

**Project Report**

**Canteen Management System**

**Semester:** Spring 2024

**Course Name:** Object Oriented Programming

**Course Code:** CSE110

**Section:** 08

**Submitted to:**

Mahamudul Hasan  
Senior Lecturer   
Department of Computer Science & Engineering

**Submitted by:**

Samiu Eshika Upoma (2020-1-60-082)

Zarin Tasnim Nuzhat (2020-1-60-211)

Tanjilul Haq (2020-1-60-214)

**Introduction:** Canteen management system is a comprehensive software solution designed to streamline and automate canteen operations. Developed using JSwing, this application provides an intuitive and user-friendly interface for common users and administrators. The system includes a login mechanism to ensure secure access, allowing users to navigate to the respective dashboard after successful authentication. Key features of the system include inventory updates, order management, order history tracking, raw material management, and raw material distribution. Each module is meticulously designed to ensure efficient management of canteen operations, with all data securely stored and managed via text files.

**Main Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

public class MainFrame extends JFrame {

public MainFrame() {

setTitle("Canteen Management System");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(null);

new WelcomeFrame();

}

public static void main(String[] args) {

new MainFrame();

}

}

**Welcome Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class WelcomeFrame extends JFrame {

public WelcomeFrame() {

setTitle("Welcome to Canteen Management System");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(null);

JLabel welcomeLabel = new JLabel("Welcome to Canteen Management System");

welcomeLabel.setBounds(350, 150, 300, 30);

add(welcomeLabel);

JButton loginButton = new JButton("Login");

loginButton.setBounds(300, 250, 100, 30);

add(loginButton);

JButton adminButton = new JButton("Admin");

adminButton.setBounds(550, 250, 100, 30);

add(adminButton);

loginButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new LoginFrame();

dispose();

}

});

adminButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new AdminFrame();

dispose();

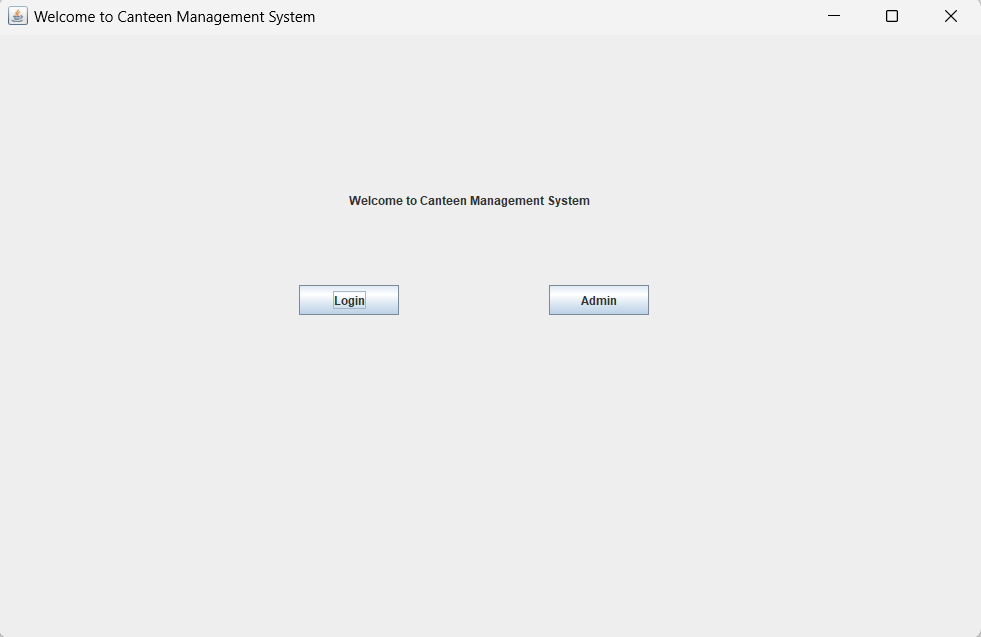
}

});

setVisible(true);

}

}



**Log in Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.IOException;

public class LoginFrame extends JFrame {

public LoginFrame() {

setTitle("Login");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLayout(null);

JLabel userLabel = new JLabel("Username:");

JLabel passLabel = new JLabel("Password:");

JTextField userField = new JTextField();

JPasswordField passField = new JPasswordField();

JButton loginButton = new JButton("Login");

userLabel.setBounds(400, 20, 80, 25);

userField.setBounds(500, 20, 150, 25);

passLabel.setBounds(400, 60, 80, 25);

passField.setBounds(500, 60, 150, 25);

loginButton.setBounds(450, 150, 100, 30);

add(userLabel);

add(userField);

add(passLabel);

add(passField);

add(loginButton);

loginButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String username = userField.getText();

String password = new String(passField.getPassword());

if (username.isEmpty() || password.isEmpty()) {

JOptionPane.showMessageDialog(null, "Please fill the both fields", "Warning", JOptionPane.WARNING\_MESSAGE);

} else {

if (authenticate(username, password)) {

new HomePageFrame(username);

dispose();

} else {

JOptionPane.showMessageDialog(null, "Login Failed");

}

}

}

});

setVisible(true);

}

private boolean authenticate(String username, String password) {

try (BufferedReader reader = new BufferedReader(new FileReader("users.txt"))) {

String line;

while ((line = reader.readLine()) != null) {

String[] userDetails = line.split(",");

if (userDetails[0].equals(username) && userDetails[1].equals(password)) {

return true;

}

}

} catch (IOException ex) {

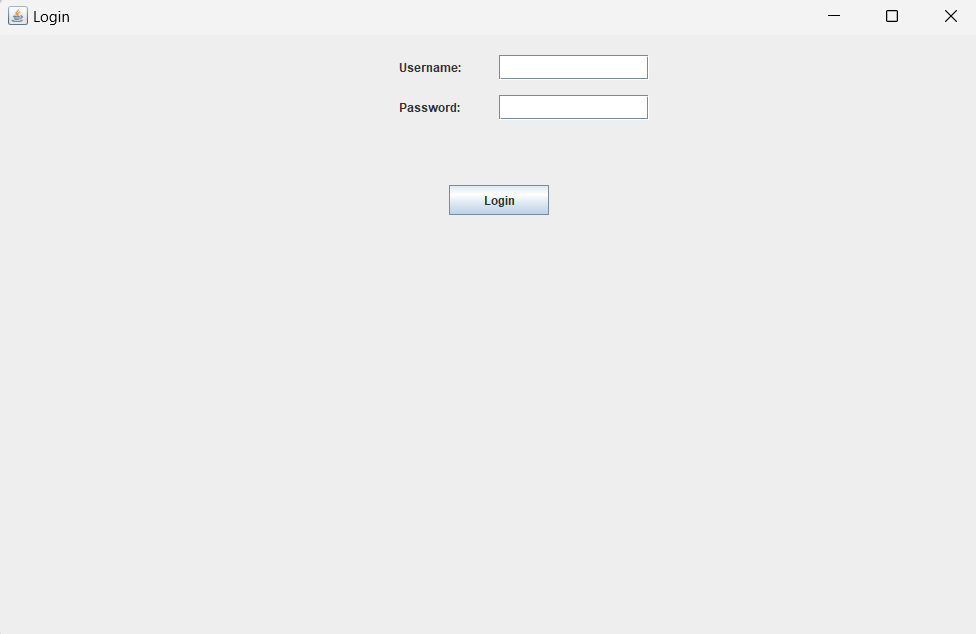
ex.printStackTrace();

}

return false;

}

}



**Admin Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.\*;

public class AdminFrame extends JFrame {

public AdminFrame() {

setTitle("Admin Information");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLayout(null);

JLabel adminLabel = new JLabel("Admin Information:");

adminLabel.setBounds(20, 20, 200, 25);

add(adminLabel);

JLabel adminImage1 = new JLabel(resizeImageIcon("upoma.jpg", 70, 70));

adminImage1.setBounds(20, 50, 70, 70);

add(adminImage1);

JTextArea adminDetails1 = createTextArea("ID: 101\nName: Upoma\nEmail: upoma@example.com\nPhone: 1234567890");

adminDetails1.setBounds(100, 50, 250, 90);

add(adminDetails1);

JLabel adminImage2 = new JLabel(resizeImageIcon("Rafi.jpg", 70, 70));

adminImage2.setBounds(20, 190, 70, 70);

add(adminImage2);

JTextArea adminDetails2 = createTextArea("ID: 102\nName: Rafi\nEmail: rafi@example.com\nPhone: 1234567899");

adminDetails2.setBounds(100, 190, 250, 90);

add(adminDetails2);

JLabel adminImage3 = new JLabel(resizeImageIcon("zarin.jpg", 70, 70));

adminImage3.setBounds(20, 330, 70, 70);

add(adminImage3);

JTextArea adminDetails3 = createTextArea("ID: 103\nName: Zarin\nEmail: zarin@example.com\nPhone: 1234567891");

adminDetails3.setBounds(100, 330, 250, 90);

add(adminDetails3);

setVisible(true);

}

private JTextArea createTextArea(String text) {

JTextArea textArea = new JTextArea(text);

textArea.setEditable(false);

textArea.setBackground(null);

textArea.setBorder(null);

return textArea;

}

private ImageIcon resizeImageIcon(String path, int width, int height) {

ImageIcon originalIcon = new ImageIcon(path);

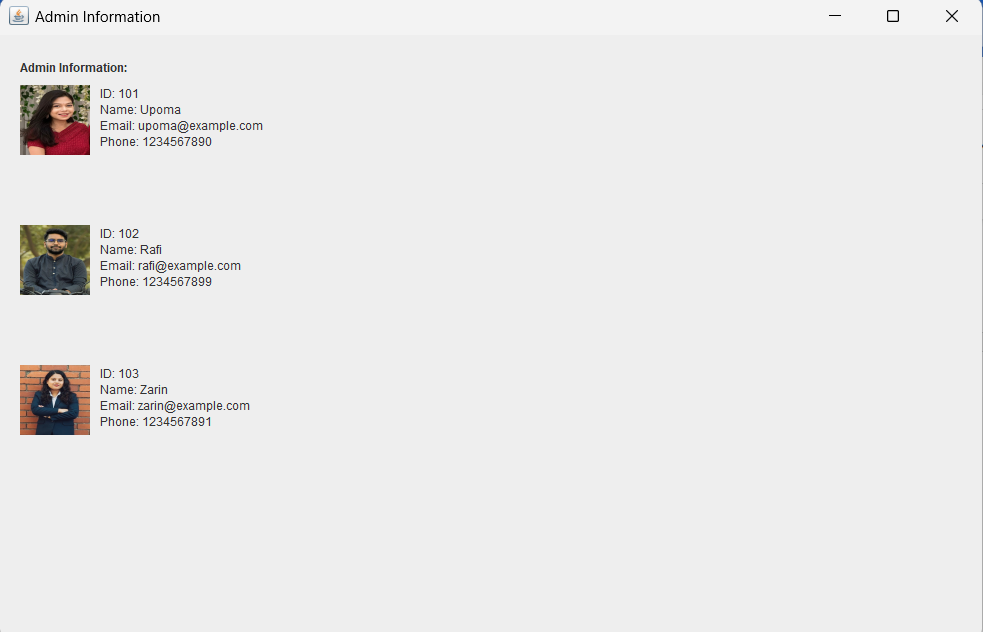
Image img = originalIcon.getImage();

Image resizedImage = img.getScaledInstance(width, height, Image.SCALE\_SMOOTH);

return new ImageIcon(resizedImage);

}

}



**Homepage Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class HomePageFrame extends JFrame {

private String username;

public HomePageFrame(String username) {

this.username = username;

setTitle("Home Page");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLayout(null);

JLabel welcomeLabel = new JLabel("Welcome " + username + "!");

welcomeLabel.setBounds(450, 60, 300, 30);

add(welcomeLabel);

JButton stockUpdateButton = new JButton("Stock Update");

JButton takeOrderButton = new JButton("Take Order");

JButton orderHistoryButton = new JButton("Order History");

JButton rawMaterialsButton = new JButton("Raw Materials");

JButton distributionButton = new JButton("Distribution");

stockUpdateButton.setBounds(400, 110, 200, 30);

takeOrderButton.setBounds(400, 160, 200, 30);

orderHistoryButton.setBounds(400, 210, 200, 30);

rawMaterialsButton.setBounds(400, 260, 200, 30);

distributionButton.setBounds(400, 310, 200, 30);

add(stockUpdateButton);

add(takeOrderButton);

add(orderHistoryButton);

add(rawMaterialsButton);

add(distributionButton);

stockUpdateButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new StockUpdateFrame();

}

});

takeOrderButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new TakeOrderFrame();

}

});

orderHistoryButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new OrderHistoryFrame();

}

});

rawMaterialsButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new RawMaterialsFrame().setVisible(true);

}

});

distributionButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

new DistributionFrame().setVisible(true);

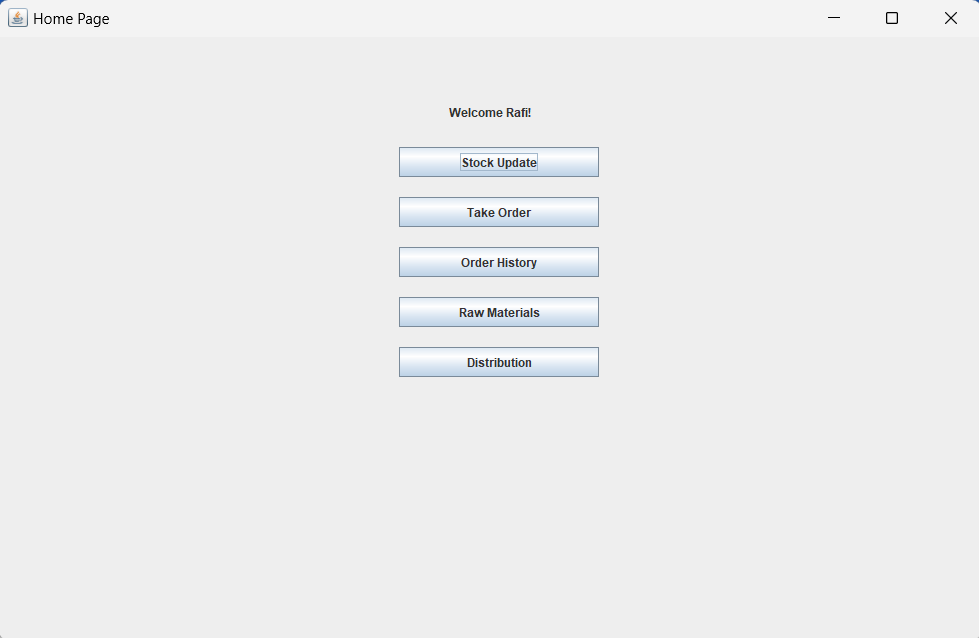
}

});

setVisible(true);

}

}



**Stoke Update Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.event.\*;

import java.io.\*;

import java.util.ArrayList;

import java.util.List;

public class StockUpdateFrame extends JFrame {

private JTextArea stockArea;

private List<JCheckBox> itemCheckBoxes;

private List<StockItem> stock;

public StockUpdateFrame() {

setTitle("Stock Update");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLayout(null);

JLabel stockLabel = new JLabel("Stock Update:");

stockLabel.setBounds(20, 20, 200, 25);

add(stockLabel);

itemCheckBoxes = new ArrayList<>();

stock = new ArrayList<>();

loadStock();

stockArea = new JTextArea();

updateStockArea();

stockArea.setBounds(20, 50, 350, 350);

stockArea.setEditable(false);

add(stockArea);

int yPosition = 50;

for (StockItem item : stock) {

JCheckBox checkBox = new JCheckBox(item.getName());

checkBox.setBounds(400, yPosition, 150, 25);

add(checkBox);

itemCheckBoxes.add(checkBox);

yPosition += 30;

}

JButton increaseButton = new JButton("Increase Stock");

increaseButton.setBounds(20, 420, 150, 30);

increaseButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

for (int i = 0; i < itemCheckBoxes.size(); i++) {

if (itemCheckBoxes.get(i).isSelected()) {

increaseStock(stock.get(i).getName(), 1);

}

}

updateStockArea();

saveStock();

}

});

add(increaseButton);

setVisible(true);

}

private void loadStock() {

try (BufferedReader reader = new BufferedReader(new FileReader("stock.txt"))) {

String line;

while ((line = reader.readLine()) != null) {

String[] parts = line.split(",");

stock.add(new StockItem(parts[0], Integer.parseInt(parts[1]), Double.parseDouble(parts[2])));

}

} catch (IOException ex) {

ex.printStackTrace();

}

}

private void saveStock() {

try (BufferedWriter writer = new BufferedWriter(new FileWriter("stock.txt"))) {

for (StockItem item : stock) {

writer.write(item.getName() + "," + item.getQuantity() + "," + item.getPrice());

writer.newLine();

}

} catch (IOException ex) {

ex.printStackTrace();

}

}

private void updateStockArea() {

StringBuilder stockText = new StringBuilder();

for (StockItem item : stock) {

stockText.append(item.getName()).append(": ").append(item.getQuantity()).append(", Price: ").append(item.getPrice()).append("\n");

}

stockArea.setText(stockText.toString());

}

private void increaseStock(String itemName, int amount) {

for (StockItem item : stock) {

if (item.getName().equals(itemName)) {

item.setQuantity(item.getQuantity() + amount);

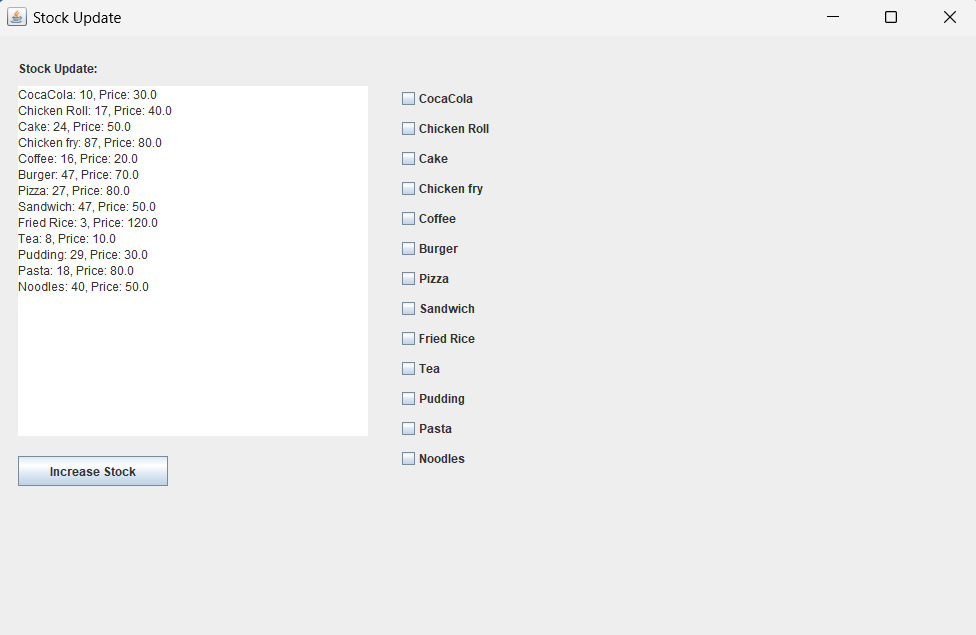
break;

}

}

}

}



**Unified code for Stock Items:**

package canteenmanagementsystem;

public class StockItem {

private String name;

private int quantity;

private double price;

private int orderQuantity;

public StockItem(String name, int quantity, double price) {

this.name = name;

this.quantity = quantity;

this.price = price;

this.orderQuantity = 0;

}

public String getName() {

return name;

}

public int getQuantity() {

return quantity;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

public int getOrderQuantity() {

return orderQuantity;

}

public void setOrderQuantity(int orderQuantity) {

this.orderQuantity = orderQuantity;

}

}

**Take Order Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.event.\*;

import java.io.\*;

import java.util.ArrayList;

import java.util.List;

public class TakeOrderFrame extends JFrame {

private List<StockItem> stock;

private List<JCheckBox> itemCheckboxes;

private List<JTextField> quantityFields;

private JLabel totalCostLabel;

private double totalCost;

public TakeOrderFrame() {

setTitle("Take Order");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLayout(null);

stock = new ArrayList<>();

itemCheckboxes = new ArrayList<>();

quantityFields = new ArrayList<>();

loadStock();

int yPosition = 20;

for (StockItem item : stock) {

JCheckBox checkBox = new JCheckBox(item.getName());

checkBox.setBounds(300, yPosition, 150, 25);

JLabel priceLabel = new JLabel("Price: " + item.getPrice());

priceLabel.setBounds(460, yPosition, 100, 25);

JTextField quantityField = new JTextField("0");

quantityField.setBounds(580, yPosition, 50, 25);

checkBox.addItemListener(new ItemListener() {

public void itemStateChanged(ItemEvent e) {

if (e.getStateChange() == ItemEvent.SELECTED) {

quantityField.setText("1");

} else {

quantityField.setText("0");

}

updateTotalCost();

}

});

quantityField.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

updateTotalCost();

}

});

add(checkBox);

add(priceLabel);

add(quantityField);

itemCheckboxes.add(checkBox);

quantityFields.add(quantityField);

yPosition += 30;

}

totalCostLabel = new JLabel("Total Cost: 0.0");

totalCostLabel.setBounds(300, yPosition, 200, 25);

add(totalCostLabel);

JButton confirmButton = new JButton("Confirm");

confirmButton.setBounds(530, yPosition + 30, 100, 30);

add(confirmButton);

confirmButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

if (checkStock()) {

updateStock();

new ConfirmOrderFrame(getOrderDetails(), totalCost);

dispose();

}

}

});

setVisible(true);

}

private void loadStock() {

try (BufferedReader reader = new BufferedReader(new FileReader("stock.txt"))) {

String line;

while ((line = reader.readLine()) != null) {

String[] parts = line.split(",");

stock.add(new StockItem(parts[0], Integer.parseInt(parts[1]), Double.parseDouble(parts[2])));

}

} catch (IOException ex) {

ex.printStackTrace();

}

}

private void saveStock() {

try (BufferedWriter writer = new BufferedWriter(new FileWriter("stock.txt"))) {

for (StockItem item : stock) {

writer.write(item.getName() + "," + item.getQuantity() + "," + item.getPrice());

writer.newLine();

}

} catch (IOException ex) {

ex.printStackTrace();

}

}

private void updateTotalCost() {

totalCost = 0;

for (int i = 0; i < stock.size(); i++) {

int quantity = Integer.parseInt(quantityFields.get(i).getText());

stock.get(i).setOrderQuantity(quantity);

totalCost += stock.get(i).getPrice() \* quantity;

}

totalCostLabel.setText("Total Cost: " + totalCost);

}

private List<StockItem> getOrderDetails() {

List<StockItem> orderDetails = new ArrayList<>();

for (StockItem item : stock) {

if (item.getOrderQuantity() > 0) {

orderDetails.add(item);

}

}

return orderDetails;

}

private boolean checkStock() {

for (StockItem item : stock) {

if (item.getOrderQuantity() > item.getQuantity()) {

JOptionPane.showMessageDialog(this, "Out of stock: " + item.getName(), "Warning", JOptionPane.WARNING\_MESSAGE);

return false;

}

}

return true;

}

private void updateStock() {

for (StockItem item : stock) {

int newStock = item.getQuantity() - item.getOrderQuantity();

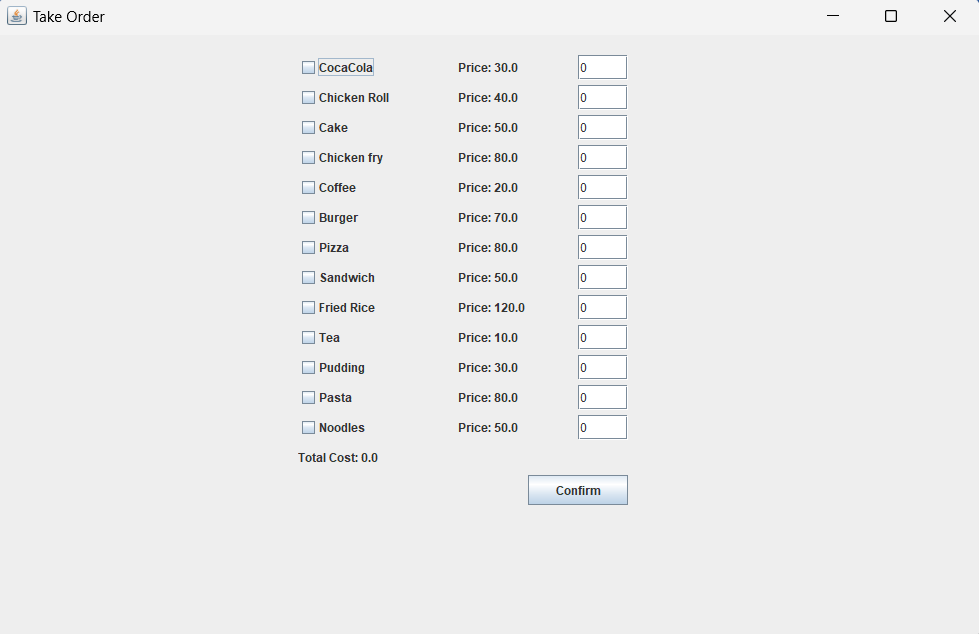
item.setQuantity(newStock);

}

saveStock();

}

}



**Confirm Order Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.List;

public class ConfirmOrderFrame extends JFrame {

private List<StockItem> orderDetails;

private double totalCost;

public ConfirmOrderFrame(List<StockItem> orderDetails, double totalCost) {

this.orderDetails = orderDetails;

this.totalCost = totalCost;

setTitle("Confirm Order");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLayout(null);

JLabel confirmLabel = new JLabel("Confirm Order:");

confirmLabel.setBounds(350, 20, 200, 25);

add(confirmLabel);

JTextArea orderSummary = new JTextArea();

orderSummary.setBounds(350, 50, 350, 150);

StringBuilder summary = new StringBuilder();

for (StockItem item : orderDetails) {

summary.append(item.getName()).append(": ").append(item.getOrderQuantity()).append("\n");

}

summary.append("\nTotal Cost: ").append(totalCost);

orderSummary.setText(summary.toString());

orderSummary.setEditable(false);

add(orderSummary);

JButton confirmButton = new JButton("Confirm");

confirmButton.setBounds(480, 220, 100, 30);

add(confirmButton);

confirmButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

OrderHistoryFrame.addOrder(orderDetails, totalCost);

OrderHistoryFrame orderHistoryFrame = new OrderHistoryFrame();

orderHistoryFrame.setVisible(true);

dispose();

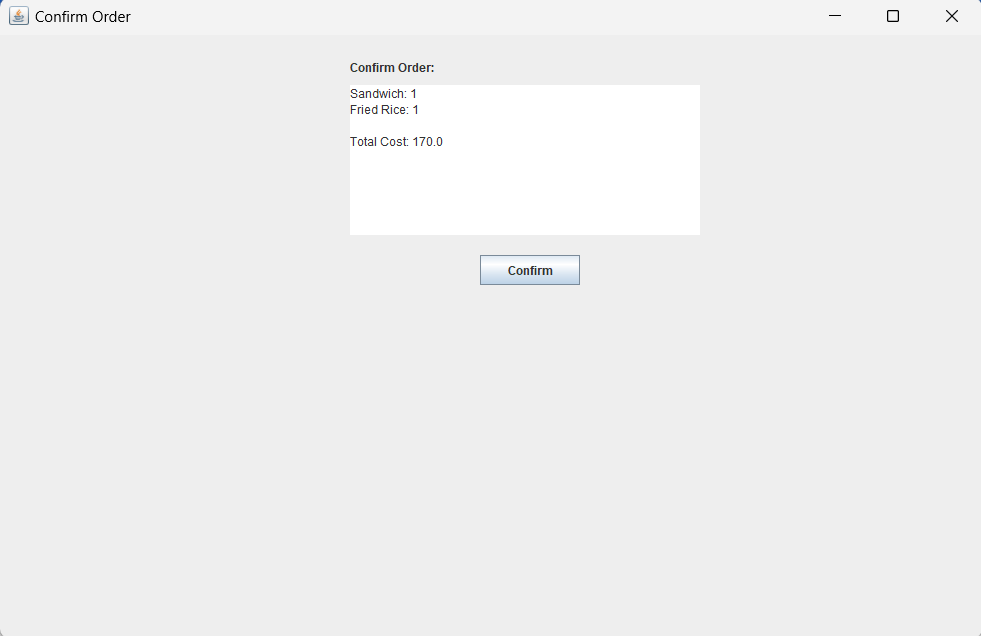
}

});

setVisible(true);

}

}



**Order History Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.io.\*;

import java.time.LocalDateTime;

import java.time.format.DateTimeFormatter;

import java.util.ArrayList;

import java.util.List;

public class OrderHistoryFrame extends JFrame {

private List<String[]> orderHistory;

private JTable orderTable;

private JLabel totalCostLabel;

private double totalOrderCost;

public OrderHistoryFrame() {

orderHistory = new ArrayList<>();

setTitle("Order History");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLayout(new BorderLayout());

JLabel historyLabel = new JLabel("Order History:");

historyLabel.setHorizontalAlignment(SwingConstants.CENTER);

add(historyLabel, BorderLayout.NORTH);

DefaultTableModel model = new DefaultTableModel();

model.addColumn("Items");

model.addColumn("Total Quantity");

model.addColumn("Total Cost");

model.addColumn("Time");

orderTable = new JTable(model);

JScrollPane scrollPane = new JScrollPane(orderTable);

add(scrollPane, BorderLayout.CENTER);

totalCostLabel = new JLabel("Total Order Cost: 0.0");

totalCostLabel.setHorizontalAlignment(SwingConstants.CENTER);

add(totalCostLabel, BorderLayout.SOUTH);

loadOrderHistory();

populateTable();

updateTotalOrderCost();

setVisible(true);

}

private void populateTable() {

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

for (String[] order : orderHistory) {

model.addRow(order);

}

}

private void updateTotalOrderCost() {

totalOrderCost = 0;

for (String[] order : orderHistory) {

totalOrderCost += Double.parseDouble(order[2]);

}

totalCostLabel.setText("Total Order Cost: " + totalOrderCost);

}

public static void addOrder(List<StockItem> orderDetails, double totalCost) {

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");

String time = LocalDateTime.now().format(formatter);

StringBuilder items = new StringBuilder();

int totalQuantity = 0;

for (StockItem item : orderDetails) {

items.append(item.getName())

.append(" (Qty: ")

.append(item.getOrderQuantity())

.append(", Unit Price: ")

.append(item.getPrice())

.append("); ");

totalQuantity += item.getOrderQuantity();

}

if (items.length() > 0) {

items.setLength(items.length() - 2);

}

String[] order = new String[4];

order[0] = items.toString();

order[1] = String.valueOf(totalQuantity);

order[2] = String.valueOf(totalCost);

order[3] = time;

List<String[]> tempOrderHistory = new ArrayList<>();

tempOrderHistory.add(order);

saveOrderHistory(tempOrderHistory);

}

private static void saveOrderHistory(List<String[]> tempOrderHistory) {

try (BufferedWriter writer = new BufferedWriter(new FileWriter("orderHistory.txt", true))) {

for (String[] order : tempOrderHistory) {

writer.write(String.join("|||", order));

writer.newLine();

}

} catch (IOException ex) {

ex.printStackTrace();

}

}

private void loadOrderHistory() {

orderHistory.clear();

try (BufferedReader reader = new BufferedReader(new FileReader("orderHistory.txt"))) {

String line;

while ((line = reader.readLine()) != null) {

orderHistory.add(line.split("\\|\\|\\|"));

}

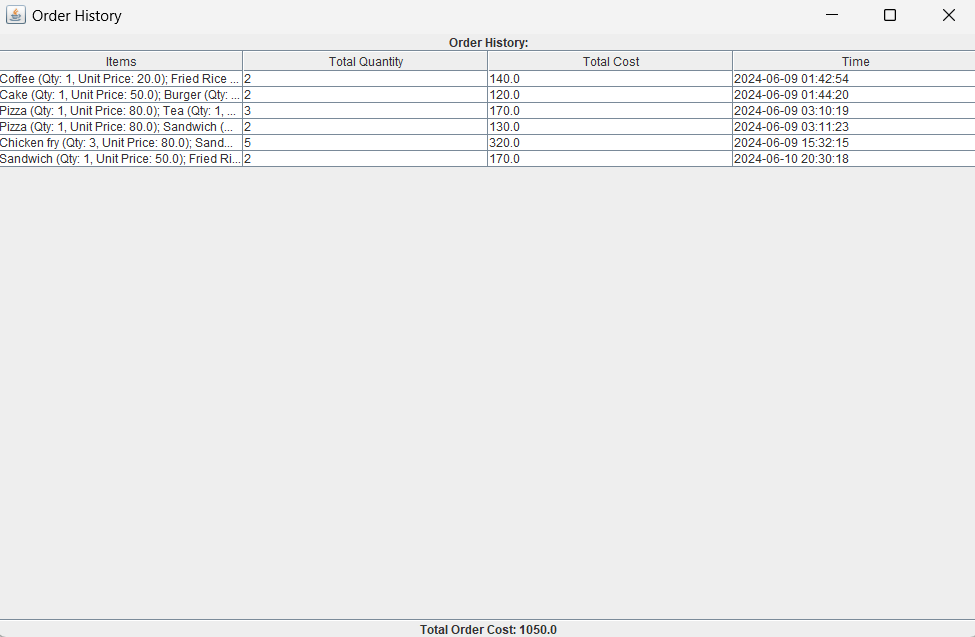
} catch (IOException ex) {

ex.printStackTrace();

}

}

}



**Raw Materials Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.io.\*;

public class RawMaterialsFrame extends JFrame {

private JTextArea rawMaterialsArea;

private JButton increaseStockButton;

private JButton updateStockButton;

private JTextField itemField;

private JTextField quantityField;

private JButton viewTotalStockButton;

public RawMaterialsFrame() {

setTitle("Raw Materials");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE);

setLayout(new BorderLayout());

rawMaterialsArea = new JTextArea();

rawMaterialsArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(rawMaterialsArea);

add(scrollPane, BorderLayout.CENTER);

JPanel inputPanel = new JPanel();

inputPanel.setLayout(new GridLayout(5, 2));

inputPanel.add(new JLabel("Item:"));

itemField = new JTextField();

inputPanel.add(itemField);

inputPanel.add(new JLabel("Quantity:"));

quantityField = new JTextField();

inputPanel.add(quantityField);

increaseStockButton = new JButton("Increase Stock");

inputPanel.add(increaseStockButton);

updateStockButton = new JButton("Update Stock");

inputPanel.add(updateStockButton);

add(inputPanel, BorderLayout.SOUTH);

increaseStockButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

increaseStock();

}

});

updateStockButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

updateStock();

}

});

loadRawMaterials();

}

private void loadRawMaterials() {

rawMaterialsArea.setText("");

try (BufferedReader br = new BufferedReader(new FileReader("raw\_materials.txt"))) {

String line;

while ((line = br.readLine()) != null) {

rawMaterialsArea.append(line + "\n");

}

} catch (IOException e) {

e.printStackTrace();

}

}

private void increaseStock() {

String item = itemField.getText();

int quantity;

try {

quantity = Integer.parseInt(quantityField.getText());

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(this, "Invalid quantity!", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

boolean itemFound = false;

File tempFile = new File("raw\_materials\_temp.txt");

File originalFile = new File("raw\_materials.txt");

try (BufferedReader br = new BufferedReader(new FileReader(originalFile));

PrintWriter pw = new PrintWriter(new FileWriter(tempFile))) {

String line;

while ((line = br.readLine()) != null) {

String[] parts = line.split(",");

if (parts[0].equals(item)) {

int currentQuantity = Integer.parseInt(parts[1]);

currentQuantity += quantity;

line = parts[0] + "," + currentQuantity;

itemFound = true;

}

pw.println(line);

}

if (!itemFound) {

pw.println(item + "," + quantity);

}

} catch (IOException e) {

e.printStackTrace();

}

if (originalFile.delete()) {

if (!tempFile.renameTo(originalFile)) {

JOptionPane.showMessageDialog(this, "Failed to rename temporary file!", "Error", JOptionPane.ERROR\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Failed to delete original file!", "Error", JOptionPane.ERROR\_MESSAGE);

}

loadRawMaterials();

}

private void updateStock() {

String item = itemField.getText();

int quantity;

try {

quantity = Integer.parseInt(quantityField.getText());

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(this, "Invalid quantity!", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

boolean itemFound = false;

File tempFile = new File("raw\_materials\_temp.txt");

File originalFile = new File("raw\_materials.txt");

try (BufferedReader br = new BufferedReader(new FileReader(originalFile));

PrintWriter pw = new PrintWriter(new FileWriter(tempFile))) {

String line;

while ((line = br.readLine()) != null) {

String[] parts = line.split(",");

if (parts[0].equals(item)) {

line = parts[0] + "," + quantity;

itemFound = true;

}

pw.println(line);

}

if (!itemFound) {

JOptionPane.showMessageDialog(this, "Item not found!", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

} catch (IOException e) {

e.printStackTrace();

}

if (originalFile.delete()) {

if (!tempFile.renameTo(originalFile)) {

JOptionPane.showMessageDialog(this, "Failed to rename temporary file!", "Error", JOptionPane.ERROR\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Failed to delete original file!", "Error", JOptionPane.ERROR\_MESSAGE);

}

loadRawMaterials();

}

private void viewTotalStock() {

StringBuilder totalStock = new StringBuilder("Total Stock:\n");

try (BufferedReader br = new BufferedReader(new FileReader("raw\_materials.txt"))) {

String line;

while ((line = br.readLine()) != null) {

totalStock.append(line).append("\n");

}

} catch (IOException e) {

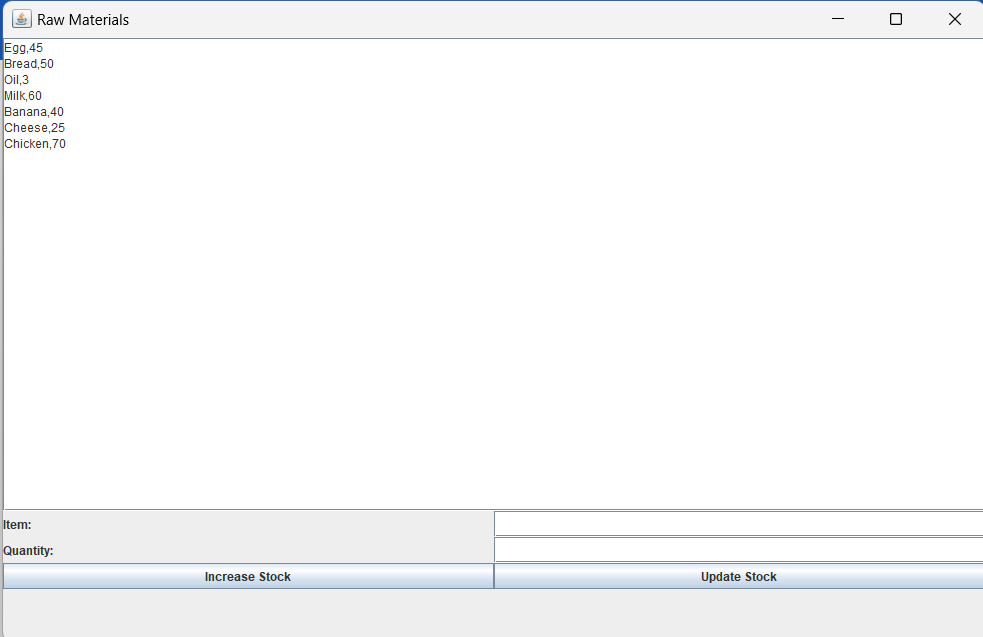
e.printStackTrace();

}

JOptionPane.showMessageDialog(this, totalStock.toString(), "Total Stock", JOptionPane.INFORMATION\_MESSAGE);

}

}



**Distribute Raw Materials Frame:**

package canteenmanagementsystem;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.io.\*;

import java.util.\*;

public class DistributionFrame extends JFrame {

private JTextArea distributionArea;

private JComboBox<String> chiefComboBox;

private JComboBox<String> itemComboBox;

private JTextField quantityField;

private JButton distributeButton;

private JButton viewHistoryButton;

public DistributionFrame() {

setTitle("Distribution Management");

setSize(1000, 650);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLayout(new BorderLayout());

distributionArea = new JTextArea();

distributionArea.setEditable(false);

JScrollPane scrollPane = new JScrollPane(distributionArea);

add(scrollPane, BorderLayout.CENTER);

JPanel inputPanel = new JPanel();

inputPanel.setLayout(new GridLayout(4, 2));

inputPanel.add(new JLabel("Chief No:"));

chiefComboBox = new JComboBox<>(new String[]{"Chief1", "Chief2", "Chief3"});

inputPanel.add(chiefComboBox);

inputPanel.add(new JLabel("Item:"));

itemComboBox = new JComboBox<>();

loadItemsIntoComboBox();

inputPanel.add(itemComboBox);

inputPanel.add(new JLabel("Quantity:"));

quantityField = new JTextField();

inputPanel.add(quantityField);

distributeButton = new JButton("Distribute");

inputPanel.add(distributeButton);

viewHistoryButton = new JButton("View History");

inputPanel.add(viewHistoryButton);

add(inputPanel, BorderLayout.SOUTH);

distributeButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

distributeItems();

}

});

viewHistoryButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

viewDistributionHistory();

}

});

loadDistributionHistory();

}

private void loadDistributionHistory() {

distributionArea.setText("");

try (BufferedReader br = new BufferedReader(new FileReader("distribution\_history.txt"))) {

String line;

while ((line = br.readLine()) != null) {

distributionArea.append(line + "\n");

}

} catch (IOException e) {

e.printStackTrace();

}

}

private void loadItemsIntoComboBox() {

try (BufferedReader br = new BufferedReader(new FileReader("raw\_materials.txt"))) {

String line;

while ((line = br.readLine()) != null) {

String[] parts = line.split(",");

itemComboBox.addItem(parts[0]);

}

} catch (IOException e) {

e.printStackTrace();

}

}

private void distributeItems() {

String chief = (String) chiefComboBox.getSelectedItem();

String item = (String) itemComboBox.getSelectedItem();

int quantity;

try {

quantity = Integer.parseInt(quantityField.getText());

} catch (NumberFormatException e) {

JOptionPane.showMessageDialog(this, "Invalid quantity!", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

boolean itemFound = false;

File tempFile = new File("raw\_materials\_temp.txt");

File originalFile = new File("raw\_materials.txt");

try (BufferedReader br = new BufferedReader(new FileReader(originalFile));

PrintWriter pw = new PrintWriter(new FileWriter(tempFile))) {

String line;

while ((line = br.readLine()) != null) {

String[] parts = line.split(",");

if (parts[0].equals(item)) {

int currentQuantity = Integer.parseInt(parts[1]);

if (currentQuantity >= quantity) {

currentQuantity -= quantity;

line = parts[0] + "," + currentQuantity;

itemFound = true;

} else {

JOptionPane.showMessageDialog(this, "Not enough stock!", "Error", JOptionPane.ERROR\_MESSAGE);

return;

}

}

pw.println(line);

}

} catch (IOException e) {

e.printStackTrace();

}

if (itemFound) {

if (originalFile.delete()) {

if (!tempFile.renameTo(originalFile)) {

JOptionPane.showMessageDialog(this, "Failed to rename temporary file!", "Error", JOptionPane.ERROR\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Failed to delete original file!", "Error", JOptionPane.ERROR\_MESSAGE);

}

try (PrintWriter pw = new PrintWriter(new FileWriter("distribution\_history.txt", true))) {

pw.println(chief + "," + item + "," + quantity + "," + new Date());

} catch (IOException e) {

e.printStackTrace();

}

loadDistributionHistory();

} else {

JOptionPane.showMessageDialog(this, "Item not found!", "Error", JOptionPane.ERROR\_MESSAGE);

}

}

private void viewDistributionHistory() {

StringBuilder history = new StringBuilder("Distribution History:\n");

try (BufferedReader br = new BufferedReader(new FileReader("distribution\_history.txt"))) {

String line;

while ((line = br.readLine()) != null) {

history.append(line).append("\n");

}

} catch (IOException e) {

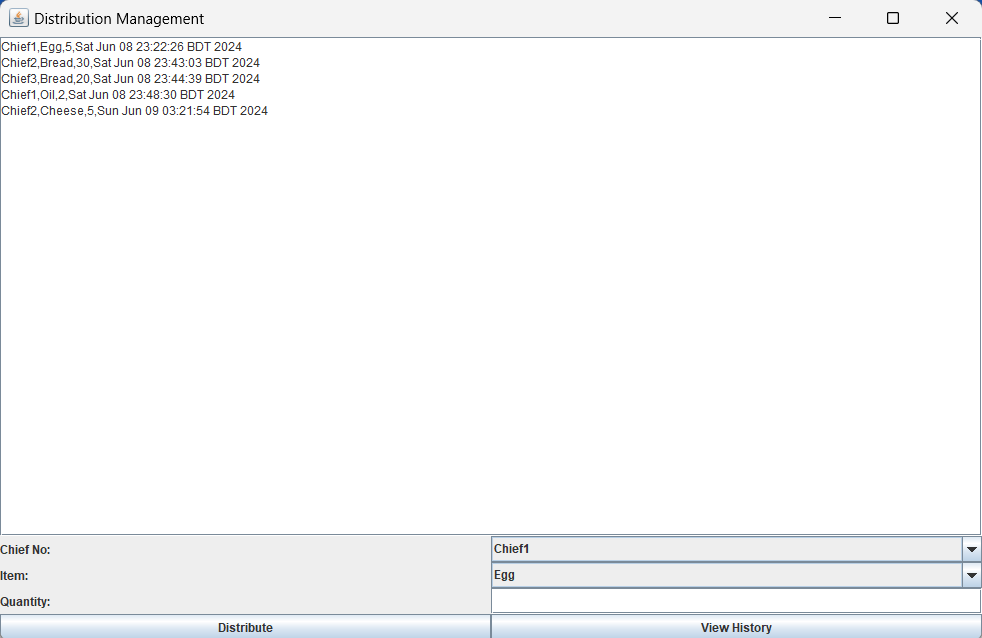
e.printStackTrace();

}

JOptionPane.showMessageDialog(this, history.toString(), "Distribution History", JOptionPane.INFORMATION\_MESSAGE);

}

}



**Conclusion:** The Canteen Management System project effectively demonstrates the potential of using JSwing to develop powerful and user-friendly management software. By integrating essential functions such as inventory updates, order processing and raw material management, the system ensures smooth and efficient operations in the canteen. The use of text files for data storage and retrieval ensures simplicity and reliability in processing important information. Overall, this project not only improves the management of canteen operations but also demonstrates the practical application of Java in creating dynamic and functional software solutions.