```
import java.io.FileWriter;
import java.io.IOException;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;
import java.util.regex.Pattern;

public class Main {
    static ArrayList<String> usernames = new ArrayList<>();
    static ArrayList<String> passwords = new ArrayList<>();
    static String currentUser = "";

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ArrayList<String> productNames = new ArrayList<>();
        ArrayList<Double> productPrices = new ArrayList<>();
        ArrayList<Integer> productQuantities = new ArrayList<>();

        System.out.println("============================");
        System.out.println("      WELCOME TO ADIlicious");
        System.out.println("============================");

        boolean loggedIn = false;

        while (!loggedIn) {
            System.out.println("\n1. Sign Up");
            System.out.println("2. Log In");
            System.out.print("Enter your choice: ");
            int option = scanner.nextInt();
```

```
    scanner.nextLine();

    if (option == 1) {
        signup(scanner);
    } else if (option == 2) {
        loggedIn = login(scanner);
    } else {
        System.out.println("Invalid choice. Please try again.");
    }
}

boolean exit = false;
while (!exit) {
    System.out.println("\n============================");
    System.out.println("1. Add Product");
    System.out.println("2. Remove Product");
    System.out.println("3. View Cart and Checkout");
    System.out.println("4. Exit");
    System.out.print("Enter your choice: ");

    int choice = scanner.nextInt();
    scanner.nextLine();

    switch (choice) {
        case 1:
            addProduct(scanner, productNames, productPrices, productQuantities);
            break;
        case 2:
            removeProduct(scanner, productNames, productPrices, productQuantities);
```

```
            break;
        case 3:
            checkout(scanner, productNames, productPrices, productQuantities);
            break;
        case 4:
            exit = true;
            System.out.println("Thank you for shopping at ADIlicious!");
            break;
        default:
            System.out.println("Invalid choice. Please try again.");
      }
   }
}

public static void signup(Scanner scanner) {
   System.out.println("\n========== USER SIGNUP ==========");
   String username, password;

   while (true) {
      System.out.print("Enter username (5-15 characters, letters/numbers only): ");
      username = scanner.nextLine();
      if (Pattern.matches("^[a-zA-Z0-9]{5,15}$", username)) break;
      else System.out.println("Invalid Username Format. Please Try Again.");
   }

   while (true) {
      System.out.print("Enter password (8-20 characters, at least 1 uppercase and 1 number): ");
      password = scanner.nextLine();
      if (Pattern.matches("^(?=.*[A-Z])(?=.*\\d).{8,20}$", password)) break;
```

```
        else System.out.println("Invalid Password Format. Please Try Again.");
    }

    usernames.add(username);
    passwords.add(password);
    System.out.println("Signup successful!");
}

public static boolean login(Scanner scanner) {
    System.out.println("\n========== USER LOGIN ==========");

    System.out.print("Enter username: ");
    String username = scanner.nextLine();
    System.out.print("Enter password: ");
    String password = scanner.nextLine();

    for (int i = 0; i < usernames.size(); i++) {
        if (usernames.get(i).equals(username) && passwords.get(i).equals(password)) {
            System.out.println("Login successful!");
            currentUser = username;
            return true;
        }
    }

    System.out.println("Invalid credentials. Please try again.");
    return false;
}
```

```
   public static void addProduct(Scanner scanner, ArrayList<String> names, ArrayList<Double> prices,
ArrayList<Integer> quantities) {
      System.out.print("Enter product name: ");
      String name = scanner.nextLine();
      System.out.print("Enter product price: ");
      double price = scanner.nextDouble();
      System.out.print("Enter quantity: ");
      int quantity = scanner.nextInt();

      names.add(name);
      prices.add(price);
      quantities.add(quantity);

      System.out.println(quantity + " x " + name + " added to the cart.");
   }

   public static void removeProduct(Scanner scanner, ArrayList<String> names, ArrayList<Double> prices,
ArrayList<Integer> quantities) {
      if (names.isEmpty()) {
         System.out.println("Your cart is empty.");
         return;
      }

      System.out.println("\nCurrent Cart:");
      for (int i = 0; i < names.size(); i++) {
         System.out.println((i + 1) + ". " + quantities.get(i) + " x " + names.get(i) + " - " + prices.get(i) + "
each");
      }

      System.out.print("Enter product number to remove: ");
```

```
        int index = scanner.nextInt() - 1;

        if (index >= 0 && index < names.size()) {
            System.out.println(names.get(index) + " removed from cart.");
            names.remove(index);
            prices.remove(index);
            quantities.remove(index);
        } else {
            System.out.println("Invalid selection.");
        }
    }

    public static void checkout(Scanner scanner, ArrayList<String> names, ArrayList<Double> prices,
ArrayList<Integer> quantities) {
        if (names.isEmpty()) {
            System.out.println("Your cart is empty. Add items first.");
            return;
        }

        double total = 0;
        StringBuilder receipt = new StringBuilder();

        receipt.append("\n===== TRANSACTION RECEIPT =====\n");
        receipt.append("Cashier: ").append(currentUser).append("\n");
        receipt.append("Date/Time:
").append(LocalDateTime.now().format(DateTimeFormatter.ofPattern("yyyy-MM-dd
HH:mm:ss"))).append("\n\n");
        receipt.append("Items:\n");

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

```
    double cost = quantities.get(i) * prices.get(i);
    receipt.append(quantities.get(i)).append(" x ").append(names.get(i))
            .append(" @ ").append(prices.get(i)).append(" = ")
            .append(String.format("%.2f", cost)).append(" PHP\n");
    total += cost;
}

receipt.append("\nTotal: ").append(String.format("%.2f", total)).append(" PHP\n");

double payment;
while (true) {
    try {
        System.out.print("Enter payment amount: ");
        payment = Double.parseDouble(scanner.nextLine());

        if (payment < total) {
            System.out.println("Insufficient amount! Please enter a valid payment.");
        } else {
            break;
        }
    } catch (NumberFormatException e) {
        System.out.println("Invalid input. Please enter a valid number.");
    }
}

double change = payment - total;
receipt.append("Payment: ").append(String.format("%.2f", payment)).append(" PHP\n");
receipt.append("Change: ").append(String.format("%.2f", change)).append(" PHP\n");
receipt.append("==============================\n\n");
```

```
        System.out.println("Change: " + String.format("%.2f", change) + " PHP");
        System.out.println("Thank you for shopping at ADIlicious!");

        try (FileWriter writer = new FileWriter("transactions.txt", true)) {
            writer.write(receipt.toString());
        } catch (IOException e) {
            System.out.println("Failed to save transaction: " + e.getMessage());
        }
    }
}
```