****

RAMAIAH

Institute of Technology

**“SuperMarket Billing System using Java”**

**A report submitted in partial fulfillment of the requirements**

**Of**

**Mini Project**

**In**

**Sixth Semester**

**By**

|  |  |
| --- | --- |
| **1MS16IS001** | **Abhijeet Saraf** |
| **1MS16IS012** | **Ankit Anand** |
|  |  |

**DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING**

**RAMAIAH INSTITUTE OF TECHNOLOGY**

(AUTONOMOUS INSTITUTE AFFILIATED TO VTU)

VIDYA SOUDHA

M. S. RAMAIAH NAGAR, M. S. R. I. T. POST, BANGALORE – 560054

**2018-2019**

**TABLE OF CONTENTS**

1. **ABSTRACT**
2. **INTRODUCTION**

2.1Overview

2.2Objective

2.3Future Scope

2.4Technologies Used

1. **IMPLEMENTATION**
2. **FLOWCHARTS AND TABLE SCHEMAS**
3. **RESULTS**

**ABSTRACT**

This project is regarding a supermarket billing system with some added functionality. This system is built for fast data processing and bill generation for supermarket customers. The billing system consists of MySQL database and effective front end designed. The billing database is a vast collection of product name, price and other product specific data. A product when billed is searched from the database and its price is added to the bill based upon the product quantity. The billing server simultaneously starts billing for the particular cart. In case the customer removes an item from the cart, the server immediately recognizes it and aligns the billing accordingly. Once the customer acknowledges the server to provide the bill, the server provides the gross amount. The customer on paying the bill is provided an acknowledgement by the server and the goods are dispatched to the customer by the time. The advantages of using such a system is that the time taken for billing huge amount of items is reduced greatly and it is very useful and secured in big shopping complexes. The supermarket billing system is built to help supermarkets calculate and display bills and serve the customer in a faster and efficient manner. This software project consists of an effective and easy Guide to help the employee in easy bill calculation and providing an efficient customer service.

**INTRODUCTION**

* 1. **Overview:-**

The project is on Supermarket Billing Software. Supermarket is the place where customers come to purchase their daily using products and pay the amount for that. So there is a need to calculate how many how many products are sold and to generate the bill for the customer. The aim is to automate its existing manual system by the help of computerized equipment and full-fledged computer software, fulfilling the requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. For easy and convenient processing we need to develop software, which work as per our requirements. While preparing this software we got to remember few things i.e. this software must have a friendly environment, in other words it should not be much complicated to handle, it should have option for future modification in the database i.e.in the goods list, new items and workers list. For smooth functioning we had to prepare a database by which we could enter the required data in their respective locations. In this database all the record of daily entry made are stored, so that we can use them in future whenever allotted to their caretaker. Other than that no one could use this and on entering an illegal password or login name an error message get displayed**.**

* 1. **Objective:-**

This project will serve the following objectives:-

1. Add and maintain records of available products.

2. Add and maintain customer details.

3. Add and maintain description of new products.

4. Add and maintain new entered category of products.

5. Provides economic/financial reports to the owner monthly or weekly and yearly.

6. Provides a convenient solution of billing pattern.

7. Make an easy to use environment for users and customers.

**1.3.Future Scope:-**

1. This project will help the store keeper in fast billing.

2. This project enable store keeper to maintain a great database of all customers visited

and purchase product from store.

3. Project will enable to see report regarding product and category.

4. Easy to maintain in future prospect.

**1.4. Technology Used:-**

1. MYSQL
2. JDBC
3. NETBEANS

**IMPLEMENTATION**

**MainPageCode:**

Package bs;

Import java.util.\*;

Import java.awt.Toolkit;

Import java.awt.event.WindowEvent;

public class MainPage extends javax.swing.JFrame {

public MainPage() {

init Components();

this.setLocation(300,100);

}

public void close(){

WindowEventwinClosingEvent=new WindowEvent(this,WindowEvent.WINDOW\_CLOSING);

Toolkit.getDefaultToolkit().getSystemEventQueue().postEvent(winClosingEvent);

}

private void jButton1ActionPerformed(java.awt.event.ActionEventevt) {

// TODO add your handling code here:

close();

AdminLogin al = new AdminLogin();

al.setVisible(true);

}

private void jButton2ActionPerformed(java.awt.event.ActionEventevt) {

close();

EmployeeLogin el = new EmployeeLogin();

el.setVisible(true);

}

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

newMainPage().setVisible(true);

}

});

}

**Admin Login Code:**

Import java.awt.event.\*;

Import java.awt.\*;

Import java.sql.\*;

Import java.util.logging.Level;

Import java.util.logging.Logger;

Import javax.swing.JOptionPane;

public class AdminLogin extends javax.swing.JFrame {

publicAdminLogin() {

init Components();

}

public void close(){

WindowEventwinClosingEvent=new WindowEvent(this,WindowEvent.WINDOW\_CLOSING);

Toolkit.getDefaultToolkit().getSystemEventQueue().postEvent(winClosingEvent);

}

private void jButton1ActionPerformed(java.awt.event.ActionEventevt) {

Connection con;

PreparedStatementps;

try{

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/alankar","root","");

ps=con.prepareStatement("SELECT `username`,`password` FROM `admin` WHERE `username`=? AND `password`=?;");

ps.setString(1,jTextField1.getText());

ps.setString(2,jPasswordField1.getText());

ResultSetrs= ps.executeQuery();

if(rs.next())

{

System.out.println("Login in successful");

close();

AdminDash ad =new AdminDash();

ad.setVisible(true);

}

else

{

JOptionPane.showMessageDialog(null,"Invalid Username or Password");

}

}

catch (SQLException ex) {

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

}

}

private void jCheckBox2ActionPerformed(java.awt.event.ActionEventevt) {

if(jCheckBox2.isSelected())

{

jPasswordField1.setEchoChar((char)0);

}

else

{

jPasswordField1.setEchoChar('\*');

}

}

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

newAdminLogin().setVisible(true);

}

});

}

**CashierBill Code:**

package frames;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;

import javax.swing.table.TableModel;

public class addCashiers extends javax.swing.JInternalFrame {

public addCashiers() {

init Components();

show\_list();

}

private void jButton1ActionPerformed(java.awt.event.ActionEventevt) {

int id = Integer.parseInt(jTextField1.getText());

String name =jTextField2.getText();

String mno = jTextField3.getText();

String address = jTextField5.getText();

String email = jTextField4.getText();

String password = jPasswordField2.getText();

Connection con = getConnection();

String query="INSERT INTO `cashier`(`id`,`name`,`mno`,`address`,`email`,`password`) VALUES('"+jTextField1.getText()+"','"+jTextField2.getText()+"','"+jTextField3.getText()+"','"+jTextField5.getText()+"','"+jTextField4.getText()+"','"+jPasswordField2.getText()+"');";

try {

executeSQLQuery(query,"inserted");

} catch (SQLException ex) {

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

}

}

private void jButton2ActionPerformed(java.awt.event.ActionEventevt) {

// TODO add your handling code here:

//update record here

//id`,`name`,`mno`,`address`,`email`,`password`

String query="UPDATE `cashier` SET `name`='"+jTextField2.getText()+"',`mno`='"+jTextField3.getText()+"',`address`='"+jTextField5.getText()+"',`email`='"+jTextField4.getText()+"',`password`='"+jPasswordField2.getText()+"' WHERE `id`="+Integer.parseInt(jTextField1.getText());

try {

executeSQLQuery(query,"update");

} catch (SQLException ex) {

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

}

}

private void jButton3ActionPerformed(java.awt.event.ActionEventevt) {

// TODO add your handling code here:

String query="DELETE FROM `cashier` WHERE `id`="+jTextField1.getText();

try {

executeSQLQuery(query,"Delete");

} catch (SQLException ex) {

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

}

}

public Connection getConnection()

{

Connection con = null;

try{

Class.forName("com.mysql.jdbc.Driver");

con = DriverManager.getConnection("jdbc:mysql://localhost:3306/alankar","root","");

return con;

}

catch(SQLException e)

{

System.out.println(e);

} catch (ClassNotFoundException ex) {

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

}

return con;

}

publicArrayList<Cash>getList() throws SQLException

{

ArrayList<Cash>ans = new ArrayList<Cash>();

ResultSetrs;

Statement st;

try{

String query = "Select \* from cashier";

Connection con = getConnection();

st = con.createStatement();

rs = st.executeQuery(query);

while(rs.next())

{

//intid,Stringname,Stringmno,Stringaddress,String email, String password

ans.add(new Cash(rs.getInt("id"),rs.getString("name"),rs.getString("mno"),rs.getString("address"),rs.getString("email"),rs.getString("password")));

}

}

catch(Exception e)

{

System.out.println(e);

}

Return ans;

}

private void jTable1MouseClicked(java.awt.event.MouseEventevt) {

int i=jTable1.getSelectedRow(); // TODO add your handling code here:

TableModel model=jTable1.getModel();

jTextField1.setText(model.getValueAt(i, 0).toString());

jTextField2.setText(model.getValueAt(i, 1).toString());

jTextField3.setText(model.getValueAt(i, 2).toString());

jTextField4.setText(model.getValueAt(i, 4).toString());

jTextField5.setText(model.getValueAt(i, 5).toString());

}

public void show\_list()

{

try {

ArrayList<Cash> list = getList();

DefaultTableModel model=(DefaultTableModel)jTable1.getModel();

Object[] row=new Object[6];

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).getMno();

row[3]=list.get(i).getPassword();

row[4]=list.get(i).getEmail();

row[5]=list.get(i).getAddress();

model.addRow(row);

}

} catch (SQLException ex) {

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

}

}

private void executeSQLQuery(String query, String message) throws SQLException {

Connection con = getConnection();

Statement st;

try{

st= con.createStatement();

if(st.executeUpdate(query)==1)

{

DefaultTableModel model = (DefaultTableModel)jTable1.getModel();

model.setRowCount(0);

show\_list();

jTextField1.setText("");

jTextField2.setText("");

jTextField3.setText("");

jTextField4.setText("");

jTextField5.setText("");

jPasswordField2.setText("");

JOptionPane.showMessageDialog(null,message);

}

else

{

JOptionPane.showMessageDialog(null,"error in requested action !!!");

}

}

catch(Exception e)

{

System.out.println(e);

}

}

}

**FLOWCHARTS AND TABLE SCHEMAS**

Request Information

CASHIER DATABASE

SERVER

ADMIN

Response Information

**Figure1** Cashier Registration

Information

Request

STOCK DATABASE

SERVER

ADMIN

Response Information

**Figure 2** Maintain Stock detail

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **FIELD** | **TYPE** | **NULL** | **KEY** | **DEFAULT** | **EXTRA** |
| Username | Varchar(20) | YES |  | NULL |  |
| Password | Varchar(20) | YES |  | NULL |  |

**Table 1** Admin Table Description

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **FIELD** | **TYPE** | **NULL** | **KEY** | **DEFAULT** | **EXTRA** |
| ID | Int(15) | NO | PRI | NULL | Auto increment |
| Name | Varchar(20) | YES |  | NULL |  |
| Mno | Varchar(20) | YES |  | NULL |  |
| Address | Varchar(20) | YES |  | NULL |  |
| Email | Varchar(20) | YES |  | NULL |  |
| Password | Varchar(20) | YES |  | NULL |  |

**Table 2** Cashier Table Description

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **FIELD** | **TYPE** | **NULL** | **KEY** | **DEFAULT** | **EXTRA** |
| ID | Int (15) | NO | PRI | NULL |  |
| PNAME | Varchar(20) | YES |  | NULL |  |
| QUANTITY | Int (15) | YES |  | NULL |  |
| MRP | Int (15) | YES |  | NULL |  |

**Table 3** Stock Table Description

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **FIELD** | **TYPE** | **NULL** | **KEY** | **DEFAULT** | **EXTRA** |
| Bno | Int(15) | NO | PRI | NULL | Auto increment |
| Cname | Varchar(20) | YES |  | NULL |  |
| Amt | Int(15) | YES |  | NULL |  |
| Pdate | Date | YES |  | NULL |  |

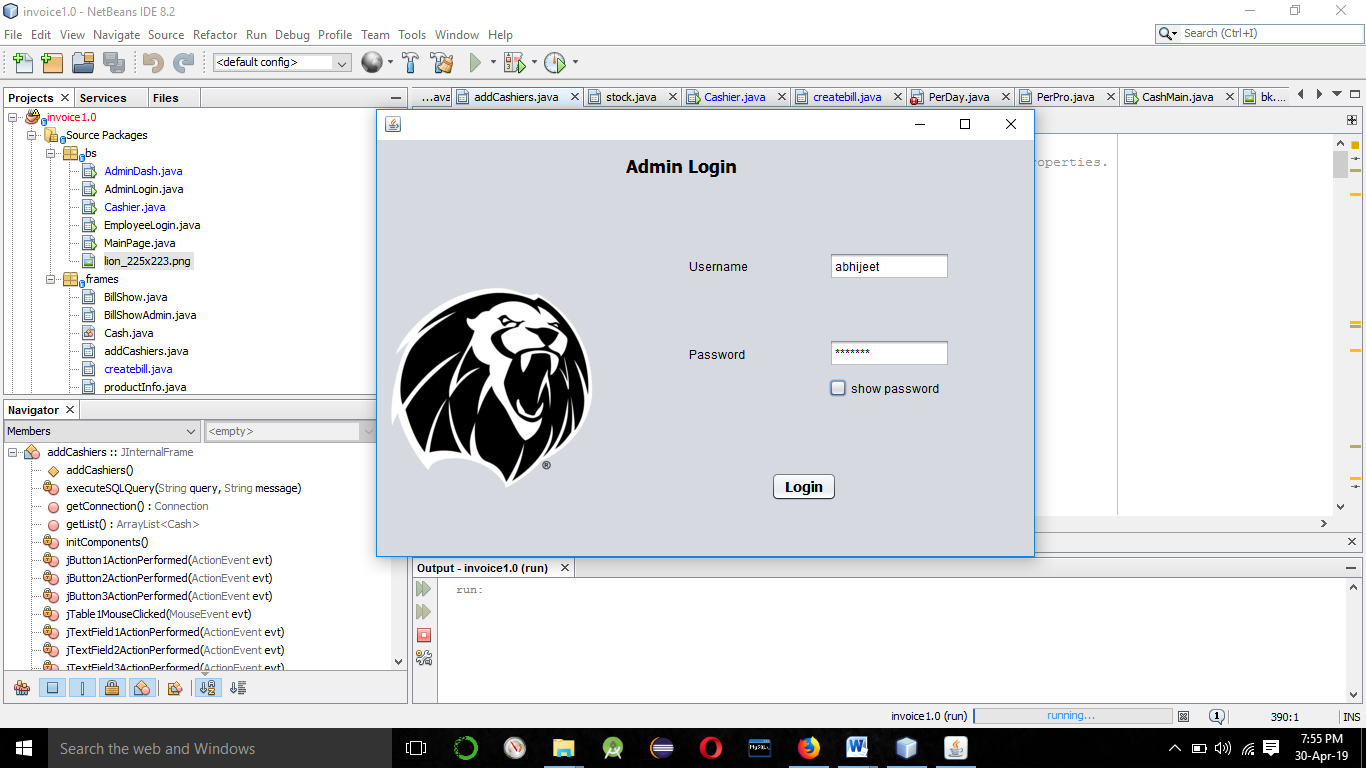
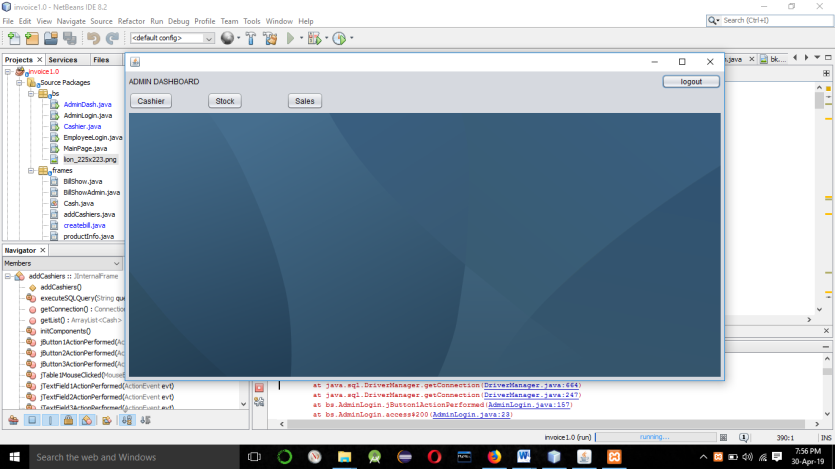
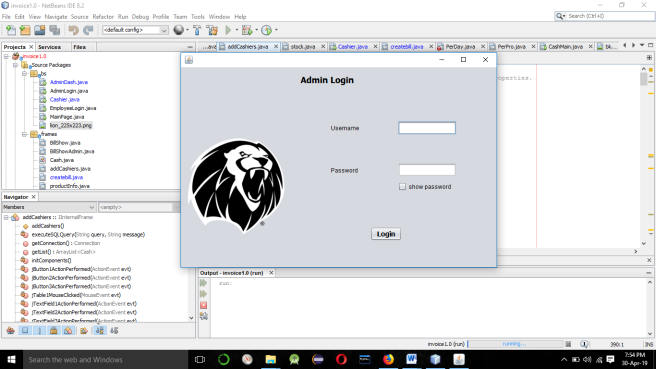
**Table 4** Bill Table Description

**RESULTS**

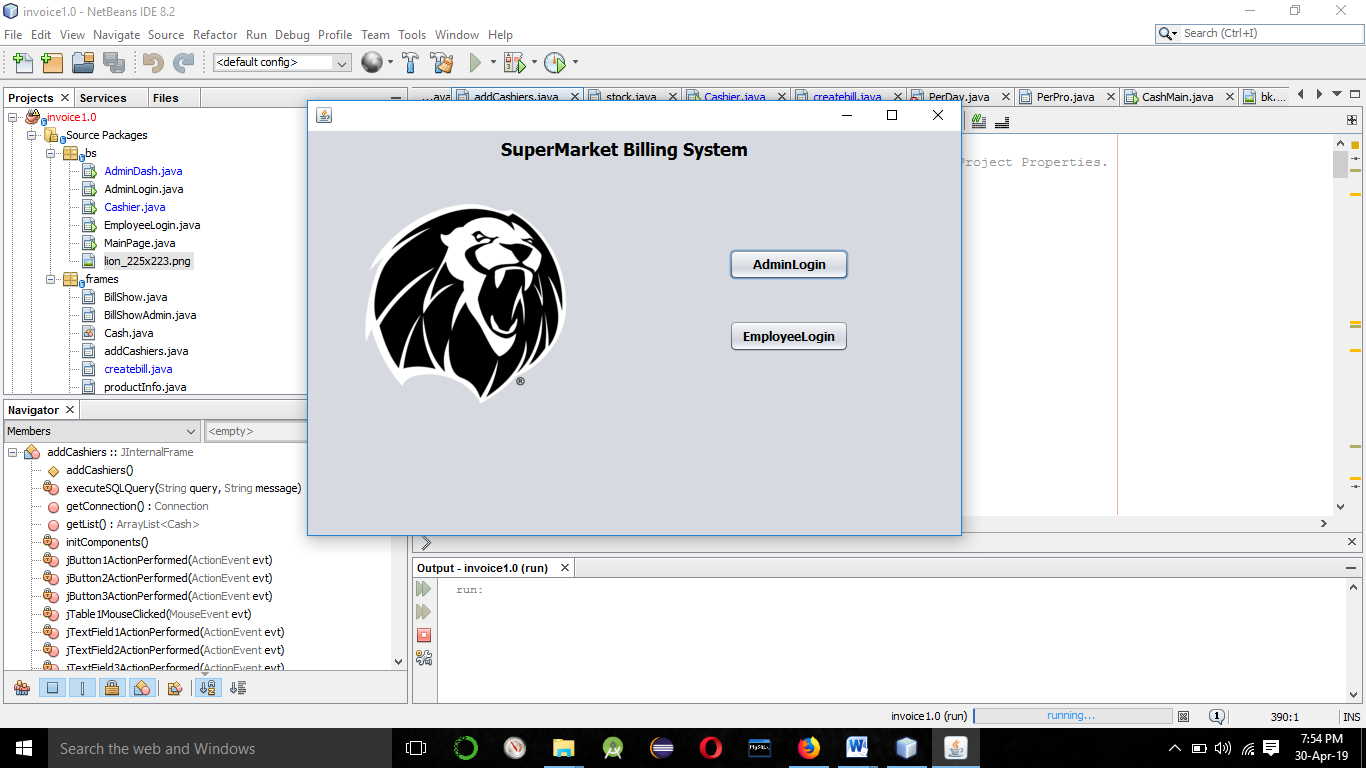
Supermarket Management System software developed for a company has been designed to reduce the time taken to handle the sales activity. It is designed to replace an existing manual record system for reducing time taken for calculations and for storing data. The system uses JAVA Swing as front end and MySQL server as a backend for the database.

The system is strong to handle daily operations where the database is cleared over certain time. This system will reduce manual work, calculations and will also provide periodic reports any time.

The proposed supermarket management system is very useful for big supermarkets as well as small ones to manage their inventories, staffs, and records of purchases and sales. New features and modules can be easily added into the system, so the project is very flexible and can adapt to the requirements of the supermarket and its users.

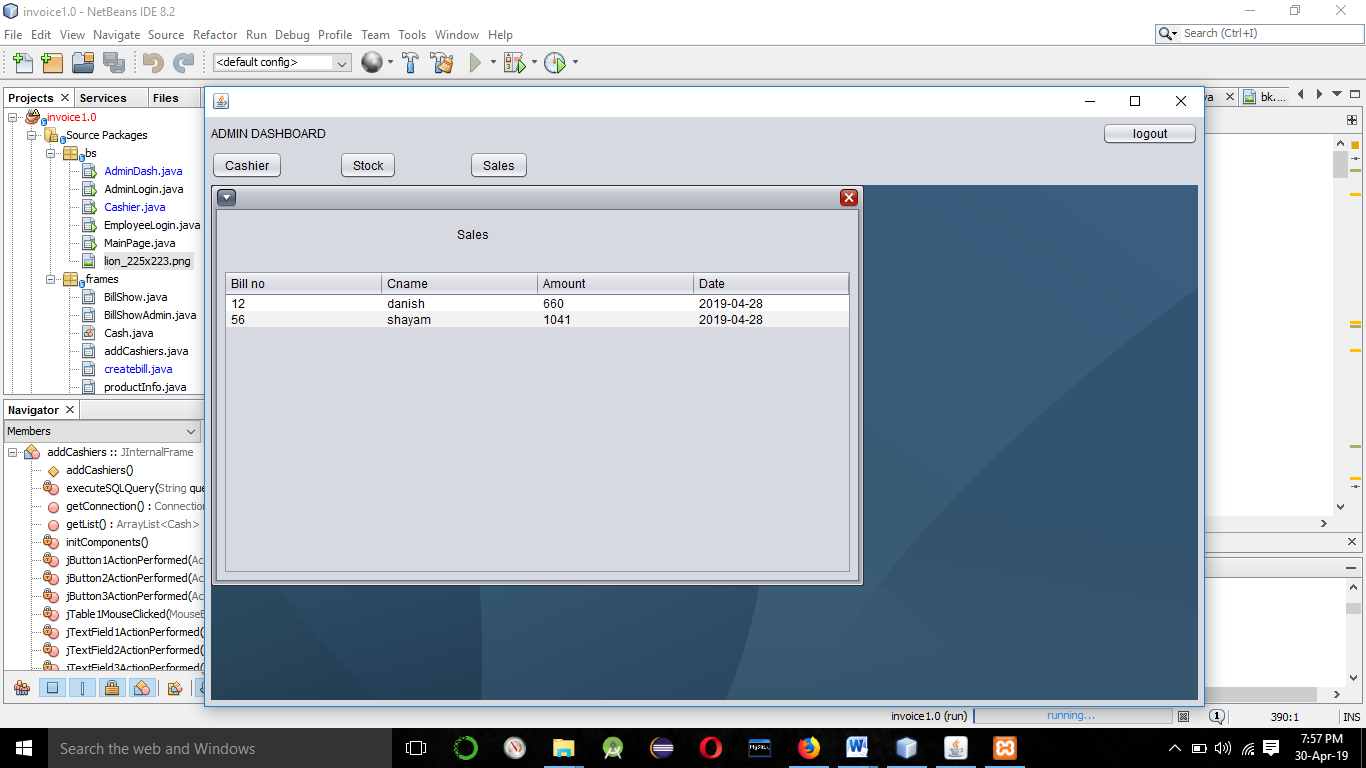
****

**OUTPUT**

****

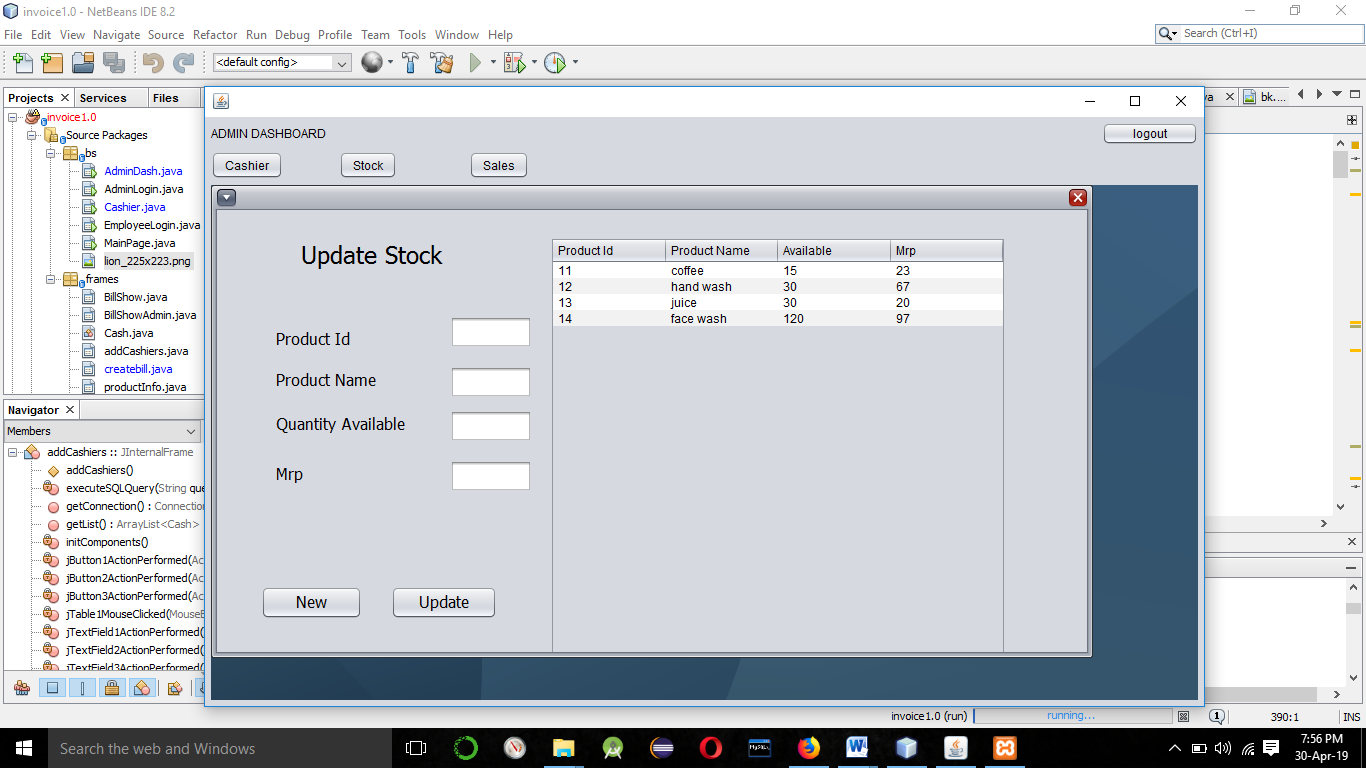
**Picture 1** – Main Page for Selective Login

.

****

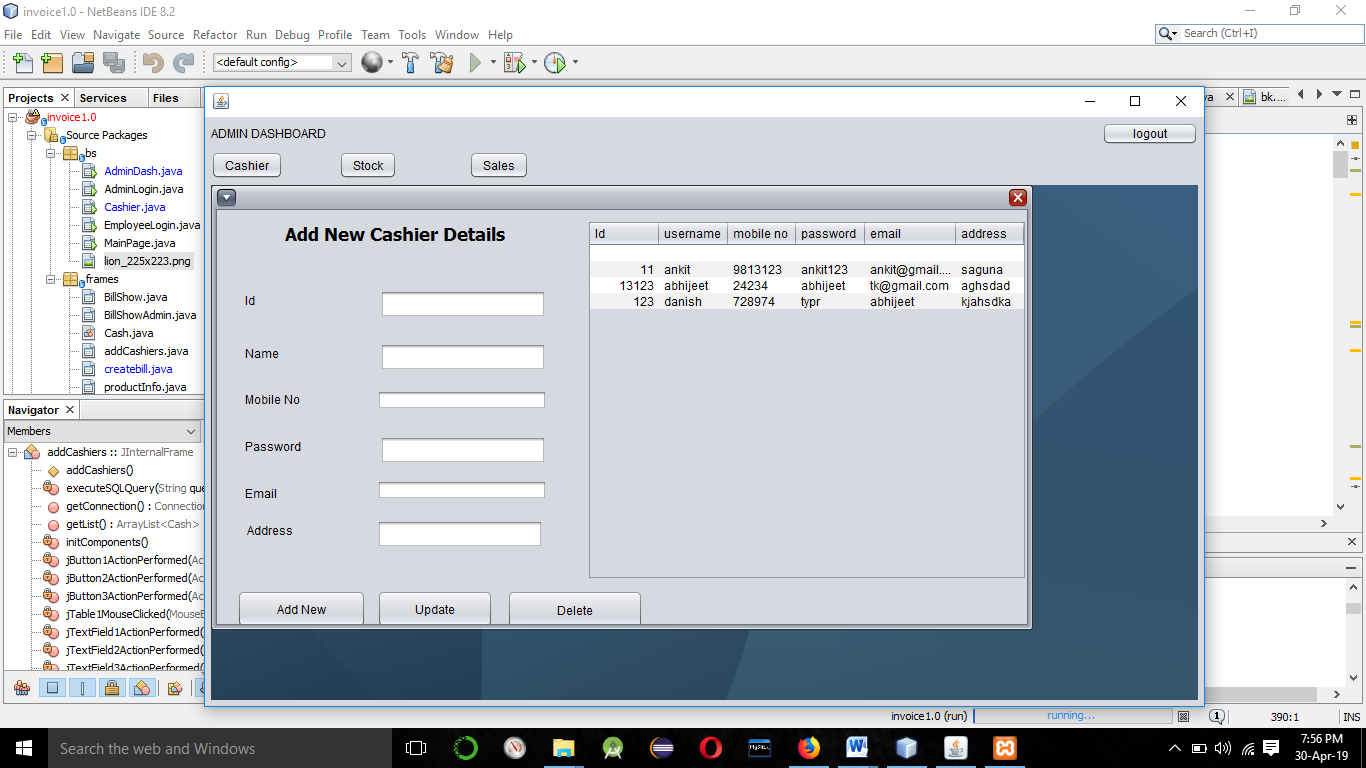
**Picture 2**– Sales Summary of Transaction

.

****

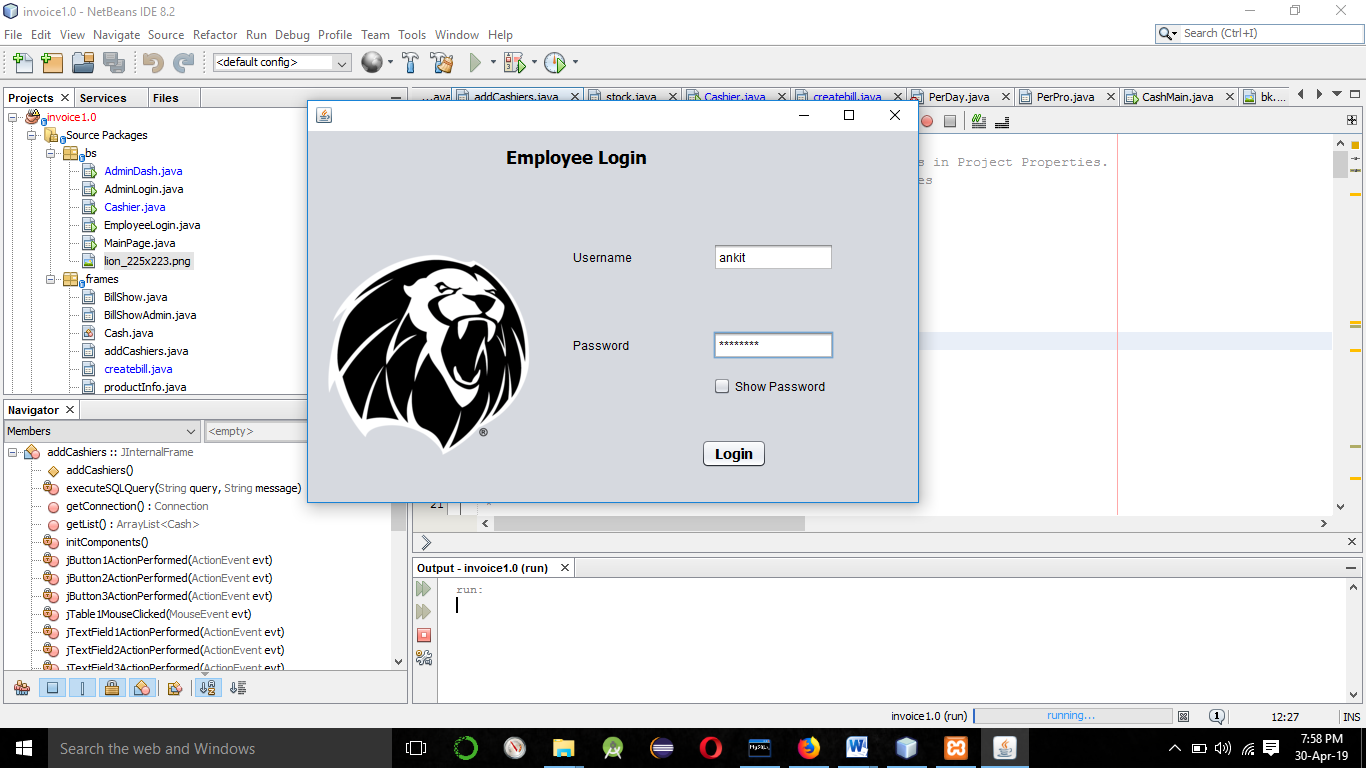
**Picture 3**– Updating/Inserting Products

.

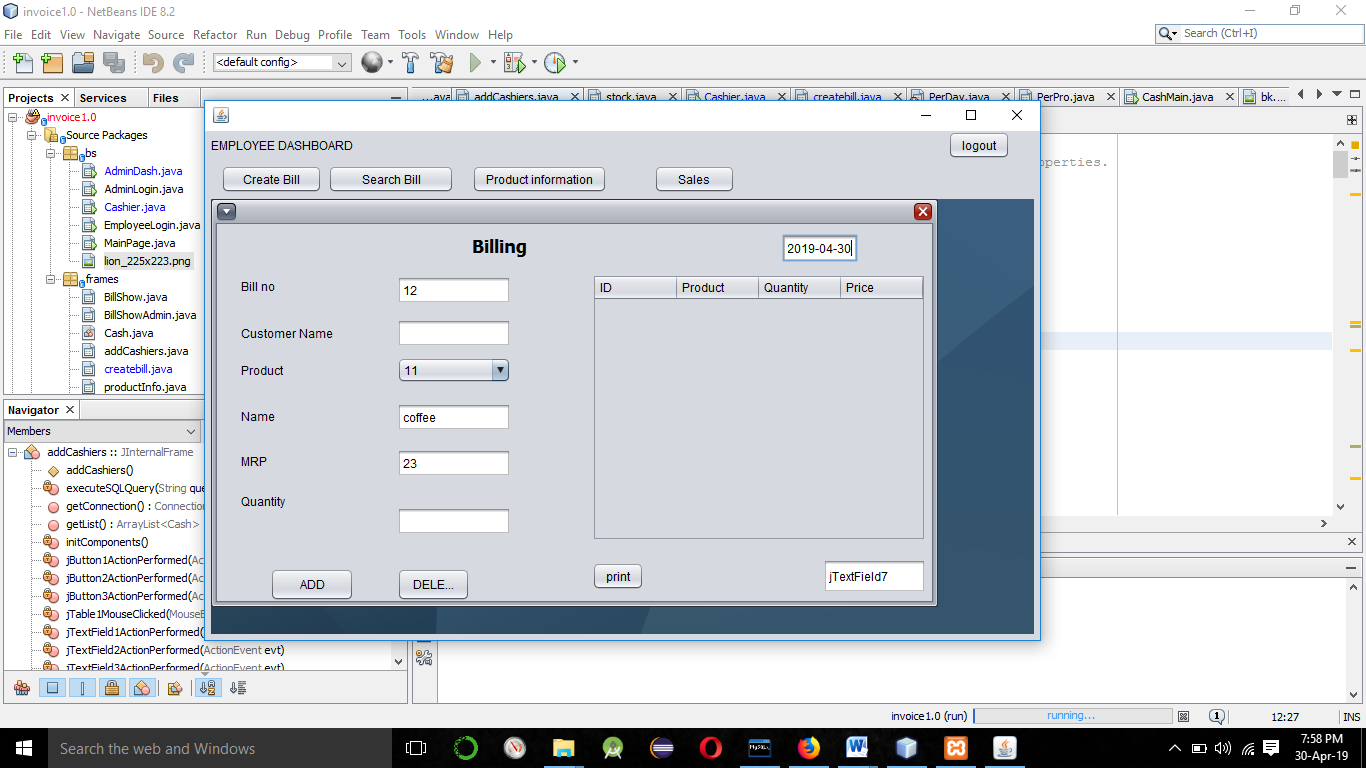
****

**Picture 4 –** Adding/Modifying Cashiers To Database

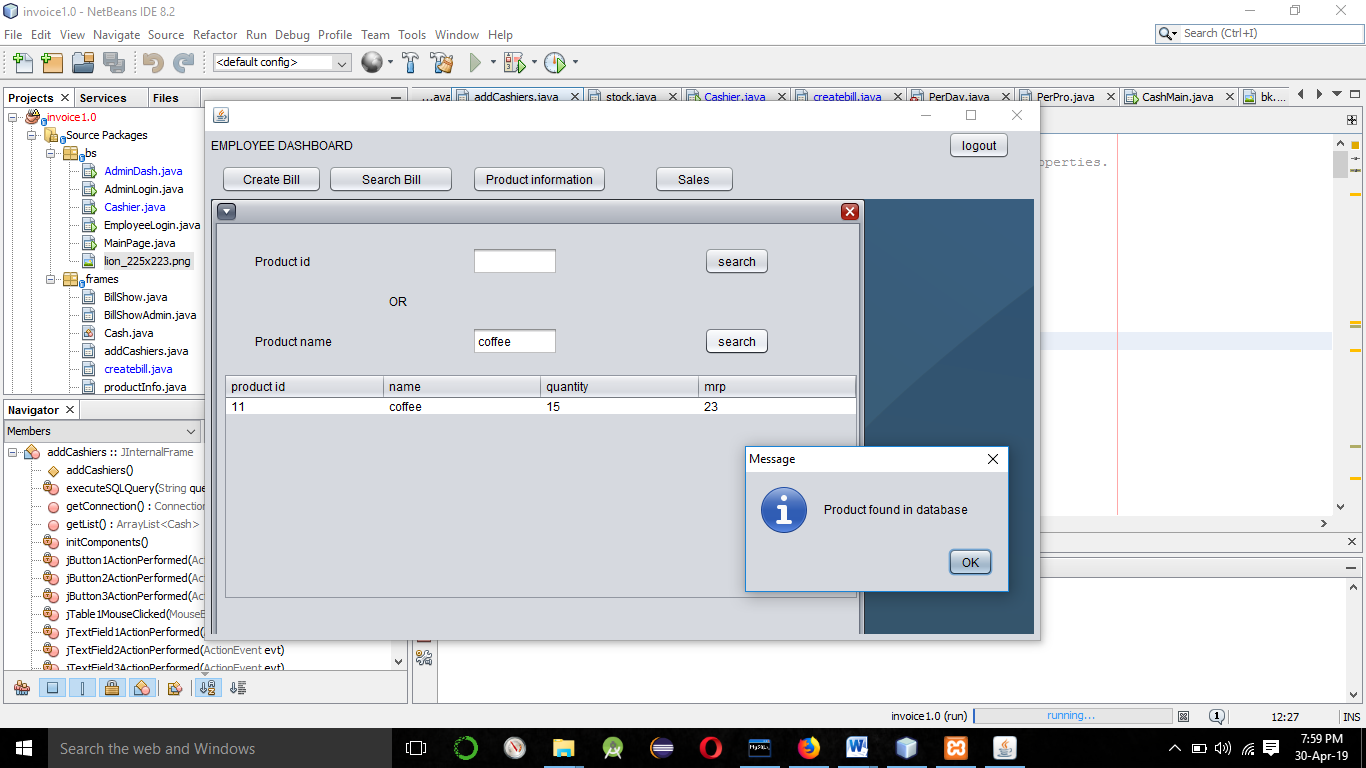
.

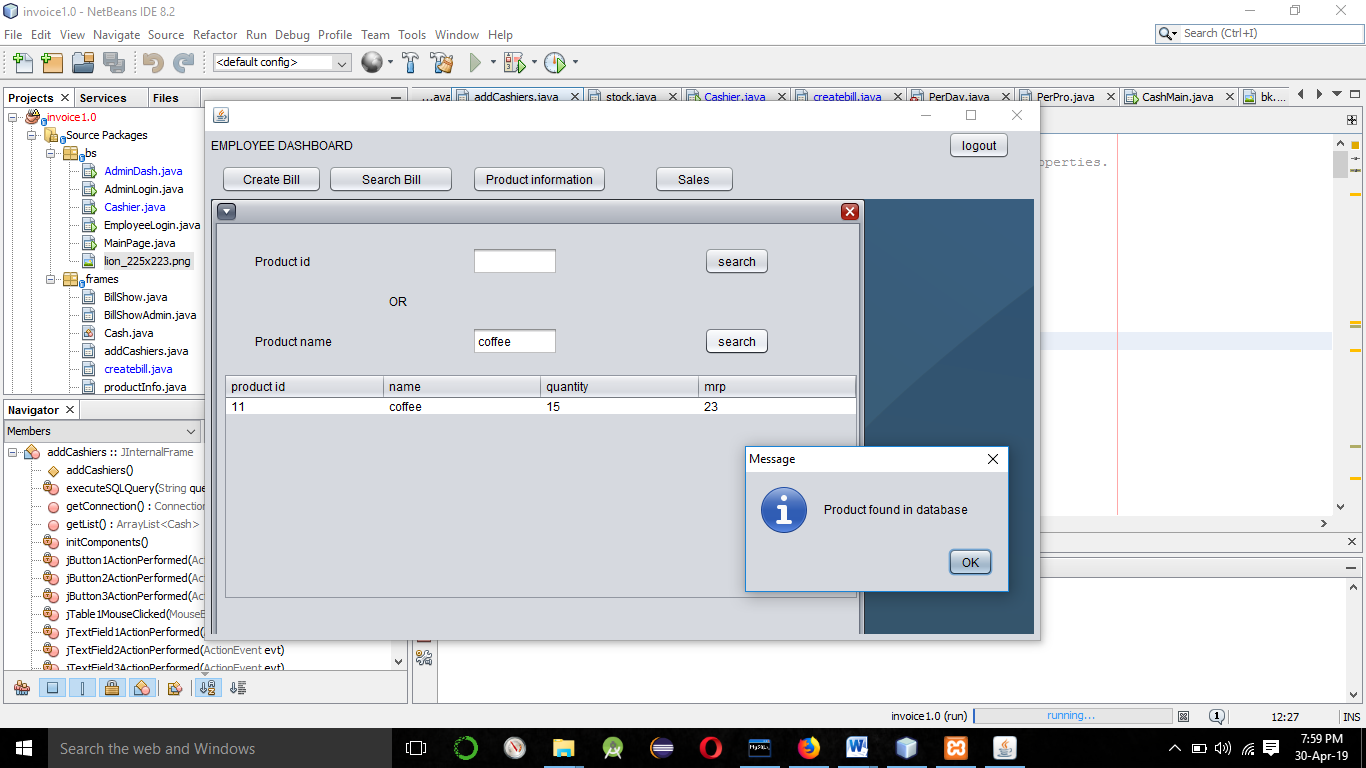
****

**Picture5 –**Login Page for cashiers/Employees.

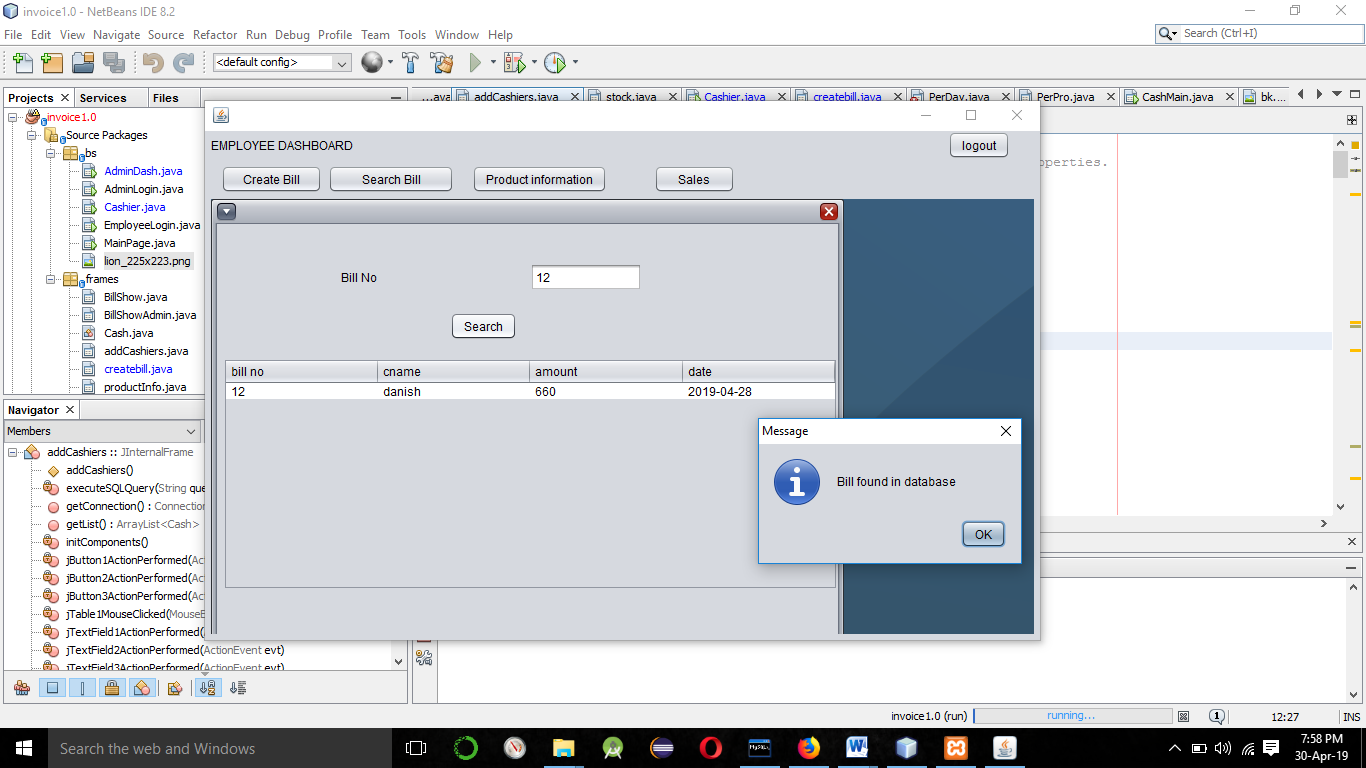
****

**Picture6 –**Create Bill Interface .

****

****

**Picture 7**-Searching Product in Stock

**Searching**

**Picture 8**-Searching Bills