I have worked on a java project named 'Smart IT Cafeteria' as a part of my 3rd semester OOP course. It was a digitalized restaurant management system for IT Cafeteria of University of Chittagong to make ordering food, getting receipt and storing employees details easier, faster and smarter. Basically, my role was of a programmer in this project.

A detail report on this project is described below:

Introduction:

This project introduces us a new era of restaurant management. Simply, a digitalized way of serving the customer , counting cost of foods(including VAT), managing the information of employees. It also provides a smarter way to the customer to select their desired items from a digitalized menu , which makes our cafeteria one step smarter and easier to the authority and also to the customers.

Purpose of Our Project :

→ The main purpose of our project is to give a smarter and easier way to a restaurant manager to perform his duty perfectly and saving his valuable time .

→ Other purpose of our project is to make the customers feel comfort while selecting their meal and paying their bill.

→ To make the whole system fair. (as the cost are counted manually and digitally , their will be almost no question of corruption)

Project Requirements:

- Excellence in Modern Java.
- Enough knowledge in MySQL.

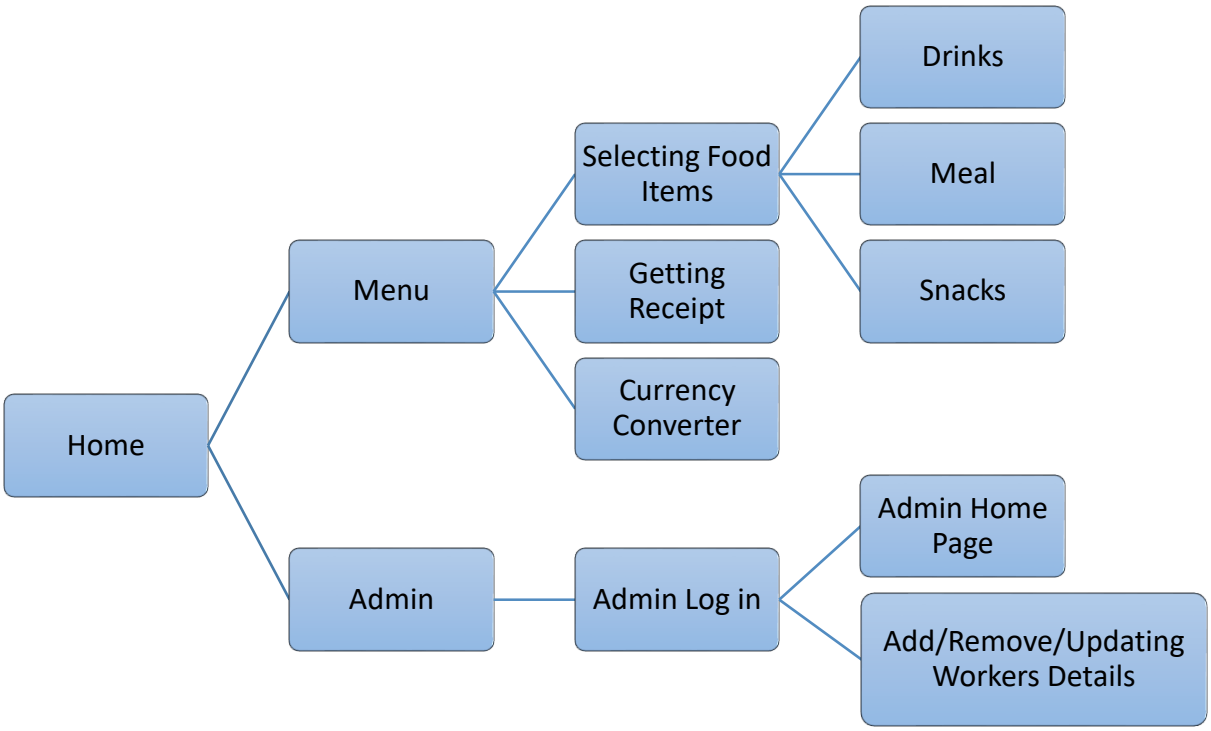
Tools Used:

1. Netbeans8.2 IDE

2.JDK 1.8.0

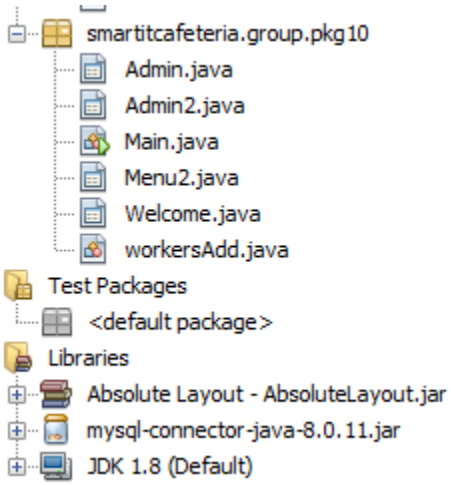
3.XAMPP for database's server

Project Diagram:



Description:

Our java project is made of one main class and four other class named “Welcome”, “Menu”, “Admin”, “Admin2” and “workersAdd”.



When the project is run, the “Main.java” class which is the main class make the “Welcome.java” class visible and we can see the home page on screen.

Home Page:

This is the “Welcome.java” class. This page includes a “Home” button which make the program stayed at home page, a “Menu” button which leads the program to “Menu.java” and shows the Menu Page, and an “Admin” button which leads the program to “Admin.java” and shows Admin Log In page.

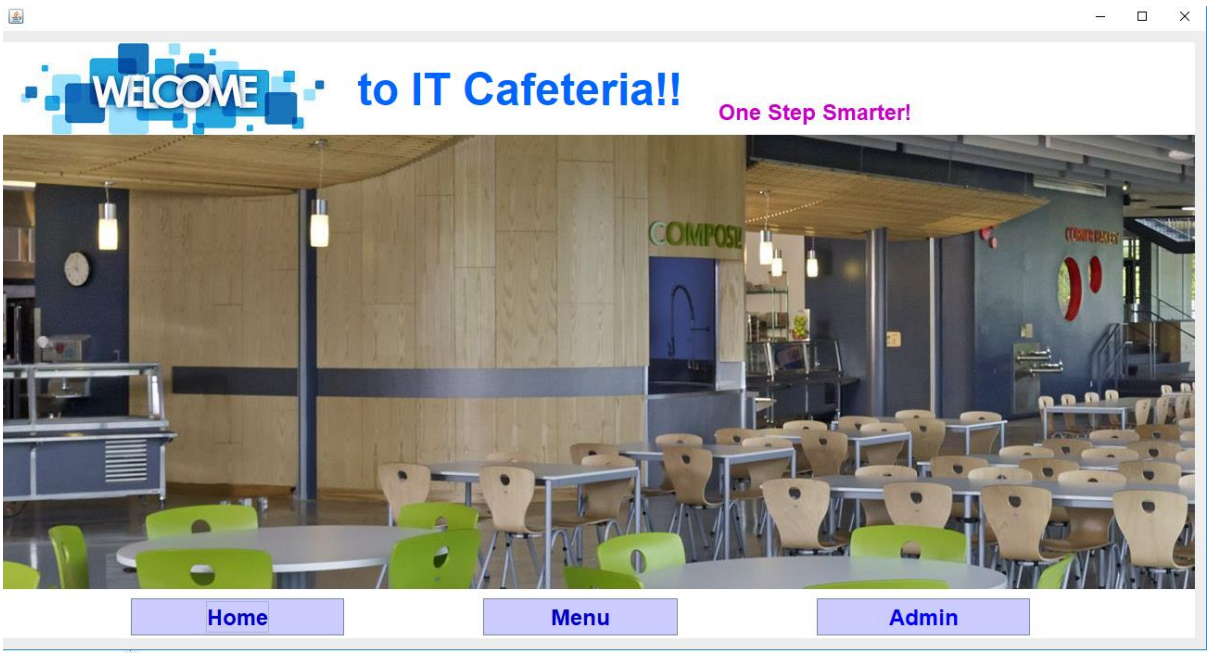


Figure : Screenshot of Home Page.

Sample Code of Home Page’s Buttons:

```
private void jBtnMenuActionPerformed(java.awt.event.ActionEvent evt) {

    Menu2 obj = new Menu2();
```

```
        obj.setVisible(true);
    }
```

Menu Page :

The menu page has food items on the left sided tabbed pane . When the user enterd the number of item in the text box and then select the food item it shows the total cost of that type of item in the Cost of Drinks/Meals/ Snacks box. After clicking the “Subtotal” button, the subtotal box shows the total cost of drinks, meals and snacks. Then if the user clicks “Tax” , 20% tax it will show how much tax is added and after that by clicking “Total” button, the user can see the total amount. On the right sided tabbed pane there are receipt and currency converter. After clicking the “Receipt” button, receipt is showed in the receipt pane. The currency converter helps the foreigner to pay money in USA or Canadian dollar by converting ammount of tk to dollar. The “Reset” button resets all the textboxes.

Figure: Screenshot of Menu Page.

Sample Code of Selecting Food Items:

```
private void Soft_DrinksMouseClicked(java.awt.event.MouseEvent evt) {
        double cdrink = Double.parseDouble(jTextCostofDrinks.getText());
        double soft_drinks = Double.parseDouble(jTextSoftDrinks.getText());
        double isoft_drinks = 15.00;
        Soft_Drinks.setSelected(true);
        if(Soft_Drinks.isSelected()) {
           i[5] = (soft_drinks*isoft_drinks);
           double p = i[5]+ cdrink;
           String pdrink = String.format("%.2f",p);
           jTextCostofDrinks.setText(pdrink);
        }
    }
```

Sample Code of Subtotal Button:

```
private void SubTotalMouseClicked(java.awt.event.MouseEvent evt) {
         double cTotal1 = Double.parseDouble(jTextCostofDrinks.getText());
        double cTotal2 = Double.parseDouble(jTextCostofMeal.getText());
        double cTotal3 = Double.parseDouble(jTextCostofSnacks.getText());
         double ctotalAll = cTotal1+cTotal2+cTotal3;
        String itotal = String.format("%.2f", ctotalAll);
        jTextSubTotal.setText(itotal);
    }
```

Sample Code of Receipt Button:

```
private void jReceiptActionPerformed(java.awt.event.ActionEvent evt) {
        int d1= Integer.parseInt(jTextTea.getText());
        int d2= Integer.parseInt(jTextCoffee.getText());
        int d3= Integer.parseInt(jTextLemonade.getText());
        // .......
        cost[0] = Double.parseDouble(jTextSubTotal.getText());
        cost[1] = Double.parseDouble(jTextTax.getText());
        cost[2] = Double.parseDouble(jTextTotal.getText());
        Calendar timer = Calendar.getInstance();
        timer.getTime();
        SimpleDateFormat tTime = new SimpleDateFormat("HH:MM:SS");
        SimpleDateFormat tDate = new SimpleDateFormat("dd-mm-yyyy");
        jTextAreaReceipt.append("\tChittagong University IT Cafetaria\n\n"+
                                "Reference:\t\t"+ refs+
                                "\n\nTea:\t"+d1+"p\t"+i[0]
                                +"\nCoffee:\t"+d2+"p\t"+i[1]
                                +"\nLemonade:\t"+d3+"p\t"+i[2]
                                +"\nOrange Juice:\t" +d4+"p\t"+i[3]
                                //.....
                                    +"\n\nSubtotal:\t\t"+cost[0]
                                    +"\nTax:\t\t"+cost[1]
                                    +"\n\nTotal:\t\t"+cost[2]
                                    + "\n\nDate:\t\t"+tTime.format(timer.getTime())
                                     +"\nTime:\t\t"+tDate.format(timer.getTime())+ "\n\nThank You");

    }
```

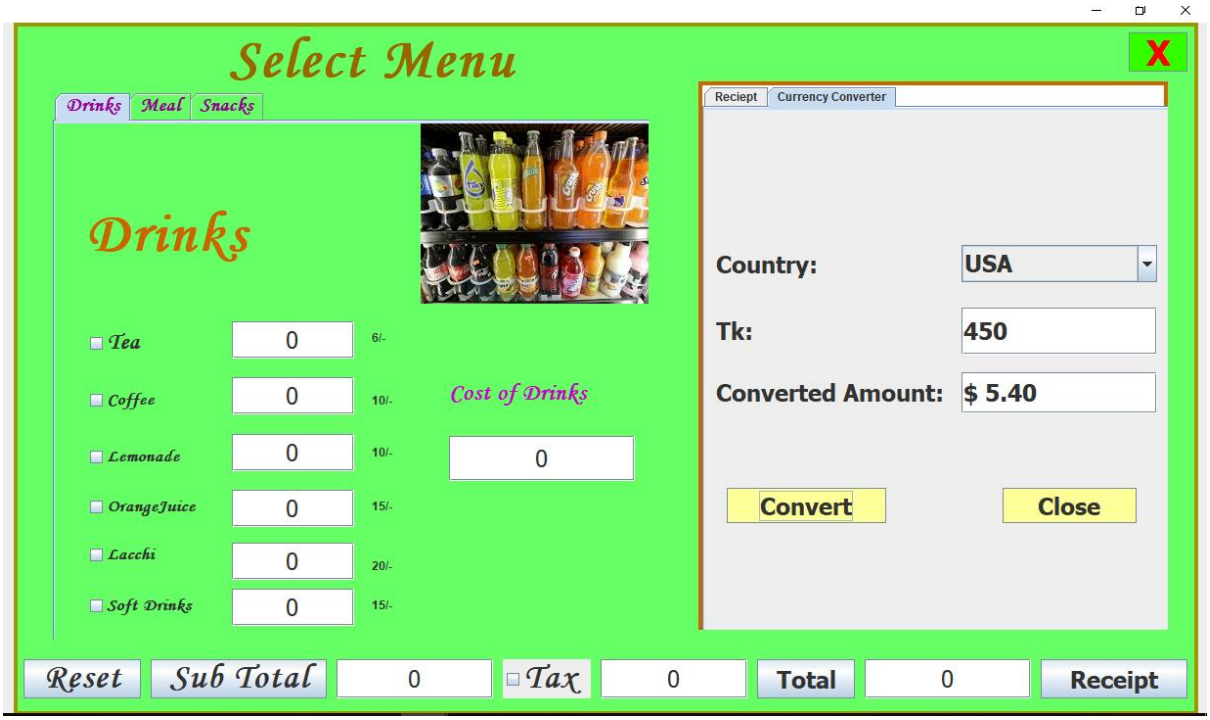


Figure: Currency Converting.

Sample Code of Convert Button of Currency Converter:

```
private void jConvertActionPerformed(java.awt.event.ActionEvent evt) {
    double Bangladeshi_tk = Double.parseDouble(jTextConvert.getText());
    if(jComboCurrency.getSelectedItem().equals("USA")) {
       String cConvert1 = String.format("$ %.2f", Bangladeshi_tk * US_Dollar);
       jTextConverted.setText(cConvert1);
    }
    if(jComboCurrency.getSelectedItem().equals("Canada")){
       String cConvert2 = String.format("$ %.2f", Bangladeshi_tk * Canadian_Dollar);
       jTextConverted.setText(cConvert2);
    }
    if(jComboCurrency.getSelectedItem().equals("Choose One")){
       jTextConverted.setText(null);
```

```
    }
}
```

Admin Log In Page:

There is username and password field in the admin log in page. If the username and password is matched , then after clicking the “Log In” button, an instance of “Admin2.java” is created and the admin home page is shown. The “Reset” button makes the username and passworl field null and the “Exit” button takes to project home page.

Figure: Admin log in Page.

Admin Home Page:

This is a tabbed pane of “Admin2.java” class. This page welcomes admin and also there is a log out button which leads admin to project home page.

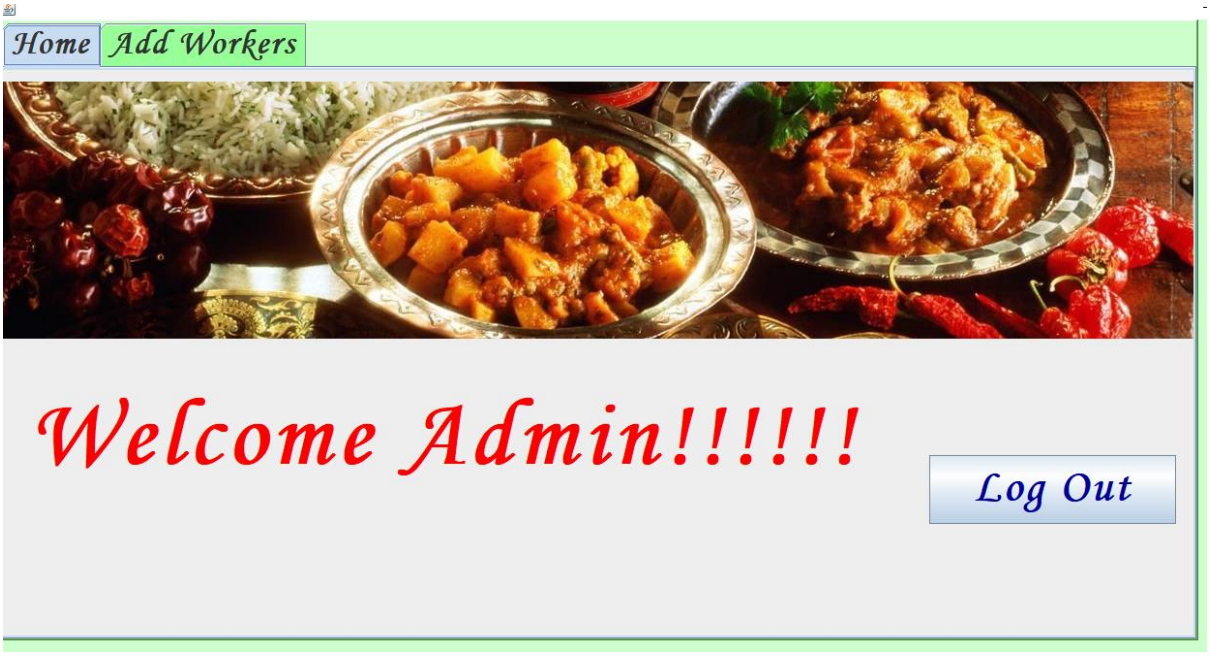


Figure : Admin Home Page.

The another tabbed pane of “Admin2.java” class is Add Workers Page. Here the admin can insert, update or delete worker’s details. For storing workers details er have used database using MYSQL language and XAMP software. There is a JTable frame which displays the workers details.

Add Workers Page:

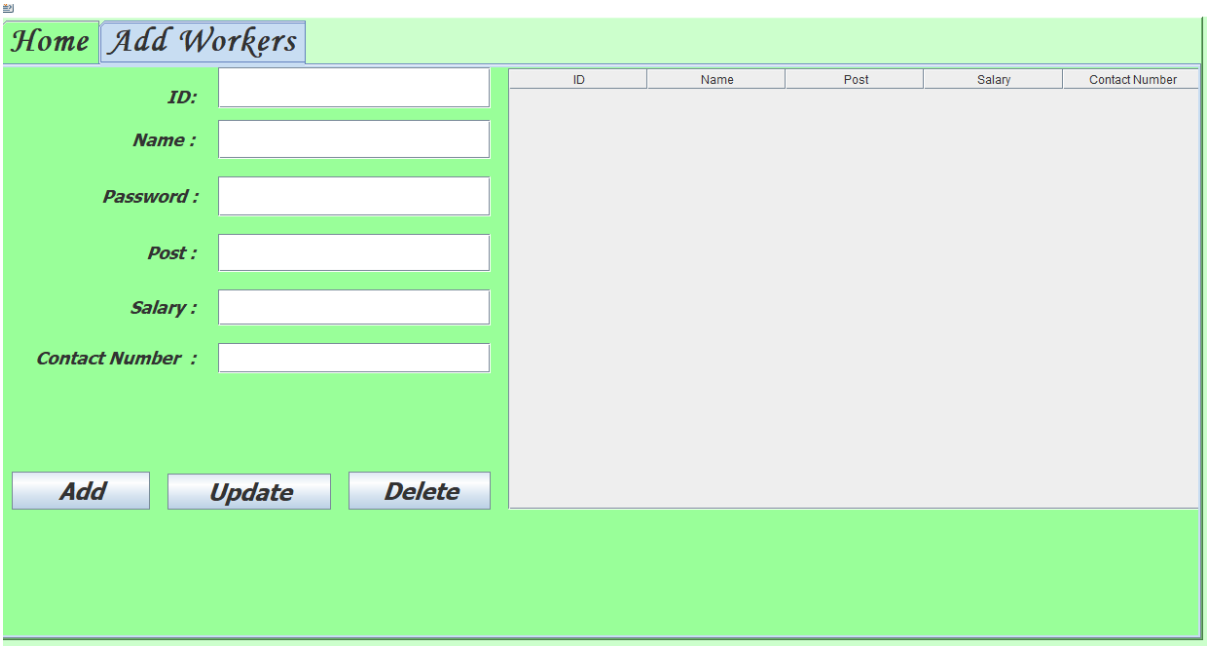


Figure: Add Workers Page

Sample Code for Database Connection:

```
public Connection getConnection(){
      Connection con = null;
       try{
         con = DriverManager.getConnection("jdbc:mysql://127.0.0.1/workersdb","root", "");
         return con;
      } catch (SQLException ex){
         Logger.getLogger(Admin2.class.getName()).log(Level.SEVERE,null,ex);
         return null;
      }
  }
```

Sample Code for Showing Worker's Details in JTable:

```
public ArrayList<workersAdd> getWorkersList(){
       ArrayList<workersAdd> workersList = new ArrayList<workersAdd>();
       Connection con = getConnection();
       String query = "SELECT * FROM workersdetails";
       Statement st;
       ResultSet rs;
       try{
          st = con.createStatement();
          rs = st.executeQuery(query);
          workersAdd workrsadd;
          while(rs.next()) {
```

```
            workrsadd = new workersAdd(rs.getInt("ID"), rs.getString("Name"),
rs.getString("Post"), Double.parseDouble(rs.getString("Salary")),
rs.getString("Contact_Number"));
            workersList.add(workrsadd);
        }
    }catch(SQLException ex){
        Logger.getLogger(Admin2.class.getName()).log(Level.SEVERE,null, ex);
    }
    return workersList;
  }
public void Show_workersdetails_in_JTable() {
    ArrayList<workersAdd> list = getWorkersList();
    DefaultTableModel model = (DefaultTableModel)jTable_workersdetails.getModel();
    model.setRowCount(0);
    Object[] row = new Object[5];
    for(int i=0; i< list.size(); i++){
        row[0] = list.get(i).getId();
        row[1] = list.get(i).getName();
        row[2] = list.get(i).getPost();
        row[3] = list.get(i).getSalary();
        row[4] = list.get(i).getContact();
        model.addRow(row);
    }
  }
```

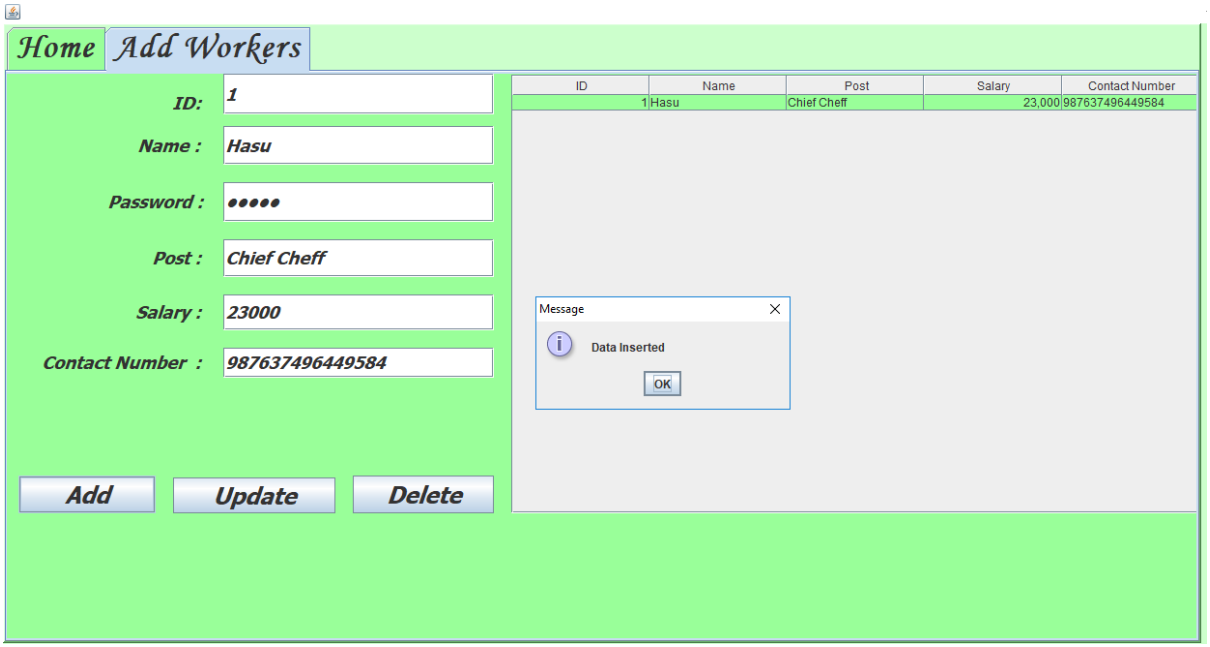


Figure: Inserting Data

Sample Code for Inserting Data:

```
public boolean checkInput(){
    if(jTextWID.getText()== null ||jTextWName.getText()== null ||
        jPasswordWPass.getText()== null ||jTextWPost.getText()== null ||
        jTextWSalary.getText()== null ||jTextWContactNumber.getText()== null ) {
      return false;
    }else {
      try{
        Float.parseFloat(jTextWSalary.getText());
        return true;
      }catch(Exception ex) {
        return false;
      }
    }
  }
```

```
private void jBtnAddActionPerformed(java.awt.event.ActionEvent evt) {

if(checkInput()){

        try{

           Connection con = getConnection();

           PreparedStatement ps = con.prepareStatement("INSERT INTO workersdetails(ID,
Name, Password, Post, Salary, Contact_Number) values(?,?,?,?,?,?)");

           ps.setInt(1, Integer.parseInt(jTextWID.getText()));

           ps.setString(2, jTextWName.getText());

           ps.setString(3, jPasswordWPass.getText());

           ps.setString(4, jTextWPost.getText());

           ps.setDouble(5, Double.parseDouble(jTextWSalary.getText()));

           ps.setString(6, jTextWContactNumber.getText());

           ps.executeUpdate();

           Show_workersdetails_in_JTable();

           JOptionPane.showMessageDialog(null, "Data Inserted");

        }catch(Exception ex){

           JOptionPane.showMessageDialog(null, ex.getMessage());

        }

     }else

     {

        JOptionPane.showMessageDialog(null, "One or more field are empty");

     }

  }
```

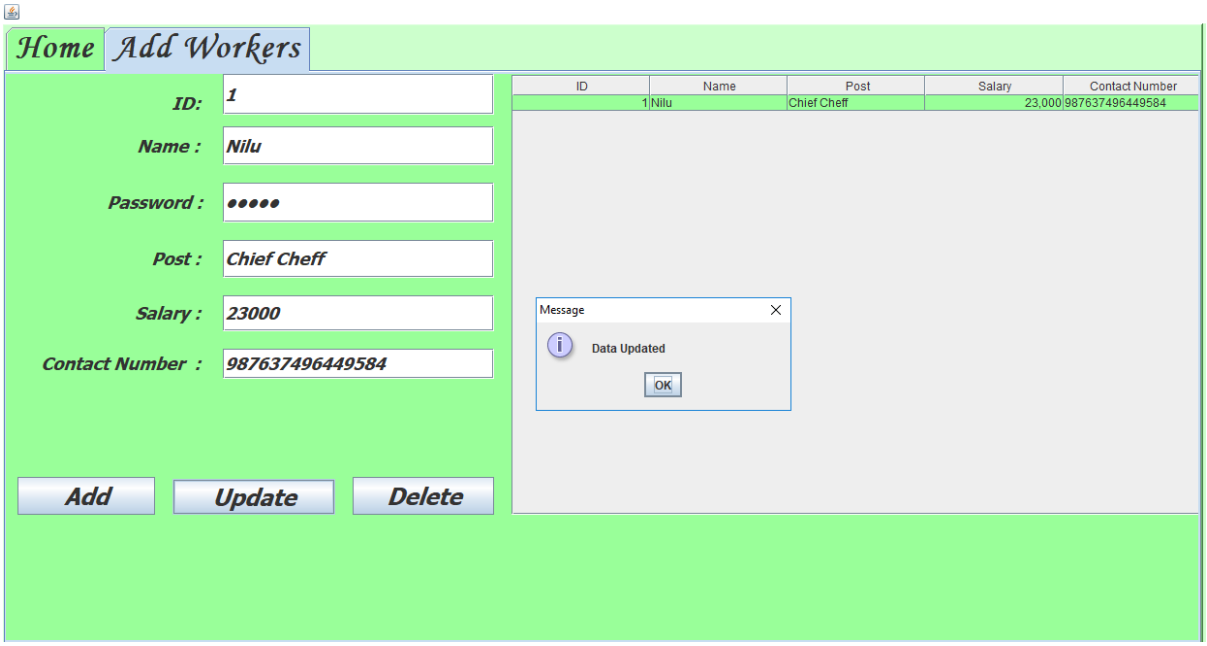


Figure: Data Updating

Sample Code of Updating Data:

```
private void jBtnUpdateActionPerformed(java.awt.event.ActionEvent evt) {
    if(checkInput() && jTextWID.getText()!= null) {
        String UpdateQuery = null;
        PreparedStatement ps = null;
        try{
            Connection con = getConnection();
            UpdateQuery = "UPDATE workersdetails SET Name = ?, Password = ?, Post = ?,  Salary = ?, Contact_Number= ? WHERE ID = ?  ";
            ps = con.prepareStatement(UpdateQuery);
            ps.setString(1, jTextWName.getText());
            ps.setString(2, jPasswordWPass.getText());
            ps.setString(3, jTextWPost.getText());
            ps.setDouble(4, Double.parseDouble(jTextWSalary.getText()));
            ps.setString(5, jTextWContactNumber.getText());
```

```
            ps.setInt(6, Integer.parseInt(jTextWID.getText()));
            ps.executeUpdate();
            Show_workersdetails_in_JTable();

            JOptionPane.showMessageDialog(null, "Data Updated");
        }catch(Exception ex){
            Logger.getLogger(Admin2.class.getName()).log(Level.SEVERE,null,ex);
        }
    }else {
        JOptionPane.showMessageDialog(null,"One or more fields are empty or wrong");
    }
}
```

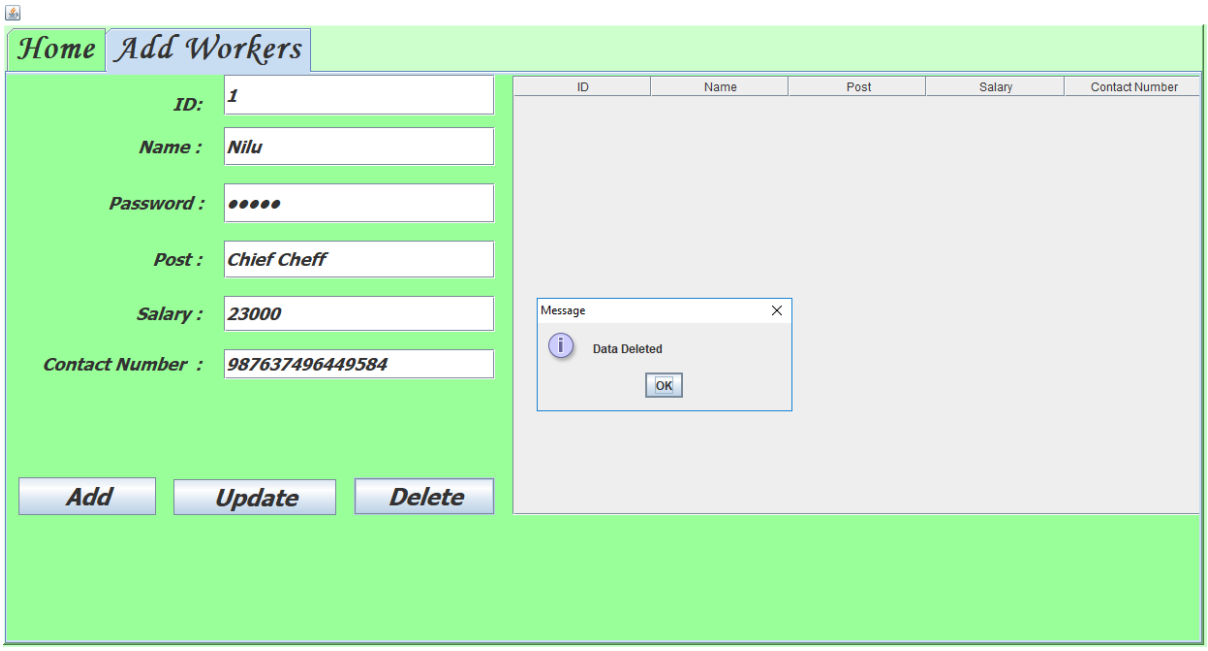


Figure: Data Deleting.

Sample Code of Deleting Data:

```
private void jBtnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
      if(!jTextWID.getText().equals("")){
         try{
            Connection con = getConnection();
            PreparedStatement ps = con.prepareStatement("DELETE FROM workersdetails WHERE
ID = ?");
            int id = Integer.parseInt(jTextWID.getText());
            ps.setInt(1, id);
            ps.executeUpdate();
            Show_workersdetails_in_JTable();
            JOptionPane.showMessageDialog(null, "Data Deleted");
         }catch(SQLException ex){
            Logger.getLogger(Admin2.class.getName()).log(Level.SEVERE, null, ex);
            JOptionPane.showMessageDialog(null, "Data Not Deleted");
         }
      }else{
         JOptionPane.showMessageDialog(null, "Data Not Deleted: No ID To Delete ");
      }
   }
```

Database:

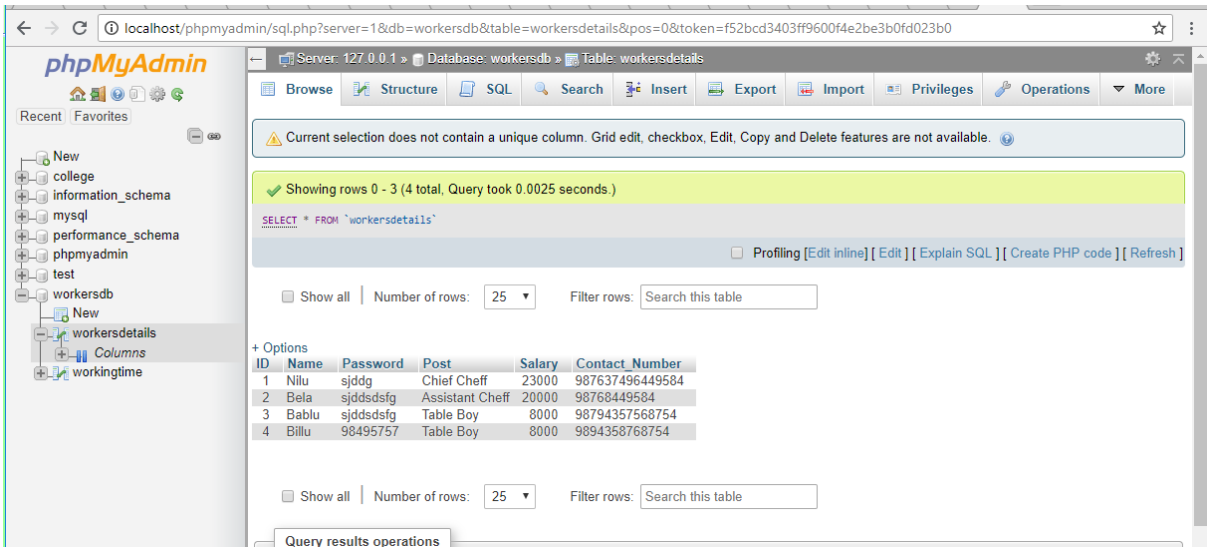


Figure: Employee's Database in XAMPP using MYSQL.

Advantages:

- Easy to use, little training needed.

- Decreased human error

- Automatic bill calculation

- Order is input to system as fast as possible

Disadvantages:

- Training costs

– Instillation and system costs

- System Maintenance

Application in Real Life:

The world is getting more digitalized day by day and Bangladesh is also walking on the way of digitalization. Digitalization makes our daily life easier, smoother an faster. So our restaurants and cafeterias should also be digitalized. Here, the advantages of the project are more than the

disadvantages. So all the restaurants and cafetaria can adapt this software for getting faster and easiar management system.

Future Aspects:

In future , our project could be further enhanced by providing following features :

1. The workers will be monitored by a smarter way. Their salary will be given by considering their exact working hour per day. After adding this feature we guarantee no workers will be able to play false with their work .

2. We will provide the customers the feature of home delivery.

3.Admin will also can add or remove food item from menu.

The inclusion of these feature would surely make our project highly demanding and would be grand success in practical field.

Conclusion:

It has been a matter of immense pleasure, honour and challenge to have this opportunity to take up this project and complete it successfully. While developing this project we've learned a lot about building a successful Java project and how to make it user friendly by hiding the complicated parts of it from the user.During the development process we studied carefully and understood the necessity of maintaining a minimal margin for error.