Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering
Semester: (Spring, Year:2021), B.Sc. in CSE (Day/Eve)

Course Title: Object-oriented programming
Course Code: 202 Section: DB

Lab Project Name: Pizza Ordering System In JAVA

Student Details

Name		ID
1.	Md.Mehedi Hassan Nayeem	213902045
2.		
3.		

Submission Date : 03/01/2023
Teacher's Name : Dr. Muhammad Aminur Rahaman

[For Teachers use only: Don't Write Anything inside this box]

Lab Project Status

Marks: …… Signature: ..... Comments:

..... Date: .....

Table of Contents

Chapter 1

Introduction

1.1 Introduction :

Today I am going to create an application in the Java language. My Project name is **Pizza ordering System** . Here a pizza seller can manage all kinds of selling information in it. They can add different kinds of orders like pizza size , quantity , pizza price and total price . They can delete any order if they want . The details are also stored in the database immediately .

1.2 Design Goals/Objective :

My Objective is to build an application for a pizza ordering system. Where a pizza seller can manage all kinds of selling information in it. They can add different kinds of orders like pizza size , quantity , pizza price and total price . They can delete any order if they want . The details are also stored in the database immediately .

Chapter 2

Performance Evaluation

2.1 Designing an interface :

I am going to explain how the project works as seller point of view

2.1.1 Main Interface :

Figure 01 : Main Dashboard

➕ When the project runs this interface will popup in front of the seller . He can see

everything that he needs . The left column shows us the size of the pizzas . There are different types of pizza available here. In the middle column we can see a table that stores the data of customs chosen . and in the last and right size column shows us four things like quantity , payment, balance and total price . Those are updateable . In the last phase we can see three buttons , add button , print button and remove button.

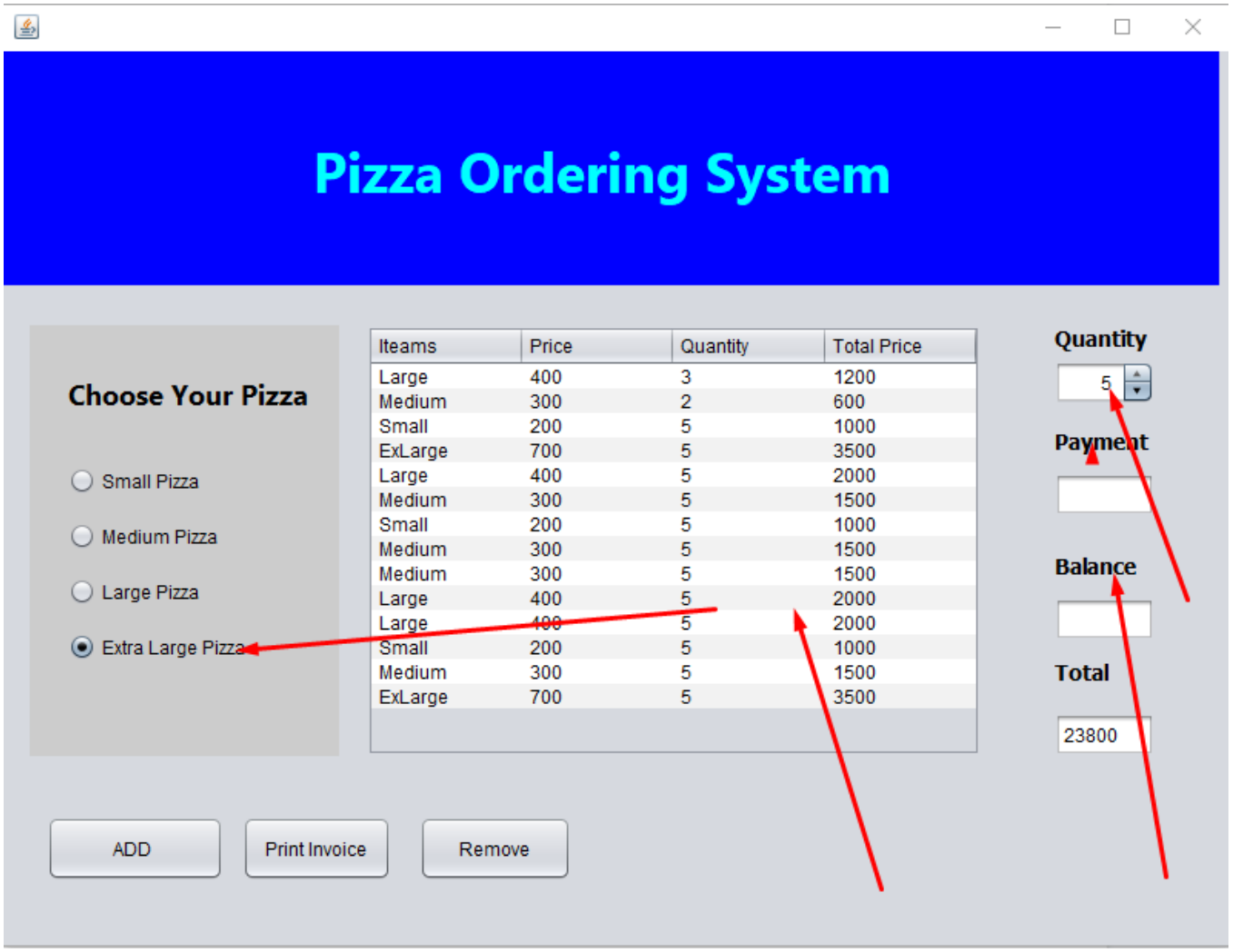


Figure 02 : Pizza added in the table

When the seller adds or chooses any size of pizza and gives the quantity of that and presses the add button , The item will automatically be added to the table . Sellers can submit payment that is received by the customer . The application automatically

calculates the amount he gives and what amount is due or the customer will receive .

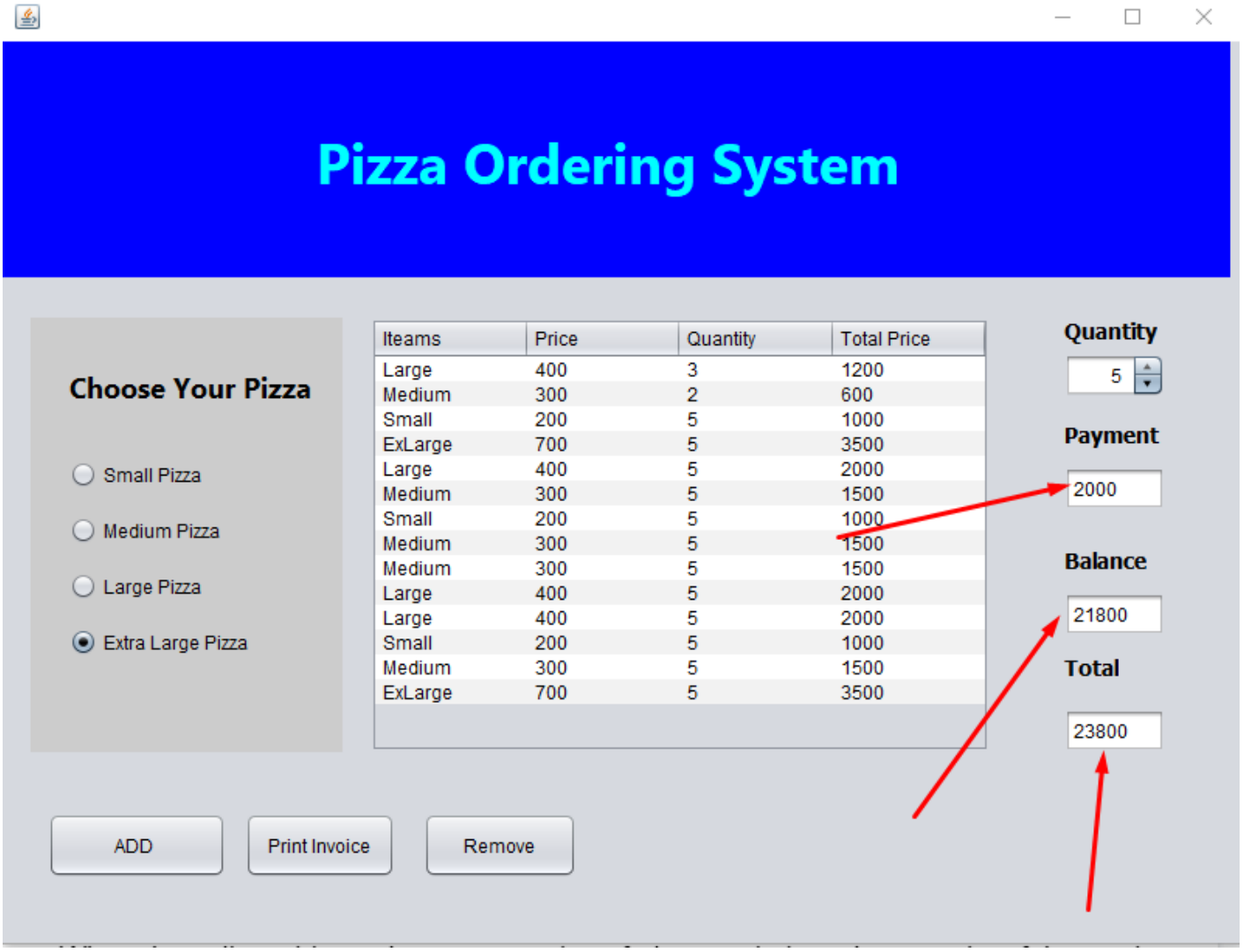


Figure 03 : calculate the payments

After pressing the print button the table information will be recorded in our database .

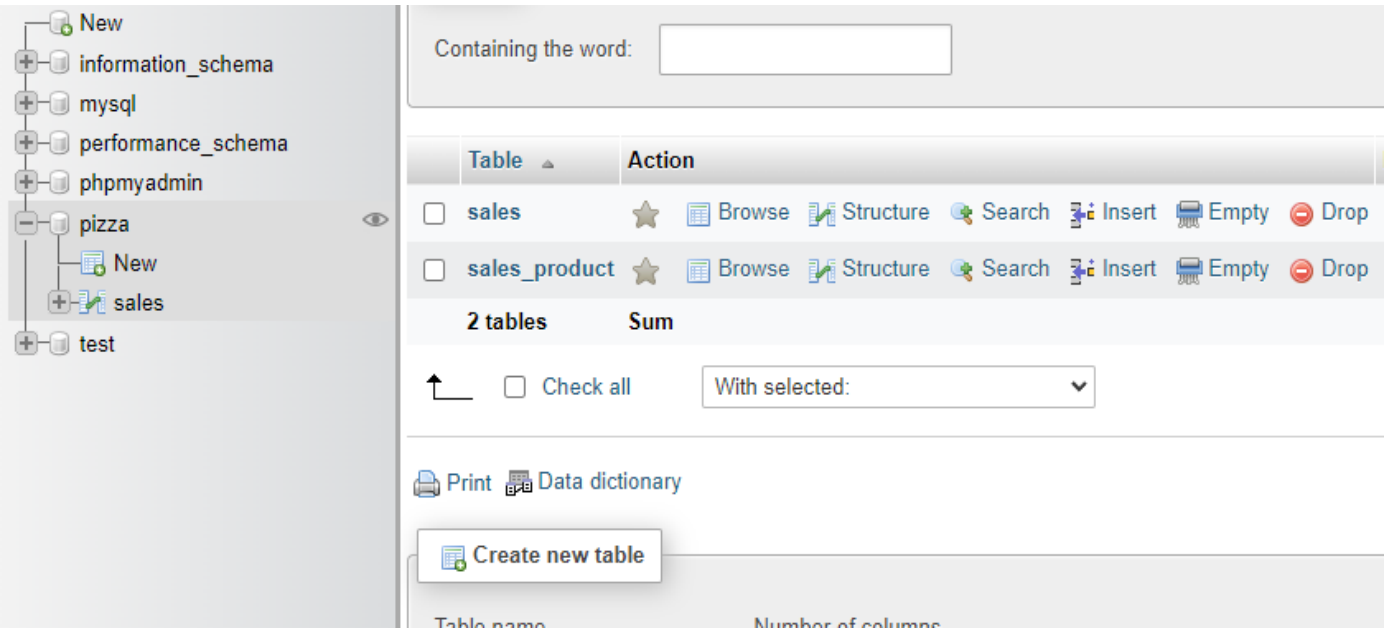
If we want to delete any row we just need to select the row and press the button .

Chapter 3
Implementation of the Project

3.1 Simulation Procedure :

Step 1: Create the database named with “pizza“. then established the database connection to download the mysql connector in order to connect Java & mysql. if you don't have an idea about downloading the mysql connector you can follow this link.

To connect the database please refer to the following code. To create the method name connect (); paste the below code inside the Connect () method.



```
DefaultTableModel model;
String item;
int price;
Connection con;
PreparedStatement pst;
PreparedStatement pst1;
public void Connect()
    {
        try {
            Class.forName("com.mysql.jdbc.Driver");
con = DriverManager.getConnection("jdbc:mysql://localhost/pizza","root","");
        } catch (ClassNotFoundException ex) {
            ex.printStackTrace();
        } catch (SQLException ex) {
           ex.printStackTrace();
        }
    }
```

After selecting the relevant pizza type the user has the option to add the qty by clicking the add button to see all Products details which will be shown in the below table.

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

```
 if(lblsmall.isSelected() == true)
        {
            item = "Small";
            price = 200;

        }
        else if(lblmed.isSelected() == true)
        {
            item = "Medium";
            price = 300;
```

```
        }
           else if(lbllarge.isSelected() == true)
        {
            item = "Large";
            price = 400;

        }
         else if(lblexlarge.isSelected() == true)
        {
            item = "ExLarge";
            price = 700;

        }

        int qty = Integer.parseInt(txtqty.getValue().toString());
        int tot = qty * price;

        model = (DefaultTableModel)jTable1.getModel();

        model.addRow(new Object[]

        {
            item,
            price,
            qty,
            tot

        });
        int sum = 0;

        for(int a=0; a<jTable1.getRowCount(); a++)
        {
            sum = sum + Integer.parseInt(jTable1.getValueAt(a,
3).toString());
        }
        txttotal.setText(Integer.toString(sum));

}
```

After that create the method Sales().we have to create two different tables to store data into the database.we also have to create the following tables from the database. sales tables consist of the following column – id,subtotal,pay,balance. sales_products tables consist of following column –id,sales_id,prodname,price,qty,total Call the Sales() method inside the Print Invoice button. after calculating the total Print receipt will be released.

Calculating the Total and balance:

```
public void sales()
    {
        int lastid =0;
        try {
            String total = txttotal.getText();
            String bal = txtbal.getText();
            String pay = txtpay.getText();
            String query = "insert into
sales(subtotal,pay,bal)values(?,?,?)";
            pst =
con.prepareStatement(query,Statement.RETURN_GENERATED_KEYS);
            pst.setString(1, total);
            pst.setString(2, pay);
             pst.setString(3, bal);
             pst.executeUpdate();
             ResultSet rs = pst.getGeneratedKeys();

            if(rs.next())
            {
                lastid = rs.getInt(1);
            }
            int row = jTable1.getRowCount();
            String query1 = "insert into
sales_product(sales_id,prodname,price,qty,total)values(?,?,?,?,?)";
```

```
            pst1 = con.prepareStatement(query1);

            String prodname = "";
            int price;
            int qty;
            int tot = 0;

            for(int i=0; i<jTable1.getRowCount(); i++)
                    {
                        prodname = (String)jTable1.getValueAt(i, 0);
                         price = (int)jTable1.getValueAt(i, 1);
                         qty = (int)jTable1.getValueAt(i, 2);
                         tot = (int)jTable1.getValueAt(i, 3);

                         pst1.setInt(1,lastid);
                          pst1.setString(2,prodname);
                          pst1.setInt(3,price);
                          pst1.setInt(4,qty);
                          pst1.setInt(5,tot);
                          pst1.executeUpdate();

                    }

            JOptionPane.showMessageDialog(this, "Sales Complete");
            HashMap a = new HashMap();
            a.put("invo", lastid);

            try {
                JasperDesign jdesign =
JRXmlLoader.load("C:\\Users\\kobinath\\Documents\\NetBeansProjects\\
Pizza\\src\\Pizza\\report1.jrxml");
                JasperReport jreport =
JasperCompileManager.compileReport(jdesign);

                JasperPrint jprint =
JasperFillManager.fillReport(jreport, a, con);
                JasperViewer.viewReport(jprint);
```

```
            } catch (JRException ex) {

Logger.getLogger(pizza.class.getName()).log(Level.SEVERE, null, ex);
            }

        } catch (SQLException ex) {

Logger.getLogger(pizza.class.getName()).log(Level.SEVERE, null, ex);
        }

    }
```

Print Invoice button :

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:

        int tot = Integer.parseInt(txttotal.getText());
        int pay = Integer.parseInt(txtpay.getText());

        int bal = pay - tot;

        txtbal.setText(String.valueOf(bal));


        sales();


    }
```

3.2 Results and Discussions

3.2.1 Results :

Users can store their data and can show or download as pdf in their system. Data store in the database so that it is more useful for sellers . anytime they can check or restore their data.

3.2.2 Analysis and Outcome : I love this project very much. I faced some problems to complete this project, like connecting with a database and sending some data in it. I also use interfaces to make them more user friendly .

Chapter 4

Conclusion

After all, the project was really good. I have to improve my skill more to make this project more beautiful and more user friendly . There are a lot of places where I have to improve . I use interfaces , print invoice systems and add databases in my project.

References

[1] *https://www.tutussfunny.com/pizza-ordering-system-in-java-with-print-receipt/*