Railway Management System

Group Members :

- Arisha Ashraf Oishi (2022-3-60-083)
- Nazratan Mazumder Niha (2022-3-60-328)
- Sajid Ahmed (2022-3-60-191)

Submission Date: 17-09-2023

Course Code: CSE110

Section: 04

Semester: Summer 2023

Submitted to:

Tanni Mittra

Senior Lecturer, Department of Computer Science and Engineering

Introduction

A Railway Management System is a comprehensive software solution designed to efficiently manage and streamline various aspects of a railway network's operations. Railway systems are vital components of transportation infrastructure, responsible for passengers' and goods' safe and reliable movement. Introducing a Railway Management System aims to modernize and optimize these operations, ensuring better performance, safety, and customer satisfaction.

Objective

The main objective of a Railway Management System is to improve the efficiency of railway operations, enhance the passenger experience, ensure safety and security, optimize resource management, streamline maintenance, manage finances effectively, promote integration with other transportation modes, address environmental concerns, and enable data-driven decision-making, ultimately contributing to the reliable and efficient functioning of the railway network.

Features and Methods

The following features have been added to the project:

- Admin
- Reserve and cancel the Ticket for the user
- Exit

The following methods of Java have been used in the project:

- Classes and object
- Association
- Inheritance
- Polymorphism
- Encapsulation
- Exception handling
- File handling

Classes and their features

Seven different classes including a main class have been used in this project. A brief description of each of the classes and their code have been given below:

• Log-in Class

This class is responsible for handling user login details. It captures and stores the user's ID and password. It also records the date when the password was set.

Code:

```
package railwayreservationsystem;


import java.io.*;

import java.util.*;

import java.time.LocalDate;


   class login{

    public String id;

    public String pass;

    public String password;

    public void getid() {

     Scanner sc = new Scanner(System.in);

      LocalDate myDate = LocalDate.now();

            System.out.println("\nUser name id:");

            id = sc.nextLine();


            System.out.println("Enter Password: ");

            password = sc.nextLine();

            System.out.println("\nYou set password on date: " + myDate);

            pass = password;

    }

}
```

● Register Class

This class is used to register new users in the system. It collects user ID and password. Similar to the login class, it also records the date when the password was set.

Code:

```
class Register{
   public String id;
     public String pass;
   public String password;
   public void getid() {
   Scanner sc = new Scanner(System.in);
 LocalDate myDate = LocalDate.now();
      System.out.println("******Register***** \n Enter your name:");
         id = sc.nextLine();
     System.out.println("Create the Password: ");
         password = sc.nextLine();
     System.out.println("\nYou set password on date: " + myDate);
         pass = password
   }
}
```

● Detail Class

This class handles the details of trains. It captures the train number, name, boarding point, destination, and date of travel.

Code:

```
class Detail {
   public long tno;
   public String tname;
   public String bp;
   public String dest;
   public int d, m, y;
   public void getDetail() {
         Scanner sc = new Scanner(System.in);
         InputStreamReader inputStrObj = new InputStreamReader(System.in);
         BufferedReader bufrObj = new BufferedReader(inputStrObj);
         try {
               System.out.println("\n--Add New details--\n");
               System.out.println("Train no: ");
               tno = sc.nextLong();
               System.out.println("Train Name: ");
               tname = bufrObj.readLine();
               System.out.println("Boarding point: ");
               bp = bufrObj.readLine();
               System.out.println("Destination pt: ");
```

```
                dest = bufrObj.readLine();

                System.out.println("Date of travel\n");

                System.out.println("Day: ");

                d = sc.nextInt();

                System.out.println("Month: ");

                m = sc.nextInt();

                System.out.println("Year: ");

                y = sc.nextInt();

            } catch (Exception e) {

                System.out.println(e);

            }

    }

}
```

● Reser Class

This class manages the reservation of tickets. It collects passenger information like name, seat number, passenger ID, and train number. It calculates the ticket amount based on seat number and adds 200 rupees.

Code:

```
class reser {

   InputStreamReader inputStrObj = new InputStreamReader(System.in);

   BufferedReader bufrObj = new BufferedReader(inputStrObj);

   public String fname;
```

```
    public int sno,pid;
    public long tno;
    public int amt;
    Scanner sc=new Scanner(System.in);
    void getresdet() throws IOException   {
    System.out.print("\n\n Enter passenger name: ");
    fname=bufrObj.readLine();
    System.out.print(" Enter passenger id=");
    pid=sc.nextInt();
    System.out.print(" Enter passenger seat no=");
    sno=sc.nextInt();
    System.out.print(" Enter passenger train no=");
    tno=sc.nextLong();
        amt = sno+ 200;
        System.out.println("You should pay: "+ amt + "rs");
    }
}
```

- **Sample Class**

This class acts as a controller or main logic handler for the program. It includes methods for user management, user login, ticket reservation, and admin operations. The `manage()` method handles user management, allowing the addition and display of user details. The `user()` method handles user login, enabling users to reserve, cancel, or inquire about tickets. The `database()` method is for admin operations, including adding and displaying train details, managing users, and displaying passenger details. The `passengerDisplay()` method displays the list of passengers. The `display()` method displays train details. The `reserve()` method handles ticket reservations, recording passenger details in a file. The `enquiry()` method displays train details for user inquiries. The `add()` method is used for user registration.

Code:

```
class Sample{

   reser b = new reser();

   login a = new login();

   Scanner sc = new Scanner(System.in);

   InputStreamReader inputStrObj = new InputStreamReader(System.in);

   BufferedReader bufrObj = new BufferedReader(inputStrObj);

         public void manage() throws IOException {

         int ch;

         DataOutputStream dos = new DataOutputStream(new
FileOutputStream("UserData.txt", true));

         DataInputStream dis = new DataInputStream(new
FileInputStream("UserData.txt"));

         System.out.println("\n---WELCOME TO THE USER MANAGEMENT
MENU---\n");

         do {

                  System.out.println("1. Add User details\n");

                  System.out.println("2. Display details\n");

                  System.out.println("3. Return to the main\n");
```

```
                    System.out.println("Enter your choice: ");

                    ch = sc.nextInt();

                    switch (ch) {

                    case 1:

                        try {

                                a.getid();

                                dos.writeUTF(a.id);

                                dos.writeUTF(a.pass);

                                dos.flush();


System.out.println("\n*****************User details added
succesfully!!*************");

                        } catch (FileNotFoundException e) {

                            System.out.println("Cannot open the output file!!");

                            return;

                        }

                        break;

                    case 2:

                        while (dis.available() > 0) {

                            try {

                                System.out.println("\n| User ID: \t
|Password:\n");

                                System.out.println("| " + dis.readUTF() + "\t | "
+ dis.readUTF() + "\n\n");
```

```
                    } catch (Exception e) {
                        System.out.println(e);
                    }
                }
                break;
            }
        } while (ch <= 2);
        dos.close();
        dis.close();
    }
    public void user() throws IOException {
        DataInputStream dis = new DataInputStream(new
FileInputStream("UserData.txt"));
            Add a= new Add();
             int ch;
             int d = 0;
             String password;
             String id;
             do{
            System.out.println("1)Login \n2)Regestation");
            System.out.println("Enter your choice:");
            ch = sc.nextInt();
             switch (ch) {
                    case 1:

                        System.out.println("\n-----Login-----\n");

                        System.out.println("Enter your ID : ");

                        id = bufrObj.readLine();

                        System.out.println("Enter your Password : ");

                        password = bufrObj.readLine();

                        try {

                         while (dis.available() > 0) {

                                String idReal = dis.readUTF();

                                String passReal = dis.readUTF();

                                if ((idReal.equals(id)) &&
(passReal.equals(password))) {

                                    do {


System.out.println("\n1.Reserve\n2.Cancel\n3.Enquiry\n4.Return to the main
menu\n");

                                        System.out.println("Enter your
choice:");

                                        ch = sc.nextInt();

                                        switch (ch) {

                                        case 1:

                                            reserve();

                                            break;

                                        case 2:
```

```
                                                  cancel();
                                                  break;
                                             case 3:
                                                  enquiry();
                                                  break;
                                             }
                                        } while (ch <= 3);
                                   } else {
                                        d = 1;
                                   }
                              }
                         } catch (Exception e) {
                              System.out.println(e);
                         }
                         if (d == 1) {
                              System.out.println("Enter your user id and password
               correctly\n");
                         }
                          dis.close();
                         break;
                              case 2:
                              a.add();
                              break;
```

```
                    }
                    }while (ch <= 2);
                     dis.close();
            }
    public void cancel() {
            int id;
            System.out.print("\n\nEnter passenger id to cancel the ticket: ");
            id=sc.nextInt();
            System.out.println("Ticket cancelled!!");
     }
            public void database() throws IOException {
                     Detail a = new Detail();
                     Scanner sc = new Scanner(System.in);
                     InputStreamReader inputStrObj = new
InputStreamReader(System.in);
                     BufferedReader bufrObj = new BufferedReader(inputStrObj);
                     DataOutputStream dos = new DataOutputStream(new
FileOutputStream("TrainData.txt", true));
                     DataInputStream dis = new DataInputStream(new
FileInputStream("TrainData.txt"));
                     int ch;
                     String c;
                     String password;
```

```
                String pass = "admin";
                System.out.println("\nEnter the Admin Password: ");
                password = sc.nextLine();
                if (!(pass.equals(password))) {
                        System.out.println("Enter the password correctly!! \n");
                        System.out.println("You are not permitted to login this
mode\n");
                }
                else {
                        do {
                                System.out.println("\n --- ADMINISTRATOR MENU
--- \n");
                                System.out.println("1. Add Train details \n");
                                System.out.println("2. Display Train details \n");
                                System.out.println("3. User Management \n");
                                System.out.println("4. Display Passenger details \n");
                                System.out.println("5. Return to Main Menu \n");
                                System.out.println("Enter your choice : ");
                                ch = sc.nextInt();
                                switch (ch) {
                                case 1:
                                        try {
                                                        a.getDetail();
                                                        dos.writeLong(a.tno);

                                                        dos.writeUTF(a.tname);

                                                        dos.writeUTF(a.bp);

                                                        dos.writeUTF(a.dest);

                                                        dos.writeInt(a.d);

                                                        dos.writeInt(a.m);

                                                        dos.writeInt(a.y);

                                                        dos.flush();


System.out.println("\n*****************Train details added
succesfully!!*************");

                                        } catch (FileNotFoundException e) {

                                                        System.out.println("Cannot open the
output file!!");

                                                        return;

                                        }

                                        break;

                                case 2:

                                        display();

                                        break;

                                case 3:

                                        manage();

                                        break;
```

```
                    case 4:

                        passengerDisplay();

                        break;

                    }

                } while (ch <= 4);

            }

            dos.close();

            dis.close();

        }

        public void passengerDisplay() throws IOException {

            DataInputStream dis1 = new DataInputStream(new
FileInputStream("UserData.txt"));

            System.out.println(" ********List of passenger*********");

            System.out.println("
===========================================================
==");

            while (dis1.available() > 0) {

                try {

                    System.out.println("\n|Pname" + "\t" + "|PId" + "\t\t"
+ "|Seat No.." + "\t "

                            + "|Train No." + "\n");



                    System.out.println("|" + dis1.readUTF() + "\t\t" + "|" +
dis1.readInt() + "\t\t" + "|"
```

```
                                        + dis1.readInt() + "\t\t" + "|" +
dis1.readLong() +  "\n");

                        } catch (Exception e) {

                                System.out.println(e);

                        }

                }

        }


        public void display() throws IOException {

                DataInputStream dis = new DataInputStream(new
FileInputStream("TrainData.txt"));

                while (dis.available() > 0) {

                        try {

                                System.out.println("\n|Train No." + "\t" + "|Train
Name" + "\t\t" + "|Boarding pt." + "\t\t "

                                        + "|Destination pt."  + "\t"+ "|Day" + "-"
+ "Month" + "-" + "Year" + "\n");



                                System.out.println("|" + dis.readLong() + "\t\t" + "|" +
dis.readUTF() + "\t" + "|"

                                        + dis.readUTF() + "\t" + "|" +
dis.readUTF()  + "\t\t"

                                        + "|" + dis.readInt() + "-" + dis.readInt()
+ "-" + dis.readInt() + "\n");

                        } catch (Exception e) {
                                System.out.println(e);

                        }

                }

        }

        public void reserve() throws Exception {

                DataOutputStream dos = new DataOutputStream(new
FileOutputStream("PassengerData.txt", true));

                try {

                        b.getresdet();

                        dos.writeUTF(b.fname);

                        dos.writeInt(b.pid);

                        dos.writeInt(b.sno);

                        dos.writeLong(b.tno);

                        dos.flush();

                        System.out.println("\n****************Ticket reserved
succesfully!!*************");

                } catch (FileNotFoundException e) {

                System.out.println("Cannot open the output file!!");

                return;

                }

                dos.close();

        }

        public void enquiry() throws IOException {
```

```
                    display();
                }
        }
```

● Add Class

This class appears to handle the addition of new users. It uses the `Register` class to collect and record user details.

Code:

```
class Add{
                Register a = new Register();
                InputStreamReader inputStrObj = new InputStreamReader(System.in);
        BufferedReader bufrObj = new BufferedReader(inputStrObj);

          Scanner sc = new Scanner(System.in);

  public void add() throws IOException {
                DataOutputStream dos = new DataOutputStream(new
FileOutputStream("UserData.txt", true));
                int ch;
                        try {
                                a.getid();
                                dos.writeUTF(a.id);
                                dos.writeUTF(a.pass);
                                dos.flush();
```

```
                                        System.out.println("\n*****************New
account opened*************");

                        } catch (FileNotFoundException e) {

                                System.out.println("Cannot open the output file!!");

                                return;

                        }

}
```

● RailwayReservationSystem Class (Main)

This is the main class where the program execution begins. It provides a menu-driven interface for users to select between admin and user modes or exit the system.

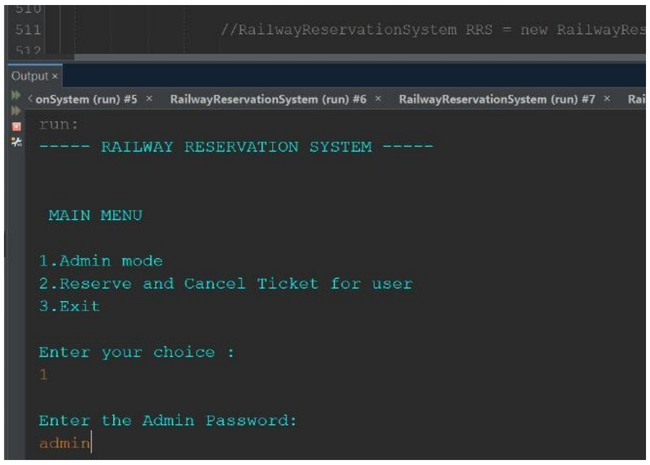
Code:

```
public class RailwayReservationSystem {



  public static void main(String args[]) throws Exception{

        Scanner sc = new Scanner(System.in);

        int ch;

              Sample d = new Sample();

        System.out.println("----- RAILWAY RESERVATION SYSTEM ----- \n");

        do{

                System.out.println("\n MAIN MENU \n");
```

```
                    System.out.println("1.Admin mode\n2.Reserve and Cancel Ticket for user\n3.Exit \n");


                    System.out.println("Enter your choice : ");

                    ch = sc.nextInt();


                    switch(ch){

                       case 1:

                            d.database();

                            break;

                       case 2:

                            d.user();

                            break;

                       default:

                            break;

                    }

            } while(ch<3);

    }

}
```

Output

When we first execute the main body, the following display is shown on the console.

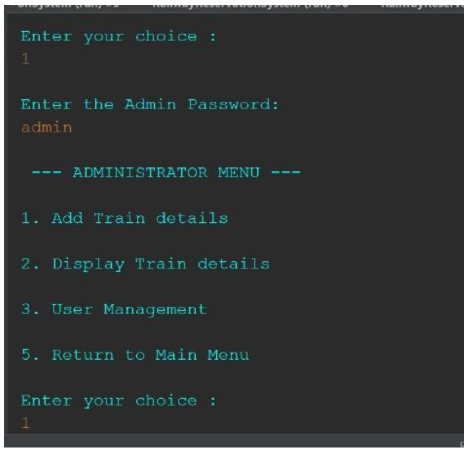


```
//RailwayReservationSystem RRS = new RailwayRes
```

Output ×
onSystem (run) #5 ×
RailwayReservationSystem (run) #6 ×
RailwayReservationSystem (run) #7 ×
Rai

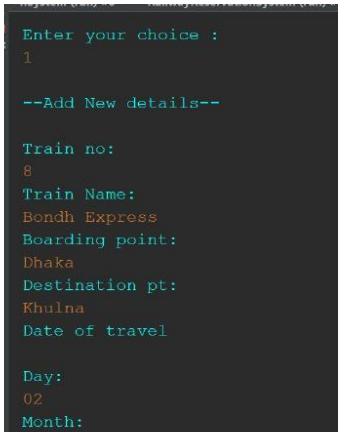
```
run:
----- RAILWAY RESERVATION SYSTEM -----


 MAIN MENU

1.Admin mode
2.Reserve and Cancel Ticket for user
3.Exit

Enter your choice :
1

Enter the Admin Password:
admin
```

At first, we choose option 1



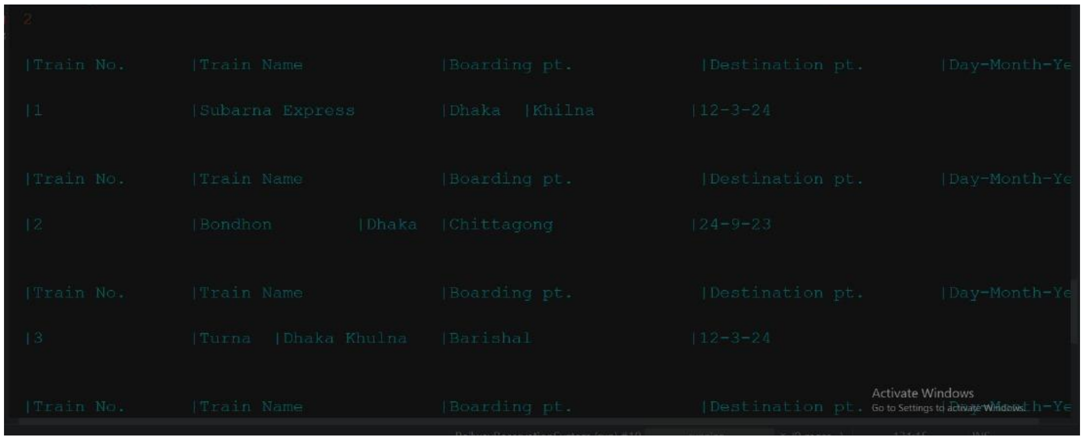
```
Enter your choice :
1

Enter the Admin Password:
admin

 --- ADMINISTRATOR MENU ---

1. Add Train details

2. Display Train details

3. User Management

5. Return to Main Menu

Enter your choice :
1
```

Here we can add details about the train



```
Enter your choice :
1

--Add New details--

Train no:
8
Train Name:
Bondh Express
Boarding point:
Dhaka
Destination pt:
Khulna
Date of travel

Day:
02
Month:
```

This window will shown to the user after the admin updates everything



```
2

|Train No.       |Train Name             |Boarding pt.            |Destination pt.       |Day-Month-Ye

|1               |Subarna Express        |Dhaka   |Khilna         |12-3-24


|Train No.       |Train Name             |Boarding pt.            |Destination pt.       |Day-Month-Ye

|2               |Bondhon         |Dhaka  |Chittagong             |24-9-23


|Train No.       |Train Name             |Boarding pt.            |Destination pt.       |Day-Month-Ye

|3               |Turna   |Dhaka Khulna  |Barishal               |12-3-24


|Train No.       |Train Name             |Boarding pt.            |Destination pt.       |Day-Month-Ye
```

Conclusion

Overall, this code implements a basic Railway Reservation System with functionalities for user registration, login, train details management, ticket reservation, and admin operations. It uses file I/O to store user, train, and passenger data. However, it's worth noting that this code may need additional improvements and error handling for robustness and security, especially in a real-world application.