**Description**

The purpose of the system is to allow a user to log in and browse the products and add items to their cart.

**Requirements**

The requirements of the system are to first allow the user to login. When the user logs in they are greeted with their unique cart and the ability to view the products available, the user may select a row in the products table to be added to the cart or they can select a row in the cart table to remove some products from the cart. The user is required to enter the quantity they would like to remove from the cart. There are checks in place to make sure the user cannot take more than the quantity of products the table has in stock.

**Database tables**

**Cartitems**

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**Customer**

Graphical user interface, application, website

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

**Product**

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**ER Diagram**

Diagram, schematic

Description automatically generated

**Interesting code snippets**

**Products table**

// products table

DefaultTableModel model = new DefaultTableModel();

model.addColumn("Product ID");

model.addColumn("Product Name");

model.addColumn("Description");

model.addColumn("Price");

model.addColumn("Quantity");

while (result.next()) {

int id = result.getInt("productId");

String name = result.getString("name");

String discription = result.getString("description");

double price = result.getDouble("price");

int quantity = result.getInt("totalQuantity");

model.addRow(new Object[]{id, name, discription, price, quantity});

}

products = new JTable(model);

grid.gridx = 0;

grid.gridy = 1;

grid.fill = GridBagConstraints.***HORIZONTAL***;

root.add(new JScrollPane(products), grid);

**add to cart button Listener**

addToCart.addActionListener(new ActionListener(){

public void actionPerformed(ActionEvent e) {

try {

Connection con = null;

PreparedStatement stat = null;

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

// query the db

int quantity = Integer.*parseInt*(quantityText.getText());

//get the id of the selected row in the products table

Object productID = products.getValueAt(products.getSelectedRow(), 0);

// Create a statement

stat = con.prepareStatement("IF EXISTS(SELECT \* FROM product WHERE productID = " + productID + " AND totalquantity >= " + quantity + ") THEN "

+ "IF EXISTS(SELECT \* FROM cartitems WHERE productID = " + productID + " AND purchased = false AND customerID = " + userID + ") THEN "

+ "UPDATE cartitems SET quantity = quantity + " + quantity + ", price = (SELECT price FROM product WHERE productID = " + productID + ") \* (quantity) WHERE productID = " + productID + " AND customerID = " + userID + "; "

+ "ELSE "

+ "INSERT INTO cartitems (productID, quantity, price, customerID) VALUES (" + productID + ", " + quantity + ", (SELECT price FROM products WHERE productID = " + productID + ") \* " + quantity + ", " + userID + "); "

+ "END IF; "

+ "UPDATE product SET totalquantity = totalquantity - " + quantity + " WHERE productID = " + productID + "; "

+ "END IF; ");

stat.executeUpdate();

AddToCart addToCart = new AddToCart(userID);

addToCart.setVisible(true);

dispose();

} catch (NumberFormatException ex) {

JOptionPane.*showMessageDialog*(root, "quantity must be a number", "Value Error", JOptionPane.***ERROR\_MESSAGE***);

} catch (SQLException ex) {

ex.printStackTrace();

}

}

});