the text fields
```

```
ComputersList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {
            rs = statement.executeQuery("SELECT * FROM
Computers");
            while (rs.next())
            {
                if
(rs.getString("CID").equals(ComputersList.getSelectedItem()))
                break;
            }
            if (!rs.isAfterLast())
            {
                cidText.setText(rs.getString("CID"));
                typeText.setText(rs.getString("TYPE"));
                countText.setText(rs.getString("COUNT"));

    manufacturerText.setText(rs.getString("MANUFACTURER"));
            }
        }
        catch (SQLException selectException)
        {
            displaySQLErrors(selectException);
        }
```

```
                }
            });
            //Handle Update Computers Button
            updateComputersButton = new Button("Update Computers");
            updateComputersButton.addActionListener(new ActionListener()
            {
                public void actionPerformed(ActionEvent e)
                {
                    try
                    {
                        Statement statement = connection.createStatement();
                        int i = statement.executeUpdate("UPDATE Computers "
                        + "SET COUNT=" + countText.getText()
                        + " WHERE cid = '" + ComputersList.getSelectedItem()
+ "'");
                        errorText.append("\nUpdated " + i + " rows
successfully");
                        ComputersList.removeAll();
                        loadComputers();
                    }
                    catch (SQLException insertException)
                    {
                        displaySQLErrors(insertException);
                    }
                }
            });
```

```
cidText = new TextField(15);
cidText.setEditable(false);
typeText = new TextField(15);
typeText.setEditable(false);
countText = new TextField(15);
manufacturerText = new TextField(15);
manufacturerText.setEditable(false);

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

Panel first = new Panel();
first.setLayout(new GridLayout(4, 2));
first.add(new Label("Computer ID:"));
first.add(cidText);
first.add(new Label("Type:"));
first.add(typeText);
first.add(new Label("Count:"));
first.add(countText);
first.add(new Label("Manufacturer:"));
first.add(manufacturerText);

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

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

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

        setTitle("Update Computers");
        setSize(500, 600);
        setLayout(new FlowLayout());
        setVisible(true);

    }
    private void displaySQLErrors(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)
    {
        ViewComputers upb = new ViewComputers();

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

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

                upb.buildGUI();
        }
}
```

Program to delete computers:

```
package java_ass2;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class DeleteComputers extends Frame
{
        Button DeleteComputersButton;
        List ComputersIDList;
        TextField cidText, typeText, countText, manufacturerText;
        TextArea errorText;
        Connection connection;
        Statement statement;
        ResultSet rs;

        public DeleteComputers()
```

```
        {
                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","ruchi","04032001
");
                  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 loadComputers()
        {
            try
            {
              rs = statement.executeQuery("SELECT * FROM computers");
              while (rs.next())
              {
                    ComputersIDList.add(rs.getString("CID"));
              }
            }
            catch (SQLException e)
            {
              displaySQLErrors(e);
            }
        }

        public void buildGUI()
        {
           ComputersIDList = new List(10);
```

```
                loadComputers();
                add(ComputersIDList);

                //When a list item is selected populate the text fields
                ComputersIDList.addItemListener(new ItemListener()
                {
                        public void itemStateChanged(ItemEvent e)
                        {
                                try
                                {
                                        rs = statement.executeQuery("SELECT * FROM
    computers");
                                        while (rs.next())
                                        {
                                                if
    (rs.getString("CID").equals(ComputersIDList.getSelectedItem()))
                                                break;
                                        }
                                        if (!rs.isAfterLast())
                                        {
                                                cidText.setText(rs.getString("CID"));
                                                typeText.setText(rs.getString("TYPE"));
                                                countText.setText(rs.getString("COUNT"));

            manufacturerText.setText(rs.getString("MANUFACTURER"));
                                        }
                                }
```

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

        //Handle Delete Computers Button
        DeleteComputersButton = new Button("Delete Computers");
        DeleteComputersButton.addActionListener(new ActionListener()
        {
                public void actionPerformed(ActionEvent e)
                {
                        try
                        {
                                Statement statement = connection.createStatement();
                                int i = statement.executeUpdate("DELETE FROM
computers WHERE CID = '" + ComputersIDList.getSelectedItem()+"'");
                                errorText.append("\nDeleted " + i + " rows
successfully");
                                cidText.setText(null);
                                typeText.setText(null);
                                countText.setText(null);
                                manufacturerText.setText(null);
                                ComputersIDList.removeAll();
                                loadComputers();
```

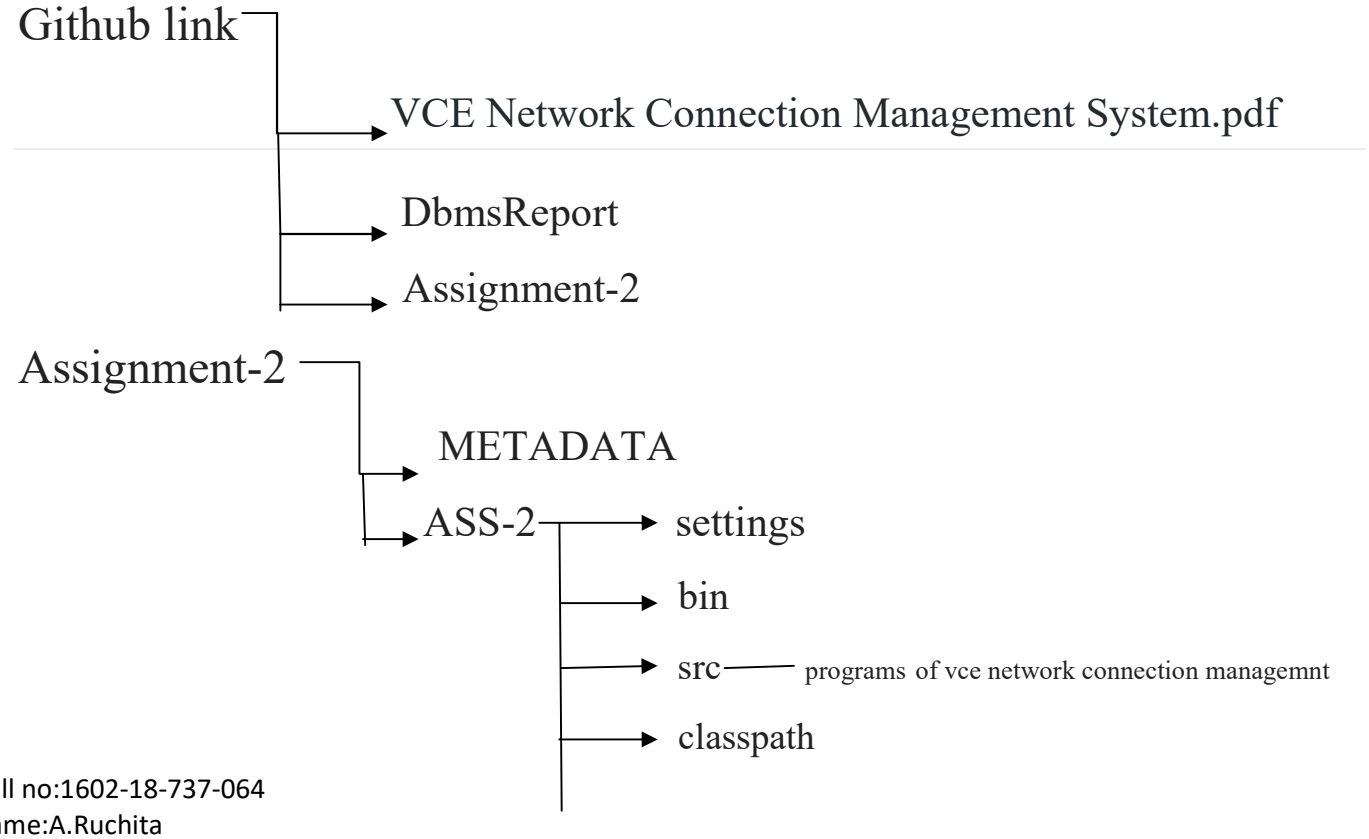
```
                }
                catch (SQLException deleteException)
                {
                        displaySQLErrors(deleteException);
                }
        }
});

cidText = new TextField(15);
typeText = new TextField(15);
countText = new TextField(15);
manufacturerText = 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("Computers ID:"));
first.add(cidText);
first.add(new Label("Type:"));
first.add(typeText);
first.add(new Label("Count:"));
first.add(countText);
first.add(new Label("Manufacturer:"));
```

```
        first.add(manufacturerText);

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

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

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

        setTitle("Remove computers");
        setSize(450, 600);
        setLayout(new FlowLayout());
        setVisible(true);

    }
    private void displaySQLErrors(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)
```

```
        {
                DeleteComputers dels = new DeleteComputers();
                dels.addWindowListener(new WindowAdapter(){
                 public void windowClosing(WindowEvent e)
                 {
                        System.exit(0);
                 }
                });
                dels.buildGUI();
        }
}
```

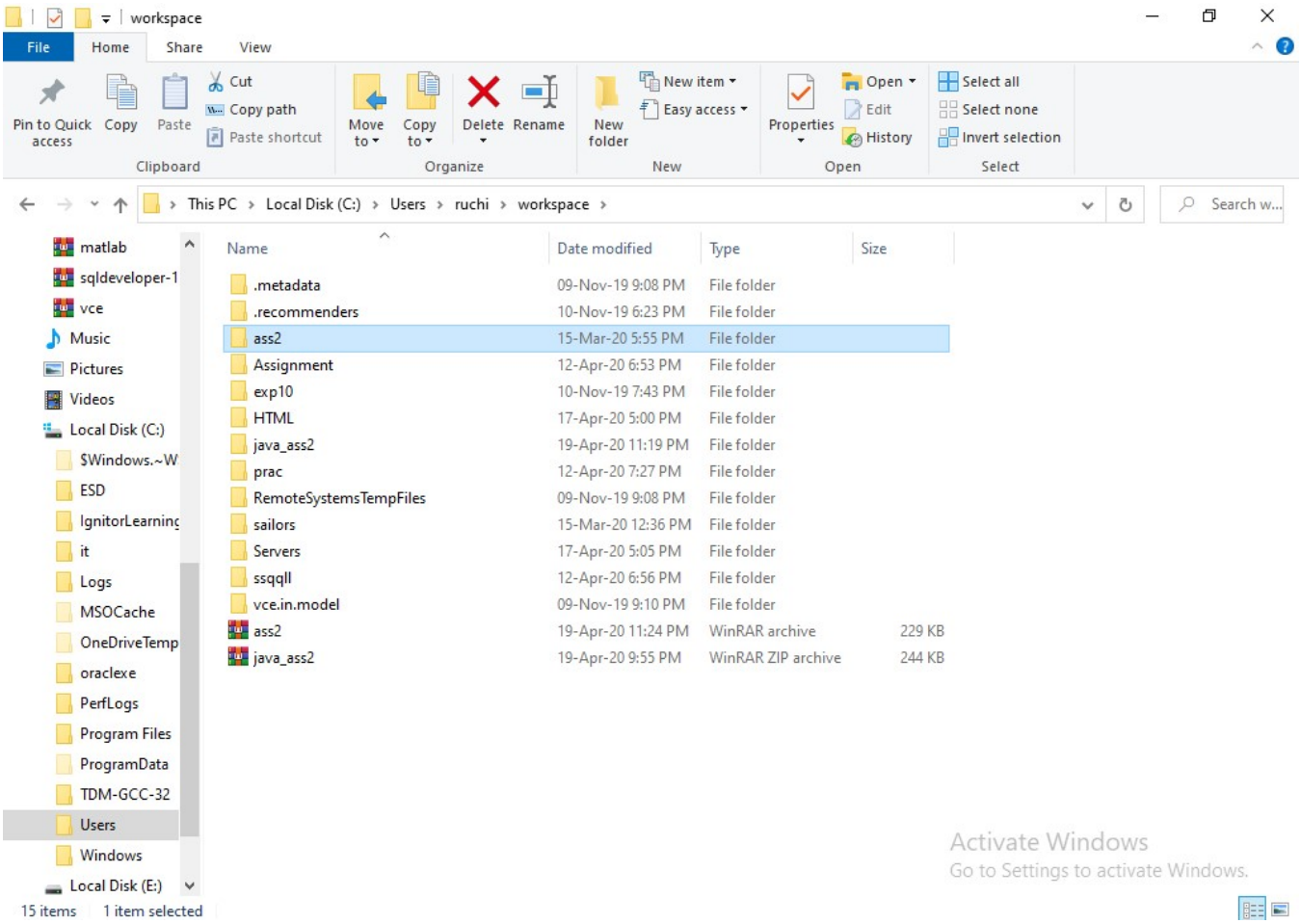
GITHUB Link:

https://github.com/ruchita0403-dot/VCE-Network-Connection-Management-System



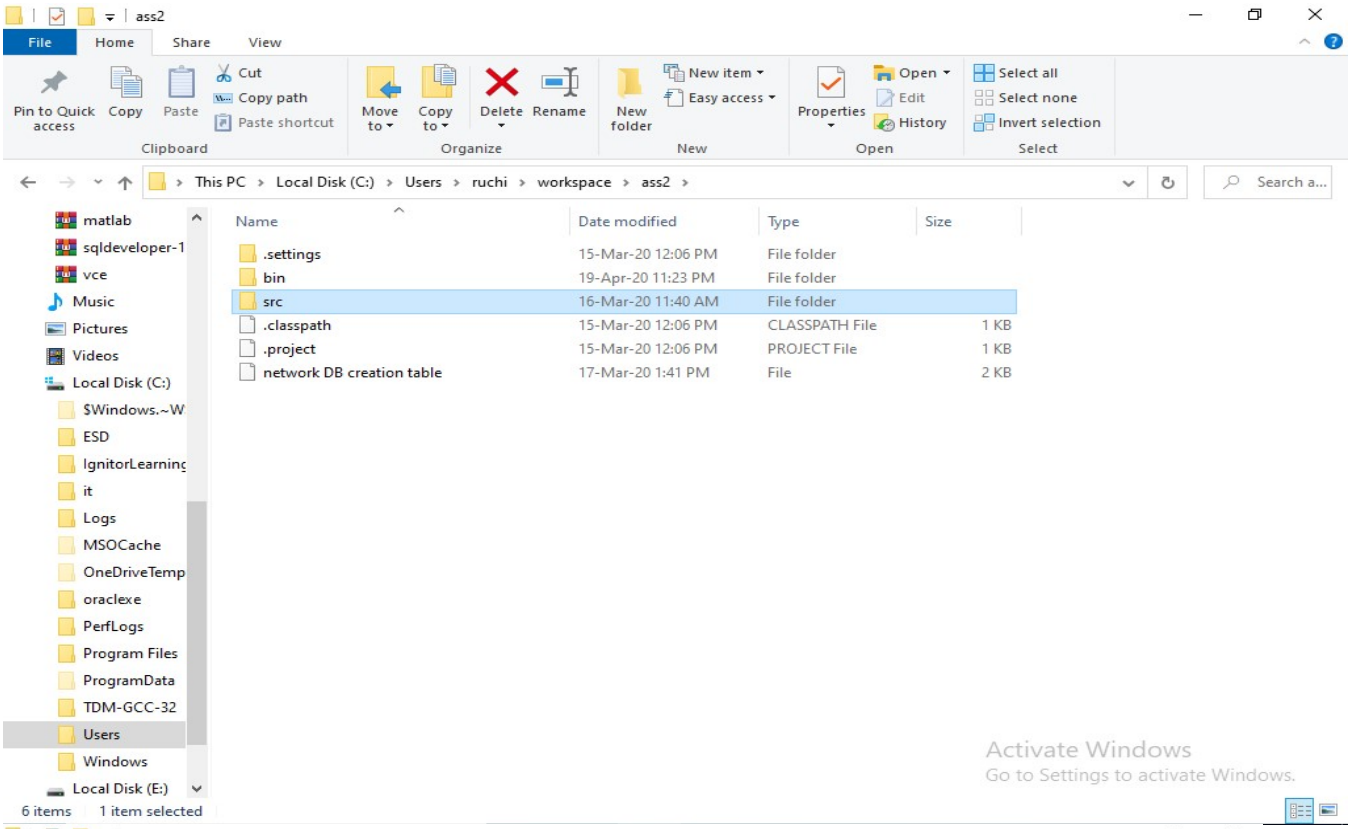
DBMS MINIPROJECT
TITLE:VCE Network Connection Management System

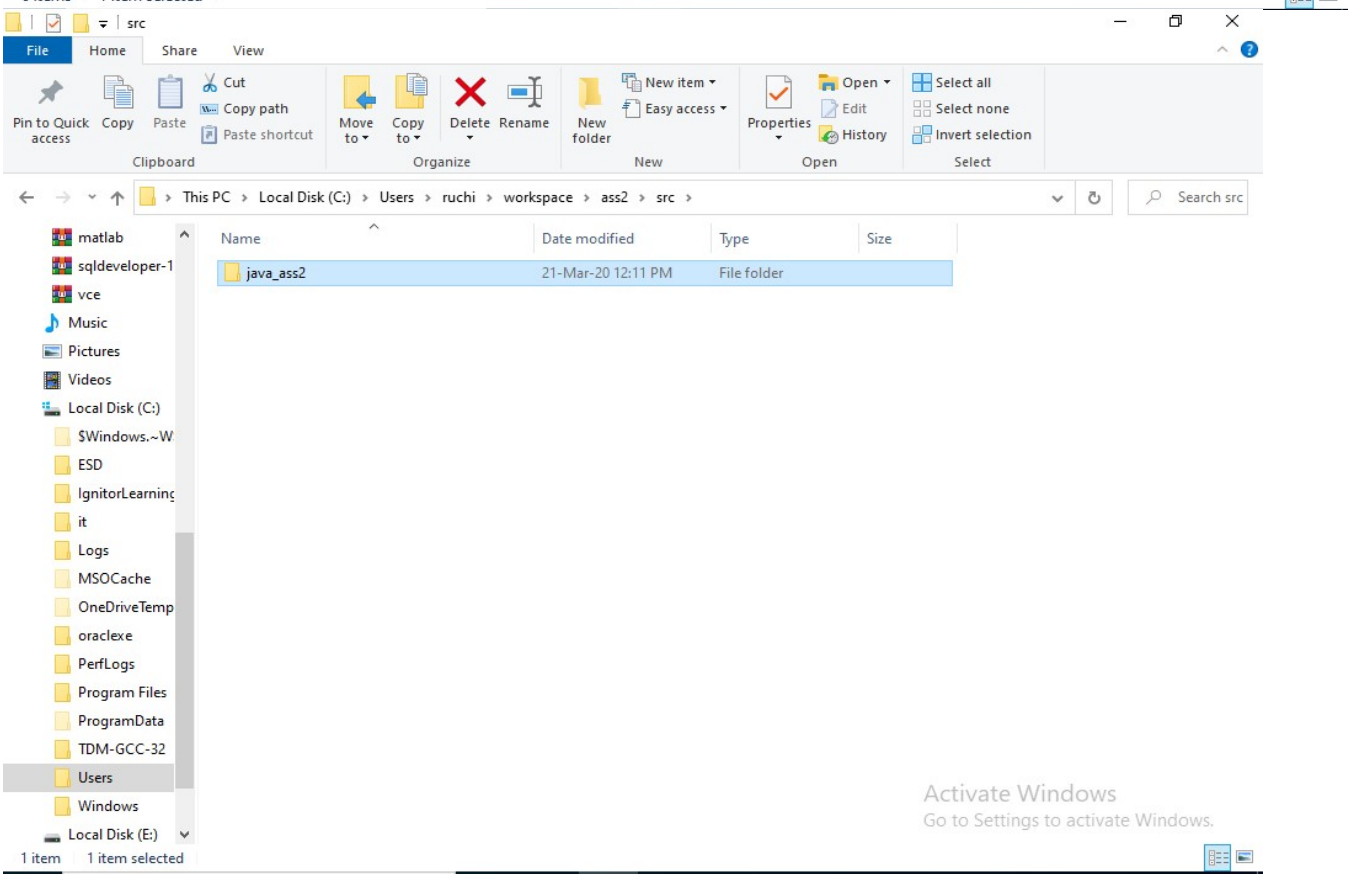
→ project



Roll no:1602-18-737-064
Name:A.Ruchita

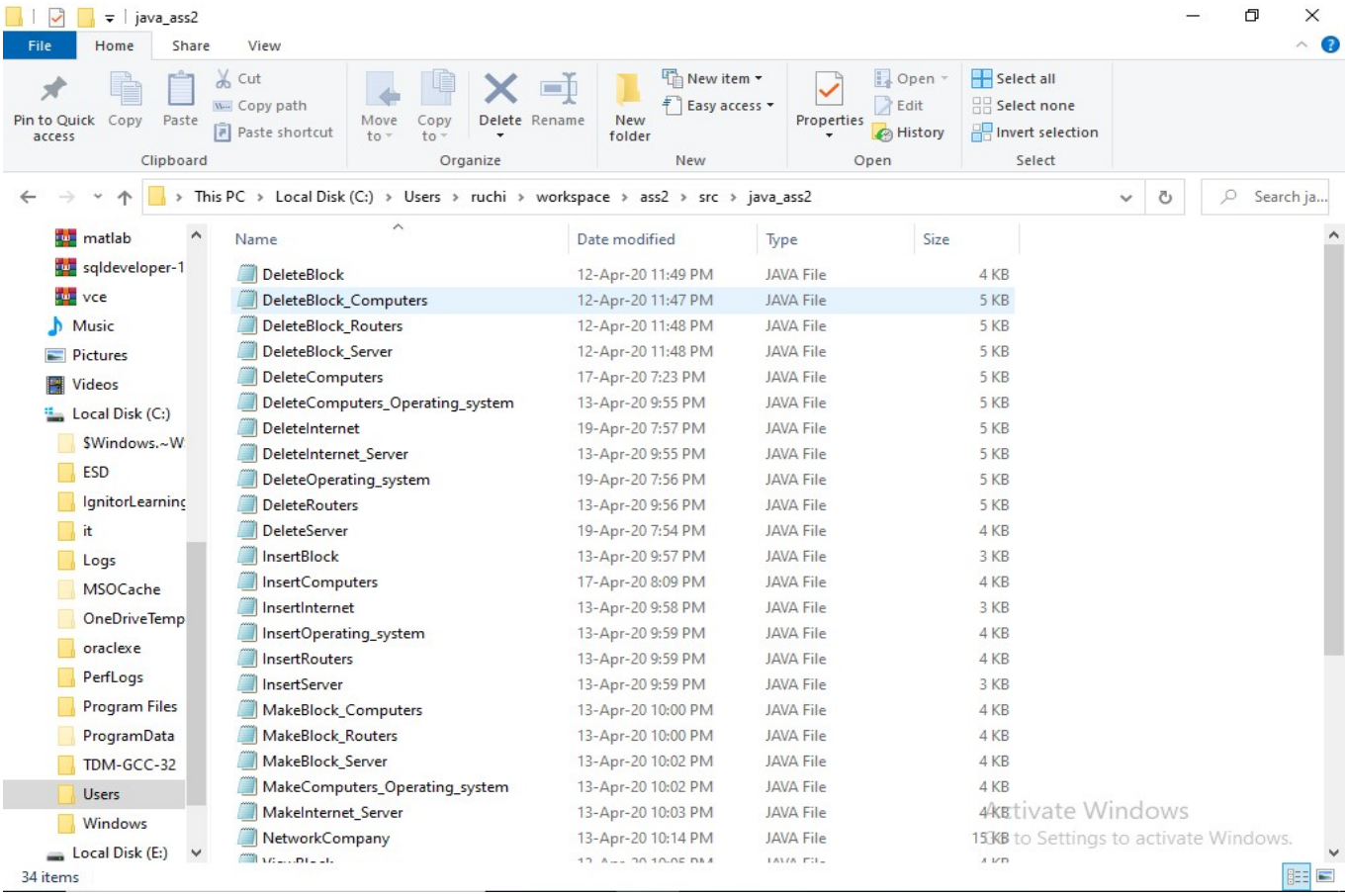
DBMS MINIPROJECT
TITLE:VCE Network Connection Management System





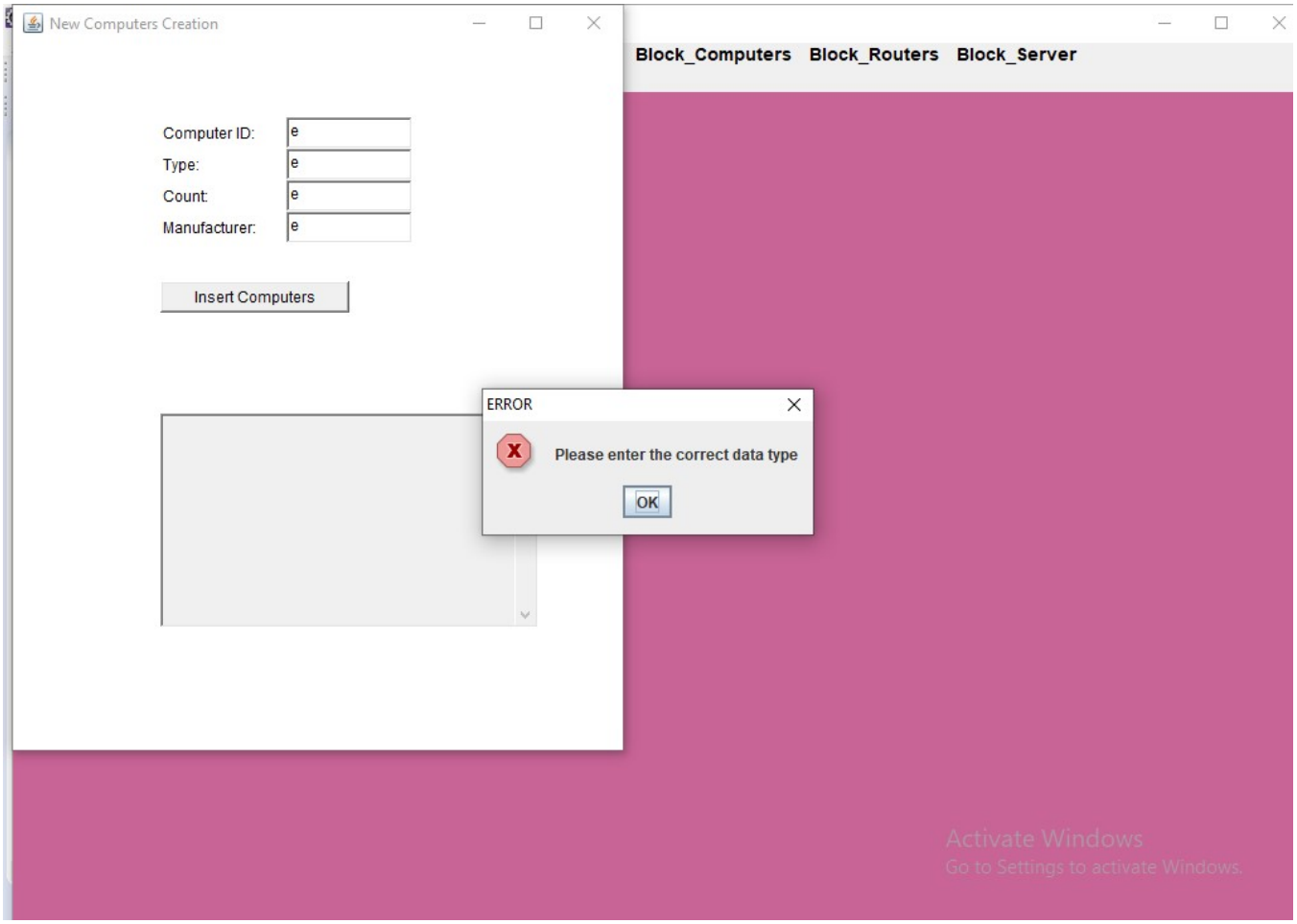
Roll no:1602-18-737-064
Name:A.Ruchita

DBMS MINIPROJECT
TITLE:VCE Network Connection Management System



Roll no:1602-18-737-064
Name:A.Ruchita

TESTING



Roll no:1602-18-737-064
Name:A.Ruchita

DML COMMANDS

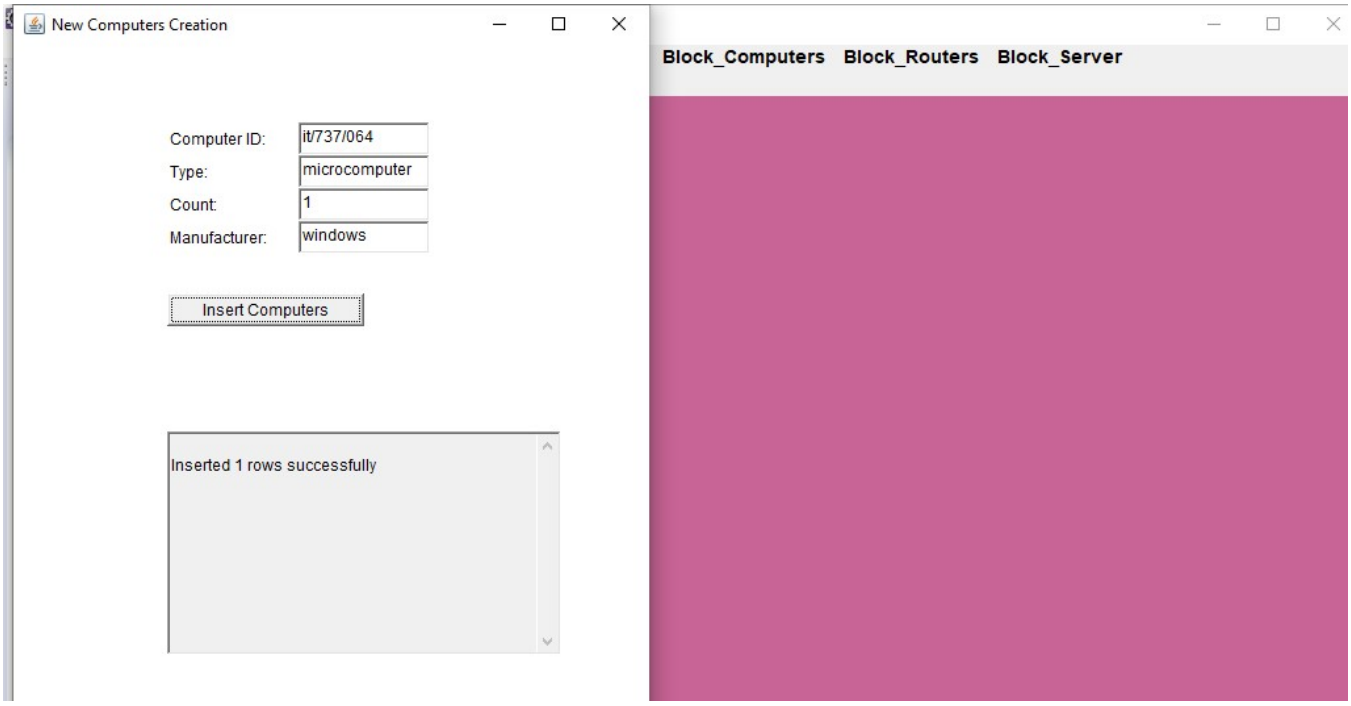
1. Insert into internet values(‘&html’,’&serv_provider’,’&mac_address’);
2. Insert into computers values (‘&type ’ , ’ & count ’ , ’ & manufacturer ’ , ’ & mode ’ ,’ &cid’);
3. Insert into server values(‘&ipaddress’);
4. Insert into block values(‘&bname’,’&hod’,’&branch’);
5. Insert into server values(‘&ipaddress’);
6. Insert into routers values(‘&website’,’&speed’,’&model’,’& username’,’&company’);
7. Insert into operating_system values(‘&osname’,’& version’,’& vendor’);
8. Insert into connected_to values(‘&mac’,’& ipadd’);
9. Insert into provides_network_to(‘&bname’,’& ipaddress’);
10. Insert into contains values (‘&cid’,’&bname’);
11. Insert into has values (‘&cid’,’& osname’);
12. Insert into are_having values (‘&website’,’& bname’);

OUTPUT SCREENSHOTS:

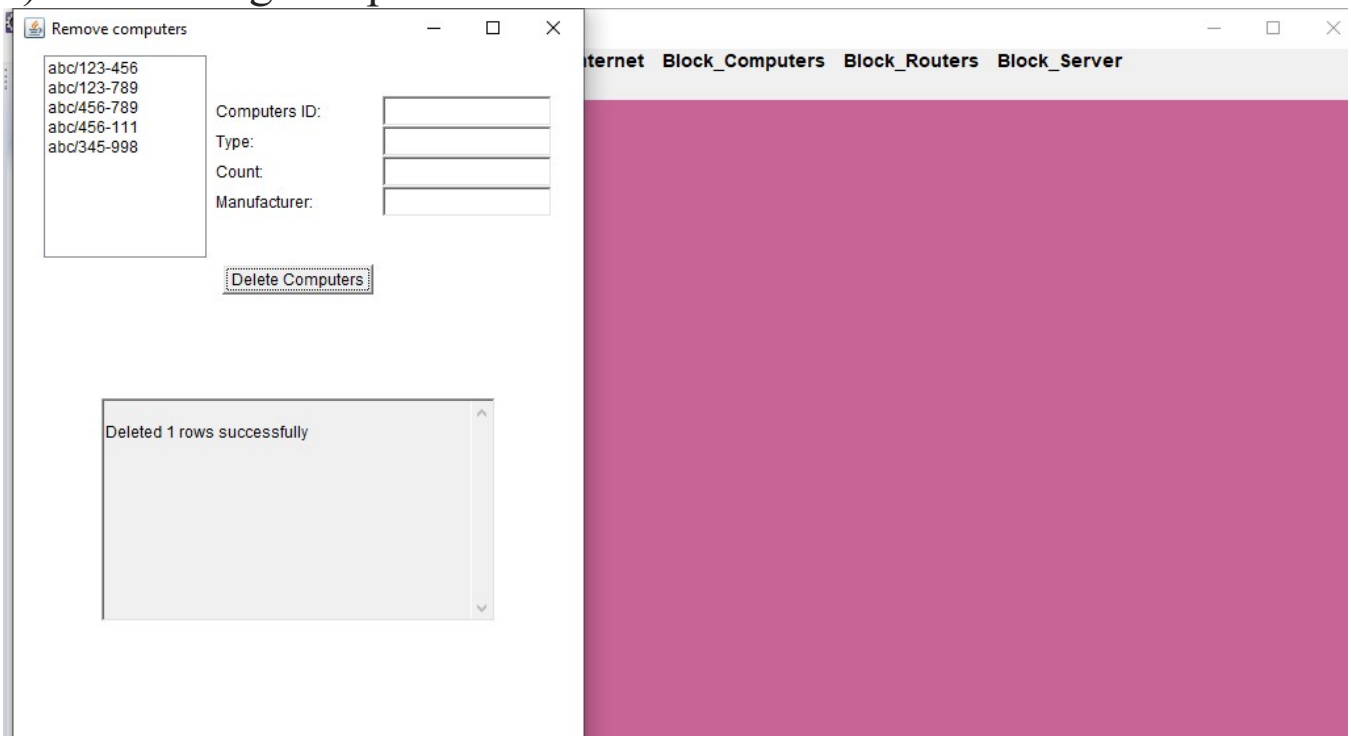
Java GUI Screenshot:

Computers

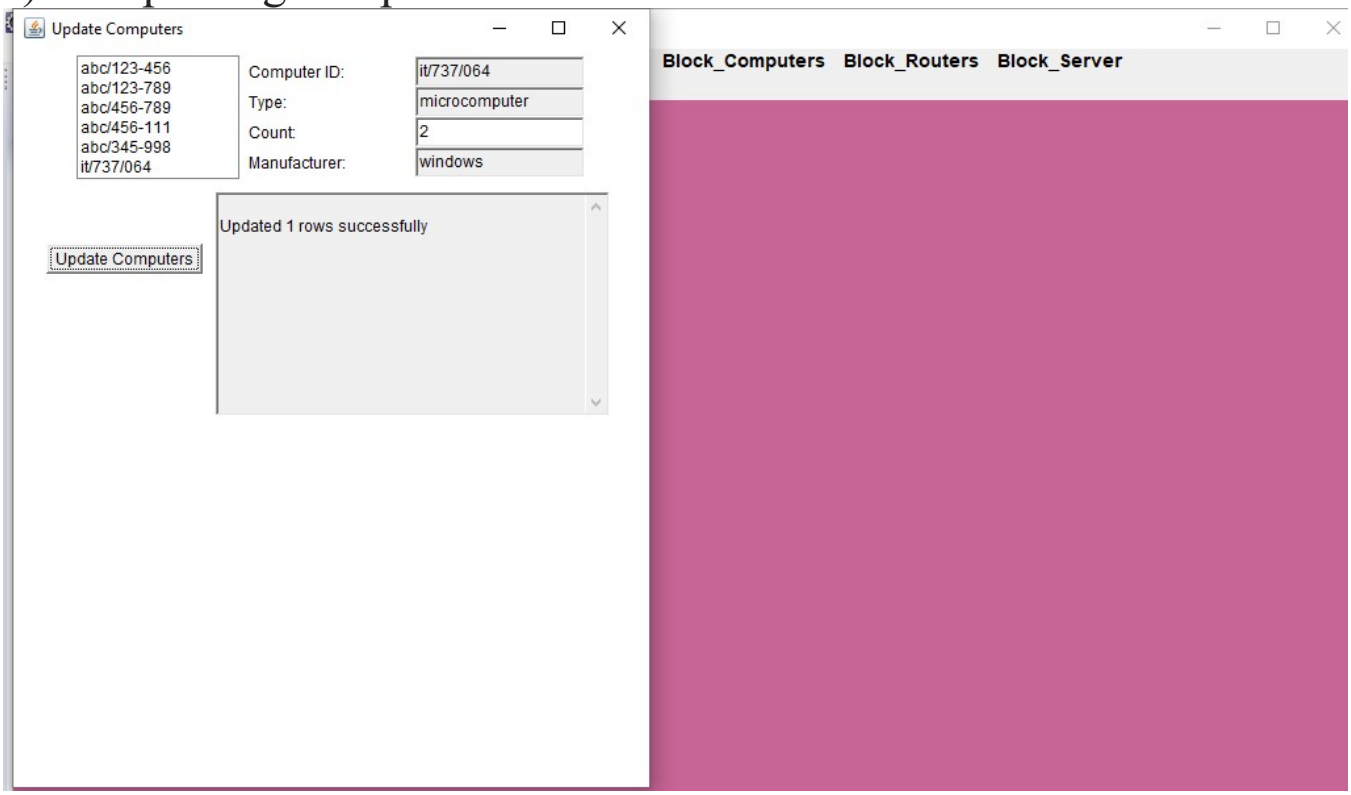
1) For inserting computers



2) for deleting computers



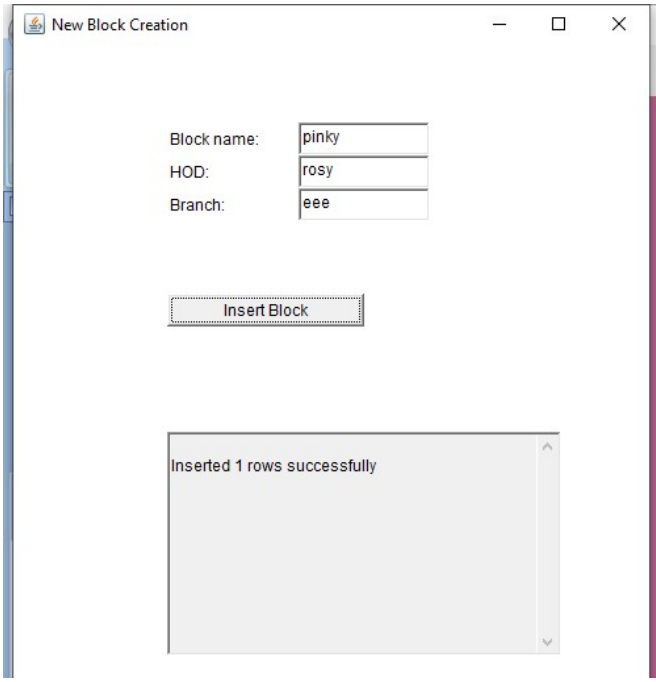
3) for updating computers

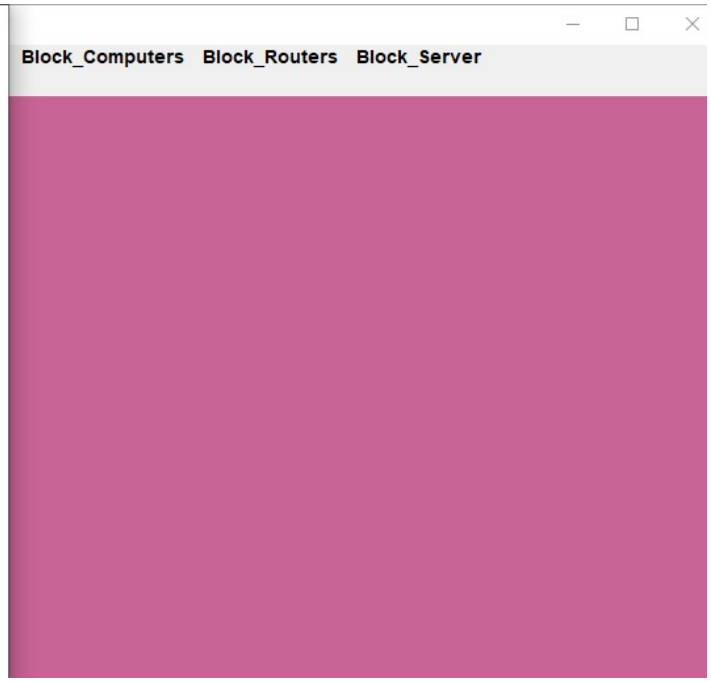


Roll no:1602-18-737-064
Name:A.Ruchita

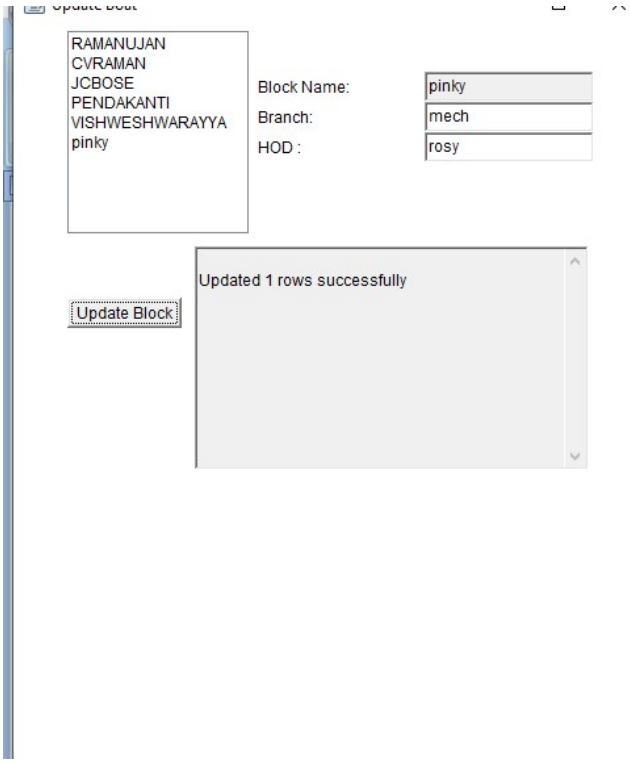
Block

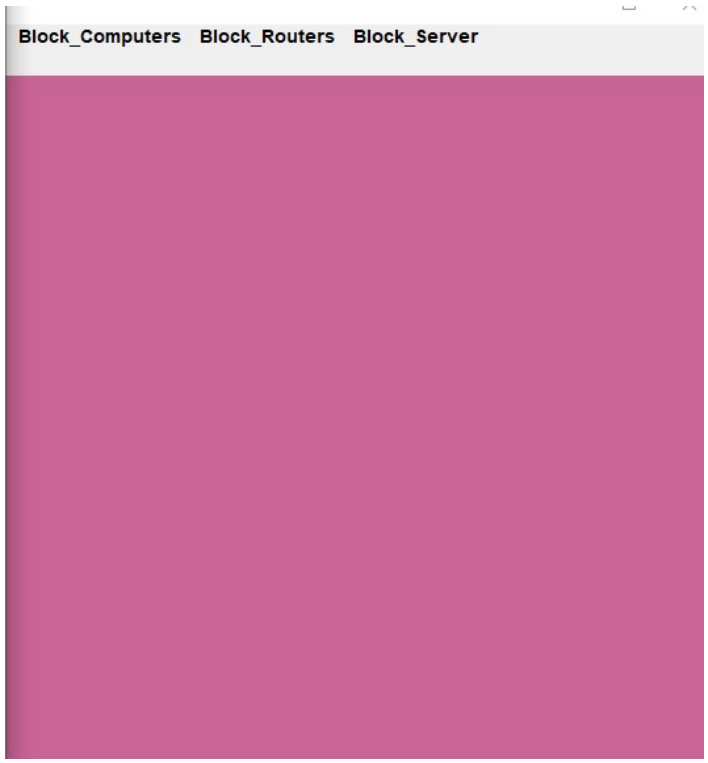
1)insert block





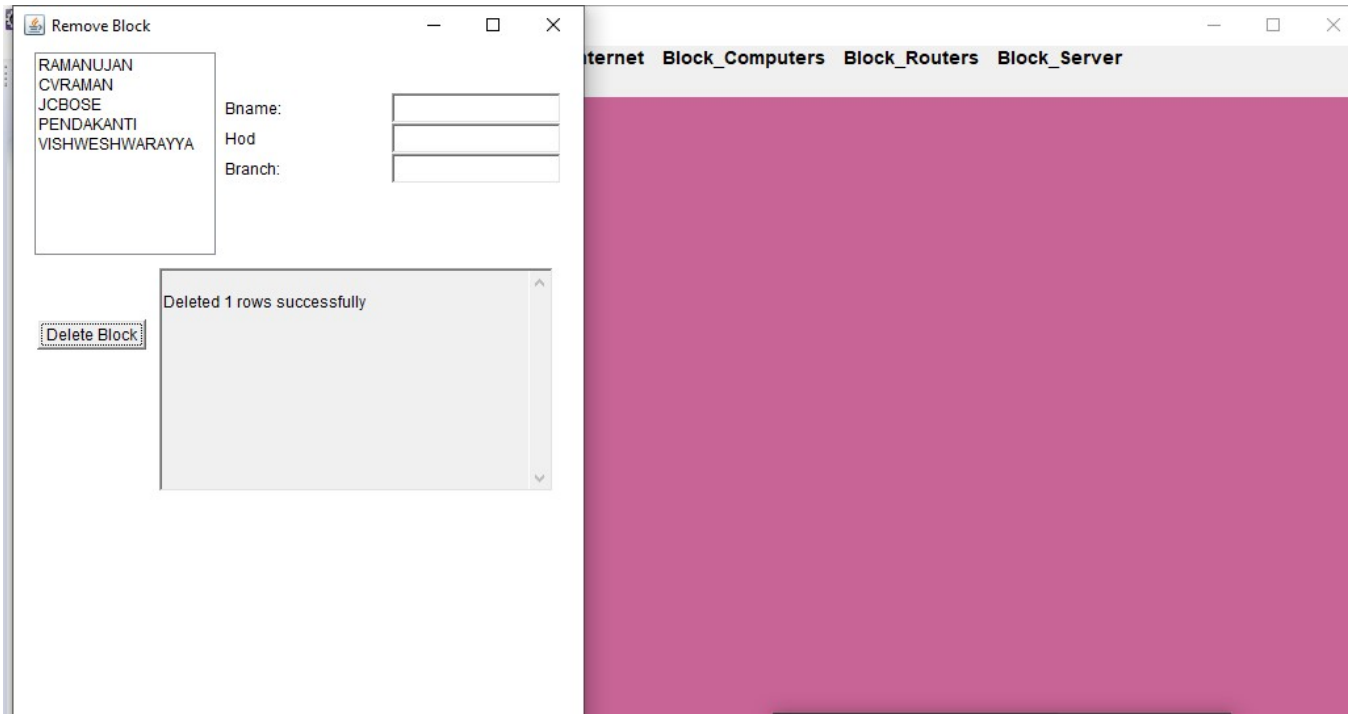
2) update block





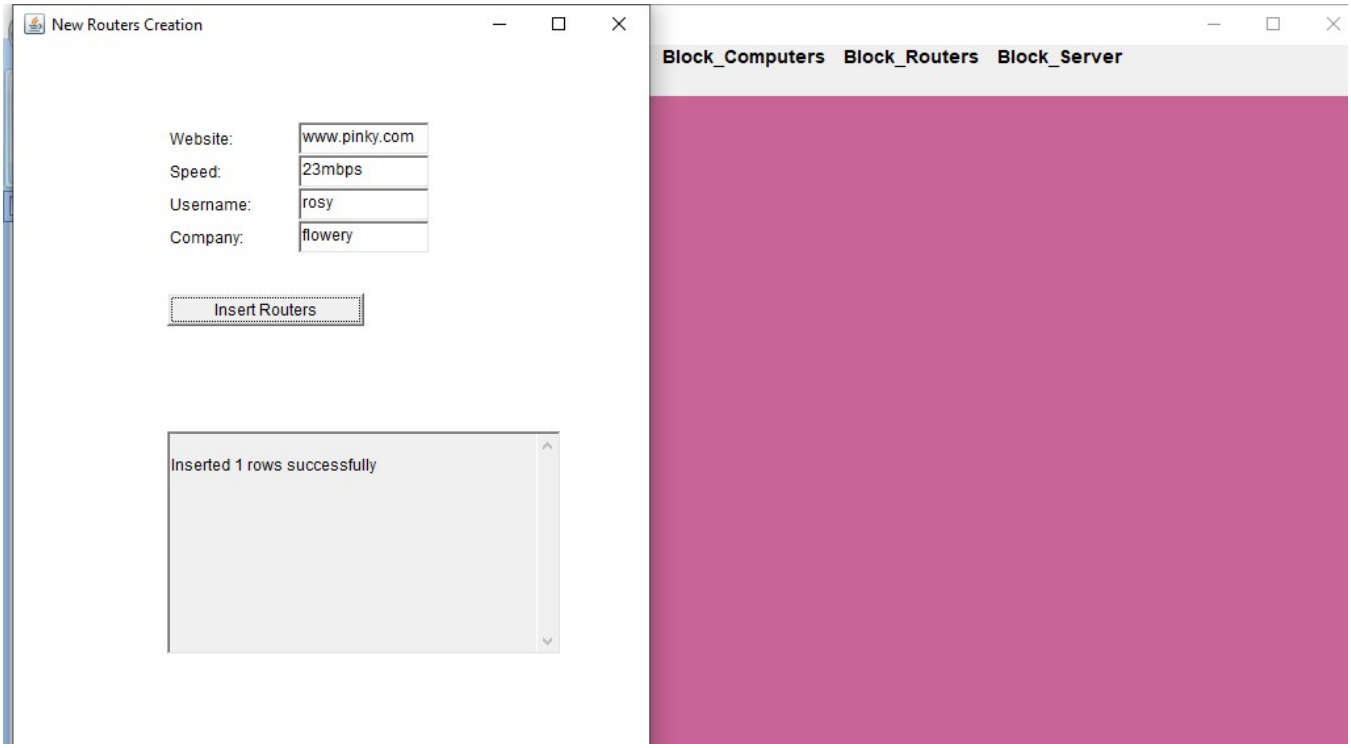
3) delete block

Roll no:1602-18-737-064
Name:A.Ruchita

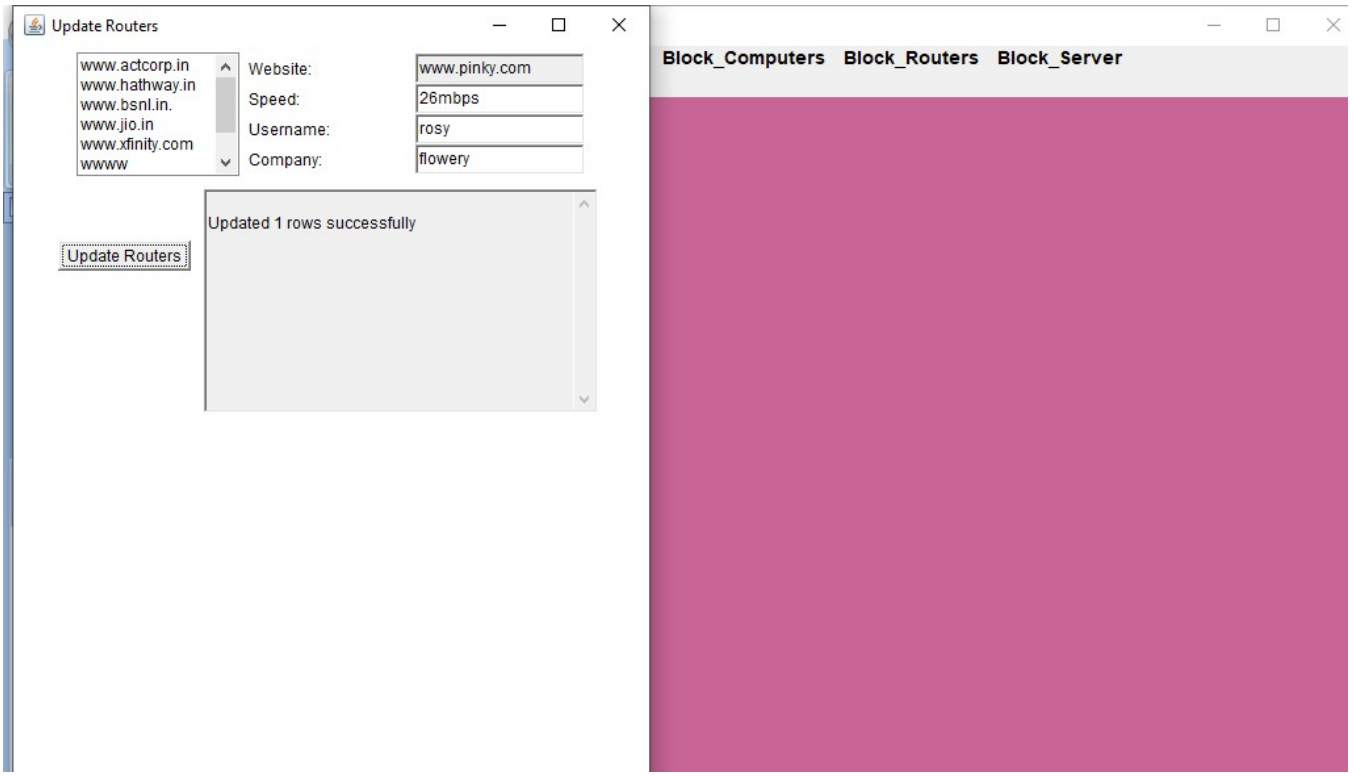


Routers

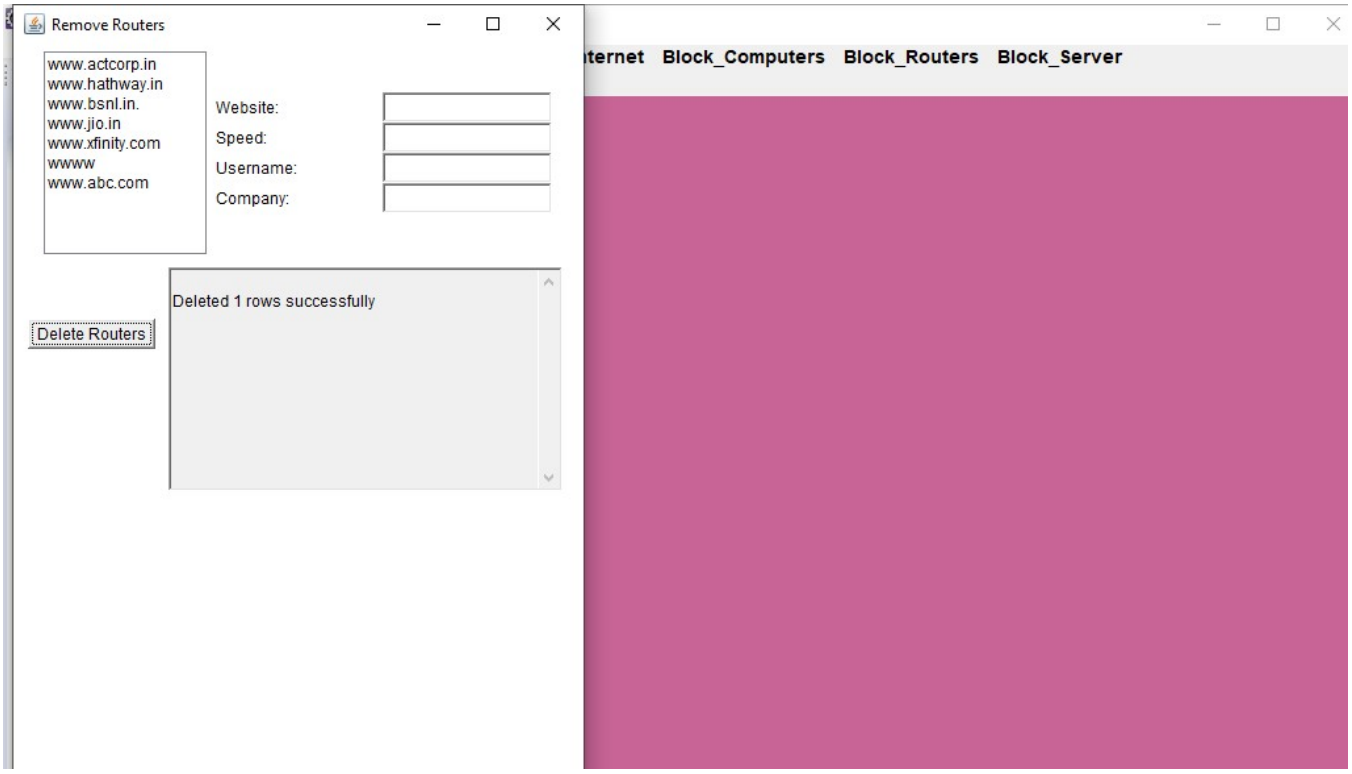
1)insert routers



2)update routers



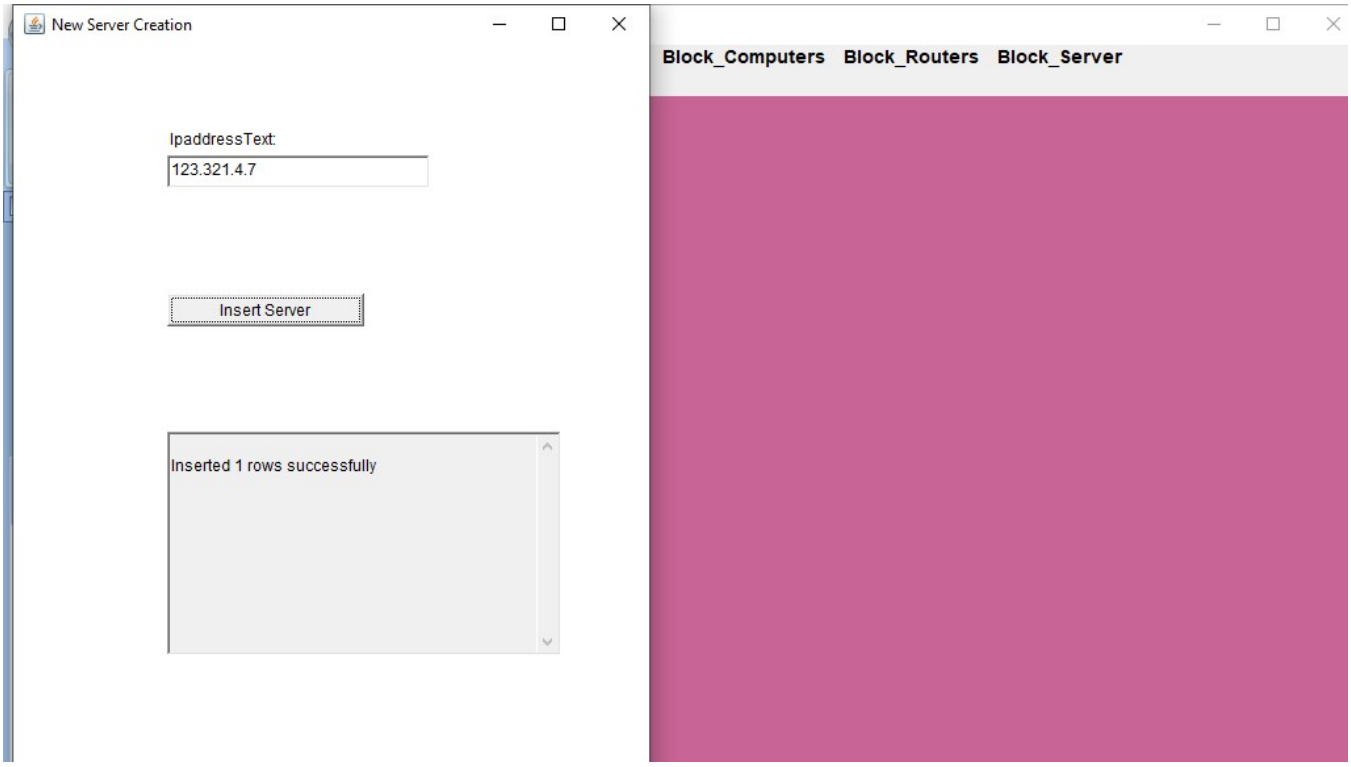
3)delete routers



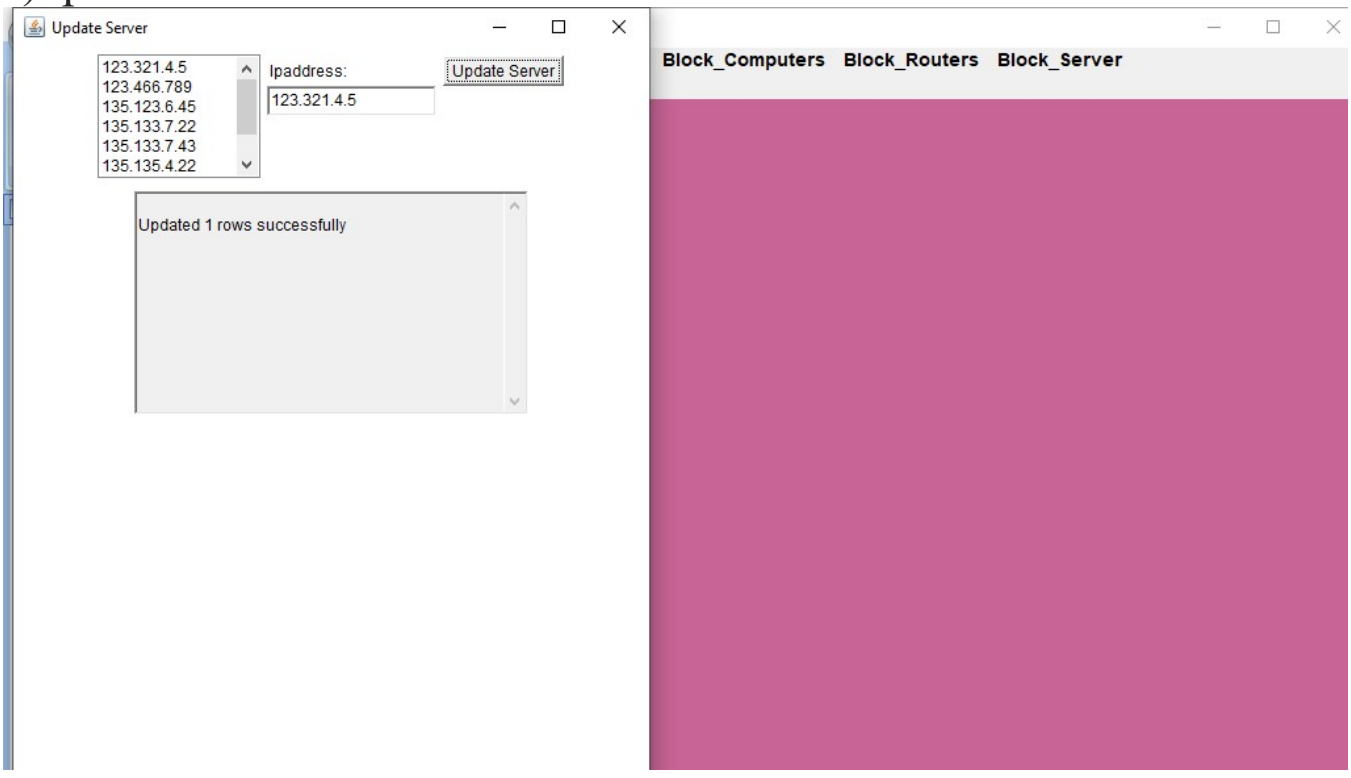
Roll no:1602-18-737-064
Name:A.Ruchita

Servers

1)insert server

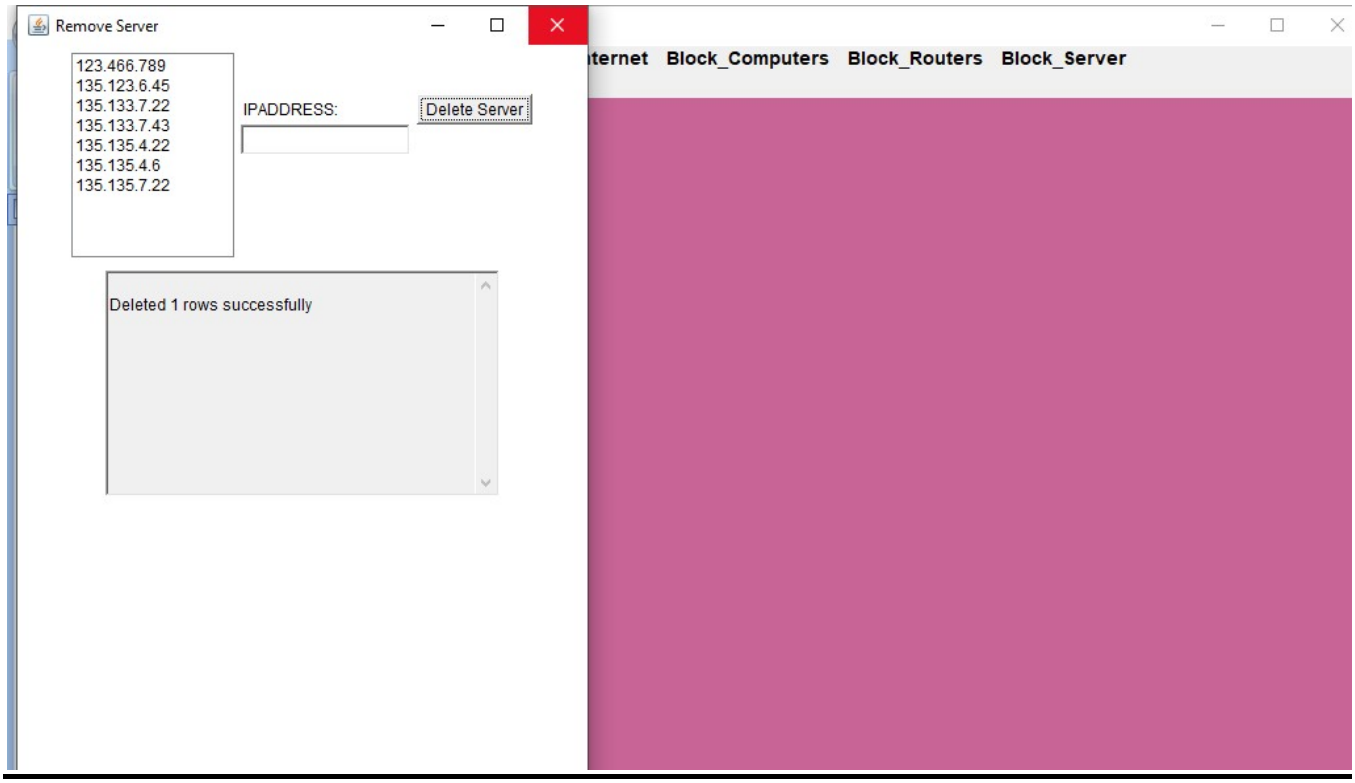


2)update server



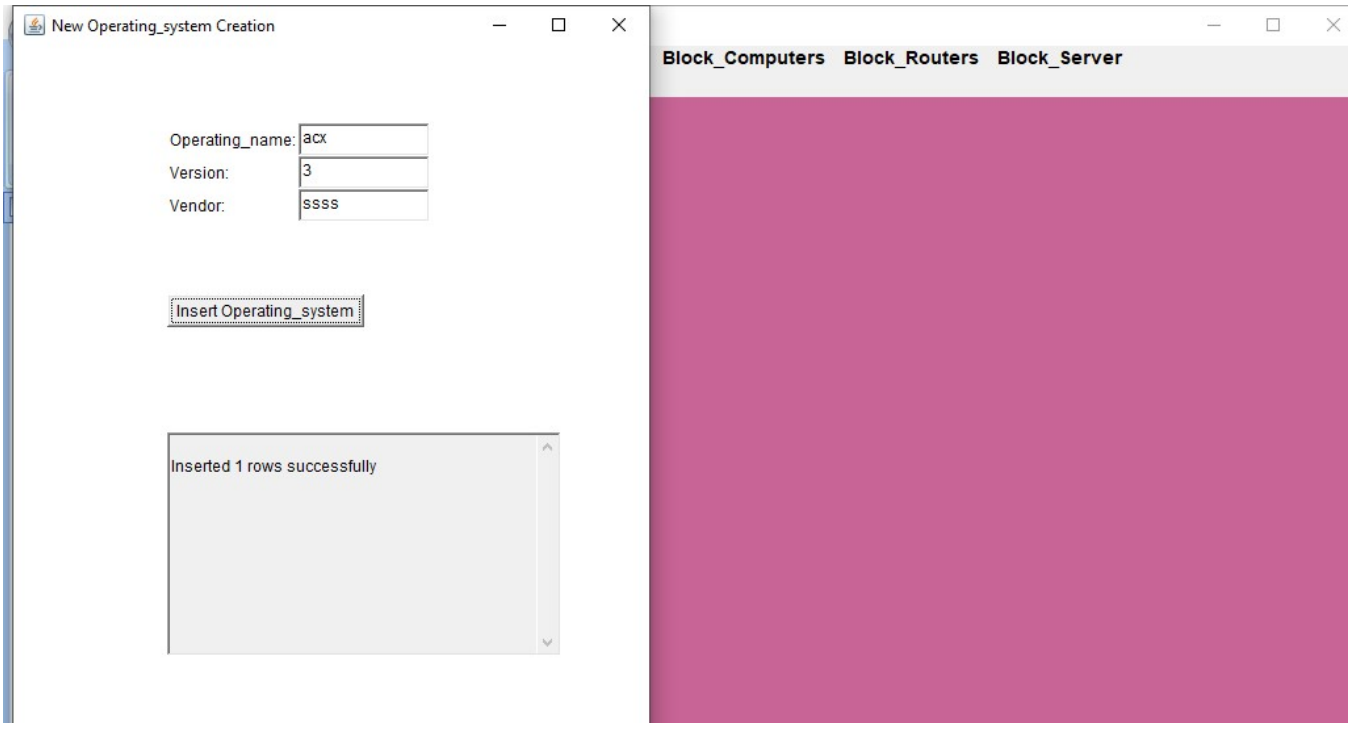
Roll no:1602-18-737-064
Name:A.Ruchita

3)delete server

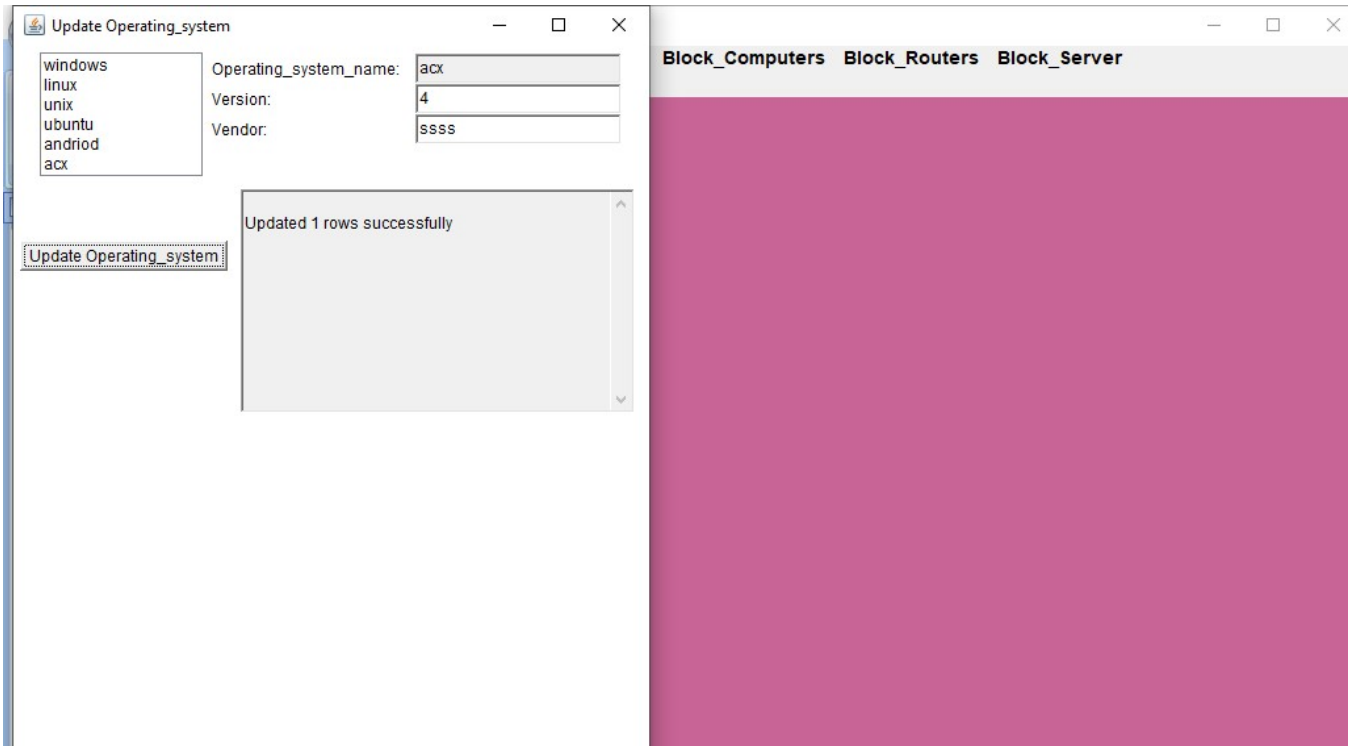


Operating System

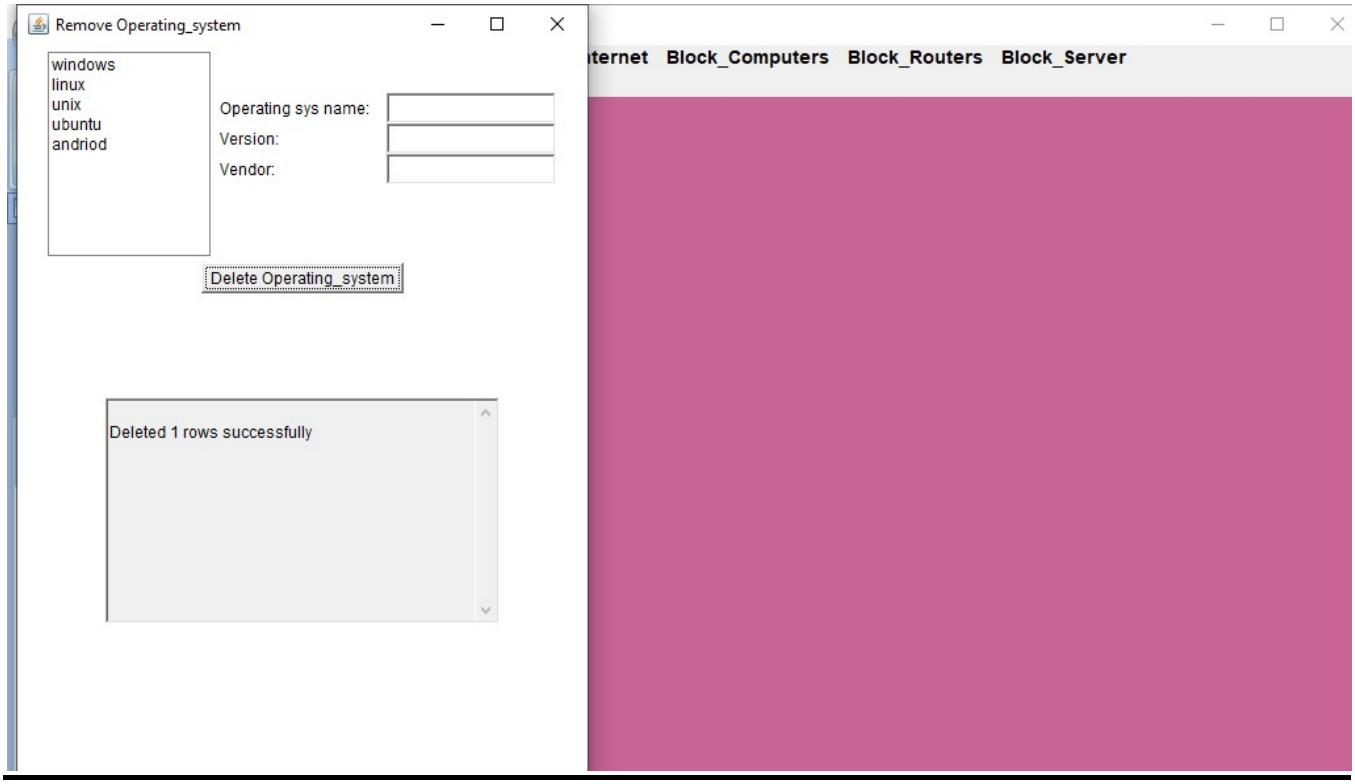
1)insert operating system



2)update operating system

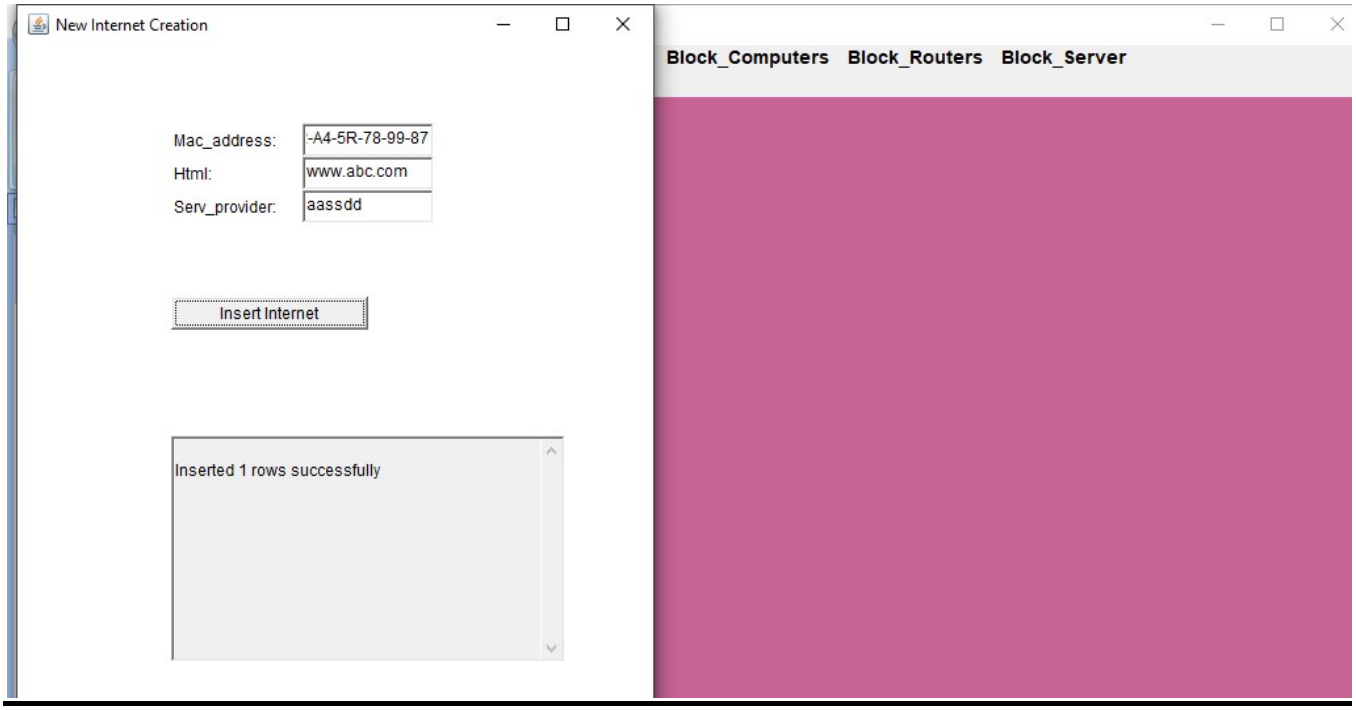


3)delete operating system

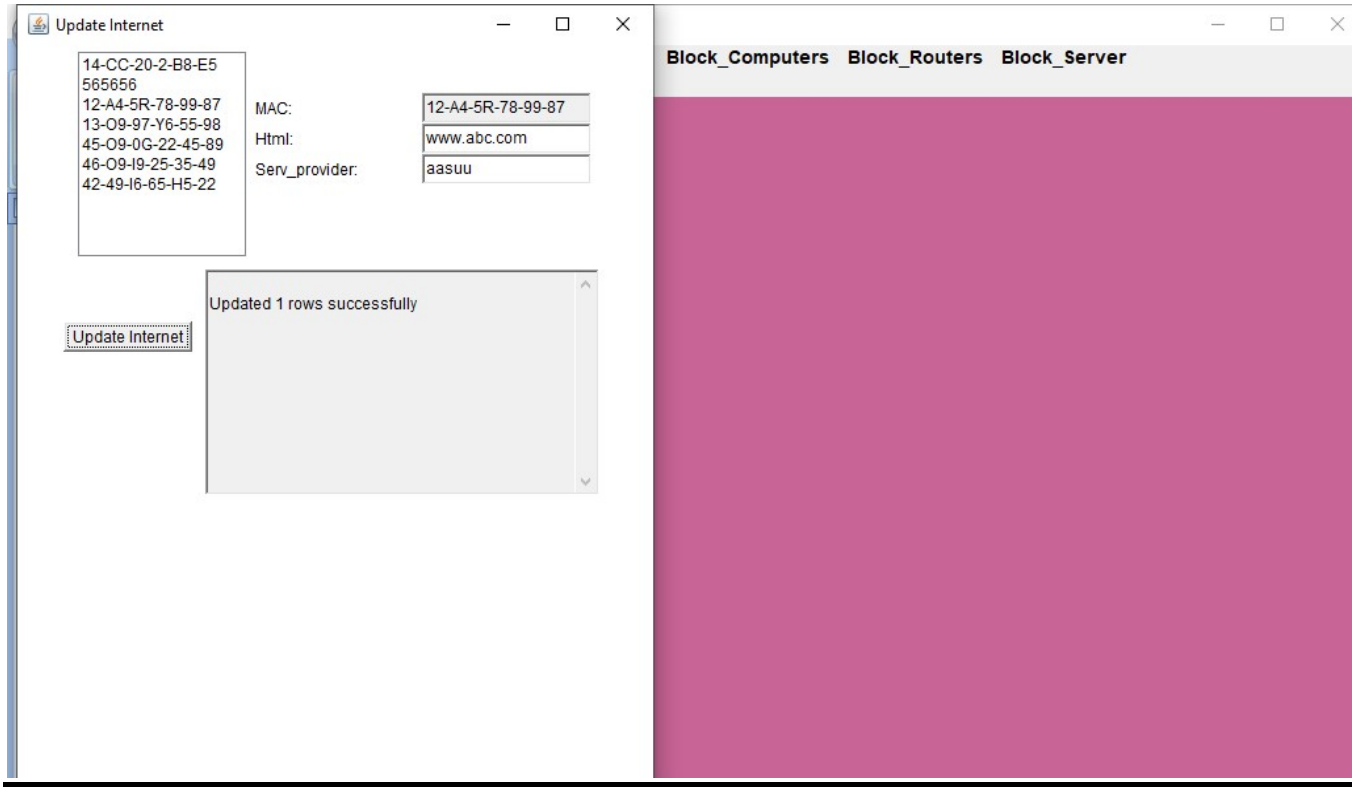


Internet

1)insert internet

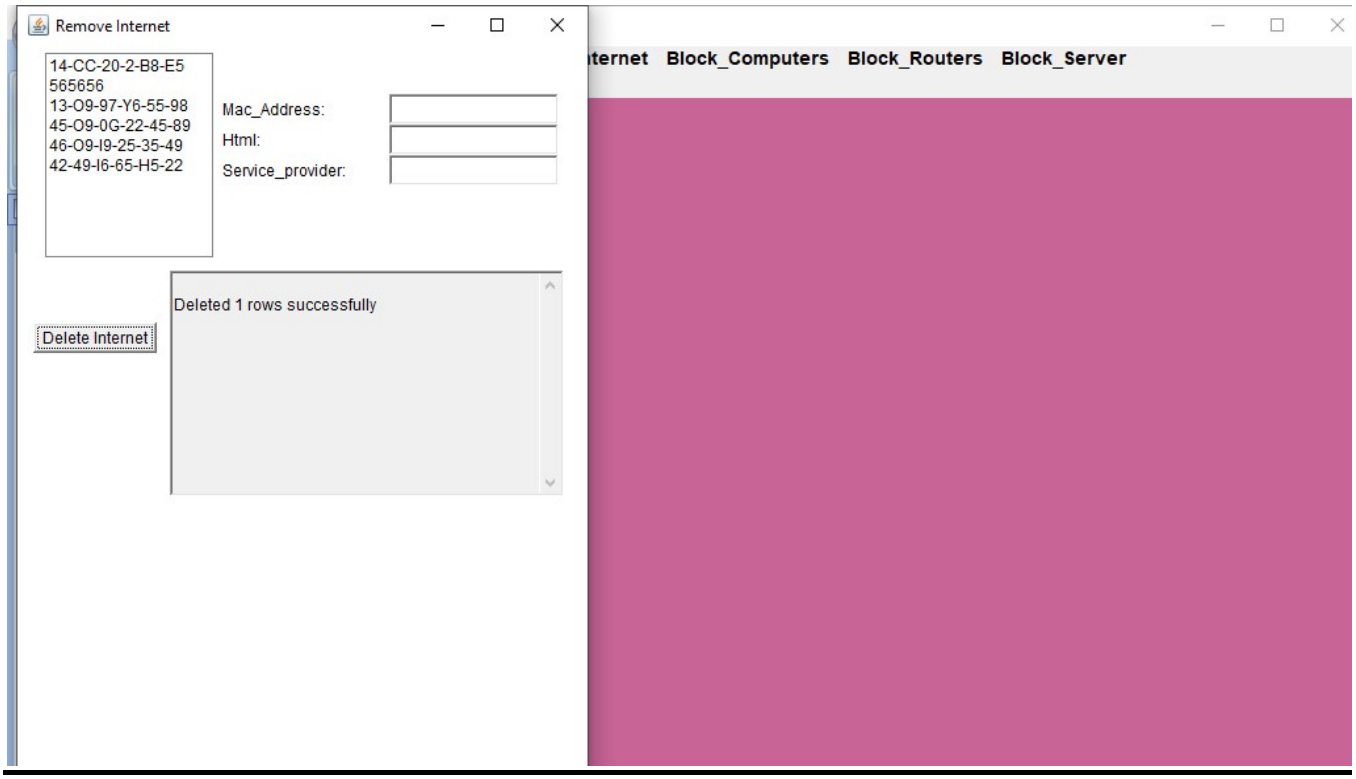


2)update internet



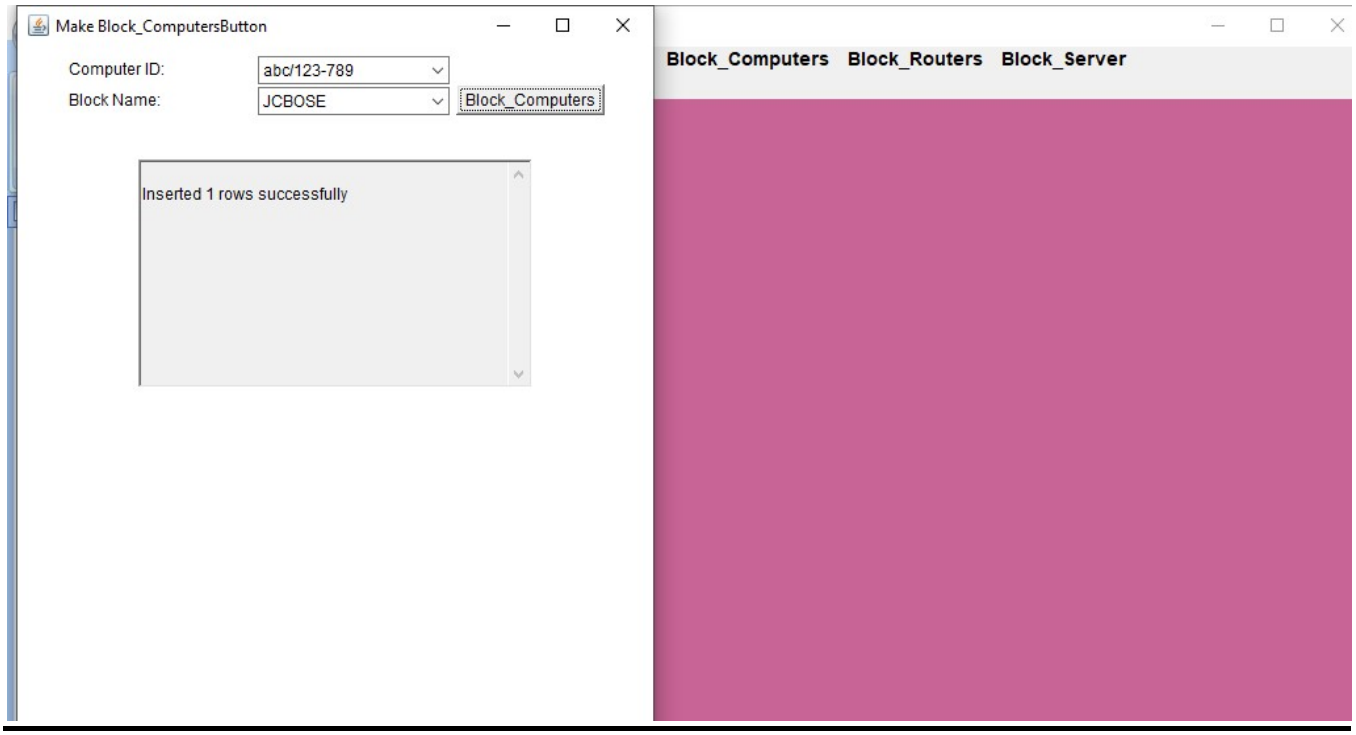
Roll no:1602-18-737-064
Name:A.Ruchita

3) delete internet

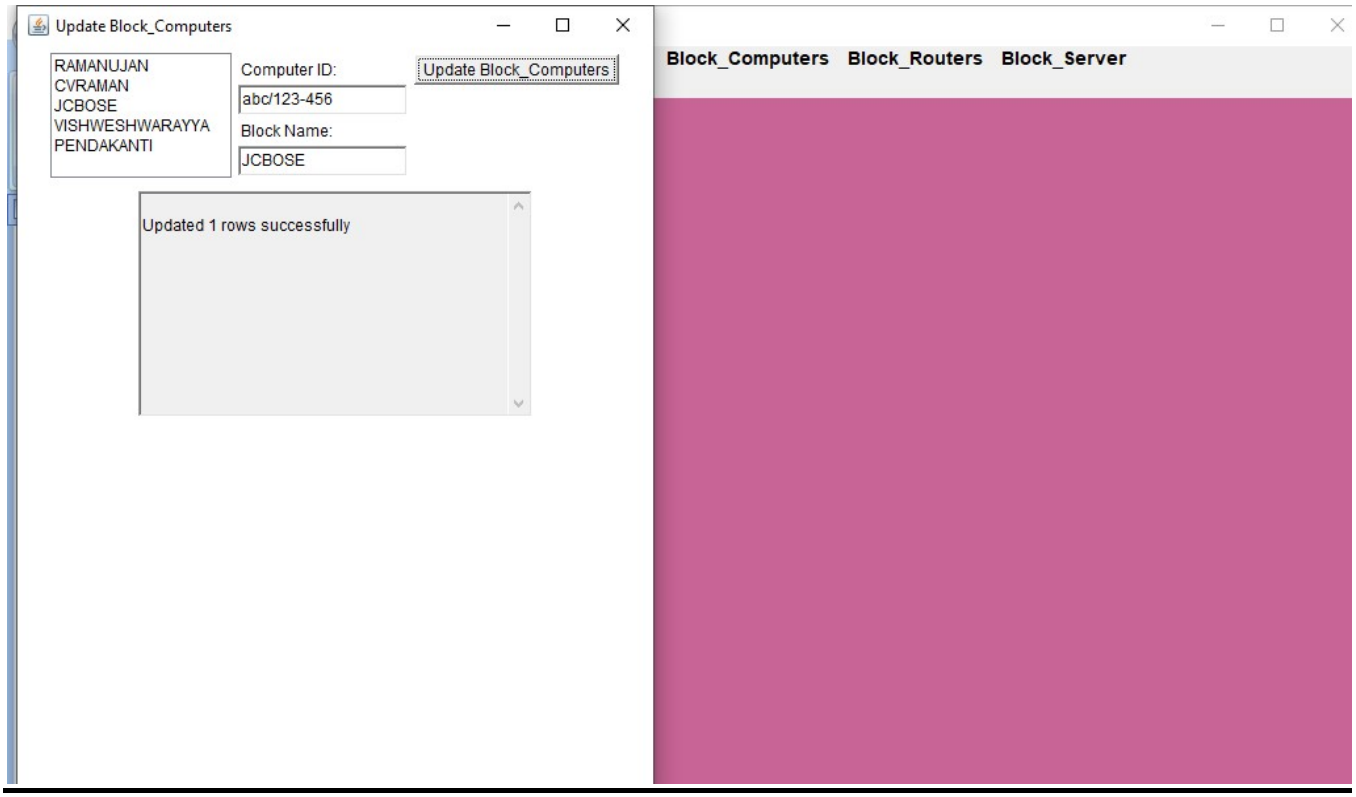


Block_computers

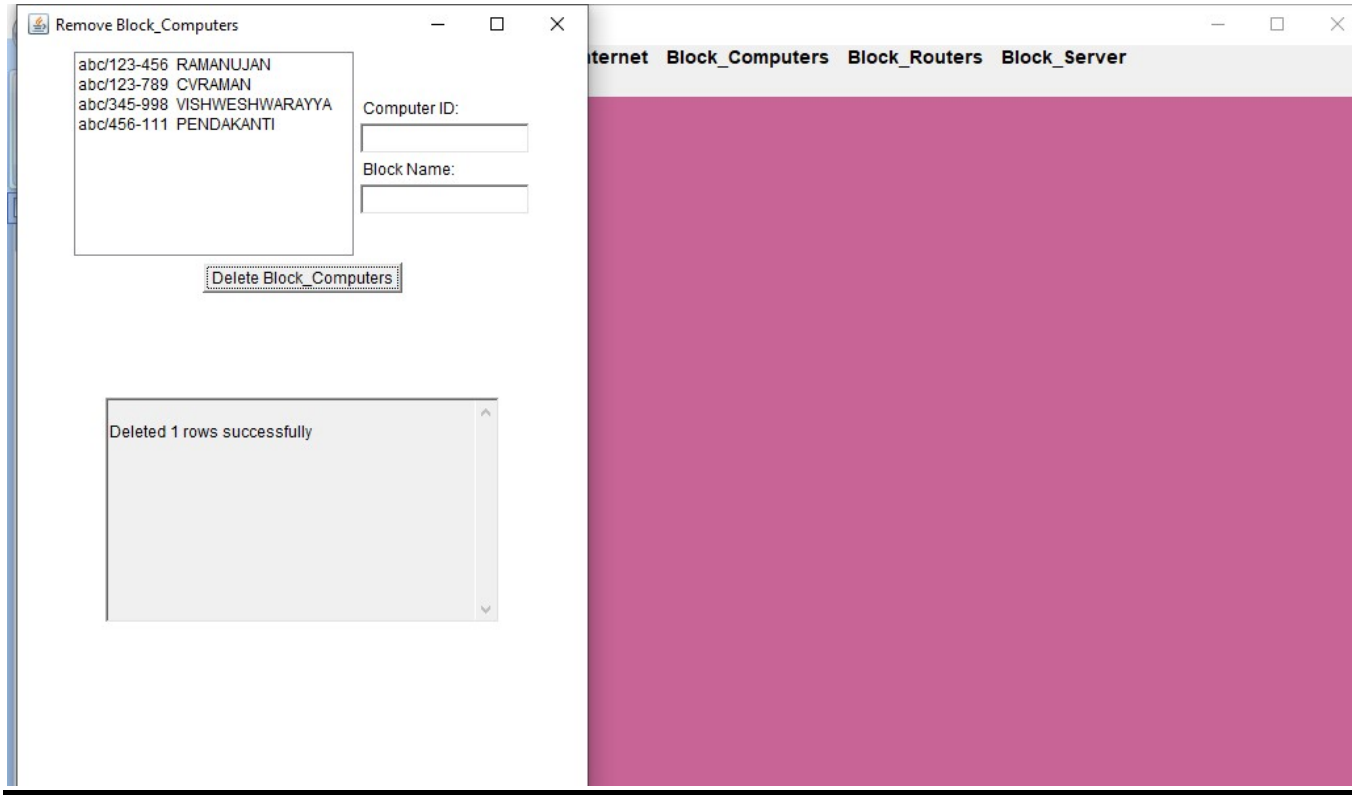
1)insert Block_computers



2)update Block_computers

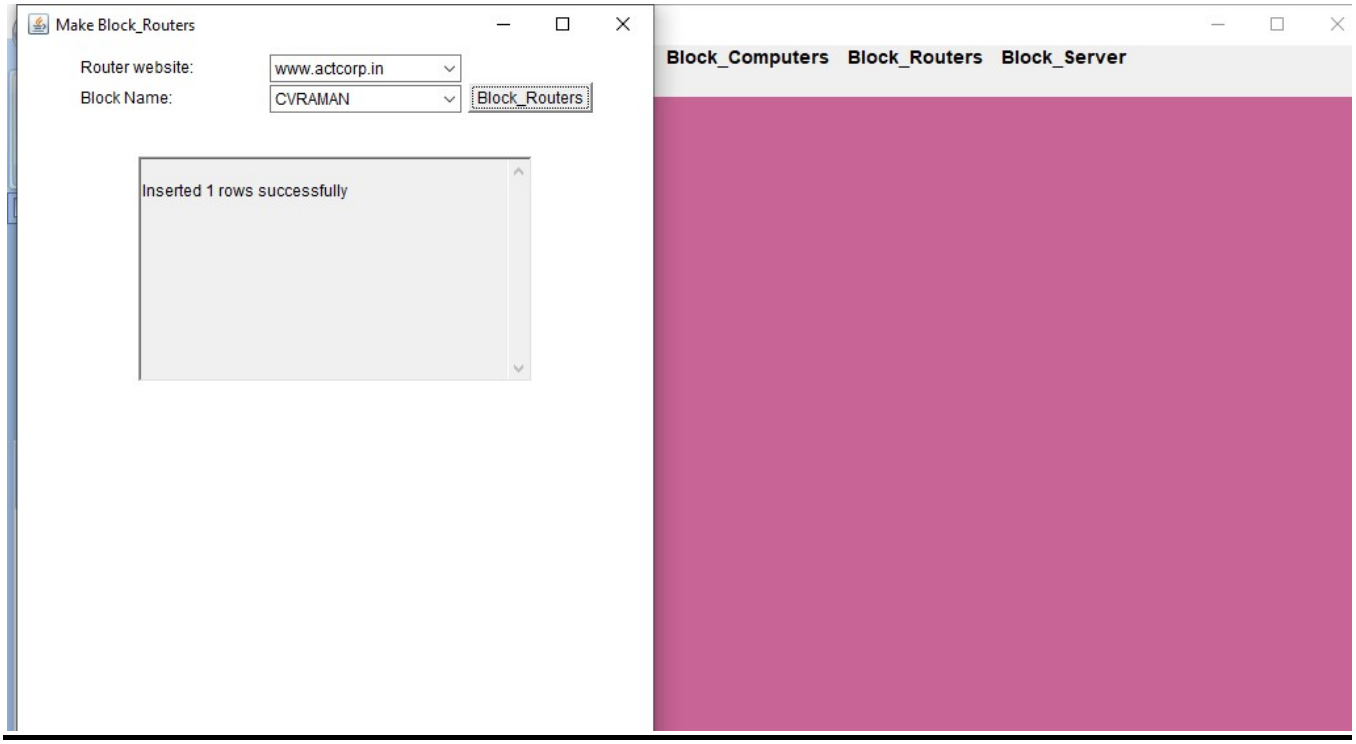


3)delete Block_computers

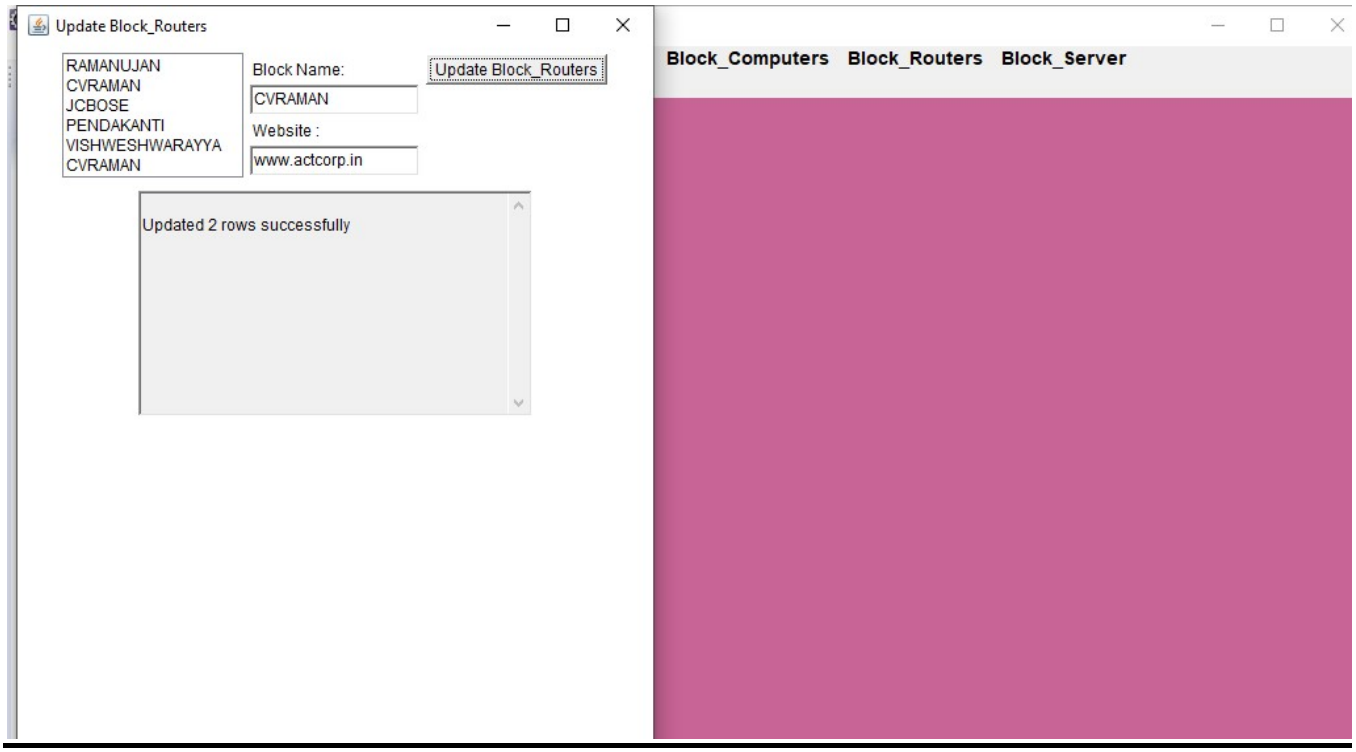


Block_routers

1)insert block_routers

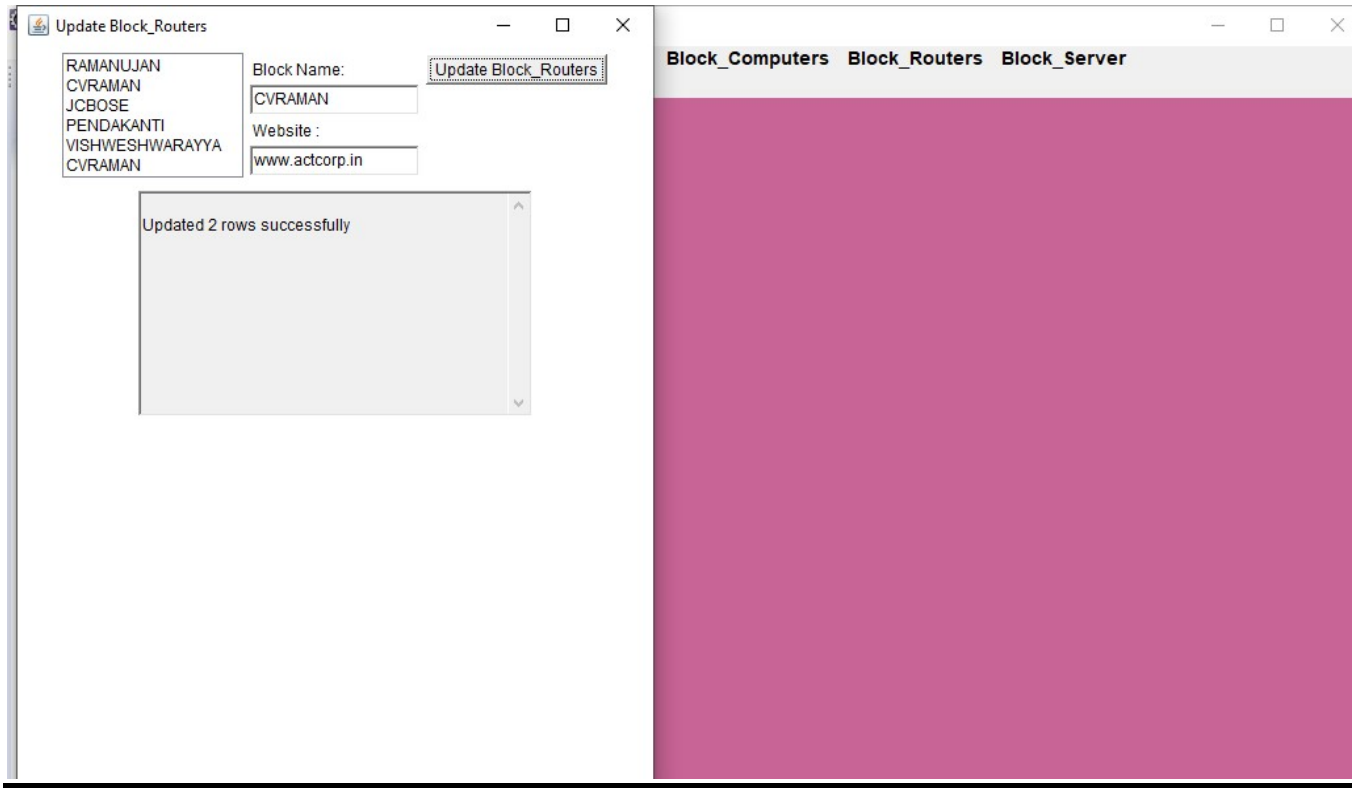


2)update block_routers



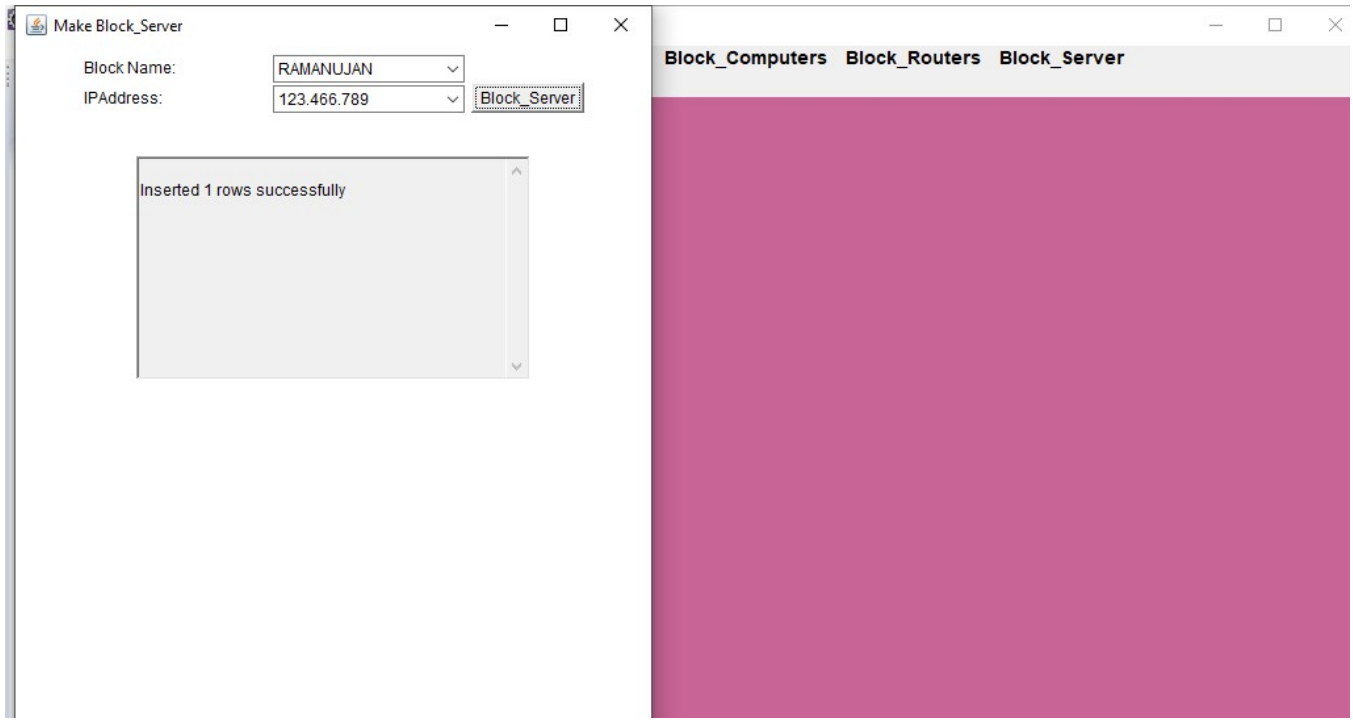
Roll no:1602-18-737-064
Name:A.Ruchita

3)delete block_routers

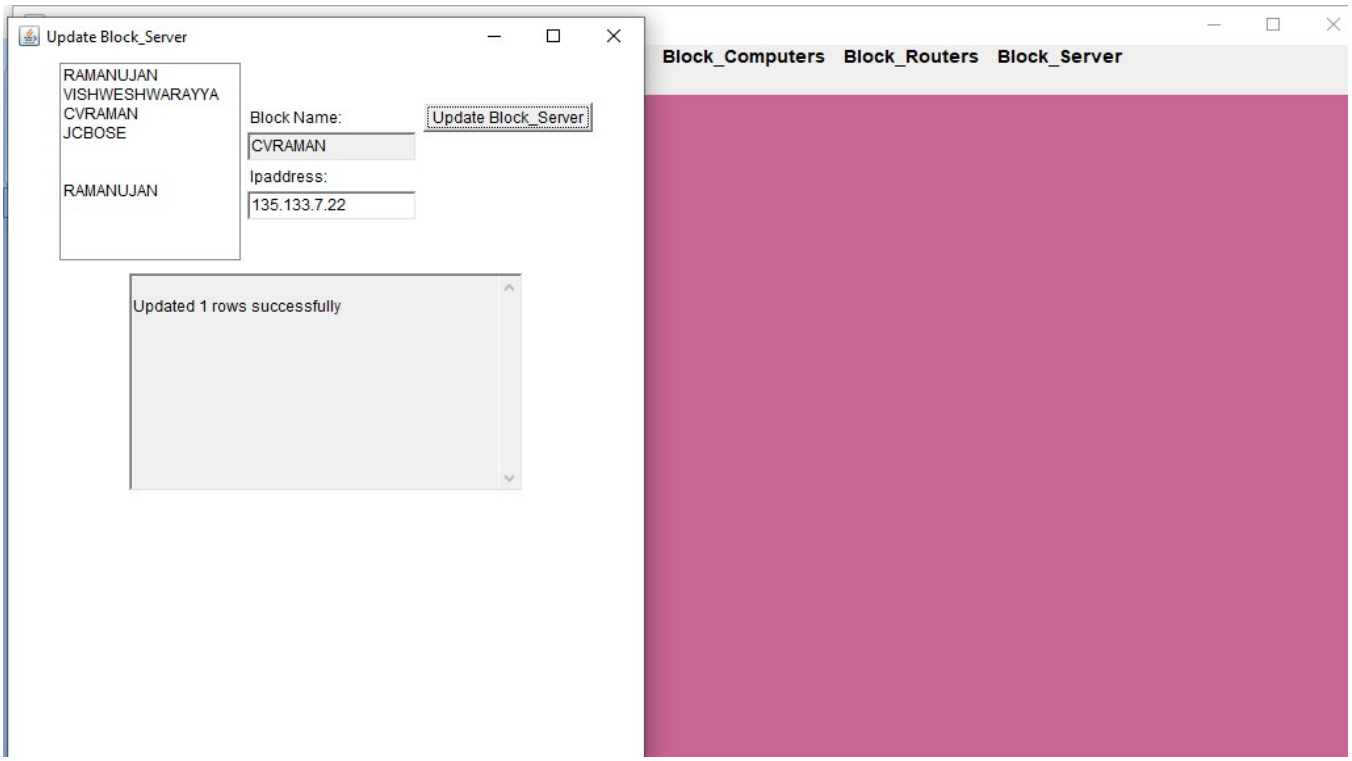


Block_server

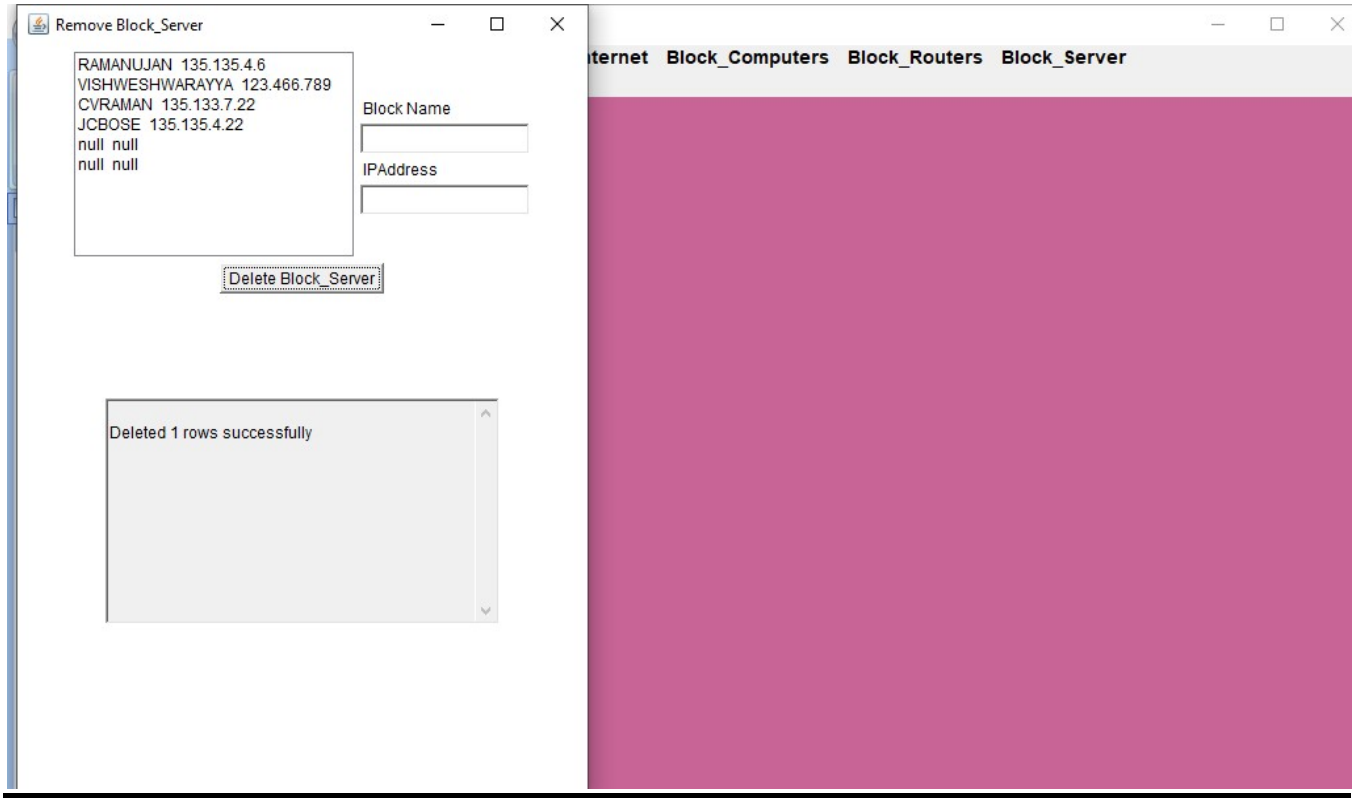
1)insert block_server



2)update block_server



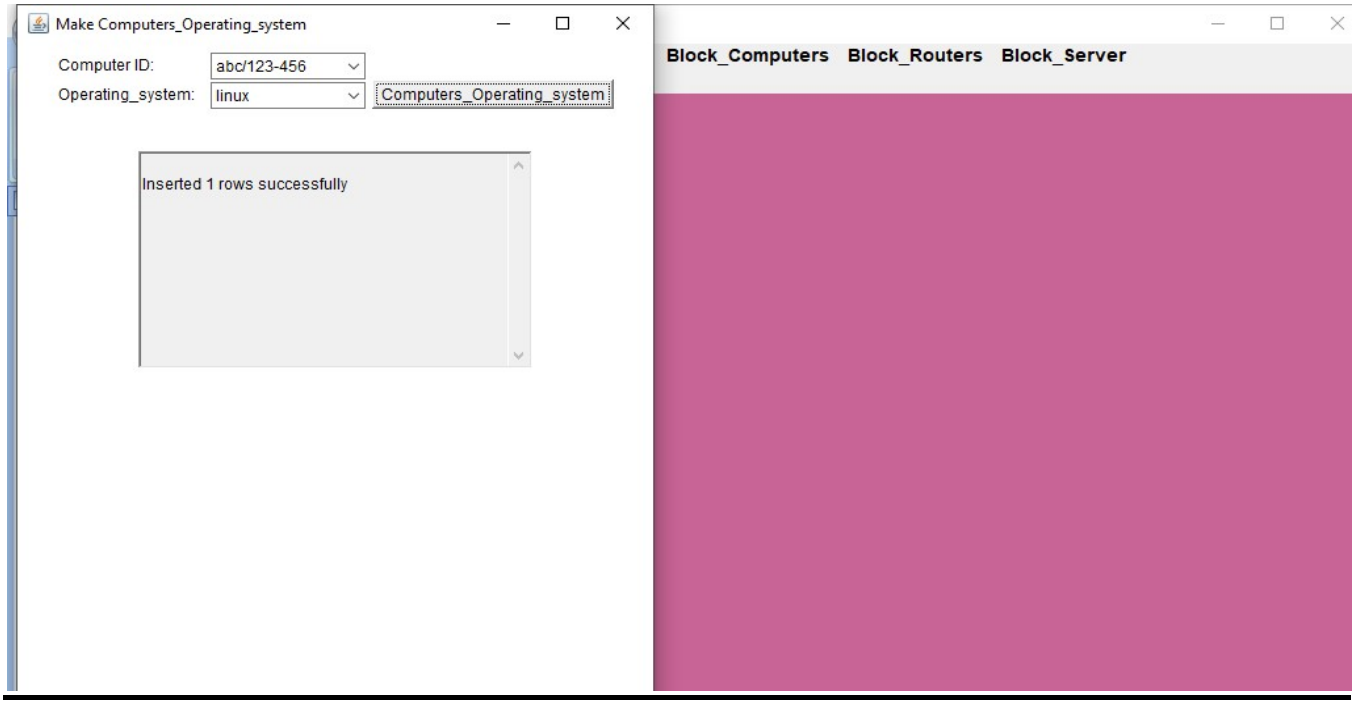
3)delete block_server



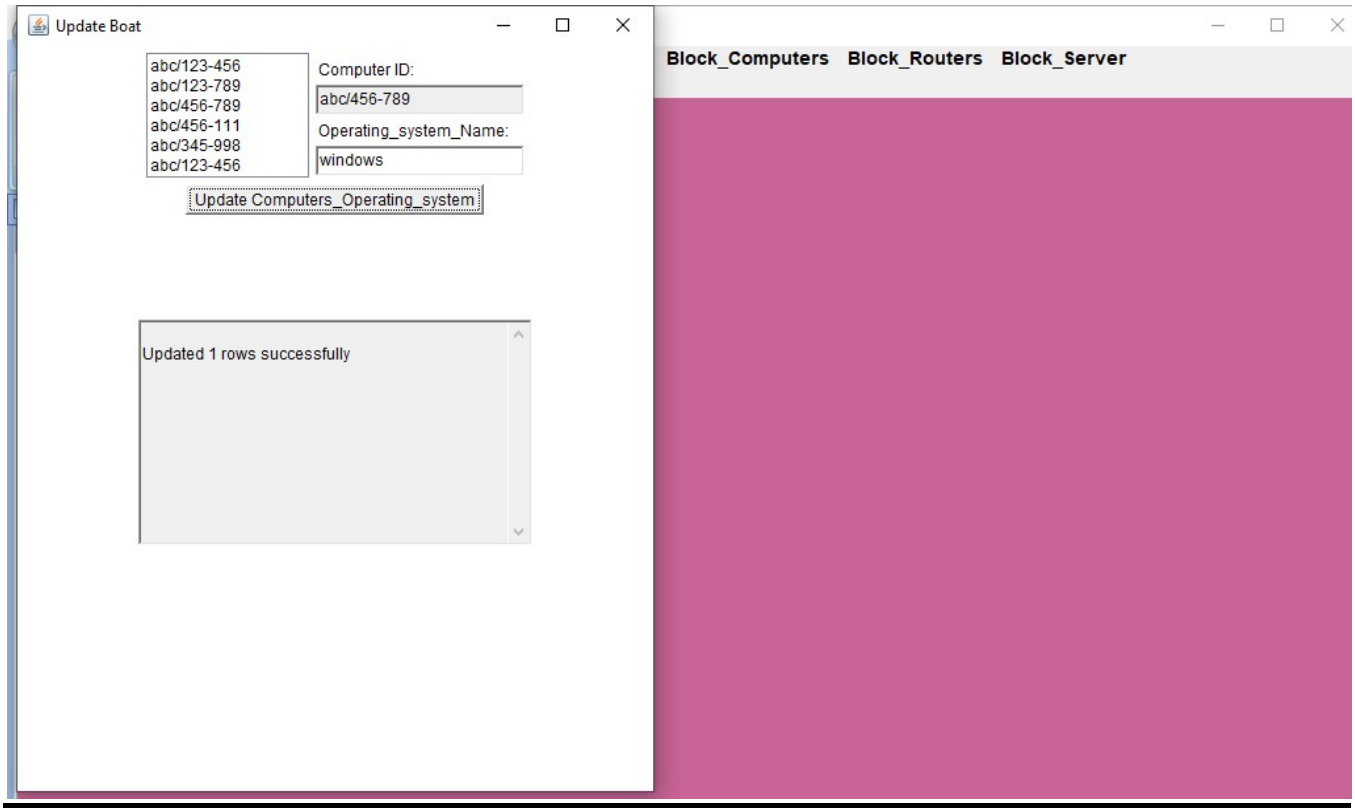
Roll no:1602-18-737-064
Name:A.Ruchita

Computer_Operating_system

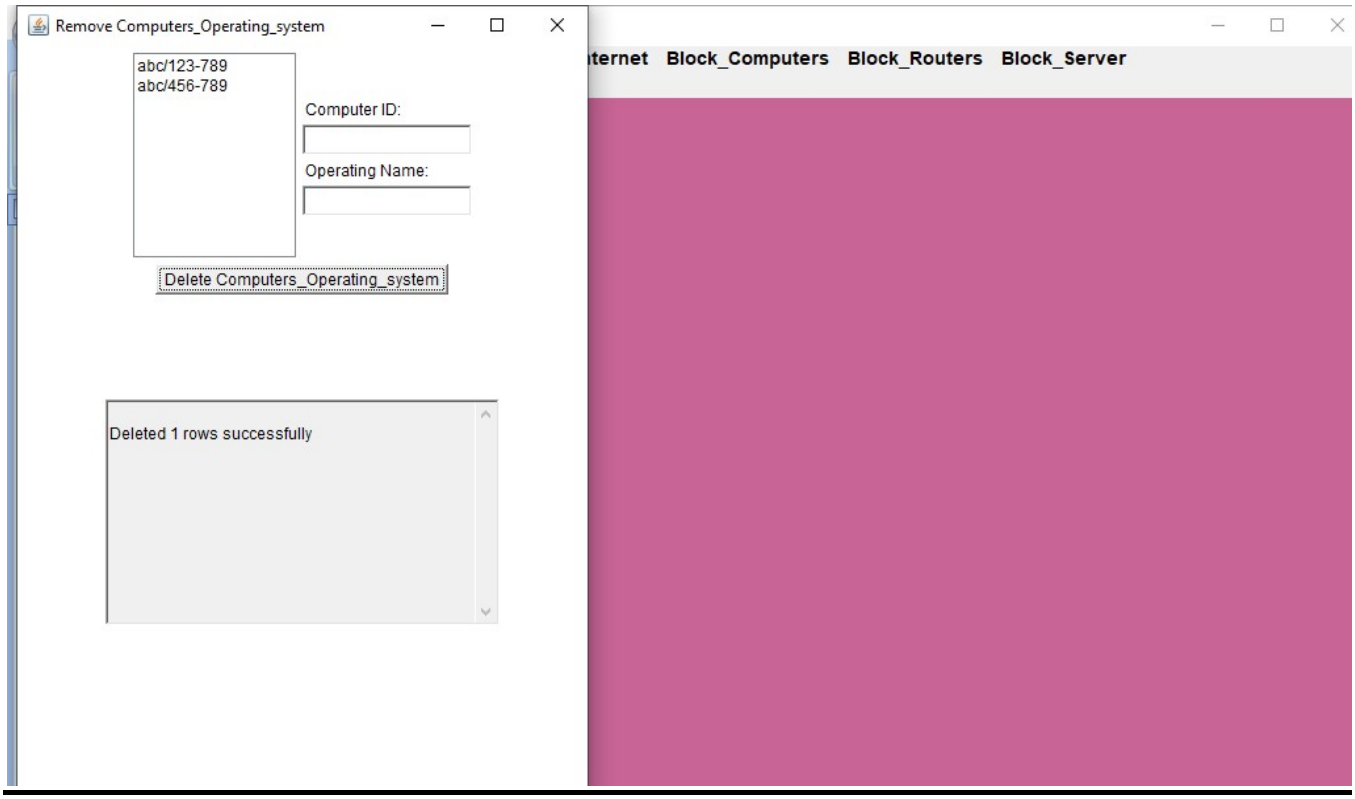
1)insert computer_operating_system



2)update computer_operating_system

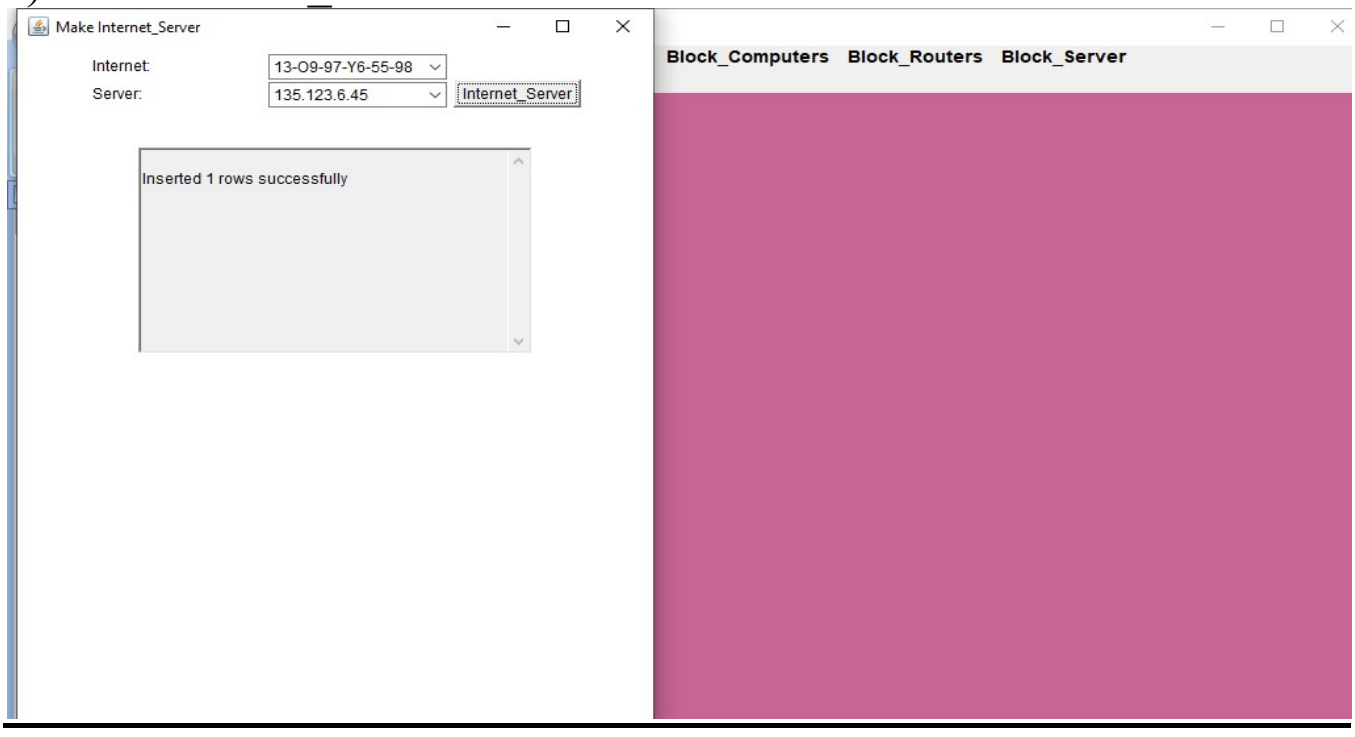


3)delete computer_operating_system

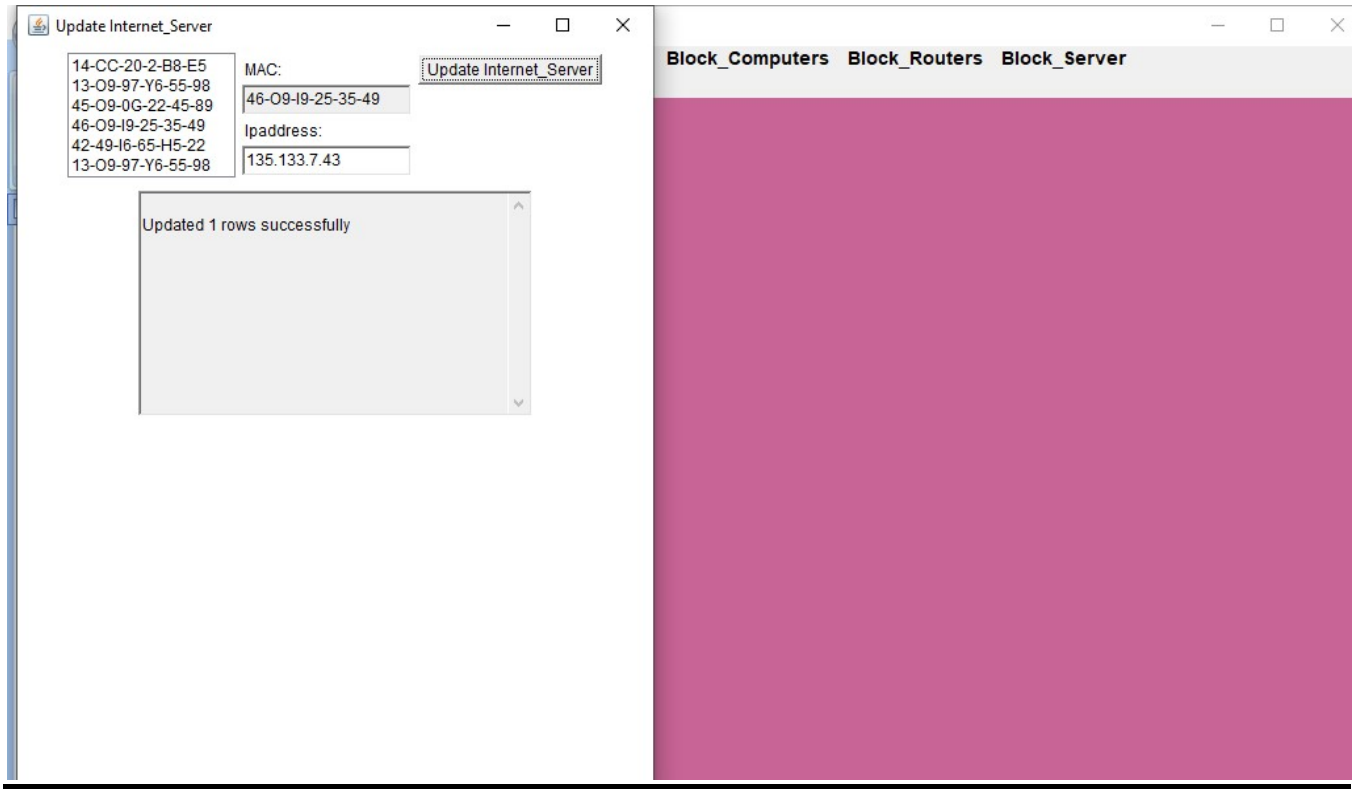


Internet_server

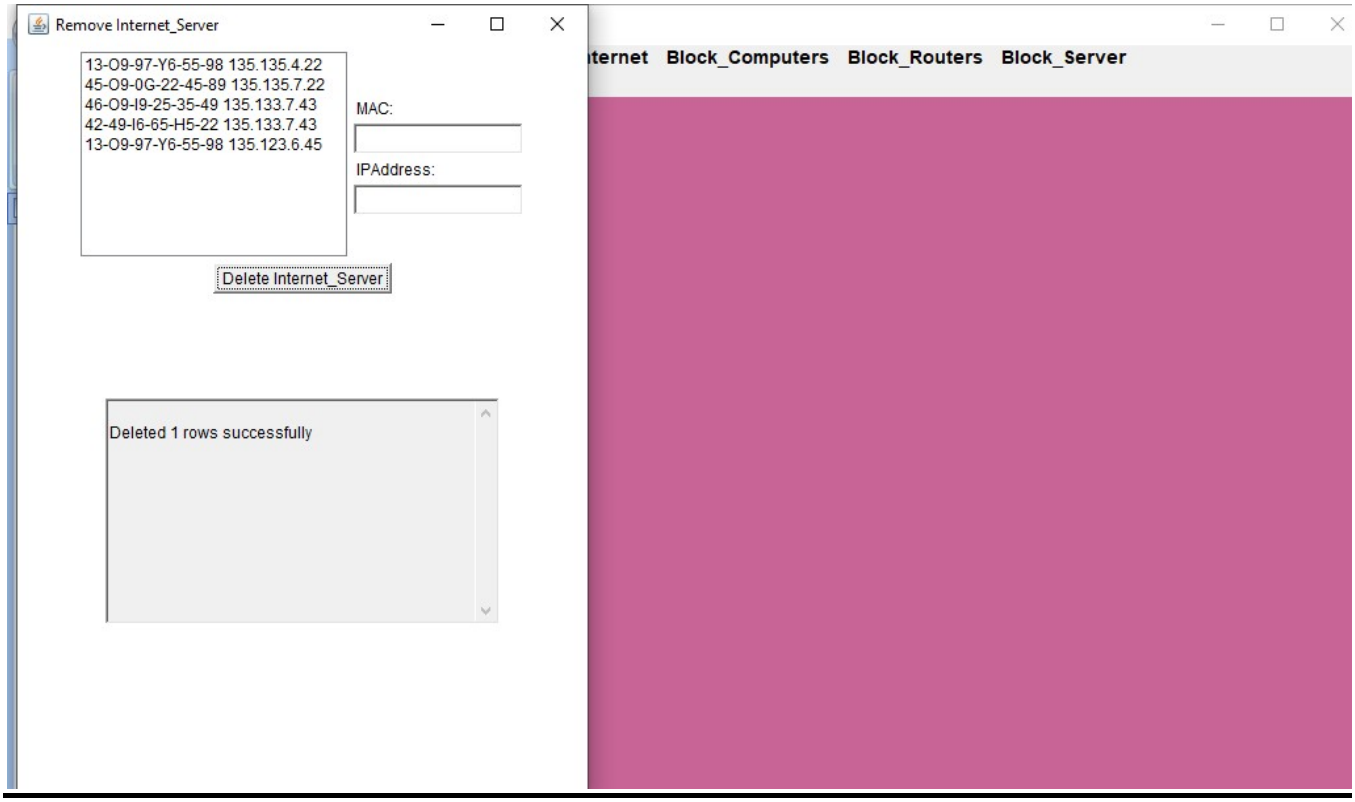
1)insert internet_server



2)update internet_server



3)delete internet_server



DISCUSSION & FUTURE WORK :

The application done till now is to store all the information related to the network connection of our college . Furthermore, other programming languages can also be used along with database by connecting SQL with it. This application can be extended further more to store network connections of other colleges, organizations etc

CONCLUSION:

Thus, a Java AWT based network connection management system is created which is connected to the Oracle 11g database. Therefore, all the entries in the form are directly updated on the network table created in the database.

REFERENCES :

https://www.oracle.com/technetwork/java/javase/documentation/index.html

https://nptel.ac.in/courses/106105175/

https://google.github.io/styleguide/javaguide.html

https://nptel.ac.in/courses/106105191/