JAVA AWT BASED- STOCK MARKET - 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

B.Harsha <1602-18-737-070>

Department of Information Technology

Vasavi College of Engineering (Autonomous)

Ibrahimbagh, Hyderabad-31

2020

BONAFIDE CERTIFICATE

This to Certify that the project report titled "Stock Market" project work of B.HARSHA bearing Roll.no: 1602-18-737-070 who carried out this project under my supervision in the IV semester for the academic year 2019-2020.

Signature *Signature*

external examiner ***internal examiner***

ABSTRACT

Stock analysis is the evaluation of a particular trading instrument,

an investment sector, or the market as a whole. Stock analysts attempt to determine the future activity of an instrument, sector, or market.

Stock analysis is a method for investors and traders to make buying and selling decisions. By studying and evaluating past and current data, investors and traders attempt to gain an edge in the markets by making informed decisions. When running stock analysis on a company's financial statements, an analyst will usually be checking for the measure of a company's profitability, liquidity, solvency, efficiency Stock analysis is the evaluation of a particular trading instrument, an investment sector, or the market as a whole. Stock analysts attempt to determine the future activity of an instrument, sector, or market.

Stock analysis is a method for investors and traders to make buying and selling decisions. By studying and evaluating past and current data, investors and traders attempt to gain an edge in the markets by making informed decisions. When running stock analysis on a company's financial statements, an analyst will usually be checking for the measure of a company's profitability, liquidity, solvency, efficiency

INTRODUCTION

➢ REQUIREMENTS FOR STOCK MARKET :

List of tables :

- SHAREHOLDERS
- COMPANIES
- HAVE
- CREATES
- PRODUCTS
- USER
- BUYS

List of attributes with their domain types:

Company: Company id:cid -Number(), Company name:cname-varchar() ,Price:-number(10) ,Capital: number(20)

Product: Item_id:number(5) ,Item_name: varchar(20), Item_type:varchar(15)

Shareholders: S_id:number(5), Gain/loss:varchar(5), S_name:varchar(10), Sharevalue:number(10), C_name:varchar2(5), C_id-number()

User: User_id:number(5), Item_id:number(5), User_name-varchar(15), Login_id:varchar(30), Password:varchar(20).

Have: Date: day-date

Creates: Date : day-date

Buys: Date : day-date

➢ SPECIFIC GOAL OF THE PROJECT:

The main goal to be achieved through this project was to provide a facility to Shareholders to add shares to their Account and Buyers to buy shares selectively.

This Application is used to Store data of various Shares of National Stock Exchange(NSE) and Bombay Stock Exchange(BSE).

➢ Architecture and technology used:

SQL Plus is the most basic Oracle Database utility with a basic command-line interface, commonly used by users, administrators and programmers.

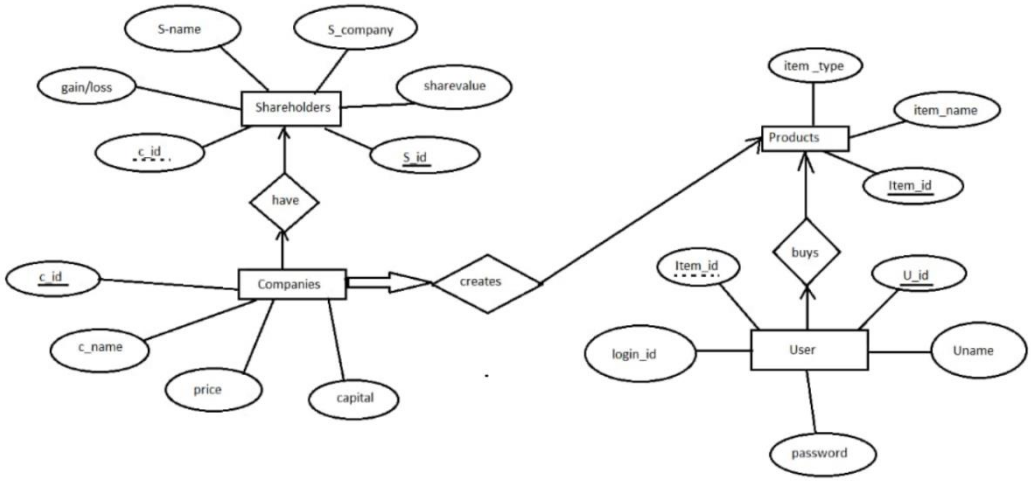
The interface of SQL Plus is used for creating the database. DDL and DML commands are implemented for operations being executed. The details of various Stocks,Products are stored in the form of tables in the database.

Eclipse 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. Eclipse is written mostly in java and its primary use is for developing Java applications, but it may also be used to develop applications in other programming languages via plug-ins, including Erlang, JavaScripts etc.

The front end application code is written in "**Java**" using Eclipse. The portal for front end application is designed through Eclipse, runs and has the capacity to connect with the database which has data inserted using SQL.

➢ DESIGN:

i)ER DIAGRAM:



DDL and DML operations are found in Stockmarket.pdf in the same repository.

Implementation

- JAVA Code : (ShareHolders Table)

1)Insert ShareHolder:

```
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class InsertShareholder extends Panel
{
	Button insertShareholderButton;
	TextField  snameText,sidText, glText, svText,cidText;
	TextArea errorText;
	Connection connection;
	Statement statement;
	public InsertShareholder()
	{
		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:orcl","it18737070","
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()
      {
            //Handle Insert Account Button
            insertShareholderButton = new Button("Insert Sailor");
            insertShareholderButton.addActionListener(new ActionListener()
            {
                  public void actionPerformed(ActionEvent e)
                  {
                        try
                        {
                          Statement statement = connection.createStatement();
                          String query= "INSERT INTO shareholders VALUES(" +
"'" + snameText.getText() + "'," + sidText.getText() + ", '" + glText.getText() +
"'," + svText.getText() + cidText.getText()+")";
                          int i = statement.executeUpdate(query);
                          errorText.append("\nInserted " + i + " rows
successfully");
                        }
                        catch (SQLException insertException)
                        {
                          displaySQLErrors(insertException);
                        }
                  }
            });


            sidText = new TextField(15);
            snameText = new TextField(15);
            glText = new TextField(15);
            svText = new TextField(15);
            cidText = new TextField(15);


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

            Panel first = new Panel();
            first.setLayout(new GridLayout(6, 2));
            first.add(new Label("ShareHolder ID:"));
            first.add(sidText);
            first.add(new Label("ShareHolder Name:"));
            first.add(snameText);
            first.add(new Label("Gain/Loss:"));
            first.add(glText);
            first.add(new Label("Share Value:"));
            first.add(svText);
            first.add(new Label("CID:"));
            first.add(cidText);
            first.setBounds(125,90,200,100);

            Panel second = new Panel(new GridLayout(4, 2));
            second.add(insertShareholderButton);
        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);

                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)
        {
                InsertShareholder sail = new InsertShareholder();

                sail.buildGUI();
        }
}
```

2)Update ShareHolder:

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

```
public class UpdateShareholder extends Panel
{
	Button updateShareholderButton;
	List shIDList;
	TextField sidText, snameText,glText, svText,cidText;
	TextArea errorText;
	Connection connection;
	Statement statement;
	ResultSet rs;

	public UpdateShareholder()
	{
		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:orcl","it18737070","
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 loadSailors()
	{
		try
		{
		  rs = statement.executeQuery("SELECT SID FROM shareholders");
		  while (rs.next())
		  {
		      shIDList.add(rs.getString("SID"));
		  }
		}
		catch (SQLException e)
		{
		  displaySQLErrors(e);
		}
	}
```

```
	public void buildGUI()
	{
	    shIDList = new List(10);
		loadSailors();
		add(shIDList);

		//When a list item is selected populate the text fields
		shIDList.addItemListener(new ItemListener()
		{
			public void itemStateChanged(ItemEvent e)
			{
				try
				{
					rs = statement.executeQuery("SELECT * FROM
shareholders where SID ="+shIDList.getSelectedItem());
					rs.next();
					sidText.setText(rs.getString("SID"));
					snameText.setText(rs.getString("SNAME"));
					glText.setText(rs.getString("GAIN_OR_LOSS"));
					svText.setText(rs.getString("SHAREVALUE"));
					cidText.setText(rs.getString("CID"));
				}
				catch (SQLException selectException)
				{
					displaySQLErrors(selectException);
				}
			}
		});


		//Handle Update Sailor Button
		updateShareholderButton = new Button("Update Sailor");
		updateShareholderButton.addActionListener(new ActionListener()
		{
			public void actionPerformed(ActionEvent e)
			{
				try
				{
					Statement statement =
connection.createStatement();
					int i = statement.executeUpdate("UPDATE
shareholders "
					+ "SET sname='" + snameText.getText() + "', "
					+ "GAIN_OR_LOSS='" + glText.getText() + "', "
					+ "SHAREVALUE ="+ svText.getText() + " WHERE sid
= "
					+ shIDList.getSelectedItem());
					errorText.append("\nUpdated " + i + " rows
successfully");
					shIDList.removeAll();
					loadSailors();
				}
				catch (SQLException insertException)
				{
					displaySQLErrors(insertException);
				}
			}
		});
```

```
            sidText = new TextField(15);
            sidText.setEditable(false);
            snameText = new TextField(15);
            glText = new TextField(15);
            svText = new TextField(15);
            cidText = new TextField(15);

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

            Panel first = new Panel();
            first.setLayout(new GridLayout(6, 2));
            first.add(new Label("Shareholder ID:"));
            first.add(sidText);
            first.add(new Label("Shareholder Name:"));
            first.add(snameText);
            first.add(new Label("Gain/Loss:"));
            first.add(glText);
            first.add(new Label("Share Value:"));
            first.add(svText);
            first.add(new Label("CID:"));
            first.add(cidText);

            Panel second = new Panel(new GridLayout(6, 1));
            second.add(updateShareholderButton);

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

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

            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)
      {
            UpdateShareholder ups = new UpdateShareholder();


            ups.buildGUI();
      }
}
```

3)Delete ShareHolder:

```
import java.awt.*;
import java.awt.event.*;
import java.sql.*;
public class DeleteShareholder extends Panel
{
	Button deleteShareholderButton;
	List shIDList;
	TextField sidText, snameText,glText, svText,cidText;
	TextArea errorText;
	Connection connection;
	Statement statement;
	ResultSet rs;

	public DeleteShareholder()
	{
		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:orcl","it18737070","
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 loadSailors()
	{
		try
		{
		  rs = statement.executeQuery("SELECT * FROM shareholders");
		  while (rs.next())
		  {
			shIDList.add(rs.getString("SID"));
		  }
		}
		catch (SQLException e)
		{
		  displaySQLErrors(e);
		}
	}

	public void buildGUI()
	{
	    shIDList = new List(10);
		  loadSailors();
		  add(shIDList);

		  //When a list item is selected populate the text fields
		  shIDList.addItemListener(new ItemListener()
		  {
			public void itemStateChanged(ItemEvent e)
			{
				try
				{
					rs = statement.executeQuery("SELECT * FROM
shareholders");
					while (rs.next())
					{
						if
(rs.getString("SID").equals(shIDList.getSelectedItem()))
						break;
					}
					if (!rs.isAfterLast())
					{
						sidText.setText(rs.getString("SID"));
						snameText.setText(rs.getString("SNAME"));

	glText.setText(rs.getString("GAIN_OR_LOSS"));

	svText.setText(rs.getString("SHAREVALUE"));
						cidText.setText(rs.getString("CID"));
					}
				}
				catch (SQLException selectException)
				{
					displaySQLErrors(selectException);
				}
			}
		});


		  //Handle Delete Sailor Button
		  deleteShareholderButton = new Button("Delete Sailor");
```

```
		deleteShareholderButton.addActionListener(new ActionListener()
		{
			public void actionPerformed(ActionEvent e)
			{
				try
				{
					Statement statement =
connection.createStatement();
					int i = statement.executeUpdate("DELETE FROM
shareholders WHERE SID = "
							+ shIDList.getSelectedItem());
					errorText.append("\nDeleted " + i + " rows
successfully");
					sidText.setText(null);
					snameText.setText(null);
					glText.setText(null);
					svText.setText(null);
					cidText.setText(null);
					shIDList.removeAll();
					loadSailors();
				}
				catch (SQLException insertException)
				{
					displaySQLErrors(insertException);
				}
			}
		});

		sidText = new TextField(15);
		snameText = new TextField(15);
		glText = new TextField(15);
		svText = new TextField(15);
		cidText = new TextField(15);

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

		Panel first = new Panel();
		first.setLayout(new GridLayout(6, 2));
		first.add(new Label("Shareholder ID:"));
		first.add(sidText);
		first.add(new Label("Shareholder Name:"));
		first.add(snameText);
		first.add(new Label("Gain/Loss:"));
		first.add(glText);
		first.add(new Label("Share Value:"));
		first.add(svText);
		first.add(new Label("CID:"));
		first.add(cidText);

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

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

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

```
            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)
      {
            DeleteShareholder dels = new DeleteShareholder();
            dels.buildGUI();
      }
}
```

Main :

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

class StockmarketGUI extends Frame implements ActionListener
{
        String msg = "";
        Label ll;
        CardLayout cardLO;

        //Create Panels for each of the menu items, welcome screen panel and home
screen panel with CardLayout
        InsertShareholder sail;
        UpdateShareholder ups;
        DeleteShareholder dels;
        InsertUser insu;
        Panel home,welcome;

        StockmarketGUI()
        {
                cardLO = new CardLayout();

                //Create an empty home panel and set its layout to card layout
                home = new Panel();
                home.setLayout(cardLO);


                ll = new Label();
                ll.setAlignment(Label.CENTER);
                ll.setText("Welcome to Stock Market Application");
```

```
                    //Create welcome panel and add the label to it
                    welcome = new Panel();
                    welcome.add(ll);

                    //create panels for each of our menu items and build them with
respective components
                    sail = new InsertShareholder(); sail.buildGUI();
                    ups = new UpdateShareholder();  ups.buildGUI();
                    dels = new DeleteShareholder();  dels.buildGUI();
                    insu = new InsertUser(); insu.buildGUI();

                    //add all the panels to the home panel which has a cardlayout
                    home.add(welcome, "Welcome");
                    home.add(sail, "InsertShareholder");
                    home.add(ups, "UpdateShareholder");
                    home.add(dels, "DeleteShareholder");
                    home.add(insu, "InsertUser");

                    // add home panel to main frame
                    add(home);

                    // create menu bar and add it to frame
                    MenuBar mbar = new MenuBar();
                    setMenuBar(mbar);

                    // create the menu items and add it to Menu
                    Menu sh = new Menu("Shareholders");
                    MenuItem item1, item2, item3;
                    sh.add(item1 = new MenuItem("Insert Shareholder"));
                    sh.add(item2 = new MenuItem("View Shareholders"));
                    sh.add(item3 = new MenuItem("Delete Shareholder"));
                    mbar.add(sh);

                    Menu com = new Menu("Company");
                    MenuItem item4, item5, item6;
                    com.add(item4 = new MenuItem("Insert Company"));
                    com.add(item5 = new MenuItem("View Companies"));
                    com.add(item6 = new MenuItem("Delete Company"));
                    mbar.add(com);

                    Menu reserve = new Menu("Users");
                    MenuItem item7, item8, item9;
                    reserve.add(item7 = new MenuItem("Add User"));
                    reserve.add(item8 = new MenuItem("View Users"));
                    reserve.add(item9 = new MenuItem("Delete User"));
                    mbar.add(reserve);

                    Menu pro = new Menu("Products");
                    MenuItem item10, item11, item12;
                    pro.add(item10 = new MenuItem("Add Product"));
                    pro.add(item11 = new MenuItem("View Products"));
                    pro.add(item12 = new MenuItem("Delete Product"));
                    mbar.add(pro);

                    Menu act = new Menu("Action");
                    MenuItem item13, item14, item15;
                    act.add(item13 = new MenuItem("Buy"));
                    act.add(item14 = new MenuItem("Hold"));
```

```
                act.add(item15 = new MenuItem("Have"));
                mbar.add(act);

                // 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);
                item11.addActionListener(this);
                item12.addActionListener(this);
                item13.addActionListener(this);
                item14.addActionListener(this);
                item15.addActionListener(this);


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

                //Frame properties
                setTitle("Stockmarket");
                Color clr = new Color(200, 200, 200);
                setBackground(clr);
                setFont(new Font("SansSerif", Font.BOLD, 14));
                setSize(500, 600);
                setVisible(true);

        }

        public void actionPerformed(ActionEvent ae)
        {
              String arg = ae.getActionCommand();
              if(arg.equals("Insert Shareholder"))
              {
                    cardLO.show(home, "InsertShareholder");
         }

             else if(arg.equals("View Shareholders"))
             {
                    cardLO.show(home, "UpdateShareholder");
             }

             else if(arg.equals("Delete Shareholder"))
             {
                    cardLO.show(home, "DeleteShareholder");
             }
             else if(arg.equals("Insert User"))
             {
```

```
                cardLO.show(home, "InsertUser");
        }
            else if(arg.equals("View Users"))
            {
                cardLO.show(home, "UpdateUser");
            }

            else if(arg.equals("Delete User"))
            {
                cardLO.show(home, "DeleteUser");
            }

            else if(arg.equals("Insert Company"))
            {
                cardLO.show(home, "InsertCompany");
            }
            else if(arg.equals("View Companies"))
            {
                cardLO.show(home, "UpdateCompany");
            }
            else if(arg.equals("Delete Company"))
            {
                cardLO.show(home, "DeleteCompany");
            }
            else if(arg.equals("Insert Product"))
            {
                cardLO.show(home, "InsertProduct");
            }
            else if(arg.equals("View Products"))
            {
                cardLO.show(home, "UpdateProduct");
            }
            else if(arg.equals("Delete Product"))
            {
                cardLO.show(home, "DeleteProduct");
            }
            else if(arg.equals("Buy"))
            {
                cardLO.show(home, "Buy");
            }
            else if(arg.equals("Hold"))
            {
                cardLO.show(home, "Hold");
            }
            else if(arg.equals("Have"))
            {
                cardLO.show(home, "Have");
            }
        }
        public static void main(String ... args)
        {
                new StockmarketGUI();
        }
}
```

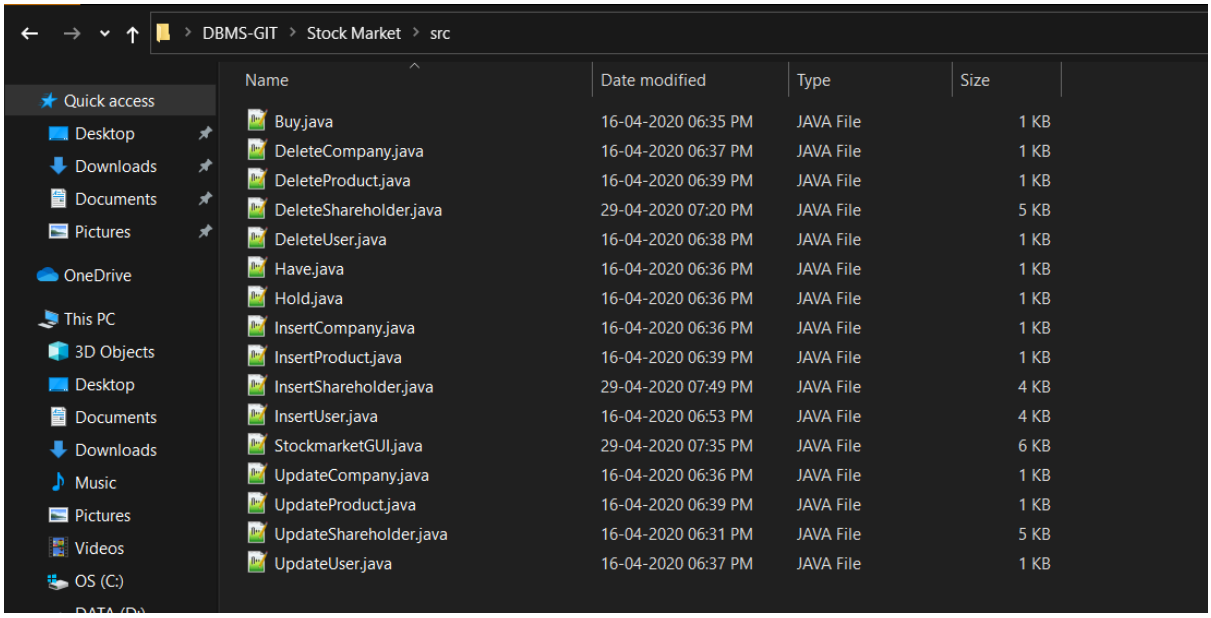
GITHUB LINK:

https://github.com/Lohith7/dbms_assingment

FOLDER STRUCTURE :

The Eclipse Project has a Folder named src having java files which will perform Insert, Update and Delete Operations.

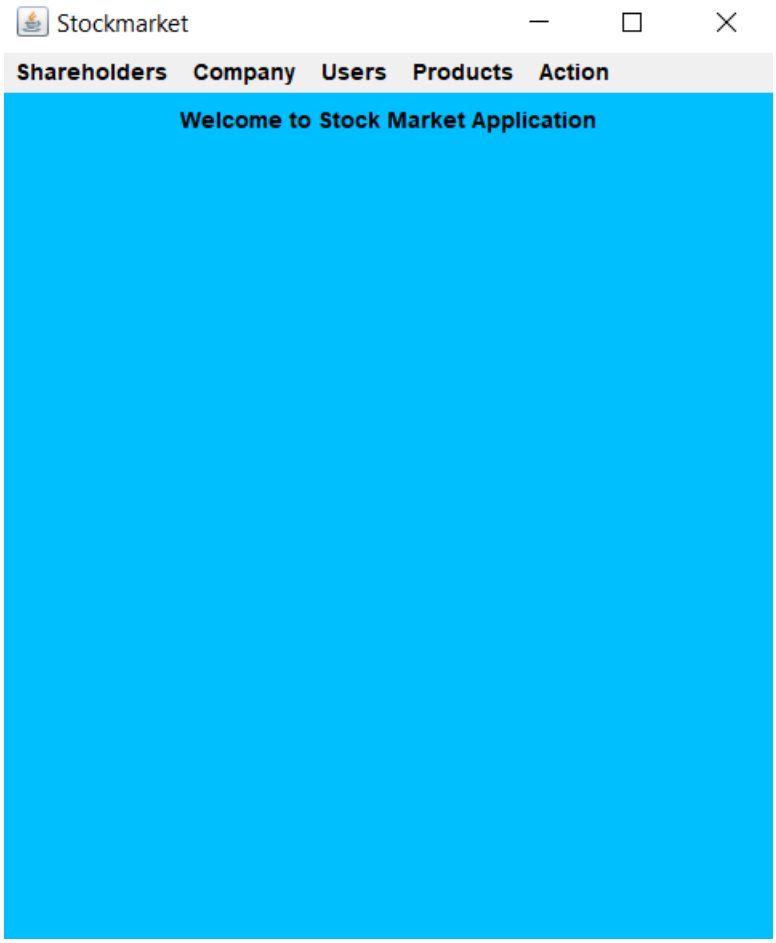
Navigate to src folder to find all files.



TESTING

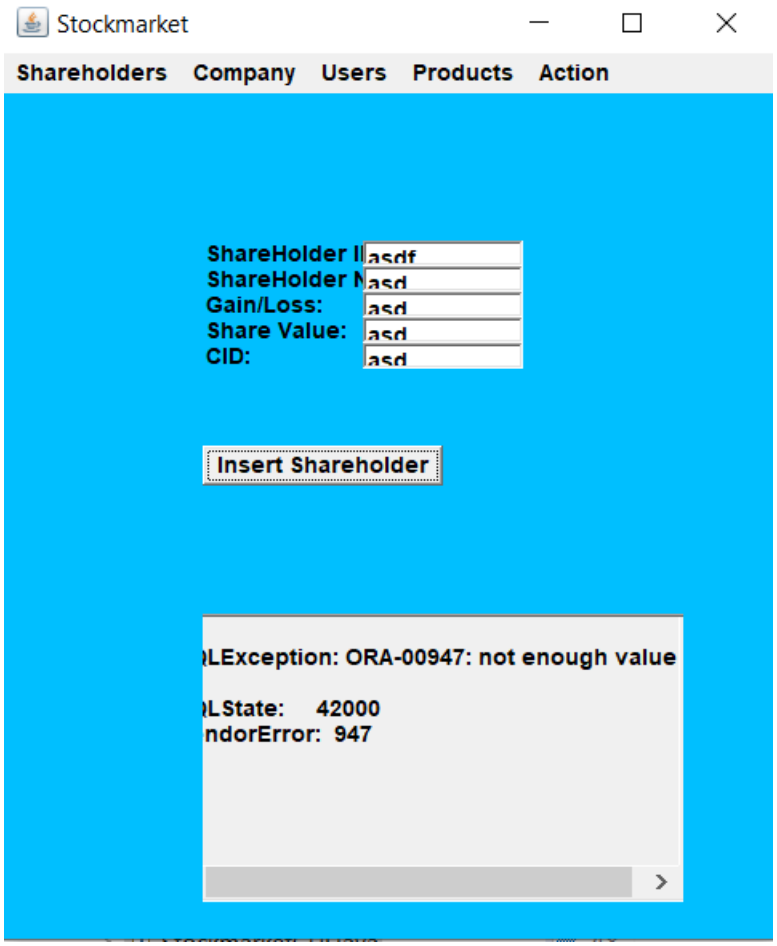
The program runs for execution of three basic operations of insertion, update and delete on 5 different table. Along with this, it also has a output column which gives the information about how many rows have been edited. Errors, syntactical or exceptional will be shown if occurred.

HOME PAGE:

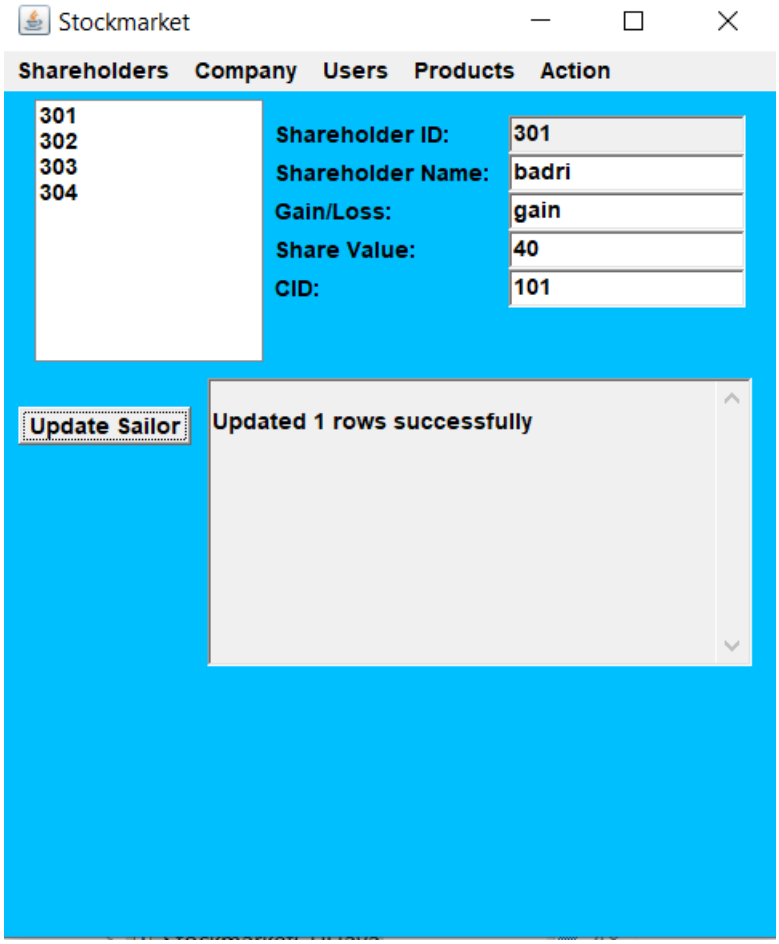


INSERT Shareholder:

Error Due to incorrect Domain type:



UPDATE ShareHolder :



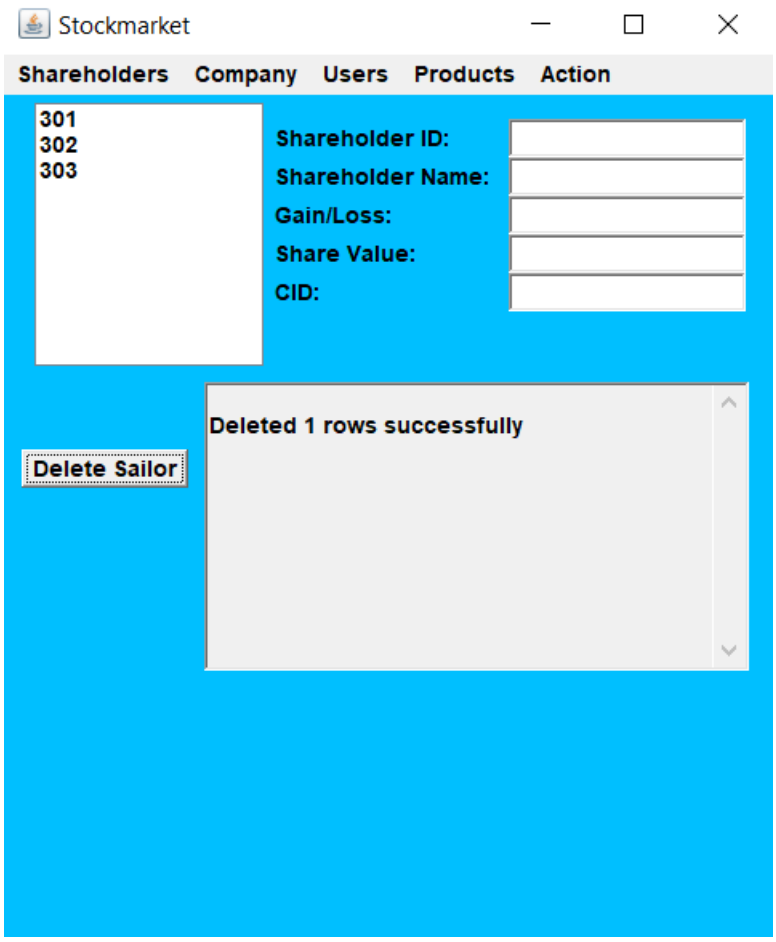
```
SQL> select * from shareholders;

SNAME                SID GAIN_OR_LOSS         SHAREVALUE        CID
-------------------- ---------- -------------------- ---------- ----------
harsha               301 gain                     40        101
teja                 302 loss                      5        102
abhi                 303 gain                     35        103
ajay                 304 loss                     20        104

SQL> select * from shareholders;

SNAME                SID GAIN_OR_LOSS         SHAREVALUE        CID
-------------------- ---------- -------------------- ---------- ----------
badri                301 gain                     40        101
teja                 302 loss                      5        102
abhi                 303 gain                     35        103
ajay                 304 loss                     20        104
```

DELETE ShareHolder:




```
SQL> select * from shareholders;

SNAME                          SID GAIN_OR_LOSS          SHAREVALUE         CID
-------------------- ---------- -------------------- ---------- ----------
badri                          301 gain                         40        101
teja                           302 loss                          5        102
abhi                           303 gain                         35        103
```

RESULTS

The DML commands, Insert, update and delete for one of the tables in given below:

For Shareholders table: (in java, as per the application)

Insert : `"INSERT INTO shareholders VALUES(" + "'" + snameText.getText() + "'," + sidText.getText() + ", '" + glText.getText() + "'," + svText.getText() + cidText.getText()+")";`

Update :
```
"UPDATE shareholders "
                            + "SET sname='" + snameText.getText() +
"', "
                            + "GAIN_OR_LOSS='" + glText.getText() +
"', "
                            + "SHAREVALUE ="+ svText.getText() + "
WHERE sid = "

                            + shIDList.getSelectedItem());
```

Delete:
```
"DELETE FROM shareholders WHERE SID = "
                                        + shIDList.getSelectedItem());
```

CONCLUSION:

1.Connection with database is established.

2.The values which are given to GUI Application are reflected in Database.

REFERENCES

https://www.moneycontrol.com/stocksmarketsindia/

https://economictimes.indiatimes.com/markets/stocks

https://money.rediff.com/index.html?

https://www.nseindia.com/