UNIVERSITY OF WESTMINSTER

INFORMATICS INSTITUTE OF TECHNOLOGY

# Informatics Institute of Technology

## School of Computing

## Software Development II Coursework Report

Module : 4COSC010C.2: Software Development II (2023)

Date of submission : 25th March 2024

Student ID : w2053222

Student First Name : Jithnuka

Student Surname : Athurugiriya

Tutorial group : G-22

Tutorial day & time : Wednesday 10.30 PM – 12.30 PM

Tutors : Mr. Ammar Raneez & Ms. Rashmi Perera

## Declaration

"I confirm that I understand what plagiarism / collusion / contract cheating is and have read and understood the section on Assessment Offences in the Essential Information for Students. The work that I have submitted is entirely my own. Any work from other authors is duly referenced and acknowledged."

Name : A.P.A.Jithnuka Nimandith Athurugiriya

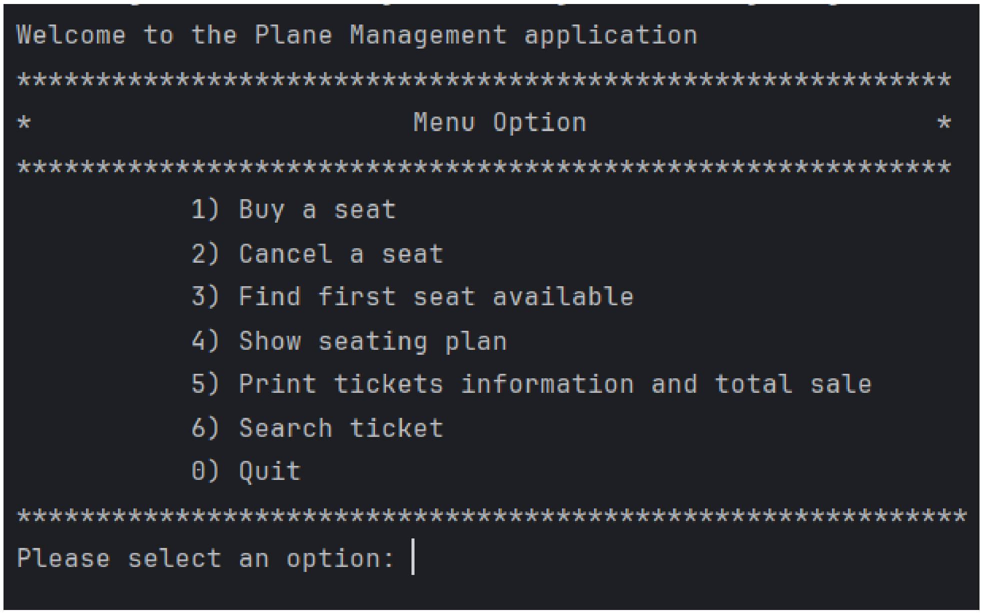
Student ID : 20222123

# Self-assessment form and test plan

## 1) Self-assessment form

| Task | Self-assessment (select one) | Comments |
|---|---|---|
| **1** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |
| **2** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |

**Insert here a screenshot of your welcome message and menu:**

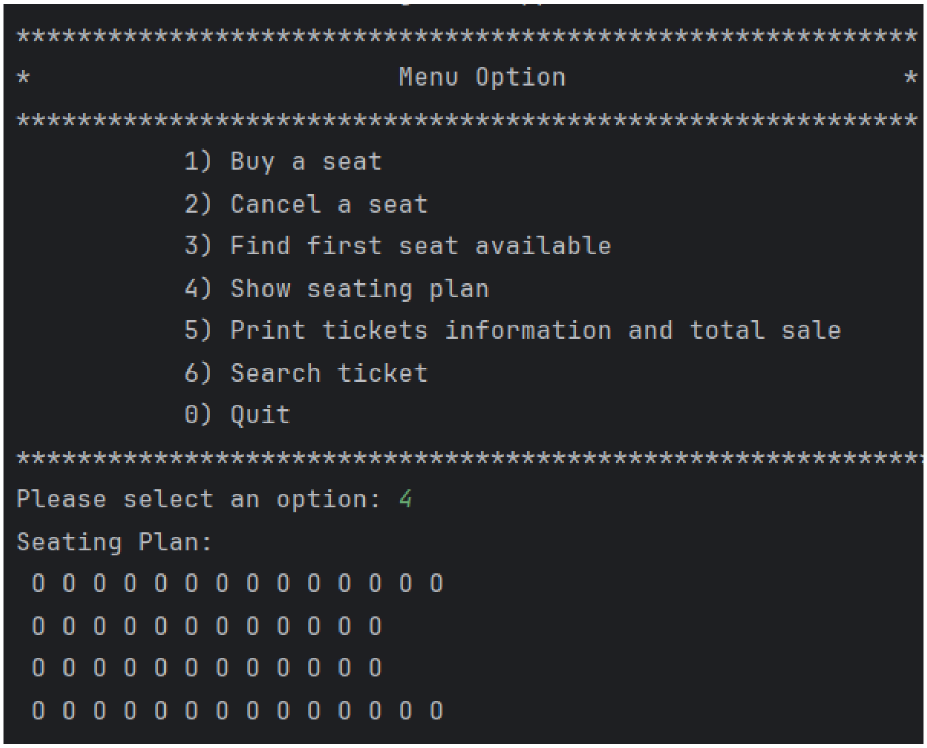


```
Welcome to the Plane Management application
***********************************************************
*                        Menu Option                      *
***********************************************************
           1) Buy a seat
           2) Cancel a seat
           3) Find first seat available
           4) Show seating plan
           5) Print tickets information and total sale
           6) Search ticket
           0) Quit
************************************************************
Please select an option: 
```

| | | |
|---|---|---|
| **3** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |
| **4** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |
| **5** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |

| | | |
|---|---|---|
| **6** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |

**Insert here a screenshot of the seating plan:**



```
***********************************************************
*                        Menu Option                      *
***********************************************************
           1) Buy a seat
           2) Cancel a seat
           3) Find first seat available
           4) Show seating plan
           5) Print tickets information and total sale
           6) Search ticket
           0) Quit
***********************************************************
Please select an option: 4
Seating Plan:
 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

| | | |
|---|---|---|
| **7** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |
| **8** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |
| **9** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |
| **10** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |
| **11** | ☐Fully implemented<br>☒Partially implemented<br>☐Not attempted | Program is running without any errors |

| | | |
|---|---|---|
| **12** | ☒Fully implemented<br>☐Partially implemented<br>☐Not attempted | Program is running without any errors |

## 2) Test Plan

Complete the test plan describing which testing you have performed on your program.
Add as many rows as you need.

### Part A Testing

| Test case / scenario | Input | Expected Output | Output | Pass/Fail |
|---|---|---|---|---|
| **Run the program. Display welcome message and the menu** | | Display Welcome message and the menu | Display Welcome message and the menu | ☒Pass<br>☐Fail |
| **Buy a ticket** | 1 | Display ticket buying progress | Display ticket buying progress | ☒Pass<br>☐Fail |
| **Enter invalid seat number for row B** | 78 | Invalid seat number for row B. Please enter a valid seat number. | Invalid seat number for row B. Please enter a valid seat number. | ☒Pass<br>☐Fail |
| **Enter invalid row letter** | t | Invalid Row. Please enter a valid Row (A-D). | Invalid Row. Please enter a valid Row (A-D). | ☒Pass<br>☐Fail |
| **Cancel a seat** | 2 | Display ticket cancelling progress | Display ticket cancelling progress | ☒Pass<br>☐Fail |
| **Find first seat availability** | 3 | The first available seat is: A1 | The first available seat is: A1 | ☒Pass<br>☐Fail |
| **Show seating plan** | 4 | Display the Seating plan with available seats with ‘0’ and others are “X”. | Display the Seating plan with available seats with ‘0’ and others are “X”. | ☒Pass<br>☐Fail |

# Part B testing

| **Test case / scenario** | **Input** | **Expected Output** | **Output** | **Pass/Fail** |
|---|---|---|---|---|
| **Print ticket information when row “A” seat “7” booked Person (Information are added randomly)** | 5 | Ticket Information:<br>Row: A<br>Seat: 5<br>Price: 200.0<br>Person Information:<br>Name: fdsgd<br>Surname: adfsgfd<br>Email: fdsg<br>Total Sale: £200 | Ticket Information:<br>Row: A<br>Seat: 5<br>Price: 200.0<br>Person Information:<br>Name: fdsgd<br>Surname: adfsgfd<br>Email: fdsg<br>Total Sale: £200 | ☒Pass<br>☐Fail |
| **Get ticket information without booking an ticket.** | 5 | Tickets Information:<br>Total Sale: £0 | Tickets Information:<br>Total Sale: £0 | ☐Pass<br>☐Fail |
| **Search ticket for row “a” seat 7**<br>**(Person Information are added randomly before)** | Enter 6 and Enter Row Letter and Seat Number | Ticket Information:<br>Row: A<br>Seat: 1<br>Price: 200.0<br>Person Information:<br>Name: df<br>Surname: sdf<br>Email: sdf | Ticket Information:<br>Row: A<br>Seat: 1<br>Price: 200.0<br>Person Information:<br>Name: df<br>Surname: sdf<br>Email: sdf | ☐Pass<br>☐Fail |
| **Search ticket for not booked seat** | Enter 6 and Enter Row letter and seat number | This seat is available. | This seat is available. | ☐Pass<br>☐Fail |
| **Save txt file for booking** | Buy a seat | Text file is created in the folder named “RowLetter”+”Seat Number”<br>Eg; A1.txt | Text file is created in the folder named “RowLetter”+”Seat Number”<br>Eg; A1.txt | ☐Pass<br>☐Fail |

Are there any specific parts of the coursework which you would like to get feedback?

You will need to demonstrate your understanding of the submitted code. Your tutor will arrange a coursework demonstration. During the coursework demonstration, your tutor will ask you to execute your program and questions on your code.

**Failure to attend the demonstration will result in 0 for the coursework.**

## 3) Code:

## PlaneManagement.java

```
import java.util.Scanner;


public class PlaneManagement {

    // Define the seat prices matrix

    public static final int[][] SEAT_PRICES = {

            {200, 200, 200, 200, 200, 150, 150, 150, 150, 180, 180, 180, 180, 180},

            {200, 200, 200, 200, 200, 150, 150, 150, 150, 180, 180, 180},

            {200, 200, 200, 200, 200, 150, 150, 150, 150, 180, 180, 180},

            {200, 200, 200, 200, 200, 150, 150, 150, 150, 180, 180, 180, 180, 180}
};


// Define the seat matrix
public static final int ROWS = 4;
public static final int[] SEATS_PER_ROW = {14, 12, 12, 14};
public static final int[][] seats = new int[ROWS][];


public static final Scanner scanner = new Scanner(System.in);
public static final Ticket[] tickets = new Ticket[52];


// Main Method
public static void main(String[] args) {
    initializeSeats();  // Calling the method to initialize seats
    System.out.println("Welcome to the Plane Management application");   // Display welcome message
    int choice;
    do {  // Show the Menu
        System.out.println("***************************************************************");
        System.out.println("*                         Menu Option                        *");
        System.out.println("***************************************************************");
        System.out.println("           1) Buy a seat");
        System.out.println("           2) Cancel a seat");
System.out.println("          3) Find first seat available");

System.out.println("          4) Show seating plan");

System.out.println("          5) Print tickets information and total sale");

System.out.println("          6) Search ticket");

System.out.println("          0) Quit");

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

System.out.print("Please select an option: ");

choice = scanner.nextInt();

scanner.nextLine();



// Calling the methods based on user choice

switch (choice) {

   case 0:

      System.out.println("Exiting program...");

      break;

   case 1:

      buy_seat();

      break;

   case 2:

      cancel_seat();

      break;
```

```
            case 3:
                find_first_available();
                break;
            case 4:
                show_seating_plan();
                break;
            case 5:
                print_tickets_info();
                break;
            case 6:
                search_ticket();
                break;
            default:
                System.out.println("Invalid option. Please select again.");
        }
    }
    while (choice != 0); // Continue until user chooses to quit
    scanner.close();
}

// Method to initialize seats
```

```
public static void initializeSeats() {

    for (int i = 0; i < ROWS; i++) {

        seats[i] = new int[SEATS_PER_ROW[i]];

        for (int j = 0; j < seats[i].length; j++) {

            seats[i][j] = 0;

        }

    }

}



// Method to buy a seat

public static void buy_seat() {

    System.out.print("Enter the row letter (A-D): ");

    char rowLetter = Character.toUpperCase(scanner.next().charAt(0));

    int row = rowLetter - 'A'; // Convert row letter to array index

    if (row < 0 || row >= ROWS) {

        System.out.println("Invalid Row. Please enter a valid Row (A-D).");

        return;

    }



    System.out.print("Enter the seat number: ");

    int seatNumber = scanner.nextInt();
```

```
scanner.nextLine();
if (seatNumber < 1 || seatNumber > SEATS_PER_ROW[row]) {
    System.out.println("Invalid seat number for row " + rowLetter + ". Please enter a valid seat number.");
    return;
}

// Check if seat is already sold
if (seats[row][seatNumber - 1] == 1) {
    System.out.println("Seat " + rowLetter + seatNumber + " is already sold.");
    return;
}

// Prompt user to enter passenger details
System.out.print("Enter passenger's name: ");
String name = scanner.next();
System.out.print("Enter passenger's surname: ");
String surname = scanner.next();
System.out.print("Enter passenger's email: ");
String email = scanner.next();
```

```
    // Create Person and Ticket objects
    Person person = new Person(name, surname, email);
    int price = SEAT_PRICES[row][seatNumber - 1];
    Ticket ticket = new Ticket(rowLetter, seatNumber, price, person);
    ticket.save();

    // Store ticket object in tickets array
    for (int i = 0; i < tickets.length; i++) {
        if (tickets[i] == null) {
            tickets[i] = ticket;
            break;
        }
    }

    // Mark seat as occupied
    seats[row][seatNumber - 1] = 1;
    System.out.println("Seat " + rowLetter + seatNumber + " has been successfully booked.");
}

// Method to cancel a seat reservation
public static void cancel_seat() {
    System.out.print("Enter the row letter (A-D): ");
    char rowLetter = Character.toUpperCase(scanner.next().charAt(0));
    int row = rowLetter - 'A'; // Convert row letter to array index
    if (row < 0 || row >= ROWS) {
        System.out.println("Invalid row. Please enter a valid row (A-D).");
        return;
    }

    System.out.print("Enter the seat number: ");
    int seatNumber = scanner.nextInt();
    scanner.nextLine();
    if (seatNumber < 1 || seatNumber > SEATS_PER_ROW[row]) {
        System.out.println("Invalid seat number for row " + rowLetter + ". Please enter a valid seat number.");
        return;
    }

    if (seats[row][seatNumber - 1] == 0) {
        System.out.println("Seat " + rowLetter + seatNumber + " is not occupied.");
        return;
    }
    // Cancel reservation and update seats matrix
    for (int i = 0; i < tickets.length; i++) {
        if (tickets[i] != null && tickets[i].getSeat() == seatNumber && tickets[i].getRow() == rowLetter) {
            tickets[i] = null;
            break;
        }
    }
    seats[row][seatNumber - 1] = 0;
    System.out.println("Seat reservation for " + rowLetter + seatNumber + " has been successfully canceled.");
}

// Method to find the first available seat
public static void find_first_available() {
    for (int i = 0; i < ROWS; i++) {
        for (int j = 0; j < SEATS_PER_ROW[i]; j++) {
            if (seats[i][j] == 0) {
                char rowLetter = (char) ('A' + i);
                System.out.println("The first available seat is: " + rowLetter + (j + 1));
                return;
            }
        }
    }
    System.out.println("Sorry, No available seats.");
}


// Method to display the seating plan
public static void show_seating_plan() {
    System.out.println("Seating Plan:");
    for (int i = 0; i < ROWS; i++) {
        char rowLetter = (char) ('A' + i);
        System.out.print(" ");
        for (int j = 0; j < SEATS_PER_ROW[i]; j++) {
            if (seats[i][j] == 0) {
                System.out.print("O ");
            } else {
                System.out.print("X ");
            }
        }
        System.out.println();
    }
```

```
}

// Method to print tickets information and total sale
public static void print_tickets_info() {
    int totalPrice = 0;
    System.out.println("Tickets Information:");
    for (Ticket ticket : tickets) {
        if (ticket != null) {
            ticket.printTicketInfo();
            totalPrice += ticket.getPrice();  // Calculate and print total sale
        }
    }
    System.out.println("Total Sale: £" + totalPrice);
}

// Method to search for a ticket by row and seat number
public static void search_ticket() {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter row letter (A-D): ");
    char rowLetter = scanner.next().toUpperCase().charAt(0);
    System.out.print("Enter seat number: ");
```

```
        int seatNumber = scanner.nextInt();
        scanner.nextLine();

        if (rowLetter < 'A' || rowLetter > 'D' || seatNumber < 1 || seatNumber > SEATS_PER_ROW[rowLetter - 'A']) {
            System.out.println("Invalid row letter or seat number.");
            return;
        }

        // Search for ticket matching input and print its information
        boolean found = false;
        for (Ticket ticket : tickets)
            if (ticket != null && ticket.getRow() == rowLetter && ticket.getSeat() == seatNumber) {
                ticket.printTicketInfo();
                found = true;
                break;
            }
        if (!found) {
            System.out.println("This seat is available.");
        }
    }
```

```
}
```

## Person.java

```
public class Person {

    private String name;

    private String surname;

    private String email;



    // Constructor

    public Person(String name, String surname, String email) {

        this.name = name;

        this.surname = surname;

        this.email = email;

    }



    // Getters and setters

    public String getName() {

        return name;

    }



    public void setName(String name) {
```

```
    this.name = name;

}



public String getSurname() {

    return surname;

}



public void setSurname(String surname) {

    this.surname = surname;

}



public String getEmail() {

    return email;

}



public void setEmail(String email) {

    this.email = email;

}



// Method to print person information
```

```
    public void printInfo() {
        System.out.println("Name: " + name);
        System.out.println("Surname: " + surname);
        System.out.println("Email: " + email);
    }
}
```

## Ticket.java

```
import java.io.FileWriter;
import java.io.IOException;

public class Ticket {
    private char row;
    private int seat;
    private double price;
    private Person person;

    // Constructor

    public Ticket(char row, int seat, double price, Person person) {
        this.row = row;
    this.seat = seat;

    this.price = price;

    this.person = person;

}



// Getters and setters



public char getRow() {

    return row;

}



public void setRow(char row) {

    this.row = row;

}



public int getSeat() {

    return seat;

}



public void setSeat(int seat) {
    this.seat = seat;

}



public double getPrice() {

    return price;

}



public void setPrice(double price) {

    this.price = price;

}



public Person getPerson() {

    return person;

}



public void setPerson(Person person) {

    this.person = person;

}



// Method to print ticket information
```

```
public void printTicketInfo() {
    System.out.println("Ticket Information:");
    System.out.println("Row: " + row);
    System.out.println("Seat: " + seat);
    System.out.println("Price: " + price);
    System.out.println("Person Information:");
    person.printInfo();
}

// Method to save ticket information to a file

public void save() {
    String fileName = row + String.valueOf(seat) + ".txt";
    try {
        FileWriter fileWriter = new FileWriter(fileName);
        fileWriter.write("Ticket Information:");
        fileWriter.write("Row: " + row);
        fileWriter.write("Seat: " + seat);
        fileWriter.write("Price: " + price);
```

```
            fileWriter.write("Person Information:");

            fileWriter.write("Name: " + person.getName());

            fileWriter.write("Surname: " + person.getSurname());

            fileWriter.write("Email: " + person.getEmail());

            fileWriter.close();

            System.out.println("Ticket information saved to " + fileName);

        } catch (IOException e) {

            System.out.println("Error occurred while saving ticket information to file.");

            e.printStackTrace();

        }

    }

}
```

<<END>>