**INVENTORY MANAGEMENT**

**A MINI-PROJECT REPORT**

***Submitted by***

**VIGNESH.B 180701280**

**YUVAHARI.J 180701293**

**VIGNESH.B 180701279**

***in partial fulfillment of the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

****

**RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI**

**CHENNAI**

**September 2019**

**BONAFIDE CERTIFICATE**

Certified that this project **“INVENTORY MANAGEMENT”** is the bonafide work of **“VIGNESH.B(280), YUVAHARI.J(293), VIGNESH.B(279)”** who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **SIGNATURE**  **Dr.N.SANKARRAM ,**  **HEAD OF THE DEPARTMENT**  Dept. of Computer Science and Engg,  Rajalakshmi Engineering College,Chennai | **SIGNATURE**  **Mrs. JANANEE V**  **ASSISTANT PROFESSOR**  Dept. of Computer Science and Engg,  Rajalakshmi Engineering College,Chennai |

This mini project report is submitted for the viva vice examination to be held on \_\_\_\_\_\_\_\_\_\_

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ABSTRACT**

An inventory management system that allows user to manage and maintain his/her inventory with ease. The inventory management system has been developed to allow users to add an inventory, delete an inventory, enter inventory quantity and other details, update inventory status and more. The inventory management system has its own intelligently managed support system that allows user to view and manage various inventories added in the system.

Inventory Management System is important to ensure quality control in businesses that handle transactions revolving around consumer goods. Without proper inventory control, a large retail store may run out of stock on an important item. A good Inventory Management System will alert the retailer when it is time to reorder. Inventory Management System is also an important means of automatically tracking large shipments.

Inventory Management System make it simple to locate and analyze inventory information in real-time with a simple database search.

**The system provides following features:**

* User may add/update/delete inventory.
* User may add/update inventory details.
* Details include cost, quantity and description.
* Includes forms for inventory inwards and outwards.
* An interactive user interface.
* A flexible inventory management system.

**TABLE OF CONTENTS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CHAPTER NO** | | **TITLE** | | | **PAGE NO** |
|  | **ABSTRACT** | | | | **3** |
| **1** | **INTRODUCTION** | | | | **5** |
| 1.1 | INTRODUCTION | | | |  |
| 1.2 | SCOPE OF THE WORK | | | |  |
| 1.3 | AIM AND OBJECTIVES OF THE PROJECT | | | |  |
| **2** | **SYSTEM SPECIFICATIONS** | | | | **6** |
| 2.1 | HARDWARE SPECIFICATIONS | | | |  |
| 2.2 | SOFTWARE SPECIFICATIONS | | | |  |
| **3** | **MODULE DESCRIPTION** | | | | **7** |
| 3.1 | APP DESCRIPTION | | | |  |
| 3.2 | REGISTRATION | | | |  |
| 3.3  3.4  3.5  3.6  3.7 | LOGIN  BILLING  UPDATING ITEMS IN INVENTORY  STATISTICAL DATA  MANUPULATING USER ACCOUNTS | | | |  |
|  | | |  |
| **4** | **SYSTEM DESIGN** | | | | **9** |
| 4.1 | ARCHITECTURE DIAGRAM | | | |  |
| **5** | **CODING** | | | | **10** |
| **6** | **SCREENSHOTS** | | | | **27** |
| **7** | **CONCLUSION** | | | | **31** |
| **8** | **REFERENCES** | | | | **32** |

**CHAPTER 1**

**INTRODUCTION**

* 1. **INTRODUCTION**

**Inventory management** is a component of supply chain **management** that involves supervising non-capitalized assets, or **inventory**, and stock items. Specifically, “**inventory management** supervises the flow of goods from manufacturers to warehouses and from these facilities to point of sale.” Inventory management is an important aspect of any successful business. It is the process of overseeing and [**controlling the flow of inventory**](https://www.unleashedsoftware.com/blog/controlling-inventory.-how-hard-is-it-really) units a business uses in the production or manufacture of goods for sale or distribution. Inventories are usually made up of a combination of goods, raw materials and finished products, and effective management of these items is essential to ensure optimal stock levels and to maximize the earning potential of the company. It also allows a business to prevent or mitigate any inventory-associated losses.

* 1. **SCOPE OF THE WORK**

The scope of an inventory system can cover many needs, including valuing the inventory, measuring the change in inventory and planning for future inventory levels. The value of the inventory at the end of each period provides a basis for financial reporting on the balance sheet. Measuring the change in inventory allows the company to determine the cost of inventory sold during the period. This allows the company to plan for future inventory needs.

* 1. **AIM AND OBJECTIVES OF THE PROJECT**

The main **objective of inventory management** is to maintain **inventory** at appropriate level to avoid excessive or shortage of **inventory** because both the cases are undesirable for business. ... To minimize investment in **inventory** at minimum level to maximize profitability.

**CHAPTER 2**

**SYSTEM SPECIFICATIONS**

**2.1 HARDWARE SPECIFICATIONS**

|  |  |  |
| --- | --- | --- |
| Processor | **:** | Intel i5 |
| Memory Size | **:** | 8GB (Minimum) |
| HDD | **:** | 1 TB (Minimum) |

**2.2 SOFTWARE SPECIFICATIONS**

|  |  |  |
| --- | --- | --- |
| Operating System | **:** | WINDOWS 10 |
| Front – End | **:** | Python |
| Back – End | **:** | Oracle |
| Language | **:** | Python |

**CHAPTER 3**

**MODULE DESCRIPTION**

**3.1 APP DESCRIPTION:**

The application Inventory Management – (MyShopee)first directs to the Login or Register screen. If you are an already existing user, click on the Login button once you are logged in you have three different option one to go to billing page you can also add discounts to the bill , another one for updating items in inventory (which can be accessed only by manager level employees) and another option to view the statistical data in table and graphical format also the admin login can be used for creating new account ,editing them , deleting accounts ,viewing account details and for accessing the statistical view page

**3.2 REGISTRATION MODULE:**

If a person needs to use this app, the person has to first register with the help of the admin by entering their username, password, name, address and email id. Once the registration is done a userid is generated using this they can log in anytime and perform various activities according to their position in the shop.

**3.3 LOGIN MODULE :**

The admin login module redirects you to the admin window where you can perform manipulation on the user accounts and the user login redirects you to the user window where one can perform manipulation activities such as billing , adding new items into the inventory , updating the existing items , viewing the statistical data ,adding discounts to the bill . the billing and other details are stored in mysql database .

**3.4 BILLING MODULE:**

The shopkeeper or the manager can perform the billing in this window . they can easily choose the items using their names and fetch their details such as its price and ID and quantity available and then add it to the bill for processing one can also add discounts to the bill using this module and proceed for payment.

**3.5 : UPDATING ITEMS IN INVENTORY:**

This module is used to add or update the contents of the inventory only the managers are allowed to perform addition and update activities the module can also be used to edit the product details if required .

**3.6 : STATISTICAL DATA MODULE :**

This module can be accessed by the employee as well as the admin it is used to show the sales details in a statistical manner it has to window one showing the sales data in table format and another one to show the sales details in graphical format

**3.7: MANUPULATING USER ACCOUNTS:**

This module is accessible only by the admin the admin can perform four main operations

1.adding new user accounts by entering the required details.

2.editing the user details and can also view the user details.

3.deleting the user accounts .

4.view the statistical data .

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 ARCHITECTURE DIAGRAM**

**CHAPTER 5**

**CODING**

**BACKEND CONNECTIVITY :**

import java.sql.\*;

import com.mysql.\*;

import java.util.\*;

import javax.swing.JOptionPane;

public class login{

Connection conn = null;

public static Connection dbconnector() {

try {

String url="jdbc:mysql://localhost:3306/minipro1";

String uname="root";

String pass="VIGN@2000";

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

Connection conn =DriverManager.getConnection(url,uname,pass);

return conn;

}

catch (Exception e){

JOptionPane.showMessageDialog(null, e);

return null;

}

}

}

Connection conn = null;

Initialize(){

conn = login.dbconnector(); //function call

}

**Basic frame structure :**

**package** miniproj2;

**import** javax.swing.\*;

**import** java.awt.\*;

**import** java.awt.event.ActionListener;

**import** java.awt.event.ActionEvent;

**import** javax.swing.GroupLayout.Alignment;

**import** javax.swing.LayoutStyle.ComponentPlacement;

**public** **class** openpage1{

JFrame frame;

**public** **static** **void** main(String[] args) {

EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**try** {

openpage1 window = **new** openpage1();

window.frame.setVisible(**true**);

window.frame.setLocationRelativeTo(**null**);

} **catch** (Exception e) {

e.printStackTrace();

}

}

});

}

**public** openpage1() {

initialize();

}

**private** **void** initialize() {

frame = **new** JFrame();

frame.getContentPane().setBackground(**new** Color(255, 255, 255));

frame.setBounds(100, 100, 1309, 838);

frame.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

JButton btnNewButton = **new** JButton("Login ");

btnNewButton.setToolTipText("Welcome page");

btnNewButton.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

demo2 li1 = **new** demo2() ;

li1.setVisible(**true**);

frame.dispose();

}

});

btnNewButton.setBackground(**new** Color(32, 178, 170));

btnNewButton.setForeground(SystemColor.***controlText***);

JLabel lblNewLabel = **new** JLabel("");

lblNewLabel.setBackground(**new** Color(0, 0, 128));

lblNewLabel.setForeground(**new** Color(0, 0, 128));

JPanel panel = **new** JPanel();

panel.setBackground(**new** Color(0, 0, 128));

JPanel panel\_1 = **new** JPanel();

panel\_1.setBackground(**new** Color(0, 128, 128));

JLabel label = **new** JLabel("");

Image img4 = **new** ImageIcon(**this**.getClass().getResource("/logo\_1.jpg")).getImage().getScaledInstance(900 , 200 , Image.***SCALE\_DEFAULT***);

label.setIcon(**new** ImageIcon (img4));

JButton btnClose = **new** JButton("Close");

btnClose.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

frame.dispose();

back1 b = **new** back1();

b.dispose();

}

});

**REGISTRATION MODULE:**

btnAddRecord.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

**try** {

String p1= passwordField\_1.~~getText~~();

String p2=passwordField.~~getText~~();

String p3= textField.getText();

String p4= textField\_1.getText();

String p5= textField\_2.getText();

String p6= textField\_4.getText();

String p7= textField\_5.getText();

**if**(p1.contentEquals("") || p2.equals("") || p3.contentEquals("") || p4.equals("") || p5.contentEquals("") || p6.contentEquals("") || p7.equals("")){

JOptionPane.*showMessageDialog*(**null**, " please enter all the details");}

**else** {

**boolean** flag = **true**;

String check="select name1 ,email\_id from log";

Statement st=conn.createStatement();

ResultSet rs=st.executeQuery(check);

**while**(rs.next()) {

String n=rs.getString("name1");

String n1=rs.getString("email\_id");

**if**(n.contentEquals(textField\_2.getText())){

flag=**false**;

JOptionPane.*showMessageDialog*(**null**, " User Name alredy exists");

textField\_2.setText("");

**break**;}

**if**(n1.contentEquals(textField\_4.getText())){

flag=**false**;

JOptionPane.*showMessageDialog*(**null**, " Email\_id alredy exists");

textField\_4.setText("");

**break**;}

}

**if**(flag) {

**if**(p1.contentEquals(p2)) {

String sql = "INSERT INTO log values (null,?, ?, ?, ?, ?, ?)";

PreparedStatement statement = conn.prepareStatement(sql);

statement.setString(1,textField.getText() );

statement.setString(2,textField\_1.getText() );

statement.setString(3,textField\_2.getText());

statement.setString(4, passwordField\_1.~~getText~~());

statement.setString(5, textField\_5.getText());

statement.setString(6, textField\_4.getText());

statement.executeUpdate();

JOptionPane.*showMessageDialog*(**null**, " Record added Successfully");

textField.setText("");

textField\_1.setText("");

textField\_2.setText("");

passwordField\_1.setText("");

textField\_4.setText("");

textField\_5.setText("");

passwordField.setText("");

}

**else** {

JOptionPane.*showMessageDialog*(**null**, " Password Mismatched");}

}

}

}

**catch** (SQLException ex) {

ex.printStackTrace();

}

}

});

**LOGIN MODULE :**

Connection conn = null ;

Conn = login.dbconnector();

**public** **void** actionPerformed(ActionEvent arg0) {

**try** {

String query="select \* from log where userid='"+username.getText()+"'AND password1='"+pass.~~getText~~()+"' AND usertype='ADMIN';";

Statement st=connectionobj.createStatement();

ResultSet rs = st.executeQuery(query);

findaname();

**if**(rs.next()) {

JOptionPane.*showMessageDialog*(**null**, name+" has Logged in successfully");

ad a = **new** ad();

a.setVisible(**true**);

dispose();

}

**else** {

JOptionPane.*showMessageDialog*(**null**, "Login failed");

//dispose();

}

rs.close();

connectionobj.close();

} **catch**(Exception e) {System.***out***.println(e);}

}

});

**BILLING MODULE:**

(Main functions)

**public** billingwin() {

*sum1* = 0;

conn222 = login.*dbconnector*();

connectionobj = login.*dbconnector*();

connectionobj2 = login.*dbconnector*();

conn3 = login.*dbconnector*();

conn4 = login.*dbconnector*();

conn5 = login.*dbconnector*();

c = login.*dbconnector*();

connection1 = login.*dbconnector*();

initialize();

}

**public** billingwin(**int** x , String y ) {

*sum1* = 0;

conn222 = login.*dbconnector*();

connectionobj = login.*dbconnector*();

connectionobj2 = login.*dbconnector*();

conn3 = login.*dbconnector*();

conn4 = login.*dbconnector*();

conn5 = login.*dbconnector*();

*string1* = y ;

*int1* = x ;

connection1 = login.*dbconnector*();

c = login.*dbconnector*();

initialize();

}

**public** **void** fillcombobox()

{

**try** {

String query = "select \* from product;";

java.sql.PreparedStatement pst1 = conn222.prepareStatement(query);

ResultSet rs19 = pst1.executeQuery();

**while**(rs19.next())

{

comboBox1.addItem(rs19.getString("prodname"));

}

}

**catch**(Exception e) {e.printStackTrace();}

}

**public** **void** refreshtable()

{

**try** {

String query = "select prodID,prodname,productQuantity from product where productQuantity <=10;";

java.sql.PreparedStatement pst =connectionobj.prepareStatement(query);

ResultSet rs = pst.executeQuery();

table.setModel(DbUtils.*resultSetToTableModel*(rs));

rs.close();

}

**catch**(Exception e) {e.printStackTrace();}

}

**public** **void** currentdate()

{

Calendar cal = **new** GregorianCalendar();

**int** month = cal.get(Calendar.***MONTH***);

**int** year = cal.get(Calendar.***YEAR***);

**int** day = cal.get(Calendar.***DAY\_OF\_MONTH***);

thisdate = day+"-"+(month+1)+"-"+year;

}

**public** String cdate()

{

Calendar cal = **new** GregorianCalendar();

**int** month = cal.get(Calendar.***MONTH***);

**int** year = cal.get(Calendar.***YEAR***);

**int** day = cal.get(Calendar.***DAY\_OF\_MONTH***);

String tdate1 = year+"-"+(month+1)+"-"+day;

**return** tdate1 ;

}

**public** **boolean** check1(**int** ch1 , String ch2)

{

**try** {

String query = "select \* from sales1 where prodID = "+ch1+" AND saledate = '"+ch2+"' ;";

java.sql.PreparedStatement pst =conn4.prepareStatement(query);

ResultSet rs = pst.executeQuery();

**if**(rs.next())

{

p2 = Integer.*parseInt*(rs.getString("profit"));

s4 = Integer.*parseInt*(rs.getString("prodQSold"));

**return** **true**;

}

}

**catch**(Exception e) {e.printStackTrace();}

**return** **false** ;

}

**public** **void** updatesales()

{

**try**

{

String s1 = "select \* from product where prodID = "+prodID+";";

java.sql.PreparedStatement pst11 =conn3.prepareStatement(s1);

ResultSet rs123 = pst11.executeQuery();

**if**(rs123.next())

{

prodID = Integer.*parseInt*(rs123.getString("prodID"));

productquantity = Integer.*parseInt*(rs123.getString("productquantity"));

prodname = rs123.getString("prodname");

buyp = Integer.*parseInt*(rs123.getString("buyprice"));

sellp = Integer.*parseInt*(rs123.getString("sellprice"));

}

}

**public** **boolean** refreshquantity()

{

**try** {

prodID = Integer.*parseInt*(txtItemno.getText());

String query2 = "select productquantity from product where prodID="+txtItemno.getText()+";";

java.sql.PreparedStatement pst1 =connectionobj.prepareStatement(query2);

ResultSet rs12 = pst1.executeQuery(query2);

**if**(rs12.next())

{

localQ = rs12.getString("productquantity");

}

localQ1 =(Integer.*parseInt*(localQ))-(Integer.*parseInt*(txtQuant.getText()));

**if**(localQ1>=0)

{

orderval = Integer.*parseInt*(txtQuant.getText());

String query123 = "update product set productquantity='"+localQ1+"' where prodID = '"+txtItemno.getText()+"';";

java.sql.PreparedStatement pst =connectionobj.prepareStatement(query123);

**int** rs1 = pst.executeUpdate(query123);

rs12.close();

refreshtable();

updatesales();

**return** **true**;

}

**else**

{

JOptionPane.*showMessageDialog*(**null**,"The product is out of stock");

}

}

**catch**(Exception e) {e.printStackTrace();}

**return** **false**;

}

**UPDATING ITEMS IN INVENTORY:**

(Adding new items)

JButton btnAddItem = **new** JButton("Add Item ");

btnAddItem.setToolTipText("Add item to the inventory");

btnAddItem.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

**try** {

String n = **null** ;

String query9 = "insert into Product1 values("+n+" , '"+txtpn.getText()+"' , "+txtquant.getText()+" , '"+txtDealer.getText()+"' , "+txtbp.getText()+" , "+txtsp.getText()+" , "+txtgst.getText()+" ) ; " ;

java.sql.PreparedStatement pst =conn5.prepareStatement(query9);

**int** rs1 = pst.executeUpdate(query9);

JOptionPane.*showMessageDialog*(**null**, "Updatd Item quantity = "+txtquant.getText());

**try** {

String q10 = "select max(prodID) from Product1 ; ";

java.sql.PreparedStatement pst9 =co.prepareStatement(q10);

ResultSet rs12 = pst9.executeQuery();

**if**(rs12.next())

{

txtprodID.setText(rs12.getString(1));

}

}

**catch**(Exception e) {System.***out***.println(e);}

refreshtable(txtprodID.getText());

conn5.close();

}

**catch**(Exception e1) {

JOptionPane.*showMessageDialog*(**null**, "All the above feilds must be filled ! ");

System.***out***.println(e1);

}

}

});

(Updating existing items)

JButton btnAddItem = **new** JButton("Update quantity");

btnAddItem.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent e) {

prodID = Integer.*parseInt*(txtproductid.getText());

s22 = textQuantity.getText();

**int** v21 = Integer.*parseInt*(s21);

**int** v22 = Integer.*parseInt*(s22);;

**int** quant =v21+v22;

**try** {

String query9 = "update Product1 set productquantity="+quant+" where prodID = "+txtproductid.getText()+" ;";

java.sql.PreparedStatement pst =connectionobj.prepareStatement(query9);

**int** rs1 = pst.executeUpdate(query9);

JOptionPane.*showMessageDialog*(**null**, "Updatd Item quantity = "+quant);

refreshtable(prodID);

**if**(rs1 != 0 ) {

relable();

}

//connectionobj.close();

}

**catch**(Exception e1) {

System.***out***.println(e1);

}

}

});

**STATISTICAL DATA MODULE:**

**(Table format )**

**public** **void** currentdate()

{

Calendar cal = **new** GregorianCalendar();

**int** month = cal.get(Calendar.***MONTH***);

**int** year = cal.get(Calendar.***YEAR***);

**int** day = cal.get(Calendar.***DAY\_OF\_MONTH***);

date\_txt.setText(day+"-"+(month+1)+"-"+year);

}

**public** **void** actionPerformed(ActionEvent e) {

currentdate();

**try** {

DateFormat df = **new** SimpleDateFormat("yyyy-MM-dd");

datecon.setText(df.format(datecon1.getDate()));

String dc = datecon.getText();

String query = "select prodID,prodname,prodQSold,buyp,sellp,profit,saledate,sellerID from sales1 where saledate = '"+dc+"';";

java.sql.PreparedStatement pst =conn.prepareStatement(query);

ResultSet rs = pst.executeQuery();

table1.setModel(DbUtils.*resultSetToTableModel*(rs));

rs.close();

}

**catch**(Exception e1) {e1.printStackTrace();}

}

});

**(Graph format )**

JButton btnGraph = **new** JButton("GENERATE GRAPH");

btnGraph.setBackground(**new** Color(255, 215, 0));

btnGraph.setBounds(183, 144, 245, 75);

btnGraph.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

String query ="select dateofsale , totalp from bill1";

JDBCCategoryDataset dataset = **null**;

**try** {

dataset = **new** JDBCCategoryDataset("jdbc:mysql://localhost:3306/minipro1","com.mysql.jdbc.Driver","root","VIGN@2000");

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

} **catch** (SQLException e) {

e.printStackTrace();

}

**try** {

dataset.executeQuery(query);

} **catch** (SQLException e) {

e.printStackTrace();

//totalp int , dateofsale date , discout int , userID

}

JFreeChart chart=ChartFactory.*createLineChart*("GRAPH SHOWING SALES VS DATE", "DATE", "AMOUNT", dataset,PlotOrientation.***VERTICAL***,**true**,**true**,**false**);

ChartPanel chartPanel= **new** ChartPanel(chart);

chartPanel.setPreferredSize(**new** java.awt.Dimension(500,270));

ApplicationFrame f= **new** ApplicationFrame("chart");

f.setContentPane(chartPanel);

f.pack();

f.setVisible(**true**);

f.setExtendedState(JFrame.***MAXIMIZED\_BOTH***);

}

});

**MANUPULATING USER ACCOUNTS :**

**(Editing)**

btnSubmit.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

**try** {

String unknown=**null**;

**boolean** flag =**true**;

Statement st=conn.createStatement();

String check="select email\_id from log";

ResultSet rs=st.executeQuery(check);

**while**(rs.next()) {

String n1=rs.getString("email\_id");

**if**(n1.contentEquals(textField\_1.getText())){

flag=**false**;

unknown="username";

**break**;

}

}

**if**(flag!=**false**){

JOptionPane.*showMessageDialog*(**null**, " No such user is found ");

textField.setText("");

passwordField.setText("");

passwordField\_1.setText("");

textField\_1.setText("");

}

**if**(unknown.contentEquals("username")) {

**try** {

Statement stt=connn.createStatement();

String p1=textField.getText();

String p2=passwordField.~~getText~~();

String p3=passwordField\_1.~~getText~~();

String w1=textField\_1.getText();

**if**(p1.contentEquals("")|| p2.contentEquals("")||p3.contentEquals("")|| w1.contentEquals("")) {

JOptionPane.*showMessageDialog*(**null**, " please enter all the details");

}

**else** {

**if**(p2.contentEquals(p3)) {

String s="update log set name1='"+p1+"' ,password1='"+p2+"' where email\_id='"+w1+"';";

stt.executeUpdate(s);

JOptionPane.*showMessageDialog*(**null**, "Record Changed successfull");

}

**else** {

JOptionPane.*showMessageDialog*(**null**, "Password mismatched");

}

}

} **catch** (SQLException e) {

e.printStackTrace();

}

}

}**catch** (SQLException e) {

e.printStackTrace();

}

}

});

**(Deleting)**

btnDelete.addActionListener(**new** ActionListener() {

**public** **void** actionPerformed(ActionEvent arg0) {

**try** {

String unknown=**null**;

**boolean** flag =**true**;

Statement st=conn.createStatement();

String check="select name1 from log";

ResultSet rs=st.executeQuery(check);

**while**(rs.next()) {

String n=rs.getString("name1");

**if**(n.contentEquals(textField.getText())){

flag =**false**;

unknown="email";

**break**;

}

}

**if**(flag!=**false**){

JOptionPane.*showMessageDialog*(**null**, " No such user is found ");

textField.setText("");

}

**if**(unknown.contentEquals("email")) {

**try** {

Statement stt=connn.createStatement();

String p1=textField.getText();

**if**(p1.contentEquals("")) {

JOptionPane.*showMessageDialog*(**null**, " please enter the name");

}

**else** {

String s="delete from log where name1='"+p1+"';";

stt.executeUpdate(s);

JOptionPane.*showMessageDialog*(**null**, "Record Deleted successfull");

textField.setText("");

Image img1 = **new** ImageIcon(**this**.getClass().getResource("/shread1.jpg")).getImage().getScaledInstance(158 , 155 , Image.***SCALE\_DEFAULT***);

label.setIcon(**new** ImageIcon(img1));

}

} **catch** (SQLException e) {

e.printStackTrace();

}

}

}**catch** (SQLException e) {

e.printStackTrace();

}

}

});

**CHAPTER 6**

**SCREEN SHOTS**

|  |
| --- |
| REGISTRATION MODULE |
| LOGIN MODULE    BILLING MODULE :    UPDATING ITEMS IN INVENTORY MODULE :    STATISTICAL DATA MODULE :    (GRAPHICAL)    MANUPULATING USER ACCOUNTS MODULE |



**CHAPTER 7**

**CONCLUSION AND FUTURE ENHANCEMENT**

**CONCLUSION Inventory management** has to do with keeping accurate records of goods that are ready to be sold out. This often means having enough stock of goods to the **inventory** totals as well as subtracting the most recent sold out product to buyers

However, this project can be improved in such a way that it turns into a real time application. It can be created into an App that could be helpful and beneficial to many people to efficiently manage their inventory.