# 

**Lab Assignment #4**

**Submitted to: Professor SHAHID BHATTI**

**Submitted by:**

**1:Muhammad Saad Badar (sp24-bse-082)  
  
2:Hafiz Abdullah Aslam (sp24-bse-028)  
  
3:Sohaib Athar (sp24-bse-115)**

Code Consists of 6 classes  
1: Main  
2: Dashboard  
3: Item  
4: AccountManager  
5: Account  
6: Style.CSS

Code is as Follows:

**Main Class**

package org.project;

import javafx.application.Application;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.CheckBox;

import javafx.scene.control.Hyperlink;

import javafx.scene.control.Label;

import javafx.scene.control.PasswordField;

import javafx.scene.control.TextField;

import javafx.scene.layout.StackPane;

import javafx.scene.layout.VBox;

import javafx.stage.Stage;

public class Main extends Application {

    public AccountManager accountManager = new AccountManager();

    Dashboard dashboard = new Dashboard();

    VBox loginForm, signUpForm, title, forgotPasswordForm;

    Stage stage = new Stage();

    @Override

    public void start(Stage stage) {

        loginForm = createLoginForm();

        title = titleForm();

        signUpForm = createRegisterForm();

        forgotPasswordForm = createForgotPasswordForm();

        StackPane root = new StackPane();

        StackPane.setAlignment(loginForm, Pos.CENTER\_RIGHT);

        StackPane.setAlignment(signUpForm, Pos.CENTER\_RIGHT);

        StackPane.setAlignment(forgotPasswordForm, Pos.CENTER\_RIGHT);

        StackPane.setAlignment(title, Pos.CENTER\_LEFT);

        loginForm.setMaxSize(300, 200);

        signUpForm.setMaxSize(300, 200);

        title.setMaxSize(300, 600);

        forgotPasswordForm.setMaxSize(300, 200);

        signUpForm.setVisible(false);

        forgotPasswordForm.setVisible(false);

        root.getChildren().addAll(title, loginForm, signUpForm, forgotPasswordForm);

        Scene scene = new Scene(root, 600, 400);

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

        stage.setTitle("Cafe Shop Management System");

        stage.setScene(scene);

        stage.setResizable(false);

        stage.show();

    }

    private VBox createLoginForm() {

        VBox login = new VBox(20);

        login.setPadding(new Insets(10, 50, 50, 50));

        login.setAlignment(Pos.CENTER\_LEFT);

        Label loginTitle = new Label("Login Account");

        loginTitle.getStyleClass().add("login-title");

        TextField usernameField = new TextField();

        usernameField.setPromptText("Username");

        usernameField.setPrefHeight(30);

        usernameField.setPrefWidth(200);

        PasswordField passwordField = new PasswordField();

        passwordField.setPromptText("Password");

        passwordField.setPrefHeight(30);

        passwordField.setPrefWidth(200);

        // TextField for showing password dynamically

        TextField passwordTextField = new TextField();

        passwordTextField.setManaged(false);

        passwordTextField.setVisible(false);

        CheckBox showPasswordCheckBox = new CheckBox("Show Password");

        showPasswordCheckBox.setStyle("-fx-text-fill: #000; -fx-font-family: \"Arial\";");

        passwordTextField.textProperty().bindBidirectional(passwordField.textProperty());

        // Toggle visibility on checkbox

        showPasswordCheckBox.setOnAction(e -> {

            if (showPasswordCheckBox.isSelected()) {

                passwordField.setVisible(false);

                passwordField.setManaged(false);

                passwordTextField.setVisible(true);

                passwordTextField.setManaged(true);

            } else {

                passwordTextField.setVisible(false);

                passwordTextField.setManaged(false);

                passwordField.setVisible(true);

                passwordField.setManaged(true);

            }

        });

        Hyperlink forgotPasswordLink = new Hyperlink("Forgot Password?");

        forgotPasswordLink.getStyleClass().add("forgot-password-link");

        forgotPasswordLink.setOnAction(e -> {

            loginForm.setVisible(false);

            forgotPasswordForm.setVisible(true);

        });

        Button loginButton = new Button("Login");

        loginButton.getStyleClass().add("button-login");

        loginButton.setPrefHeight(30);

        loginButton.setPrefWidth(200);

        Label warning = new Label();

        // Event handler for login button

        loginButton.setOnAction(e -> {

            String username = usernameField.getText();

            String password = passwordField.getText();

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

                warning.setText("Please fill in all fields.");

                warning.setStyle("-fx-text-fill: red; -fx-font-family: \"Arial\";");

                return;

            }

            if (accountManager.verify(username, password)) {

                warning.setText("Login successful!");

                warning.setStyle("-fx-text-fill: green; -fx-font-family: \"Arial\";");

                dashboard.showDashboard();

                usernameField.clear();

                passwordField.clear();

                this.stage = stage;

                stage.close();

            } else {

                warning.setText("Invalid username or password.");

                warning.setStyle("-fx-text-fill: red; -fx-font-family: \"Arial\";");

            }

        });

        login.getChildren().addAll(loginTitle, usernameField, passwordField, passwordTextField, showPasswordCheckBox, forgotPasswordLink, loginButton, warning);

        login.setMargin(loginTitle, new Insets(30, 0, 0, 0));

        return login;

    }

    private VBox createRegisterForm() {

        VBox register = new VBox(20);

        register.setPadding(new Insets(50, 50, 50, 50));

        register.setAlignment(Pos.CENTER\_LEFT);

        Label registerTitle = new Label("Register Account");

        registerTitle.getStyleClass().add("register-title");

        TextField usernameField = new TextField();

        usernameField.setPromptText("Username");

        usernameField.setPrefHeight(30);

        usernameField.setPrefWidth(200);

        TextField emailField = new TextField();

        emailField.setPromptText("Email");

        emailField.setPrefHeight(30);

        emailField.setPrefWidth(200);

        // Password fields

        PasswordField passwordField = new PasswordField();

        passwordField.setPromptText("Password");

        passwordField.setPrefHeight(30);

        passwordField.setPrefWidth(200);

        PasswordField confirmPasswordField = new PasswordField();

        confirmPasswordField.setPromptText("Confirm Password");

        confirmPasswordField.setPrefHeight(30);

        confirmPasswordField.setPrefWidth(200);

        // TextFields for showing passwords dynamically

        TextField passwordTextField = new TextField();

        passwordTextField.setManaged(false);

        passwordTextField.setVisible(false);

        TextField confirmPasswordTextField = new TextField();

        confirmPasswordTextField.setManaged(false);

        confirmPasswordTextField.setVisible(false);

        // Synchronize text properties

        passwordTextField.textProperty().bindBidirectional(passwordField.textProperty());

        confirmPasswordTextField.textProperty().bindBidirectional(confirmPasswordField.textProperty());

        // Checkbox to toggle password visibility

        CheckBox showPasswordCheckBox = new CheckBox("Show Passwords");

        showPasswordCheckBox.setStyle("-fx-text-fill: #000; -fx-font-family: \"Arial\";");

        showPasswordCheckBox.setOnAction(e -> {

            if (showPasswordCheckBox.isSelected()) {

                passwordField.setVisible(false);

                passwordField.setManaged(false);

                confirmPasswordField.setVisible(false);

                confirmPasswordField.setManaged(false);

                passwordTextField.setVisible(true);

                passwordTextField.setManaged(true);

                confirmPasswordTextField.setVisible(true);

                confirmPasswordTextField.setManaged(true);

            } else {

                passwordTextField.setVisible(false);

                passwordTextField.setManaged(false);

                confirmPasswordTextField.setVisible(false);

                confirmPasswordTextField.setManaged(false);

                passwordField.setVisible(true);

                passwordField.setManaged(true);

                confirmPasswordField.setVisible(true);

                confirmPasswordField.setManaged(true);

            }

        });

        Button signupButton = new Button("Signup");

        signupButton.getStyleClass().add("button-login");

        signupButton.setPrefHeight(30);

        signupButton.setPrefWidth(200);

        Label warning = new Label();

        // Event handler for signup button

        signupButton.setOnAction(e -> {

            String username = usernameField.getText();

            String email = emailField.getText();

            String password = passwordField.getText(); // Get the password from the PasswordField

            String confirmPassword = confirmPasswordField.getText(); // Get confirm password from PasswordField

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

                warning.setText("Please fill in all fields.");

                warning.setStyle("-fx-text-fill: red; -fx-font-family: \"Arial\";");

                return;

            }

            if (password.length() < 8) {

                warning.setText("Password must be at least 8 characters long.");

                warning.setStyle("-fx-text-fill: red; -fx-font-family: \"Arial\";");

                return;

            }

            if (!password.equals(confirmPassword)) {

                warning.setText("Passwords do not match.");

                warning.setStyle("-fx-text-fill: red; -fx-font-family: \"Arial\";");

                return;

            }

            if (accountManager.register(username, password, email)) {

                warning.setText("Account created successfully!");

                warning.setStyle("-fx-text-fill: green; -fx-font-family: \"Arial\";");

                loginForm.setVisible(true);

                signUpForm.setVisible(false);

            } else {

                warning.setText("Username already exists.");

                warning.setStyle("-fx-text-fill: red; -fx-font-family: \"Arial\";");

            }

        });

        register.getChildren().addAll(registerTitle, usernameField, emailField, passwordField, passwordTextField,

                confirmPasswordField, confirmPasswordTextField, showPasswordCheckBox, signupButton, warning);

        return register;

    }

    private VBox titleForm() {

        VBox switchPane = new VBox(20);

        switchPane.setAlignment(Pos.CENTER);

        switchPane.getStyleClass().add("shift-pane");

        Label logo = new Label("\uD83C\uDF7D");

        logo.setStyle("-fx-font-size: 100; -fx-text-fill: white;");

        Label systemTitle = new Label("Food Hive");

        systemTitle.getStyleClass().add("title");

        Button createAccountButton = new Button("Create new Account");

        createAccountButton.getStyleClass().add("button-create-account");

        createAccountButton.setOnAction(e -> {

            signUpForm.setVisible(true);

            loginForm.setVisible(false);

        });

        Button alreadyAccountButton = new Button("Already have Account");

        alreadyAccountButton.getStyleClass().add("button-create-account");

        alreadyAccountButton.setOnAction(e -> {

            loginForm.setVisible(true);

            signUpForm.setVisible(false);

            stage.close();

        });

        switchPane.getChildren().addAll(logo, systemTitle, createAccountButton, alreadyAccountButton);

        return switchPane;

    }

    private VBox createForgotPasswordForm() {

        VBox forgotPassword = new VBox(10);

        forgotPassword.setPadding(new Insets(10, 50, 50, 50));

        forgotPassword.setAlignment(Pos.CENTER\_LEFT);

        Label forgotPasswordTitle = new Label("Forgot Password");

        forgotPasswordTitle.getStyleClass().add("forgot-password-title");

        Label enterEmailTitle = new Label("Enter your Email.");

        TextField emailField = new TextField();

        emailField.setPrefWidth(200);

        emailField.setPrefHeight(30);

        Button submitButton = new Button("Submit");

        submitButton.getStyleClass().add("button-submit");

        submitButton.setPrefWidth(200);

        submitButton.setPrefHeight(20);

        Label statusLabel = new Label();

        // VBox to hold the new password fields

        VBox resetPasswordForm = new VBox(20);

        resetPasswordForm.setVisible(false);

        resetPasswordForm.setManaged(false);

        TextField newPasswordField = new TextField();

        newPasswordField.setPromptText("New Password");

        newPasswordField.setPrefWidth(200);

        newPasswordField.setPrefHeight(20);

        TextField confirmPasswordField = new TextField();

        confirmPasswordField.setPromptText("Confirm New Password");

        confirmPasswordField.setPrefWidth(200);

        confirmPasswordField.setPrefHeight(20);

        Button savePasswordButton = new Button("Save Password");

        savePasswordButton.getStyleClass().add("button-save");

        savePasswordButton.setPrefWidth(200);

        savePasswordButton.setPrefHeight(20);

        savePasswordButton.setOnAction(e -> {

            String newPassword = newPasswordField.getText();

            String confirmPassword = confirmPasswordField.getText();

            if (newPassword.isEmpty() || confirmPassword.isEmpty()) {

                statusLabel.setText("Please fill in all fields.");

                statusLabel.setStyle("-fx-text-fill: red;");

                return;

            }

            if (newPassword.length() < 8) {

                statusLabel.setText("Password must be at least 8 characters long.");

                statusLabel.setStyle("-fx-text-fill: red;");

                return;

            }

            if (!newPassword.equals(confirmPassword)) {

                statusLabel.setText("Passwords do not match.");

                statusLabel.setStyle("-fx-text-fill: red;");

                return;

            }

            // Update the password in the account manager

            accountManager.updatePassword(emailField.getText(), newPassword);

            // Show success message

            statusLabel.setText("Password saved successfully!");

            statusLabel.setStyle("-fx-text-fill: green;");

            // Hide all forms and go back to the login page

            forgotPassword.setVisible(false);

            resetPasswordForm.setVisible(false);

            loginForm.setVisible(true);

        });

        resetPasswordForm.getChildren().addAll(newPasswordField, confirmPasswordField, savePasswordButton);

        submitButton.setOnAction(e -> {

            String email = emailField.getText();

            if (email.isEmpty()) {

                statusLabel.setText("Please enter an email address.");

                statusLabel.setStyle("-fx-text-fill: red;");

            } else if (!accountManager.isEmailRegistered(email)) { // Check email validity

                statusLabel.setText("Email not registered.");

                statusLabel.setStyle("-fx-text-fill: red;");

            } else {

                statusLabel.setText("Email verified. Enter your new password.");

                statusLabel.setStyle("-fx-text-fill: green;");

                resetPasswordForm.setVisible(true);

                resetPasswordForm.setManaged(true);

            }

        });

        Button backButton = new Button("Back to Login");

        backButton.getStyleClass().add("button-back");

        backButton.setPrefWidth(200);

        backButton.setPrefHeight(20);

        // Event handling for the back button

        backButton.setOnAction(e -> {

            // Reset the visibility and fields when going back

            resetPasswordForm.setVisible(false);

            resetPasswordForm.setManaged(false);

            statusLabel.setText("");

            emailField.clear();

            newPasswordField.clear();

            confirmPasswordField.clear();

            forgotPassword.setVisible(false);

            loginForm.setVisible(true);

        });

        forgotPassword.getChildren().addAll(forgotPasswordTitle, enterEmailTitle, emailField, submitButton, statusLabel, resetPasswordForm, backButton);

        forgotPassword.setMargin(forgotPasswordTitle, new Insets(25, 0, 0, 0));

        return forgotPassword;

    }

    public static void main(String[] args) {

        launch(args);

    }

}

**Dashboard Class**

package org.project;

import java.net.URL;

import javafx.collections.ListChangeListener;

import javafx.collections.ObservableList;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.Node;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.Label;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.image.ImageView;

import javafx.scene.layout.HBox;

import javafx.scene.layout.Priority;

import javafx.scene.layout.StackPane;

import javafx.scene.layout.TilePane;

import javafx.scene.layout.VBox;

import javafx.stage.Stage;

public class Dashboard {

    Item item = new Item();

    TableView<Item> tableView = new TableView<>();

    Scene scene;

    // Create three vertical boxes

    static VBox vBox1 = new VBox(20);

    static VBox vBox2 = new VBox(20);

    static VBox vBox3 = new VBox(20);

    static VBox showRestaurant = new VBox(20);

    public void showDashboard() {

        // New stage for the dashboard

        Stage dashboardStage = new Stage();

        dashboardStage.setTitle("Cafe Shop Management System");

        vBox2.setVisible(true);

        showRestaurant.setVisible(false);

        // Create the top HBox for the label "FOOD HIVE"

        HBox topHBox = new HBox(20);

        topHBox.getStyleClass().add("top-hbox-style");

        topHBox.setAlignment(Pos.CENTER);

        Label foodHiveLabel = new Label("FOOD HIVE");

        foodHiveLabel.getStyleClass().add("food-hive-label");

        Label logo = new Label("\uD83C\uDF7D");

        logo.setStyle("-fx-font-size: 50; -fx-text-fill: #ffce1b;");

        topHBox.getChildren().addAll(logo, foodHiveLabel);

        vBox1.setMaxWidth(150);

        vBox3.setMaxWidth(300);

        vBox1.getStyleClass().add("vbox1-style");

        vBox2.getStyleClass().add("vbox2-style");

        vBox3.getStyleClass().add("vbox3-style");

        // Setup vBox1 and vBox3 content using the new methods

        setupVBox1();

        setupVBox2();

        setupVBox3();

        StackPane mainScreen = new StackPane(vBox2, showRestaurant);

        mainScreen.setPrefSize(1050, 600);

        HBox hBox = new HBox(5, vBox1, mainScreen, vBox3);

        hBox.setAlignment(Pos.CENTER);

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

        HBox.setHgrow(vBox1, Priority.ALWAYS);

        HBox.setHgrow(vBox2, Priority.ALWAYS);

        HBox.setHgrow(vBox3, Priority.ALWAYS);

        VBox mainVBox = new VBox(10, topHBox, hBox);

        VBox.setVgrow(hBox, Priority.ALWAYS);

        Scene scene = new Scene(mainVBox, 1100, 600);

        hBox.prefWidthProperty().bind(scene.widthProperty());

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

        dashboardStage.setScene(scene);

        dashboardStage.setFullScreen(true);

        dashboardStage.show();

    }

    private void setupVBox1() {

        vBox1.setPadding(new Insets(20));

        vBox1.setAlignment(Pos.CENTER);

        Button homeButton = new Button("Home");

        Button restaurantsButton = new Button("Restaurants");

        Button categoriesButton = new Button("Categories");

        homeButton.getStyleClass().add("home-button");

        homeButton.setOnAction(event -> {

            vBox2.setVisible(true);

            showRestaurant.setVisible(false);

        });

        homeButton.setPrefSize(200, 40);

        restaurantsButton.getStyleClass().add("restaurants-button");

        restaurantsButton.setPrefSize(200, 40);

        restaurantsButton.setOnAction(event -> {

            vBox2.setVisible(false);

            showRestaurant.setVisible(true);

            showRestaurants();

        });

        categoriesButton.getStyleClass().add("categories-button");

        categoriesButton.setPrefSize(200, 40);

        VBox buttonBox = new VBox(20, homeButton, restaurantsButton, categoriesButton);

        buttonBox.setAlignment(Pos.CENTER);

        vBox1.getChildren().add(buttonBox);

    }

    private void setupVBox2() {

        vBox2.getChildren().clear();

        vBox2.setPadding(new Insets(5));

        vBox2.setAlignment(Pos.CENTER);

        TilePane tilePane = new TilePane();

        tilePane.setPadding(new Insets(5));

        tilePane.setHgap(10);

        tilePane.setVgap(10);

        tilePane.setPrefColumns(3);

        tilePane.setAlignment(Pos.CENTER);

        for (int i = 0; i < 9; i++) {

            VBox itemBox = new VBox(5);

            itemBox.setAlignment(Pos.CENTER);

            itemBox.setStyle("-fx-border-color: lightgray; -fx-border-width: 1; -fx-padding: 10px;");

            // Loading image from the resource path

            ImageView imageView = null;

            String imagePath = "image" + (i + 1) + ".png";

            URL resource = getClass().getResource(imagePath);

            if (resource == null) {

                System.out.println("Image not found: " + imagePath);

            } else {

                imageView = new ImageView(resource.toExternalForm());

            }

            // Setting image view properties

            imageView.setFitWidth(300);

            imageView.setFitHeight(120);

            imageView.setPreserveRatio(true);

            // Item names and prices

            String[] itemNames = {"Burger", "Karahi", "Cake", "Pizza", "Biryani", "Ice Cream", "Nuggets", "Kabab", "Shakes"};

            double[] itemPrices = {5.99, 8.99, 7.49, 4.50, 6.00, 3.50, 2.99, 3.00, 1.99};

            // Item name and price

            String itemName = itemNames[i];

            double itemPrice = itemPrices[i];

            // Create labels for the item name and price

            Label itemNameLabel = new Label(itemName);

            itemNameLabel.setStyle("-fx-font-size: 14; -fx-font-weight: bold; -fx-text-fill: #ffce1b;");

            Label itemPriceLabel = new Label("$" + itemPrice);

            itemPriceLabel.setStyle("-fx-font-size: 12; -fx-text-fill: white;");

            Label titleLabel = new Label("Select Quantity");

            titleLabel.setStyle("-fx-font-size: 16; -fx-font-weight: bold;");

            HBox counterBox = new HBox(10);

            counterBox.setAlignment(Pos.CENTER);

            // Counter buttons and quantity label

            Button minusButton = new Button("-");

            Button plusButton = new Button("+");

            minusButton.getStyleClass().add("minus-button");

            Label quantityLabel = new Label("0");

            plusButton.getStyleClass().add("plus-button");

            quantityLabel.setStyle("-fx-font-size: 16; -fx-background-color:transparent; -fx-text-fill:#ffce1b; ");

            counterBox.getChildren().addAll(quantityLabel);

            final int[] quantity = {0};

            minusButton.setOnAction(e -> {

                if (quantity[0] > 0) {

                    quantity[0]--;

                    quantityLabel.setText(String.valueOf(quantity[0]));

                }

            });

            plusButton.setOnAction(e -> {

                quantity[0]++;

                quantityLabel.setText(String.valueOf(quantity[0]));

            });

            Button addToOrderButton = new Button("Add to Cart");

            addToOrderButton.setPrefSize(100, 20);

            addToOrderButton.setStyle("-fx-background-color: #ffce1b;  -fx-text-fill: black; -fx-border-radius: 5px;-fx-cursor: hand;");

            addToOrderButton.setOnAction(event -> {

                if (quantity[0] > 0) {

                    addItemToTable(itemName, quantity[0], itemPrice \* quantity[0]);

                    quantity[0] = 0;

                    quantityLabel.setText(String.valueOf(quantity[0]));

                }

            });

            HBox namePriceBox = new HBox(10);

            namePriceBox.setAlignment(Pos.CENTER);

            namePriceBox.getChildren().addAll(itemNameLabel, itemPriceLabel);

            HBox hBoxcart = new HBox(10);

            hBoxcart.setAlignment(Pos.CENTER);

            hBoxcart.getChildren().addAll(minusButton, addToOrderButton, plusButton);

            itemBox.getChildren().addAll(namePriceBox, imageView, counterBox, hBoxcart);

            tilePane.getChildren().add(itemBox);

        }

        tilePane.widthProperty().addListener((observable, oldValue, newValue) -> {

            double tileWidth = newValue.doubleValue() / 3 - tilePane.getHgap() \* 2;

            for (Node node : tilePane.getChildren()) {

                if (node instanceof VBox) {

                    VBox item = (VBox) node;

                    item.setPrefWidth(tileWidth);

                }

            }

        });

        vBox2.getChildren().add(tilePane);

    }

    private void setupVBox3() {

        vBox3.setPadding(new Insets(10));

        vBox3.setAlignment(Pos.TOP\_CENTER);

        Label orderLabel = new Label("Your Order");

        orderLabel.getStyleClass().add("orderLabel");

        TableColumn<Item, String> itemNameColumn = new TableColumn<>("Item Name");

        itemNameColumn.setMinWidth(100);

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

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

        quantityColumn.setMinWidth(70);

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

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

        priceColumn.setMinWidth(70);

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

        tableView.getColumns().addAll(itemNameColumn, quantityColumn, priceColumn);

        Label totalPriceLabel = new Label();

        totalPriceLabel.setStyle("-fx-font-size: 16; -fx-font-weight: bold; -fx-text-fill: white;");

        Runnable updateTotalPrice = () -> {

            double total = tableView.getItems().stream()

                    .mapToDouble(Item::getPrice)

                    .sum();

            totalPriceLabel.setText("Total Price: $" + String.format("%.2f", total));

        };

        tableView.getItems().addListener((ListChangeListener<Item>) change -> updateTotalPrice.run());

        updateTotalPrice.run();

        vBox3.getChildren().addAll(orderLabel, tableView, totalPriceLabel);

    }

    private void showRestaurants() {

        showRestaurant.getChildren().clear();

        showRestaurant.setPadding(new Insets(20));

        showRestaurant.setAlignment(Pos.TOP\_CENTER);

        // Sample restaurant data

        String[][] restaurants = {

            {"KFC", "image1.jpg", "Amazing broast and burgers. Must visit!"},

            {"Macdonald's", "image2.png", "Best burgers fries in town."},

            {"Butt Karahi", "image3.png", "Delicious and consistent karahi."},

            {"Village", "image4.png", "Unlimited  Buffet on daily basis."}

        };

        // Create HBox for each restaurant

        for (String[] restaurant : restaurants) {

            HBox restaurantBox = new HBox(10);

            restaurantBox.setPadding(new Insets(10));

            restaurantBox.setAlignment(Pos.CENTER\_LEFT);

            restaurantBox.setPrefHeight(600);

            restaurantBox.setStyle("-fx-border-color: lightgray; -fx-border-width: 1; -fx-padding: 10px; -fx-text-fill: black;");

            // Add restaurant name and description

            VBox infoBox = new VBox(5);

            Button nameLabel = new Button(restaurant[0]);

            nameLabel.setStyle("-fx-font-size: 16; -fx-font-weight: bold;-fx-text-fill: black;");

            Label descriptionLabel = new Label(restaurant[2]);

            descriptionLabel.setWrapText(true);

            descriptionLabel.setStyle("-fx-text-fill: black;");

            nameLabel.setOnMouseClicked(event -> {

            });

            infoBox.getChildren().addAll(nameLabel, descriptionLabel);

            // Add image and info to HBox

            restaurantBox.getChildren().addAll(infoBox);

            showRestaurant.getChildren().add(restaurantBox);

        }

    }

    private void addItemToTable(String itemName, int quantity, double itemPrice) {

        ObservableList<Item> items = tableView.getItems();

        for (Item item : items) {

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

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

                item.setPrice(item.getPrice() + (itemPrice \* quantity));

                tableView.refresh();

                return;

            }

        }

        items.add(new Item(itemName, quantity, itemPrice \* quantity));

    }

}

**AccountManager Class**

package org.project;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

import java.util.List;

public class AccountManager {

    private static final String ACCOUNTS\_FILE = "accounts.dat";

    private List<Account> accounts;

    public AccountManager() {

        accounts = new ArrayList<>();

        loadAccounts();

    }

    // Method to load accounts from the file

    private void loadAccounts() {

        File file = new File(ACCOUNTS\_FILE);

        if (file.exists()) {

            try (ObjectInputStream ois = new ObjectInputStream(new FileInputStream(file))) {

                Object data = ois.readObject();

                if (data instanceof List<?>) {

                    accounts = (List<Account>) data;

                } else {

                    System.err.println("Error: The file does not contain a valid List<Account>.");

                }

            } catch (IOException | ClassNotFoundException e) {

                System.err.println("Error loading accounts: " + e.getMessage());

            }

        }

    }

    // Method to save accounts to the file

    private void saveAccounts() {

        try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(ACCOUNTS\_FILE))) {

            oos.writeObject(accounts);

        } catch (IOException e) {

            System.err.println("Error saving accounts: " + e.getMessage());

        }

    }

    // Method to register a new account

    public boolean register(String username, String password, String email) {

        for (Account account : accounts) {

            if (account.getUsername().equals(username)) {

                return false;

            }

        }

        accounts.add(new Account(username, password, email));

        saveAccounts();

        return true;

    }

    // Method to verify login credentials

    public boolean verify(String username, String password) {

        for (Account account : accounts) {

            if (account.getUsername().equals(username) && account.getPassword().equals(password)) {

                return true;

            }

        }

        return false;

    }

    // Check if an email is registered

    public boolean isEmailRegistered(String email) {

        for (Account account : accounts) {

            if (account.getEmail().equals(email)) {

                return true;

            }

        }

        return false;

    }

    // Update password for a given email

    public void updatePassword(String email, String newPassword) {

        for (Account account : accounts) {

            if (account.getEmail().equals(email)) {

                account.setPassword(newPassword);

                saveAccounts();

                return;

            }

        }

    }

}

**Item Class**

package org.project;

public class Item {

    private  String name;

    private int quantity;

    private  double price;

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

        this.name = name;

        this.quantity = quantity;

        this.price = price;

    }

    public Item() {

    }

    public void setName(String name){

        this.name = name;

    }

    public void setPrice(double  price){

        this.price = price;

    }

    public void setQuantity(int quantity) {

        this.quantity = quantity;

    }

    public String getName() {

        return name;

    }

    public int getQuantity() {

        return quantity;

    }

    public double getPrice() {

        return price;

    }}

**Account Class**

package org.project;

import java.io.Serializable;

public class Account implements Serializable {

    private String username;

    private String password;

    private String email;

    public Account(String username, String password, String email) {

        this.username = username;

        this.password = password;

        this.email = email;

    }

    public String getUsername() {

        return username;

    }

    public String getPassword() {

        return password;

    }

    public void setPassword(String password) {

        this.password = password;

    }

    public String getEmail() {

        return email;

    }

}

**Style. CSS**

.register-title, .login-title , .forgot-password-title {  
 -fx-font-size: 20px;  
 -fx-font-weight: bold;  
 -fx-text-fill: #232323;  
 -fx-font-family: "Times New Roman";  
}  
  
.button-login, .button-register , .button-submit , .button-save ,.button-back{  
 -fx-background-color: #232323;  
 -fx-text-fill: white;  
 -fx-font-size: 14px;  
 -fx-font-weight: bold;  
 -fx-padding: 10px 20px;  
 -fx-border-radius: 5px;  
}  
  
.button-login:hover , .button-submit:hover , .button-save:hover , .button-register:hover ,.button-back:hover {  
 -fx-background-color: #3C3C3C;  
 -fx-text-fill: white;  
}  
  
/\* Hyperlink Styling \*/  
.hyperlink {  
 -fx-font-size: 12px;  
 -fx-text-fill: #232323;  
 -fx-padding: 5px 0;  
}  
  
.hyperlink:hover {  
 -fx-text-fill: #3C3C3C;  
}  
  
.button-create-account, .button-already-account {  
 -fx-background-color: transparent;  
 -fx-text-fill: white;  
 -fx-font-size: 14px;  
 -fx-padding: 10px 20px;  
 -fx-border-radius: 1px;  
 -fx-border-color: white;  
}  
  
.button-create-account:hover, .button-already-account:hover {  
 -fx-background-color: white;  
 -fx-text-fill: black;  
}  
  
.shift-pane {  
 /\*-fx-background-color: #15151e;\*/  
 -fx-background-color: linear-gradient(to bottom ,#3C3C3C , #232323);  
 -fx-padding: 50px;  
}  
  
.title {  
 -fx-font-size: 30px;  
 -fx-font-weight: bold;  
 -fx-text-fill: white;  
}  
  
.food-hive-label ,.orderLabel{  
 -fx-font-family: "Arial Black";  
 -fx-font-size: 20px;  
 -fx-text-fill: #ffce1b;  
 -fx-font-weight: bold;  
 -fx-underline: true;  
}  
  
.top-hbox-style , showRestaurant ,restaurantBox{  
 -fx-background-color: linear-gradient(to bottom ,#3C3C3C , #232323);  
}  
  
  
/\* Button Styles \*/  
.button-style {  
 -fx-pref-width: 150px;  
}  
  
/\* VBox styles \*/  
.vbox-style {  
 -fx-spacing: 20px;  
 -fx-padding: 20px;  
 -fx-alignment: center;  
}  
  
.vbox1-style, .vbox2-style, .vbox3-style{  
 -fx-border-width: 1px;  
 -fx-border-color: white;  
 -fx-background-color: linear-gradient(to bottom ,#3C3C3C , #232323);  
}  
  
/\* TableView Styles \*/  
.table-view {  
 -fx-pref-height: 350px;  
 -fx-max-height: 350px;  
}  
  
/\* Labels in the VBoxes \*/  
.vbox-label {  
 -fx-font-size: 18px;  
 -fx-font-weight: bold;  
}  
  
.restaurants-button, .categories-button, .home-button {  
 -fx-background-color: red;  
 -fx-text-fill: white;  
 -fx-font-size: 12px;  
 -fx-padding: 10px 20px;  
 -fx-border-radius: 5px;  
 -fx-cursor: hand;  
}  
  
/\* Hover effect in JavaFX \*/  
.restaurants-button:hover,  
.categories-button:hover,  
.home-button:hover {  
 -fx-background-color: yellow;  
 -fx-text-fill: #333;  
}  
  
.minus-button , .plus-button{  
 -fx-background-color: transparent;  
 -fx-text-fill: #ffce1b;  
 -fx-font-size: 20px;  
 -fx-padding: 5px 10px;  
 -fx-font-weight: bold;  
}