```
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.text.DecimalFormat;

class User {
    String username;
    String password;

    public User(String username, String password) {
        this.username = username;
        this.password = password;
    }
}

class Product {
    String name;
    int quantity;
    double price;

    public Product(String name, int quantity, double price) {
        this.name = name;
        this.quantity = quantity;
        this.price = price;
    }

    public double getTotalPrice() {
        return quantity * price;
    }
}

public class Main {
    ArrayList<User> users = new ArrayList<>();
    ArrayList<Product> products = new ArrayList<>();
    double totalPrice = 0.0;
    DecimalFormat df = new DecimalFormat("0.00");
    final String USERNAME_REGEX = "^[a-zA-Z0-9]{5,15}$";
    final String PASSWORD_REGEX = "^(?=.*[A-Z])(?=.*\\d).{8,20}$";
```

```
    Pattern usernamePattern = Pattern.compile(USERNAME_REGEX);
    Pattern passwordPattern = Pattern.compile(PASSWORD_REGEX);

    public void signup(Scanner scanner) {
        String username;
        String password;
        while (true) {
            System.out.print("Enter username (alphanumeric, 5-15 characters): ");
            username = scanner.nextLine();
            Matcher usernameMatcher = usernamePattern.matcher(username);
            if (!usernameMatcher.matches()) {
                System.out.println("Error: Username must be alphanumeric and 5-15 characters long.
Please try again.");
                continue;
            }
            boolean exists = false;
            for (User u : users) {
                if (u.username.equals(username)) {
                    System.out.println("Error: Username already taken. Please try again.");
                    exists = true;
                    break;
                }
            }
            if (!exists) {
                break;
            }
        }
        while (true) {
            System.out.print("Enter password (8-20 characters, at least one uppercase letter and
one number): ");
            password = scanner.nextLine();
            Matcher passwordMatcher = passwordPattern.matcher(password);
            if (passwordMatcher.matches()) {
                break;
            } else {
                System.out.println("Error: Password must be 8-20 characters long and contain at least
one uppercase letter and one number. Please try again.");
            }
        }
        users.add(new User(username, password));
        System.out.println("Signup successful!");
    }

    public String login(Scanner scanner) {
```

```
    if (users.isEmpty()) {
        System.out.println("No users registered yet. Please sign up first.");
        return null;
    }
    String username;
    String password;
    while (true) {
        System.out.print("Enter username: ");
        username = scanner.nextLine();
        System.out.print("Enter password: ");
        password = scanner.nextLine();
        for (User user : users) {
            if (user.username.equals(username) && user.password.equals(password)) {
                System.out.println("Login successful! Welcome, " + username + ".");
                return username;
            }
        }
        System.out.println("Invalid username or password. Please try again.");
    }
}

public void addProduct(String name, int quantity, double price) {
    products.add(new Product(name, quantity, price));
}

public void calculateTotal() {
    totalPrice = 0.0;
    for (Product product : products) {
        totalPrice += product.getTotalPrice();
    }
}

public void displayTotal() {
    System.out.println("Total Price: " + df.format(totalPrice));
}

public void acceptPayment(double payment) {
    double change = payment - totalPrice;
    System.out.println("Payment Accepted. Change: " + df.format(change));
}

public void showProducts() {
    System.out.println("\n--- Products in Cart ---");
    if (products.isEmpty()) {
```

```
            System.out.println("Cart is empty.");
        } else {
            for (Product product : products) {
                System.out.println("Product: " + product.name +
                        " | Quantity: " + product.quantity +
                        " | Price: " + df.format(product.price) +
                        " | Total: " + df.format(product.getTotalPrice()));
            }
        }
        System.out.println("-----------------------");
    }

    public void logTransaction(String username) {
        try (BufferedWriter writer = new BufferedWriter(new FileWriter("transactions.txt", true))) {
            writer.write("Transaction Time: " +
LocalDateTime.now().format(DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss")));
            writer.newLine();
            writer.write("Cashier: " + username);
            writer.newLine();
            writer.write("Items Purchased:");
            writer.newLine();
            for (Product product : products) {
                writer.write("- " + product.name + " | Qty: " + product.quantity + " | Price: " +
df.format(product.price) +
                        " | Total: " + df.format(product.getTotalPrice()));
                writer.newLine();
            }
            writer.write("Total Amount: " + df.format(totalPrice));
            writer.newLine();
            writer.write("------------------------------------------------");
            writer.newLine();
        } catch (IOException e) {
            System.out.println("Error logging transaction: " + e.getMessage());
        }
    }

    public void runCashRegister(Scanner scanner, String username) {
        boolean continueTransaction = true;
        String[] menuItems = {"Chicken Joy (Bucket)", "Spaghetti", "Burger Steak", "Palabok
Fiesta", "Jolly Hotdog"};
        double[] prices = {499.99, 299.99, 379.99, 249.99, 199.99};

        while (continueTransaction) {
            System.out.println("\n--- Jollibee Cash Register ---");
```

```
System.out.println("Menu:");
for (int i = 0; i < menuItems.length; i++) {
    System.out.println((i + 1) + ". " + menuItems[i] + " - " + df.format(prices[i]));
}
System.out.println("----------------------------");

int choice = -1;
boolean validChoice = false;
while (!validChoice) {
    System.out.print("Enter the number of the product to add (or 0 to checkout): ");
    try {
        int inputChoice = Integer.parseInt(scanner.nextLine());
        if (inputChoice == 0) {
            choice = -1;
            validChoice = true;
        } else if (inputChoice >= 1 && inputChoice <= menuItems.length) {
            choice = inputChoice - 1;
            validChoice = true;
        } else {
            System.out.println("Invalid selection. Try again.");
        }
    } catch (NumberFormatException e) {
        System.out.println("Invalid input. Please enter a valid number.");
    }
}

if (choice == -1) {
    if (products.isEmpty()) {
        System.out.println("Cart is empty. Add items before checking out.");
        break;
    } else {
        showProducts();
        calculateTotal();
        displayTotal();
        double payment = 0.0;
        boolean validPayment = false;
        while (!validPayment) {
            try {
                System.out.print("Enter payment amount: ");
                payment = Double.parseDouble(scanner.nextLine());
                if (payment >= totalPrice) {
                    validPayment = true;
                    acceptPayment(payment);
                    logTransaction(username);
```

```
                    System.out.println("Transaction saved to transactions.txt");
                } else {
                    System.out.println("Insufficient funds.");
                }
            } catch (NumberFormatException e) {
                System.out.println("Invalid amount. Try again.");
            }
        }
        break;
    }
} else {
    int quantity = 0;
    boolean validQuantity = false;
    while (!validQuantity) {
        System.out.print("Enter quantity: ");
        try {
            quantity = Integer.parseInt(scanner.nextLine());
            if (quantity > 0) {
                validQuantity = true;
            } else {
                System.out.println("Quantity must be positive.");
            }
        } catch (NumberFormatException e) {
            System.out.println("Invalid quantity input.");
        }
    }
    addProduct(menuItems[choice], quantity, prices[choice]);
    System.out.println(quantity + " x " + menuItems[choice] + " added to cart.");
    showProducts();
    calculateTotal();
    displayTotal();
}
}
}

public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    Main system = new Main();
    boolean exitProgram = false;

    while (!exitProgram) {
        System.out.println("\n===== Welcome =====");
        System.out.println("1. Signup");
        System.out.println("2. Login");
```

```
            System.out.println("3. Exit");
            System.out.print("Choose an option: ");
            String choice = scanner.nextLine();

            switch (choice) {
                case "1":
                    system.signup(scanner);
                    break;
                case "2":
                    String username = system.login(scanner);
                    if (username != null) {
                        boolean anotherTransaction;
                        do {
                            Main session = new Main();
                            session.runCashRegister(scanner, username);
                            boolean validResponse = false;
                            anotherTransaction = false;
                            while (!validResponse) {
                                System.out.print("\nStart a new transaction? (yes/no): ");
                                String response = scanner.nextLine().toLowerCase();
                                if (response.equals("no")) {
                                    validResponse = true;
                                    System.out.println("Logging out...");
                                } else if (response.equals("yes")) {
                                    validResponse = true;
                                    anotherTransaction = true;
                                } else {
                                    System.out.println("Invalid response.");
                                }
                            }
                        } while (anotherTransaction);
                    }
                    break;
                case "3":
                    exitProgram = true;
                    System.out.println("Thank you for shopping!");
                    break;
                default:
                    System.out.println("Invalid choice.");
            }
        }
        scanner.close();
    }
}
```