

**ASSIGNMENT #4**

**Assigned Date: 10/12/24**

**Due Date: 20/12/24**

**M UMAR AMIN(SP24-BSE-055)**

**ALI HASSAN(SP24-BSE-140)**

**HAMMAD ALEEM(SP24-BSE-073)**

# SIGNUP .JAVA

**package com.example;**

**import javafx.application.Application;**

**import javafx.geometry.Insets;**

**import javafx.geometry.Pos;**

**import javafx.scene.Scene;**

**import javafx.scene.control.\*;**

**import javafx.scene.layout.\*;**

**import javafx.stage.Stage;**

**import java.io.\*;**

**public class signupButton extends Application {**

**private void saveEmailToFile(String email, String fileName) throws IOException {**

**File file = new File(fileName);**

**if (!file.exists()) {**

**file.createNewFile(); // Ensure the file is created if it doesn't exist**

**}**

**try (FileWriter writer = new FileWriter(file, true)) {**

**writer.write(email + System.lineSeparator());**

**}**

**}**

**private boolean isEmailExists(String email, String fileName) throws IOException {**

**File file = new File(fileName);**

**if (!file.exists()) {**

**return false; // No file means no email is registered yet**

**}**

**try (BufferedReader reader = new BufferedReader(new FileReader(file))) {**

**String line;**

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

**if (line.trim().equalsIgnoreCase(email.trim())) {**

**return true; // Email exists**

**}**

**}**

**}**

**return false; // Email does not exist**

**}**

**@Override**

**public void start(Stage stage) {**

**Label titLabel = new Label("Inventory Management System");**

**titLabel.setStyle("-fx-font-size:45px");**

**Button signupButton = new Button("SIGNUP");**

**Button loginButton = new Button("LOGIN");**

**signupButton.setStyle("-fx-background-color:white;");**

**loginButton.setStyle("-fx-background-color:white;");**

**signupButton.setPrefHeight(60);**

**signupButton.setPrefWidth(150);**

**loginButton.setPrefHeight(60);**

**loginButton.setPrefWidth(150);**

**HBox topBox = new HBox(signupButton, loginButton);**

**topBox.setPadding(new Insets(150, 0, 0, 170));**

**topBox.setSpacing(10);**

**Label nameLabel = new Label("Username:");**

**nameLabel.setStyle("-fx-font-size:14px");**

**TextField nameField = new TextField();**

**nameField.setPrefWidth(250);**

**nameField.setPrefHeight(35);**

**HBox nameBox = new HBox(nameLabel, nameField);**

**nameBox.setSpacing(10);**

**nameBox.setPadding(new Insets(0,0,0,120));**

**Label emailLabel = new Label("Email:");**

**emailLabel.setStyle("-fx-font-size:15px");**

**TextField emailField = new TextField();**

**emailField.setPrefWidth(250);**

**emailField.setPrefHeight(35);**

**HBox emailBox = new HBox(emailLabel, emailField);**

**emailBox.setSpacing(37);**

**emailBox.setPadding(new Insets(0,0,0,120));**

**Label passwordLabel = new Label("Password:");**

**passwordLabel.setStyle("-fx-font-size:15px");**

**PasswordField passwordField = new PasswordField();**

**passwordField.setPrefWidth(250);**

**passwordField.setPrefHeight(35);**

**HBox passwordBox = new HBox(passwordLabel, passwordField);**

**passwordBox.setSpacing(10);**

**passwordBox.setPadding(new Insets(0,0,0,120));**

**Button actionButton = new Button("Sign Up");**

**actionButton.setPrefWidth(100);**

**actionButton.setPrefHeight(30);**

**HBox bottomBox = new HBox(actionButton);**

**bottomBox.setPadding(new Insets(0, 0, 0, 250));**

**File file = new File("products.txt");**

**if (!file.exists()) {**

**try {**

**file.createNewFile();**

**} catch (IOException e) {**

**e.printStackTrace();**

**showAlert(Alert.AlertType.ERROR, "Error", "Failed to create data file.");**

**}**

**}**

**signupButton.setOnAction(e -> {**

**loginButton.setStyle("-fx-background-color:white");**

**signupButton.setStyle("-fx-background-color:#16A085");**

**if (!nameBox.getChildren().contains(nameLabel)) {**

**nameBox.getChildren().addAll(nameLabel, nameField);**

**}**

**actionButton.setText("Sign Up");**

**actionButton.setOnAction(signupEvent -> {**

**String username = nameField.getText();**

**String email = emailField.getText();**

**String password = passwordField.getText();**

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

**showAlert(Alert.AlertType.ERROR, "Error", "All fields must be filled out!");**

**} else {**

**try {**

**if (isEmailExists(email, "products.txt")) {**

**showAlert(Alert.AlertType.ERROR, "Error", "This email is already registered!");**

**} else {**

**saveEmailToFile(email, "products.txt");**

**showAlert(Alert.AlertType.INFORMATION, "Signup Success", "Welcome, " + username + "!");**

**nameField.clear();**

**emailField.clear();**

**passwordField.clear();**

**}**

**} catch (IOException ex) {**

**showAlert(Alert.AlertType.ERROR, "Error", "An error occurred while accessing the file.");**

**ex.printStackTrace();**

**}**

**}**

**});**

**});**

**loginButton.setOnAction(e -> {**

**loginButton.setStyle("-fx-background-color:#16A085");**

**signupButton.setStyle("-fx-background-color:white");**

**nameBox.getChildren().removeAll(nameLabel, nameField);**

**actionButton.setText("Log In");**

**actionButton.setOnAction(loginEvent -> {**

**String loginEmail = emailField.getText();**

**String loginPassword = passwordField.getText();**

**if (loginEmail.isEmpty() || loginPassword.isEmpty()) {**

**showAlert(Alert.AlertType.ERROR, "Error", "Email and password cannot be empty!");**

**} else {**

**showAlert(Alert.AlertType.INFORMATION, "Login Success", "Welcome back!");**

**emailField.clear();**

**passwordField.clear();**

**InventoryManagement im1=new InventoryManagement();**

**im1.start(stage);**

**}**

**});**

**});**

**GridPane gridPane = new GridPane();**

**gridPane.add(topBox, 0, 1);**

**gridPane.add(nameBox, 0, 2);**

**gridPane.add(emailBox, 0, 3);**

**gridPane.add(passwordBox, 0, 4);**

**gridPane.add(bottomBox, 0, 5);**

**gridPane.add(titLabel,0,0);**

**gridPane.setVgap(30);**

**gridPane.setAlignment(Pos.TOP\_CENTER);**

**Scene scene = new Scene(gridPane, 1000, 700);**

**scene.getStylesheets().add(getClass().getResource("Signup.css").toExternalForm());**

**stage.setScene(scene);**

**stage.show();**

**}**

**private void showAlert(Alert.AlertType type, String title, String message) {**

**Alert alert = new Alert(type);**

**alert.setTitle(title);**

**alert.setHeaderText(null);**

**alert.setContentText(message);**

**alert.showAndWait();**

**}**

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

**launch(args);**

**}**

**}**

# INVENTORY MANGEMENT SYSTEM.JAVA

package com.example;

import javafx.application.Application;

import javafx.geometry.Insets;

import javafx.scene.Scene;

import javafx.scene.control.\*;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.image.Image;

import javafx.scene.image.ImageView;

import javafx.scene.layout.\*;

import javafx.stage.FileChooser;

import javafx.stage.Stage;

import java.io.\*;

import java.util.ArrayList;

import java.util.List;

public class InventoryManagement extends Application {

    private List<Product> products = new ArrayList<>();

    private TableView<Product> productTable = new TableView<>();

    @Override

    public void start(Stage primaryStage) {

        primaryStage.setTitle("Inventory Management System");

        // Main Layout

        BorderPane mainLayout = new BorderPane();

        // Navigation Bar

        HBox navBar = new HBox(10);

        navBar.setPadding(new Insets(10));

        navBar.setStyle("-fx-background-color:rgb(6, 27, 49); -fx-padding: 10;");

        // Buttons

        Button addButton = new Button("Add Product");

        styleButton(addButton, "#27AE60");

        Button removeButton = new Button("Remove Product");

        styleButton(removeButton, "#E74C3C");

        Button saveButton = new Button("Save to File");

        styleButton(saveButton, "#F39C12");

        Button loadButton = new Button("Load from File");

        styleButton(loadButton, "#9B59B6");

        Button buyButton = new Button("Buy Product");

        styleButton(buyButton, "#1ABC9C");

        Button sellButton = new Button("Sell Product");

        styleButton(sellButton, "#E67E22");

        Button updateButton = new Button("Update Product");

        styleButton(updateButton, "#34495E");

        Button signupButton = new Button("Signup");

        styleButton(signupButton, "#8E44AD");

        Button currencyexchangebutton =new Button("Currency Exchange");

        styleButton(currencyexchangebutton, "#8E44AD");

        Button exitButton = new Button("Exit");

        styleButton(exitButton, "#E74C3C");

        // Adding buttons to the navigation bar

        navBar.getChildren().addAll(addButton, removeButton, saveButton, loadButton, buyButton, sellButton, updateButton, signupButton,currencyexchangebutton,exitButton);

        mainLayout.setTop(navBar);

        navBar.setSpacing(30);

        // Center Area for displaying the table

        mainLayout.setCenter(createTableView());

        // Event Handlers

        addButton.setOnAction(e -> addProduct());

        removeButton.setOnAction(e -> removeProduct());

        saveButton.setOnAction(e -> saveToFile(primaryStage));

        loadButton.setOnAction(e -> loadFromFile(primaryStage));

        buyButton.setOnAction(e -> buyProduct());

        sellButton.setOnAction(e -> sellProduct());

        updateButton.setOnAction(e -> updateProduct());

        signupButton.setOnAction(e -> openSignupStage());

        currencyexchangebutton.setOnAction(e->openExchangerateStage());

        exitButton.setOnAction(e -> primaryStage.close());

        // Scene and Stage

        Scene scene = new Scene(mainLayout, 1100, 700);

        scene.getStylesheets().add(getClass().getResource("Signup.css").toExternalForm());

        primaryStage.setScene(scene);

        primaryStage.show();

    }

    private void styleButton(Button button, String color) {

        button.setStyle("-fx-background-color: " + color + "; -fx-text-fill: white; -fx-font-size: 14px; -fx-padding: 10px;");

    }

    private TableView<Product> createTableView() {

        TableColumn<Product, String> nameColumn = new TableColumn<>("Name");

        nameColumn.setCellValueFactory(new PropertyValueFactory<>("name"));

        TableColumn<Product, Double> priceColumn = new TableColumn<>("Price");

        priceColumn.setCellValueFactory(new PropertyValueFactory<>("price"));

        TableColumn<Product, Integer> quantityColumn = new TableColumn<>("Quantity");

        quantityColumn.setCellValueFactory(new PropertyValueFactory<>("quantity"));

        TableColumn<Product, String> imageColumn = new TableColumn<>("Image");

        imageColumn.setCellValueFactory(new PropertyValueFactory<>("imageUrl"));

        imageColumn.setCellFactory(param -> new TableCell<>() {

            private final ImageView imageView = new ImageView();

            @Override

            protected void updateItem(String imageUrl, boolean empty) {

                super.updateItem(imageUrl, empty);

                if (empty || imageUrl == null) {

                    setGraphic(null);

                } else {

                    imageView.setImage(new Image(imageUrl));

                    imageView.setFitHeight(50);

                    imageView.setFitWidth(50);

                    setGraphic(imageView);

                }

            }

        });

        productTable.getColumns().addAll(nameColumn, priceColumn, quantityColumn, imageColumn);

        productTable.setColumnResizePolicy(TableView.CONSTRAINED\_RESIZE\_POLICY);

        productTable.setStyle("-fx-font-size: 14px;");

        ScrollPane scrollPane = new ScrollPane(productTable);

        scrollPane.setFitToWidth(true);

        scrollPane.setFitToHeight(true);

        return productTable;

    }

    private void addProduct() {

        Stage addStage = new Stage();

        addStage.setTitle("Add Product");

        VBox layout = new VBox(10);

        layout.setPadding(new Insets(10));

        layout.setStyle("-fx-background-color:rgb(19, 189, 231);");

        // Create TextField for product name

        TextField nameField = new TextField();

        nameField.setPromptText("Enter product name");

        nameField.setStyle("-fx-background-color: #ECF0F1; -fx-text-fill: #2C3E50; -fx-prompt-text-fill: #7F8C8D;");

        // Create TextField for price

        TextField priceField = new TextField();

        priceField.setPromptText("Enter price");

        priceField.setStyle("-fx-background-color: #ECF0F1; -fx-text-fill: #2C3E50; -fx-prompt-text-fill: #7F8C8D;");

        // Create TextField for quantity

        TextField quantityField = new TextField();

        quantityField.setPromptText("Enter quantity");

        quantityField.setStyle("-fx-background-color:rgb(247, 247, 247); -fx-text-fill: #2C3E50; -fx-prompt-text-fill: #7F8C8D;");

        // Create button to select image

        Button imageButton = new Button("Select Image");

        styleButton(imageButton, "#3498DB");

        final String[] imageUrl = {null};

        imageButton.setOnAction(e -> {

            FileChooser fileChooser = new FileChooser();

            fileChooser.getExtensionFilters().add(new FileChooser.ExtensionFilter("Image Files", "\*.png", "\*.jpg", "\*.gif"));

            File file = fileChooser.showOpenDialog(addStage);

            if (file != null) {

                imageUrl[0] = file.toURI().toString();

            }

        });

        // Create button to add product

        Button addButton = new Button("Add");

        styleButton(addButton, "#27AE60");

        addButton.setOnAction(e -> {

            try {

                String name = nameField.getText();

                double price = Double.parseDouble(priceField.getText());

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

                products.add(new Product(name, price, quantity, imageUrl[0]));

                addStage.close();

                updateTable();

            } catch (NumberFormatException ex) {

                showAlert("Invalid input!", "Please enter valid product details.");

            }

        });

        layout.getChildren().addAll(new Label("Name:"), nameField,

                new Label("Price:"), priceField,

                new Label("Quantity:"), quantityField,

                imageButton, addButton);

        Scene scene = new Scene(layout, 500, 350);

        addStage.setScene(scene);

        addStage.show();

    }

    private void removeProduct() {

        Stage removeStage = new Stage();

        removeStage.setTitle("Remove Product");

        VBox layout = new VBox(10);

        layout.setPadding(new Insets(10));

        layout.setStyle("-fx-background-color:rgb(7, 249, 249);");

        TextField nameField = new TextField();

        nameField.setPromptText("Enter product name to remove");

        TextField quantityField = new TextField();

        quantityField.setPromptText("Enter quantity to remove");

        Button removeButton = new Button("Remove");

        styleButton(removeButton, "#E74C3C");

        removeButton.setOnAction(e -> {

            try {

                String name = nameField.getText();

                int quantityToRemove = Integer.parseInt(quantityField.getText());

                for (Product product : products) {

                    if (product.getName().equalsIgnoreCase(name)) {

                        if (product.getQuantity() >= quantityToRemove) {

                            product.setQuantity(product.getQuantity() - quantityToRemove);

                            if (product.getQuantity() == 0) {

                                products.remove(product);

                            }

                            showAlert("Success", "Removed " + quantityToRemove + " units of " + name);

                            removeStage.close();

                            updateTable();

                            return;

                        } else {

                            showAlert("Error", "Not enough stock to remove!");

                            return;

                        }

                    }

                }

                showAlert("Error", "Product not found!");

            } catch (NumberFormatException ex) {

                showAlert("Invalid input!", "Please enter a valid quantity.");

            }

        });

        layout.getChildren().addAll(new Label("Name:"), nameField, new Label("Quantity:"), quantityField, removeButton);

        Scene scene = new Scene(layout, 500, 350);

        removeStage.setScene(scene);

        removeStage.show();

    }

    private void saveToFile(Stage stage) {

        FileChooser fileChooser = new FileChooser();

        fileChooser.setTitle("Save Products");

        fileChooser.getExtensionFilters().add(new FileChooser.ExtensionFilter("Text Files", "\*.txt"));

        File file = fileChooser.showSaveDialog(stage);

        if (file != null) {

            try (BufferedWriter writer = new BufferedWriter(new FileWriter(file))) {

                for (Product product : products) {

                    writer.write(product.toString() + "\n");

                }

                showAlert("Success", "Products saved to file: " + file.getName());

            } catch (IOException e) {

                showAlert("Error", "Error saving to file!");

            }

        }

    }

    private void loadFromFile(Stage stage) {

        FileChooser fileChooser = new FileChooser();

        fileChooser.setTitle("Load Products");

        fileChooser.getExtensionFilters().add(new FileChooser.ExtensionFilter("Text Files", "\*.txt"));

        File file = fileChooser.showOpenDialog(stage);

        if (file != null) {

            try (BufferedReader reader = new BufferedReader(new FileReader(file))) {

                products.clear();

                String line;

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

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

                    String name = parts[0].split(": ")[1];

                    double price = Double.parseDouble(parts[1].split(": ")[1]);

                    int quantity = Integer.parseInt(parts[2].split(": ")[1]);

                    String imageUrl = parts.length > 3 ? parts[3].split(": ")[1] : null;

                    products.add(new Product(name, price, quantity, imageUrl));

                }

                showAlert("Success", "Products loaded from file: " + file.getName());

                updateTable();

            } catch (IOException e) {

                showAlert("Error", "Error loading from file!");

            }

        }

    }

    private void buyProduct() {

        Stage buyStage = new Stage();

        buyStage.setTitle("Buy Product");

        VBox layout = new VBox(10);

        layout.setPadding(new Insets(10));

        layout.setStyle("-fx-background-color:rgb(5, 205, 255);");

        TextField nameField = new TextField();

        nameField.setPromptText("Enter product name to buy");

        TextField quantityField = new TextField();

        quantityField.setPromptText("Enter quantity to buy");

        Button buyButton = new Button("Buy");

        styleButton(buyButton, "#1ABC9C");

        buyButton.setOnAction(e -> {

            String name = nameField.getText();

            try {

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

                for (Product product : products) {

                    if (product.getName().equalsIgnoreCase(name)) {

                        product.setQuantity(product.getQuantity() + quantity);

                        showAlert("Success", "Bought " + quantity + " units of " + name);

                        buyStage.close();

                        updateTable();

                        return;

                    }

                }

                showAlert("Error", "Product not found!");

            } catch (NumberFormatException ex) {

                showAlert("Invalid input!", "Please enter a valid quantity.");

            }

        });

        layout.getChildren().addAll(new Label("Name:"), nameField,

                new Label("Quantity:"), quantityField, buyButton);

        Scene scene = new Scene(layout, 500, 350);

        buyStage.setScene(scene);

        buyStage.show();

    }

    private void sellProduct() {

        Stage sellStage = new Stage();

        sellStage.setTitle("Sell Product");

        VBox layout = new VBox(10);

        layout.setPadding(new Insets(10));

        layout.setStyle("-fx-background-color:rgb(0, 201, 252);");

        TextField nameField = new TextField();

        nameField.setPromptText("Enter product name to sell");

        TextField quantityField = new TextField();

        quantityField.setPromptText("Enter quantity to sell");

        Button sellButton = new Button("Sell");

        styleButton(sellButton, "#E67E22");

        sellButton.setOnAction(e -> {

            String name = nameField.getText();

            try {

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

                for (Product product : products) {

                    if (product.getName().equalsIgnoreCase(name)) {

                        if (product.getQuantity() >= quantity) {

                            product.setQuantity(product.getQuantity() - quantity);

                            showAlert("Success", "Sold " + quantity + " units of " + name);

                            sellStage.close();

                            updateTable();

                            return;

                        } else {

                            showAlert("Error", "Not enough stock to sell!");

                            return;

                        }

                    }

                }

                showAlert("Error", "Product not found!");

            } catch (NumberFormatException ex) {

                showAlert("Invalid input!", "Please enter a valid quantity.");

            }

        });

        layout.getChildren().addAll(new Label("Name:"), nameField,

                new Label("Quantity:"), quantityField, sellButton);

        Scene scene = new Scene(layout, 500, 350);

        sellStage.setScene(scene);

        sellStage.show();

    }

    private void updateProduct() {

        Stage updateStage = new Stage();

        updateStage.setTitle("Update Product");

        VBox layout = new VBox(10);

        layout.setPadding(new Insets(10));

        layout.setStyle("-fx-background-color:rgb(0, 204, 255);");

        TextField nameField = new TextField();

        nameField.setPromptText("Enter product name to update");

        TextField newPriceField = new TextField();

        newPriceField.setPromptText("Enter new price");

        TextField newQuantityField = new TextField();

        newQuantityField.setPromptText("Enter new quantity");

        Button updateButton = new Button("Update");

        styleButton(updateButton, "#34495E");

        updateButton.setOnAction(e -> {

            String name = nameField.getText();

            try {

                double newPrice = Double.parseDouble(newPriceField.getText());

                int newQuantity = Integer.parseInt(newQuantityField.getText());

                for (Product product : products) {

                    if (product.getName().equalsIgnoreCase(name)) {

                        product.setPrice(newPrice);

                        product.setQuantity(newQuantity);

                        showAlert("Success", "Updated product: " + name);

                        updateStage.close();

                        updateTable();

                        return;

                    }

                }

                showAlert("Error", "Product not found!");

            } catch (NumberFormatException ex) {

                showAlert("Invalid input!", "Please enter valid product details.");

            }

        });

        layout.getChildren().addAll(new Label("Name:"), nameField,

                new Label("New Price:"), newPriceField,

                new Label("New Quantity:"), newQuantityField, updateButton);

        Scene scene = new Scene(layout, 500, 350);

        updateStage.setScene(scene);

        updateStage.show();

    }

    private void updateTable() {

        productTable.getItems().clear();

        productTable.getItems().addAll(products);

    }

    private void showAlert(String title, String message) {

        Alert alert = new Alert(Alert.AlertType.INFORMATION);

        alert.setTitle(title);

        alert.setHeaderText(null);

        alert.setContentText(message);

        alert.showAndWait();

    }

    private void openSignupStage() {

        signupButton  signupApp = new signupButton();

        Stage signupStage = new Stage();

        signupApp.start(signupStage);

    }

    private void openExchangerateStage() {

        Project exchangerate = new Project();

        Stage ert= new Stage();

        exchangerate.start(ert);

    }

    public static void main(String[] args) {

        launch(args);

    }

}

# APICAL.JAVA

**package com.example;**

**import java.io.BufferedReader;**

**import java.io.InputStreamReader;**

**import java.net.HttpURLConnection;**

**import java.net.URL;**

**import java.util.HashMap;**

**import java.util.Iterator;**

**import java.util.Map;**

**import org.json.JSONObject;**

**public class Apical {**

**public static Map<String, Double> getCurrencyExchangeRates(String baseCurrency) {**

**Map<String, Double> exchangeRates = new HashMap<>();**

**try {**

**String apiKey = "e3accbe4235dabde4e5b39f2";**

**String apiUrl = "https://v6.exchangerate-api.com/v6/" + apiKey + "/latest/" + baseCurrency;**

**URL url = new URL(apiUrl);**

**HttpURLConnection connection = (HttpURLConnection) url.openConnection();**

**connection.setRequestMethod("GET");**

**int statusCode = connection.getResponseCode();**

**if (statusCode == 200) {**

**BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));**

**String inputLine;**

**StringBuilder content = new StringBuilder();**

**while ((inputLine = in.readLine()) != null) {**

**content.append(inputLine);**

**}**

**in.close();**

**JSONObject jsonResponse = new JSONObject(content.toString());**

**if (jsonResponse.has("conversion\_rates")) {**

**JSONObject conversionRates = jsonResponse.getJSONObject("conversion\_rates");**

**Iterator<String> keys = conversionRates.keys();**

**while (keys.hasNext()) {**

**String key = keys.next();**

**double rate = conversionRates.getDouble(key);**

**exchangeRates.put(key, rate);**

**}**

**} else {**

**System.out.println("Error: 'conversion\_rates' not found in the API response.");**

**}**

**} else {**

**System.out.println("Error: Unable to get data. Response code: " + statusCode);**

**}**

**connection.disconnect();**

**} catch (Exception e) {**

**e.printStackTrace();**

**}**

**return exchangeRates;**

**}**

**}**

# PRODUCT.JAVA

package com.example;

public class Product {

    private String name;

    private double price;

    private int quantity;

    private String imageUrl;

    public Product(String name, double price, int quantity, String imageUrl) {

        this.name = name;

        this.price = price;

        this.quantity = quantity;

        this.imageUrl = imageUrl;

    }

    public String getName() {

        return name;

    }

    public double getPrice() {

        return price;

    }

    public void setPrice(double price) {

        this.price = price;

    }

    public int getQuantity() {

        return quantity;

    }

    public void setQuantity(int quantity) {

        this.quantity = quantity;

    }

    public String getImageUrl() {

        return imageUrl;

    }

    public void setImageUrl(String imageUrl) {

        this.imageUrl = imageUrl;

    }

    @Override

    public String toString() {

        return "Name: " + name + ", Price: " + price + ", Quantity: " + quantity + (imageUrl != null ? ", Image: " + imageUrl : "");

    }

}