Student Names: Amy Percival, Talita Oliveira and Christopher Chan

Student ID: x20163266, x19218958 and x20143087

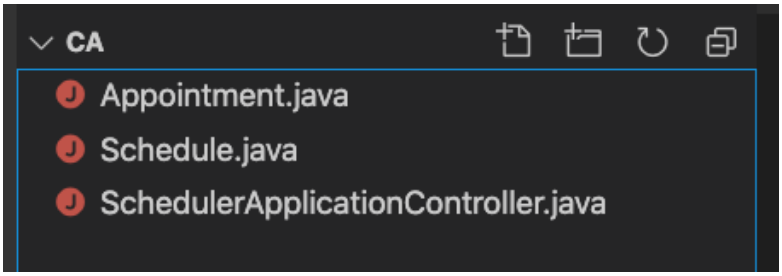
Programme: HDSDEVSEP

Module: Object Orientated Software Engineering

Lecturer: Peeyush Shankhareman

Design Document: Use Case - Make an appointment for a Patient.

For this use-case we implemented three classes in Java, the Appointment, the Schedule and the ScheduleApplicationController, which includes the main method.



Files for the make appointment use case

The ScheduleApplicationController class is the entry point for implementation of this use case, it will accept user input from the console and process the given inputs.

Appointment Class

The Appointment class has private data variables of patient Id*(int*) and reason for visiting(*String*).

```
// ------data variables -----
private int patientID;
private String reason;
```

Data Variables for Appointment Object

It also defines one constructor that passes the patientId and reason to the class.

```
// constructor
public Appointment(int patientID, String reason) {
    this.patientID = patientID;
    this.reason = reason;
}
```

One constructor for the Appointment Class

Schedule Class

The Schedule Class takes has three private data variables an array of type appointment, an array dailyScheduleCheck of type boolean and an array times of type String. When the constructor is called, each one of these arrays is initialised. The times array holds the scheduled times for appointments, the dailyScheduleCheck holds information on whether the time slot is available and the appointment array holds information about the appointments that have been made.

```
// data variables
private Appointment[] appointment;
private boolean[] dailyScheduleCheck;
private String[] times;

// Constructor
public Schedule() {

    // Array of the times avilable for appointments
    this.times = new String[] { "9am", "10am",
    "11am", "12pm", "1pm", "2pm", "3pm", "4pm" };
    // Checking if the time slots are available
    this.dailyScheduleCheck = new boolean[] { true,
    true, true, true, true, true, true, true };
    // There are 8 appointments in the daily
    schedule.
    this.appointment = new Appointment[8];
}
```

Data Variables and Constructor for Schedule Class

Schedule Class - updateSchedule()

The updateSchedule method takes an appointment object and integer timeSlot in parameters. The method checks if time for an appointment is available in the daily-Schedule array. If the timeSlot is available the flag is changed to “false” and the ap-pointment is added to the appointment array at that timeSlot.

```
// updates a schedule with and appointment
public boolean updateSchedule(Appointment app, int timeSlot) {

    if (dailyScheduleCheck[timeSlot - 1] == true) {
        // Sets the timeslot in the schedule to false and add slot
        to database
        dailyScheduleCheck[timeSlot - 1] = false;
        appointment[timeSlot - 1] = app;
        return true;
    }
    return false;
}
```

updateSchedule()

Schedule Class - printDailySchedule()

The printDailySchedule method prints the daily schedule to the console.

```
// prints the schedule
public void printDailySchedule() {
    String slot;
    for (int i = 0; i < times.length; i++) {
        if (dailyScheduleCheck[i] == true) {
            slot = "Yes";
        } else {
            slot = "No";
        }
        System.out.println((i + 1) + ". " + "Time Slot at " + times[i] + " is Available:
        " + slot);
        if (dailyScheduleCheck[i] == false) {
            System.out.println("   PatientID: " + appointment[i].getPatientID() + ". The
            Reason: "
                    + appointment[i].getReason());
        }
        System.out.println();
    }
}
```

printDailySchedule()

Schedule Class - scheduleNotFull()

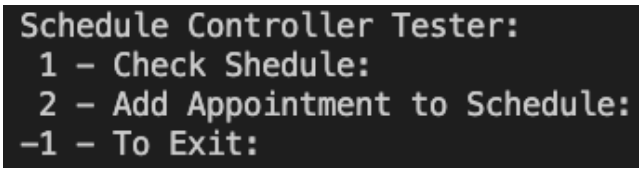
A method to check if the daily schedule is is full (has more space for appointments) or not.

```
// returns true if schedule is not full
public boolean scheduleNotFull() {
    for (int i = 0; i < dailyScheduleCheck.length; i++) {
        if (dailyScheduleCheck[i]) {
            return true;
        }
    }
    return false;
}
```

scheduleNotFull()

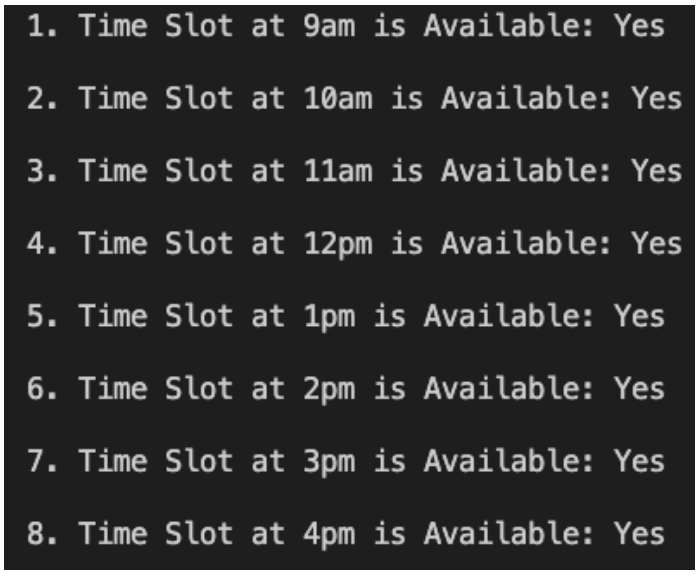
ScheduleApplicationController

The ScheduleApplicationController is the entry point for user input. It accepts inputs from the console and processes those inputs. When the application is run a new Schedule Object is created and there are three options displayed to the user.



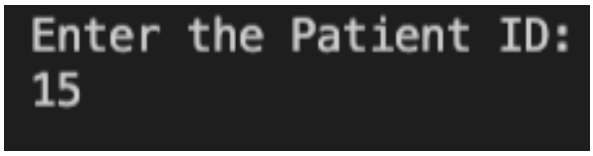
Options for schedule adding appointment

Selecting *option1* calls the printDailySchedule method defined in the schedule class. By default the daily schedule is empty. The user can add an appointment to the time slot if that slot is available.



The schedule displayed

When the user decides *option2*, to add an appointment, the user is prompted to enter a patient Id.

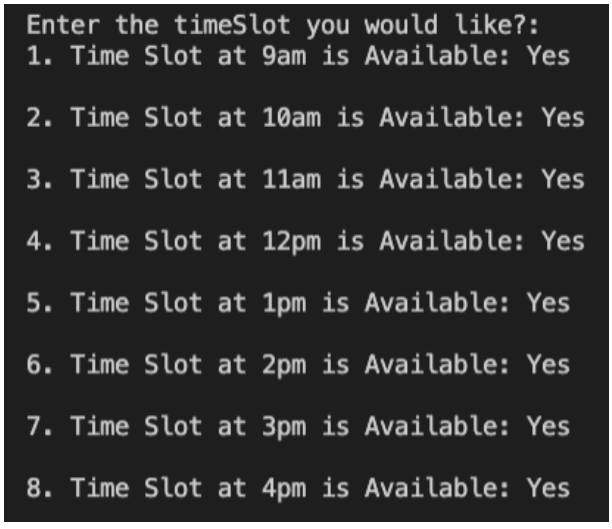


Enter the patient Id

Then prompted to enter the reason for visiting

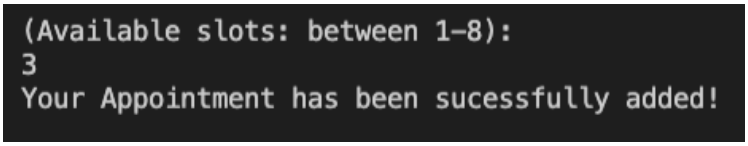
The reason of visiting the clinic

The user will then be shown the schedule again. They will be promoted to enter an available time between 1-8 that corresponds to times from 9am to 4pm.



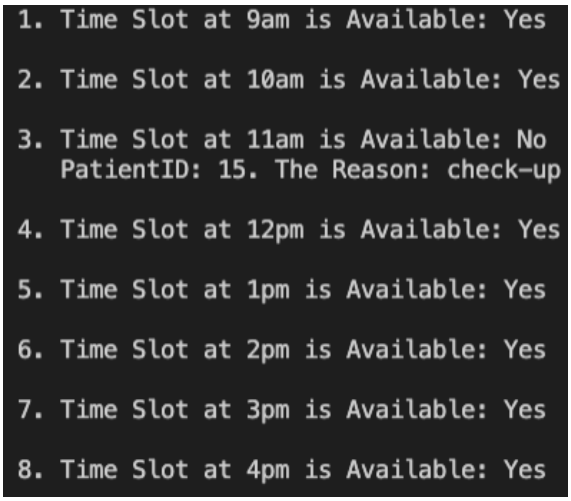
Enter the desired time slot

Once a time slot has been chosen the appointment will be confirmed.



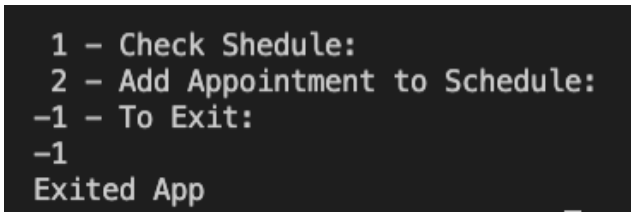
Confirmation that the appointment has been successfully added.

If the schedule is checked we can see our appointment has been added. Appointments can be added until there are no available slots in the schedule.



Checking the updated schedule

Selection *option -1* exits the console application



The app is exited