Department of Computer Science & Engineering
UNIVERSITY OF LIBERAL ARTS
BANGLADESH

Lab Final

Submitted to:
Shakib Mahmud Dipto
Faculty

Submitted by:
Sumaiya Akter
ID: 201014071

Department of CSE
Summer'24

Course code: CSE 2104
Course Title: Object Oriented Programming Lab
Section: 01

University of Liberal Arts Bangladesh

September 12, 2024

Problem 01

Class Explanation :

A number of essential classes make up the Doctor's Appointment Management System, which functions as a user-friendly and practical application for scheduling medical appointments. All doctors use the Doctor class as a template, which includes their name, area of expertise, and schedule of appointments.

The Doctor class is expanded upon by the General Practitioner and Specialist classes, which offer particular behaviors for each category of physician, such as accepting walk-in patients or demanding appointment confirmations.

The Patient class, which keeps track of appointments, name, and contact details, depicts people in need of medical attention.

Each appointment's specifics, such as the patient, doctor, date, and diagnosis, are contained in the Appointment class, which also manages file storing of this data.

Last but not least, the ClinicManagementSystem class serves as the primary controller, overseeing patient and physician interactions and enabling users to register new patients, make appointments, view available doctors, and monitor existing appointments. The integration of these classes results in a unified system that streamlines the administration of patient-doctor interactions in a clinical context.

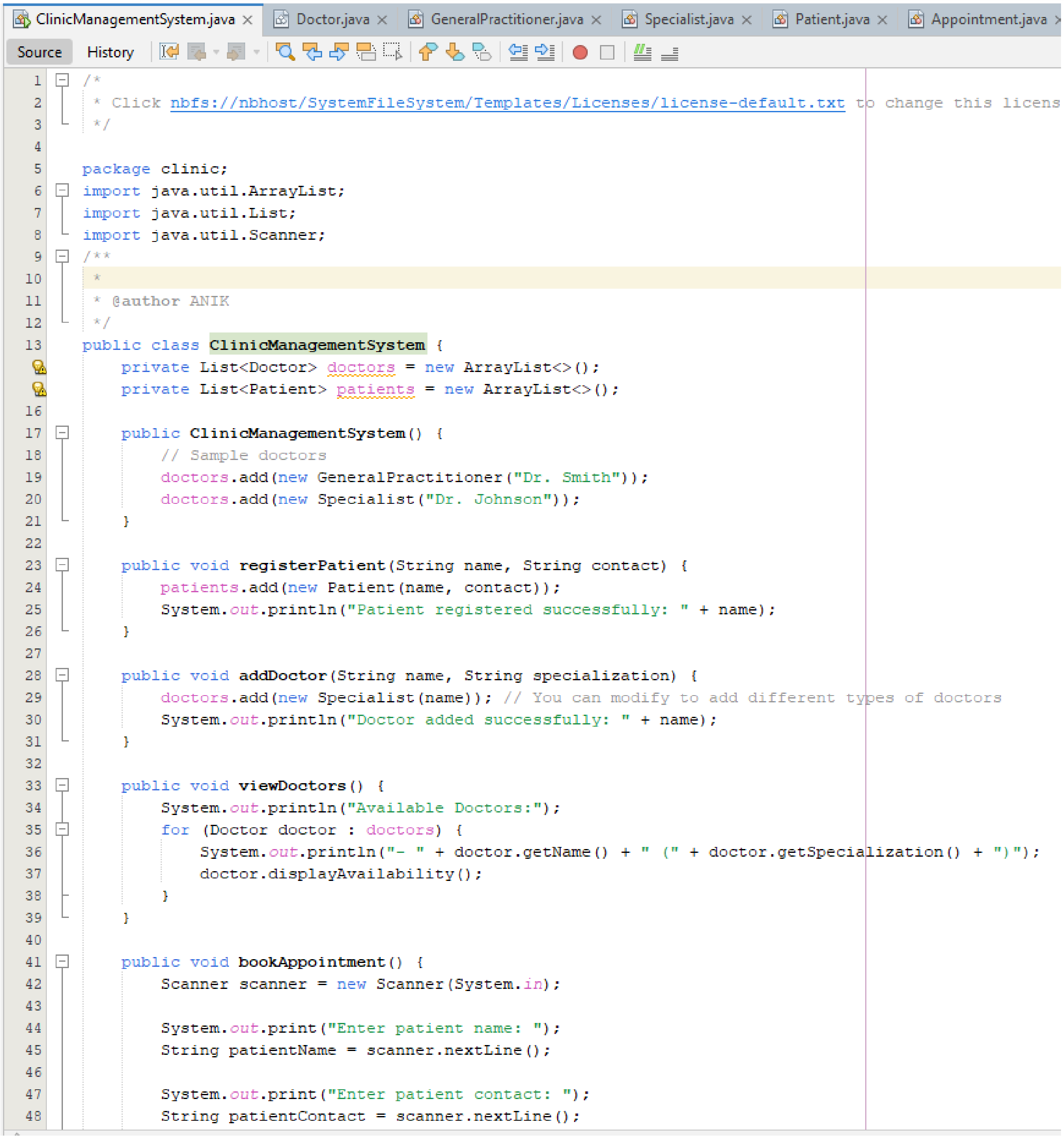
OOP concepts used:

The Doctor's Appointment Management System uses Object-Oriented Programming (OOP) concepts to create a user-friendly application that feels intuitive. It features classes that act as templates for doctors, patients, and appointments, allowing us to create specific objects for each entity. Inheritance lets us define specialized doctor types, like General Practitioners and Specialists, that share common attributes while having unique behaviors. Polymorphism allows these doctors to implement the same method, ‘displayAvailability()’, in different ways, reflecting their practices. Encapsulation keeps sensitive data secure by bundling attributes and methods within classes, while abstraction simplifies user interactions by hiding complex details, such as file handling. Together, these concepts create a cohesive system that efficiently manages the relationships between doctors, patients, and their appointments, making the process smooth and organized.

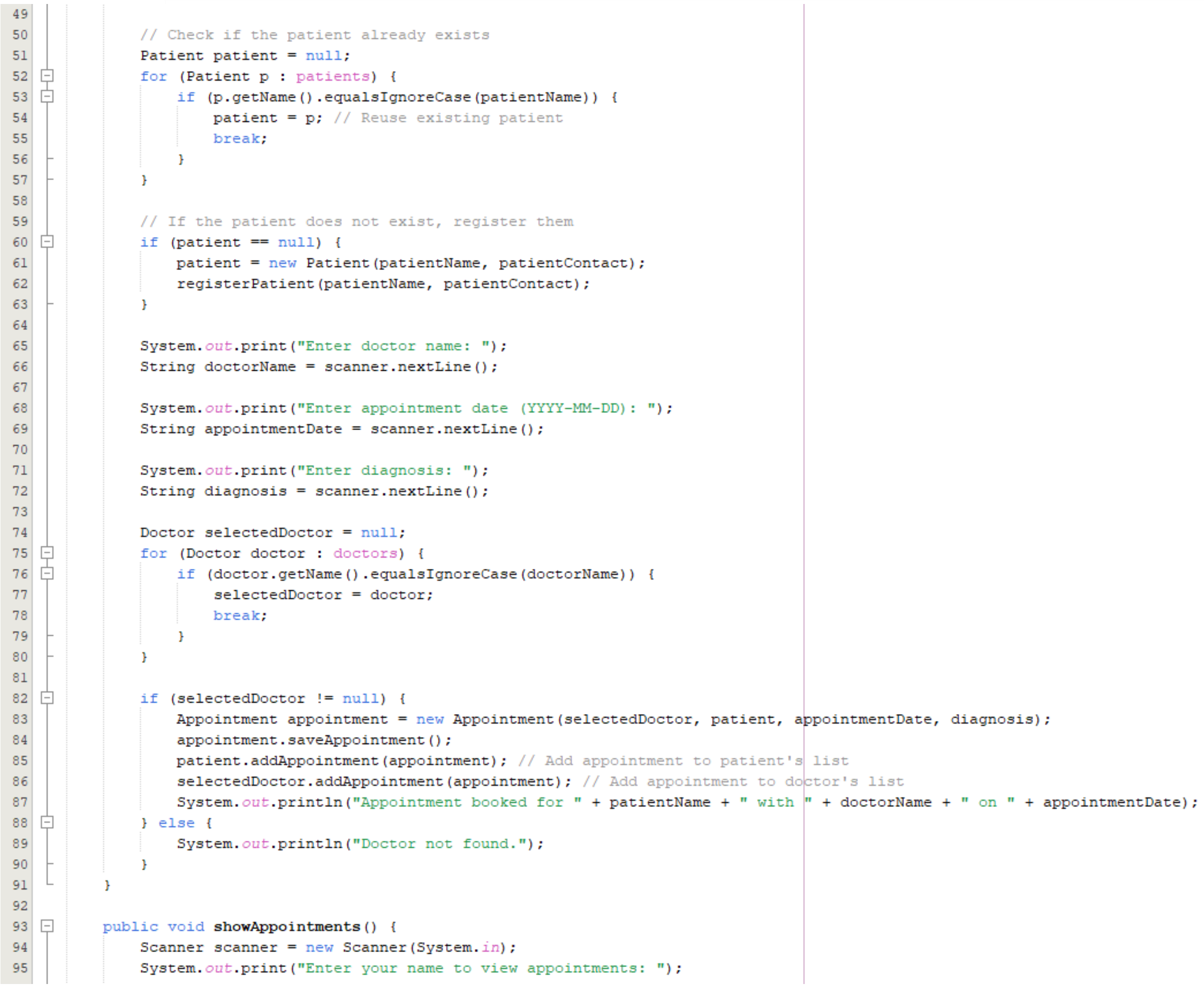
Code Screenshots:

Input:

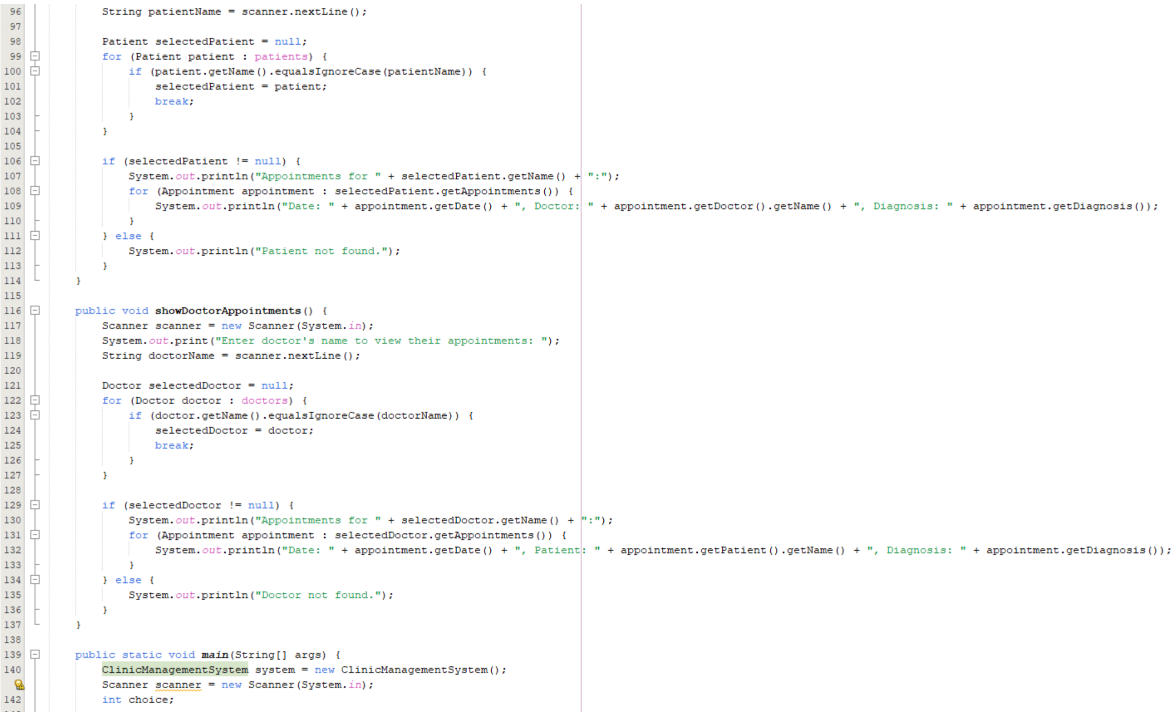
ClinicManagementSystem.java



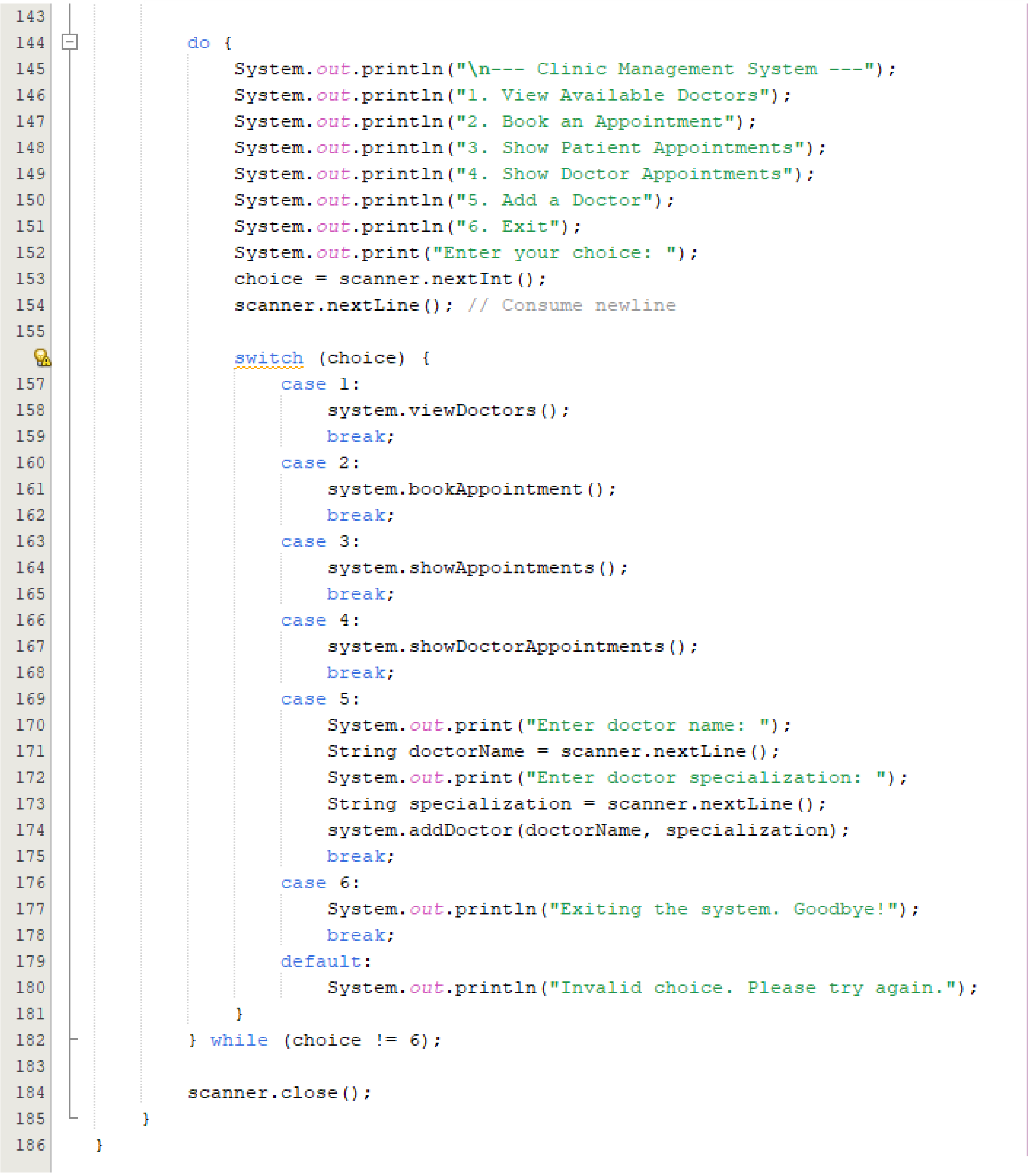
```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this licens
 */

package clinic;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
/**
 *
 * @author ANIK
 */
public class ClinicManagementSystem {
    private List<Doctor> doctors = new ArrayList<>();
    private List<Patient> patients = new ArrayList<>();

    public ClinicManagementSystem() {
        // Sample doctors
        doctors.add(new GeneralPractitioner("Dr. Smith"));
        doctors.add(new Specialist("Dr. Johnson"));
    }

    public void registerPatient(String name, String contact) {
        patients.add(new Patient(name, contact));
        System.out.println("Patient registered successfully: " + name);
    }

    public void addDoctor(String name, String specialization) {
        doctors.add(new Specialist(name)); // You can modify to add different types of doctors
        System.out.println("Doctor added successfully: " + name);
    }

    public void viewDoctors() {
        System.out.println("Available Doctors:");
        for (Doctor doctor : doctors) {
            System.out.println("- " + doctor.getName() + " (" + doctor.getSpecialization() + ")");
            doctor.displayAvailability();
        }
    }

    public void bookAppointment() {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter patient name: ");
        String patientName = scanner.nextLine();

        System.out.print("Enter patient contact: ");
        String patientContact = scanner.nextLine();
```



```java

        // Check if the patient already exists
        Patient patient = null;
        for (Patient p : patients) {
            if (p.getName().equalsIgnoreCase(patientName)) {
                patient = p; // Reuse existing patient
                break;
            }
        }

        // If the patient does not exist, register them
        if (patient == null) {
            patient = new Patient(patientName, patientContact);
            registerPatient(patientName, patientContact);
        }

        System.out.print("Enter doctor name: ");
        String doctorName = scanner.nextLine();

        System.out.print("Enter appointment date (YYYY-MM-DD): ");
        String appointmentDate = scanner.nextLine();

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

        Doctor selectedDoctor = null;
        for (Doctor doctor : doctors) {
            if (doctor.getName().equalsIgnoreCase(doctorName)) {
                selectedDoctor = doctor;
                break;
            }
        }

        if (selectedDoctor != null) {
            Appointment appointment = new Appointment(selectedDoctor, patient, appointmentDate, diagnosis);
            appointment.saveAppointment();
            patient.addAppointment(appointment); // Add appointment to patient's list
            selectedDoctor.addAppointment(appointment); // Add appointment to doctor's list
            System.out.println("Appointment booked for " + patientName + " with " + doctorName + " on " + appointmentDate);
        } else {
            System.out.println("Doctor not found.");
        }
    }

    public void showAppointments() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter your name to view appointments: ");
```

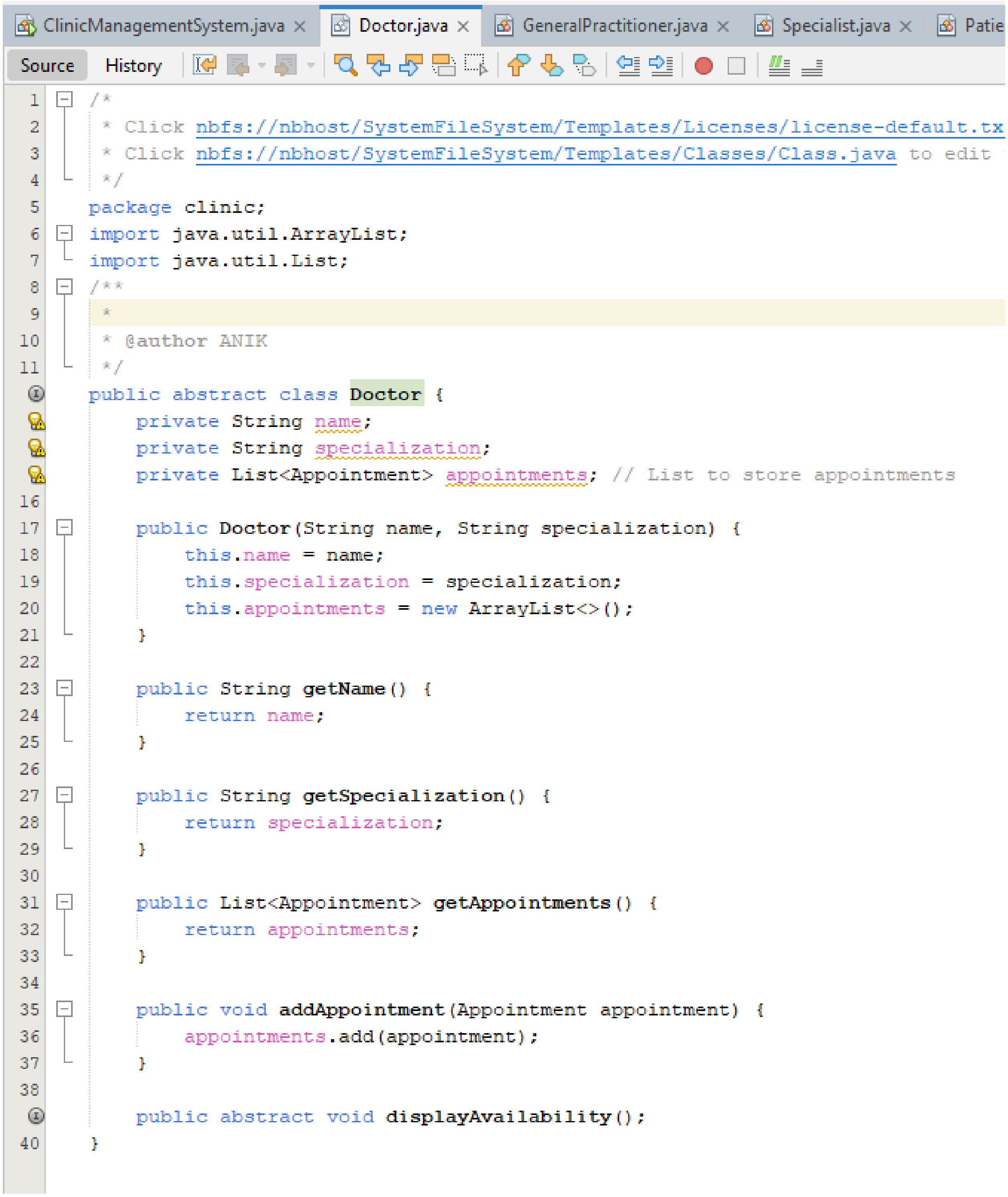


```java
        String patientName = scanner.nextLine();

        Patient selectedPatient = null;
        for (Patient patient : patients) {
            if (patient.getName().equalsIgnoreCase(patientName)) {
                selectedPatient = patient;
                break;
            }
        }

        if (selectedPatient != null) {
            System.out.println("Appointments for " + selectedPatient.getName() + ":");
            for (Appointment appointment : selectedPatient.getAppointments()) {
                System.out.println("Date: " + appointment.getDate() + ", Doctor: " + appointment.getDoctor().getName() + ", Diagnosis: " + appointment.getDiagnosis());
            }
        } else {
            System.out.println("Patient not found.");
        }
    }

    public void showDoctorAppointments() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter doctor's name to view their appointments: ");
        String doctorName = scanner.nextLine();

        Doctor selectedDoctor = null;
        for (Doctor doctor : doctors) {
            if (doctor.getName().equalsIgnoreCase(doctorName)) {
                selectedDoctor = doctor;
                break;
            }
        }

        if (selectedDoctor != null) {
            System.out.println("Appointments for " + selectedDoctor.getName() + ":");
            for (Appointment appointment : selectedDoctor.getAppointments()) {
                System.out.println("Date: " + appointment.getDate() + ", Patient: " + appointment.getPatient().getName() + ", Diagnosis: " + appointment.getDiagnosis());
            }
        } else {
            System.out.println("Doctor not found.");
        }
    }

    public static void main(String[] args) {
        ClinicManagementSystem system = new ClinicManagementSystem();
        Scanner scanner = new Scanner(System.in);
        int choice;
```



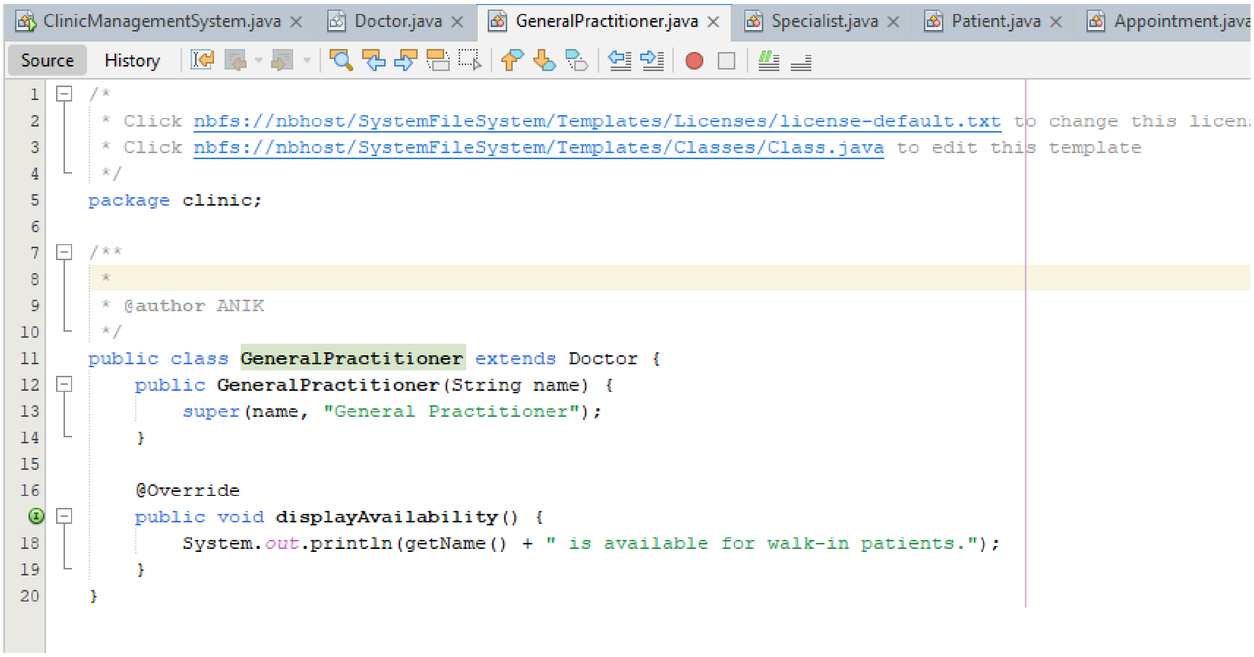
```java

        do {
            System.out.println("\n--- Clinic Management System ---");
            System.out.println("1. View Available Doctors");
            System.out.println("2. Book an Appointment");
            System.out.println("3. Show Patient Appointments");
            System.out.println("4. Show Doctor Appointments");
            System.out.println("5. Add a Doctor");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline

            switch (choice) {
                case 1:
                    system.viewDoctors();
                    break;
                case 2:
                    system.bookAppointment();
                    break;
                case 3:
                    system.showAppointments();
                    break;
                case 4:
                    system.showDoctorAppointments();
                    break;
                case 5:
                    System.out.print("Enter doctor name: ");
                    String doctorName = scanner.nextLine();
                    System.out.print("Enter doctor specialization: ");
                    String specialization = scanner.nextLine();
                    system.addDoctor(doctorName, specialization);
                    break;
                case 6:
                    System.out.println("Exiting the system. Goodbye!");
                    break;
                default:
                    System.out.println("Invalid choice. Please try again.");
            }
        } while (choice != 6);

        scanner.close();
    }
}
```

Doctor.java



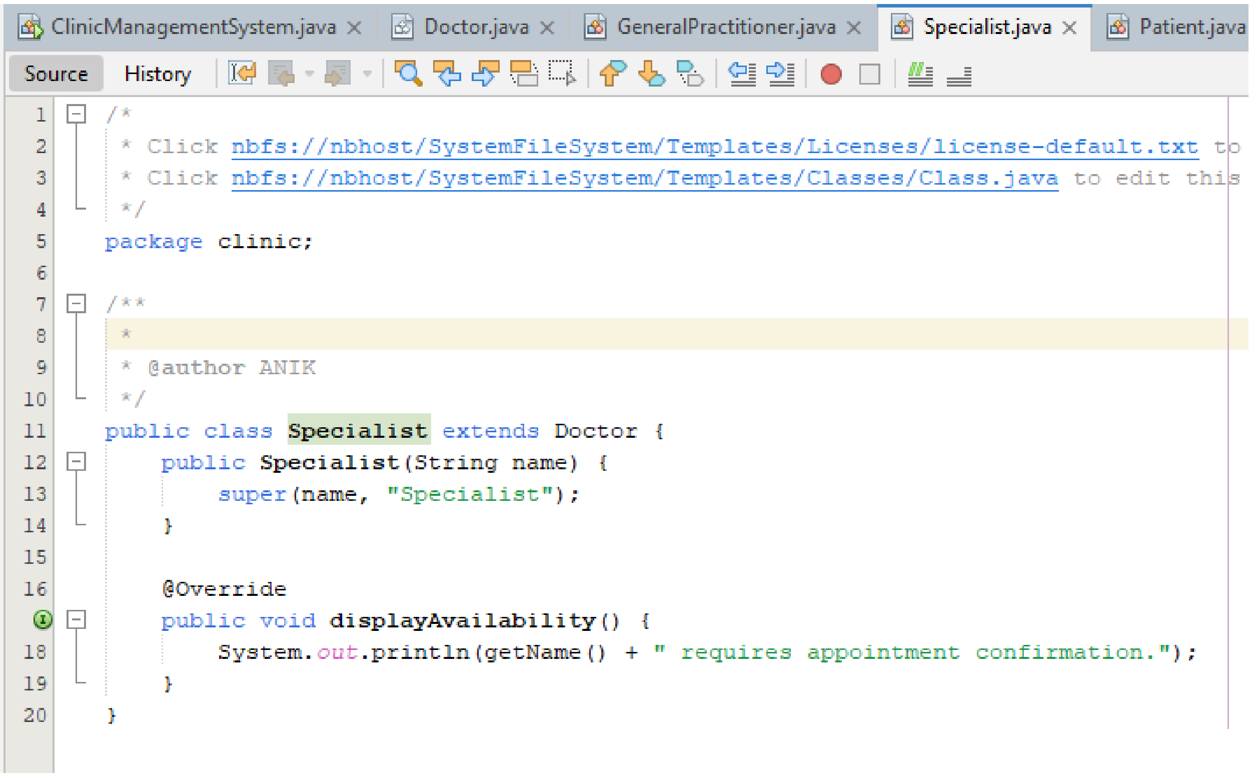
```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.tx
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit
 */
package clinic;
import java.util.ArrayList;
import java.util.List;
/**
 *
 * @author ANIK
 */
public abstract class Doctor {
    private String name;
    private String specialization;
    private List<Appointment> appointments; // List to store appointments

    public Doctor(String name, String specialization) {
        this.name = name;
        this.specialization = specialization;
        this.appointments = new ArrayList<>();
    }

    public String getName() {
        return name;
    }

    public String getSpecialization() {
        return specialization;
    }

    public List<Appointment> getAppointments() {
        return appointments;
    }

    public void addAppointment(Appointment appointment) {
        appointments.add(appointment);
    }

    public abstract void displayAvailability();
}
```

GeneralPractitioner.java



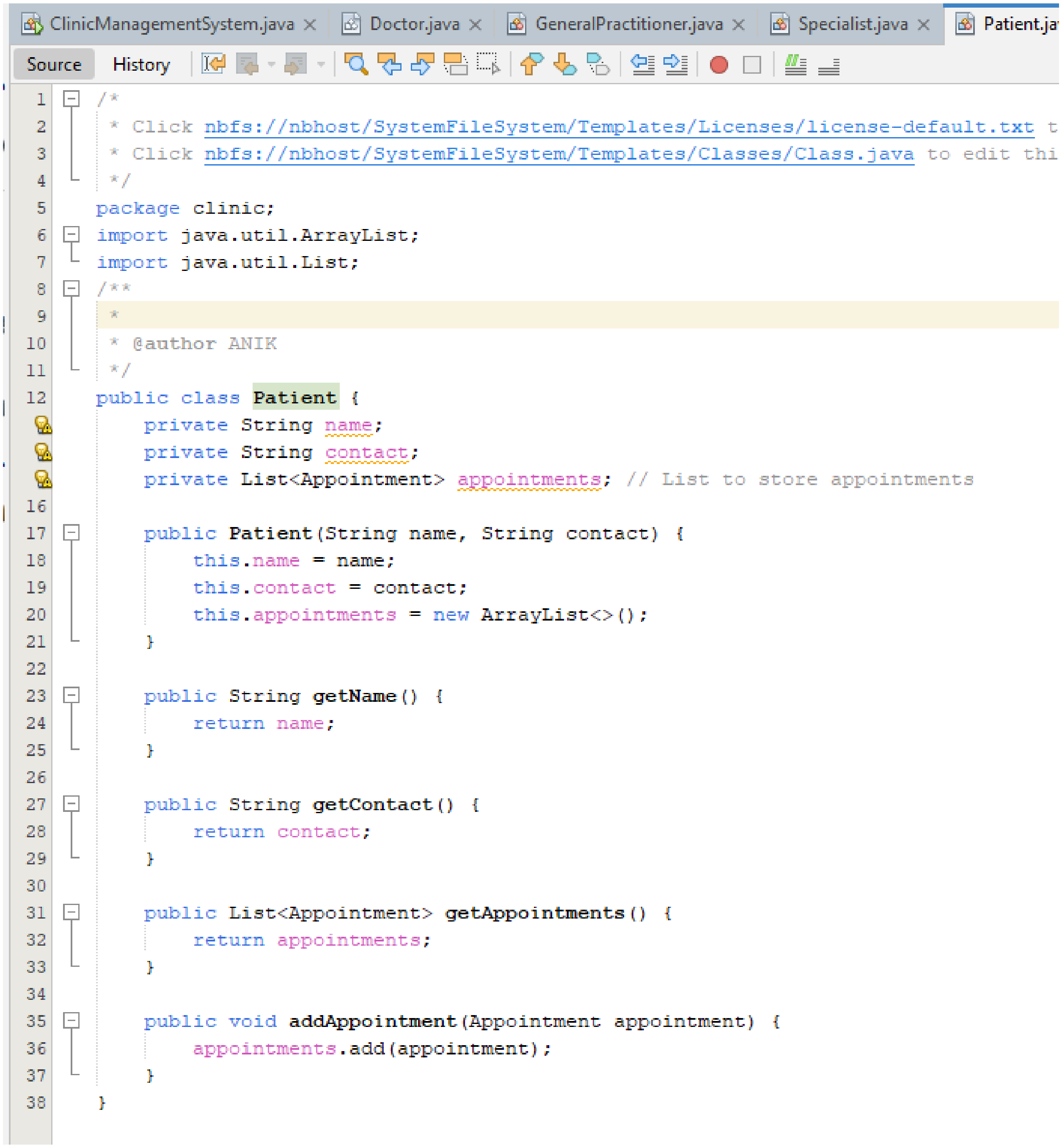
```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this licen
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package clinic;

/**
 *
 * @author ANIK
 */
public class GeneralPractitioner extends Doctor {
    public GeneralPractitioner(String name) {
        super(name, "General Practitioner");
    }

    @Override
    public void displayAvailability() {
        System.out.println(getName() + " is available for walk-in patients.");
    }
}
```

Specialist.java



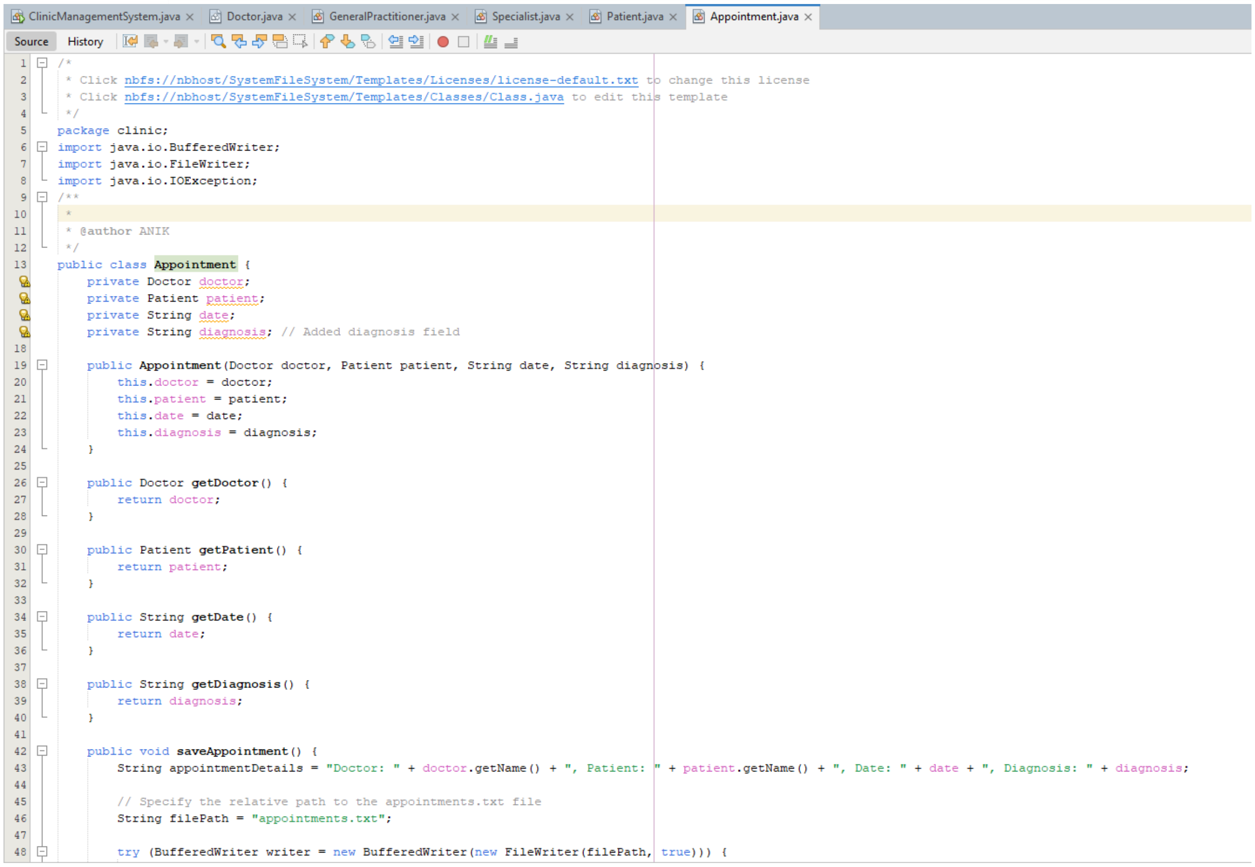
```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
 */
package clinic;

/**
 *
 * @author ANIK
 */
public class Specialist extends Doctor {
    public Specialist(String name) {
        super(name, "Specialist");
    }

    @Override
    public void displayAvailability() {
        System.out.println(getName() + " requires appointment confirmation.");
    }
}
```

Patient.java

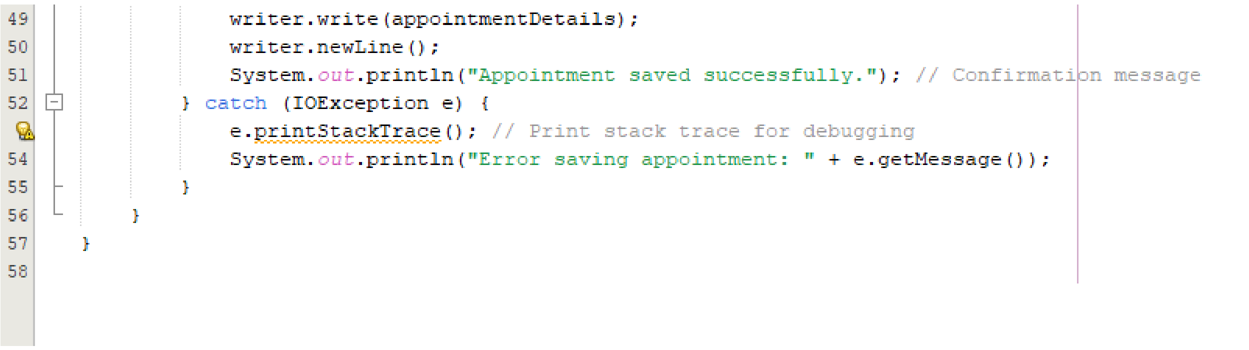


```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt t
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit thi
 */
package clinic;
import java.util.ArrayList;
import java.util.List;
/**
 *
 * @author ANIK
 */
public class Patient {
    private String name;
    private String contact;
    private List<Appointment> appointments; // List to store appointments

    public Patient(String name, String contact) {
        this.name = name;
        this.contact = contact;
        this.appointments = new ArrayList<>();
    }

    public String getName() {
        return name;
    }

    public String getContact() {
        return contact;
    }

    public List<Appointment> getAppointments() {
        return appointments;
    }

    public void addAppointment(Appointment appointment) {
        appointments.add(appointment);
    }
}
```

Appointment.java

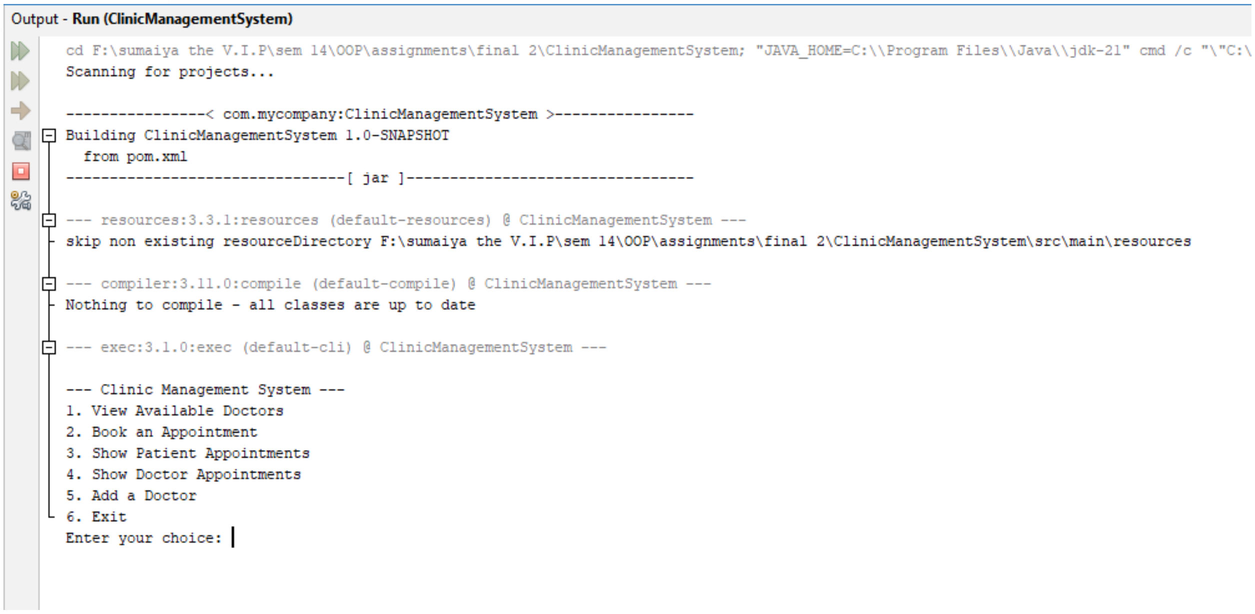


```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package clinic;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
/**
 *
 * @author ANIK
 */
public class Appointment {
    private Doctor doctor;
    private Patient patient;
    private String date;
    private String diagnosis; // Added diagnosis field

    public Appointment(Doctor doctor, Patient patient, String date, String diagnosis) {
        this.doctor = doctor;
        this.patient = patient;
        this.date = date;
        this.diagnosis = diagnosis;
    }

    public Doctor getDoctor() {
        return doctor;
    }

    public Patient getPatient() {
        return patient;
    }

    public String getDate() {
        return date;
    }

    public String getDiagnosis() {
        return diagnosis;
    }

    public void saveAppointment() {
        String appointmentDetails = "Doctor: " + doctor.getName() + ", Patient: " + patient.getName() + ", Date: " + date + ", Diagnosis: " + diagnosis;

        // Specify the relative path to the appointments.txt file
        String filePath = "appointments.txt";

        try (BufferedWriter writer = new BufferedWriter(new FileWriter(filePath, true))) {
```

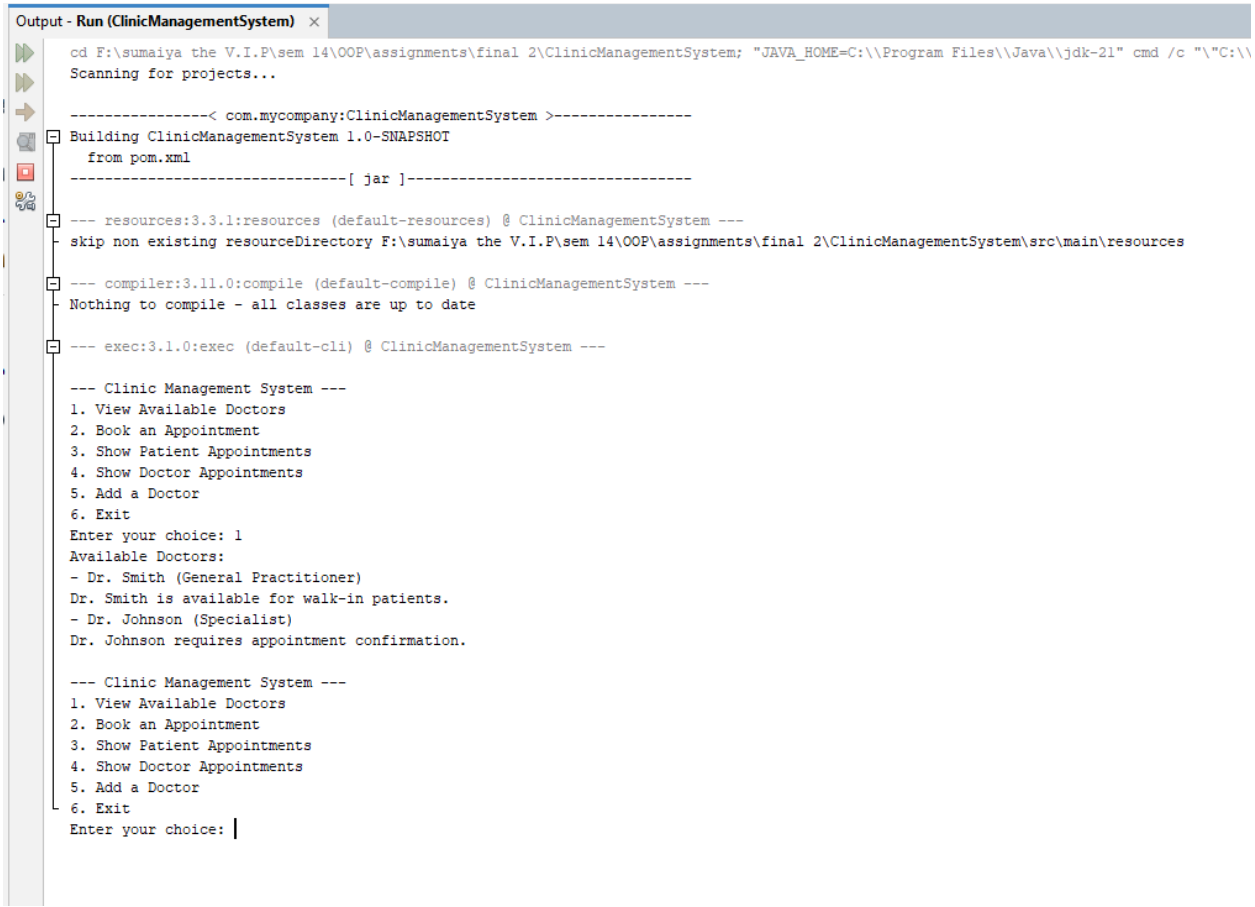


```java
            writer.write(appointmentDetails);
            writer.newLine();
            System.out.println("Appointment saved successfully."); // Confirmation message
        } catch (IOException e) {
            e.printStackTrace(); // Print stack trace for debugging
            System.out.println("Error saving appointment: " + e.getMessage());
        }
    }
}
```

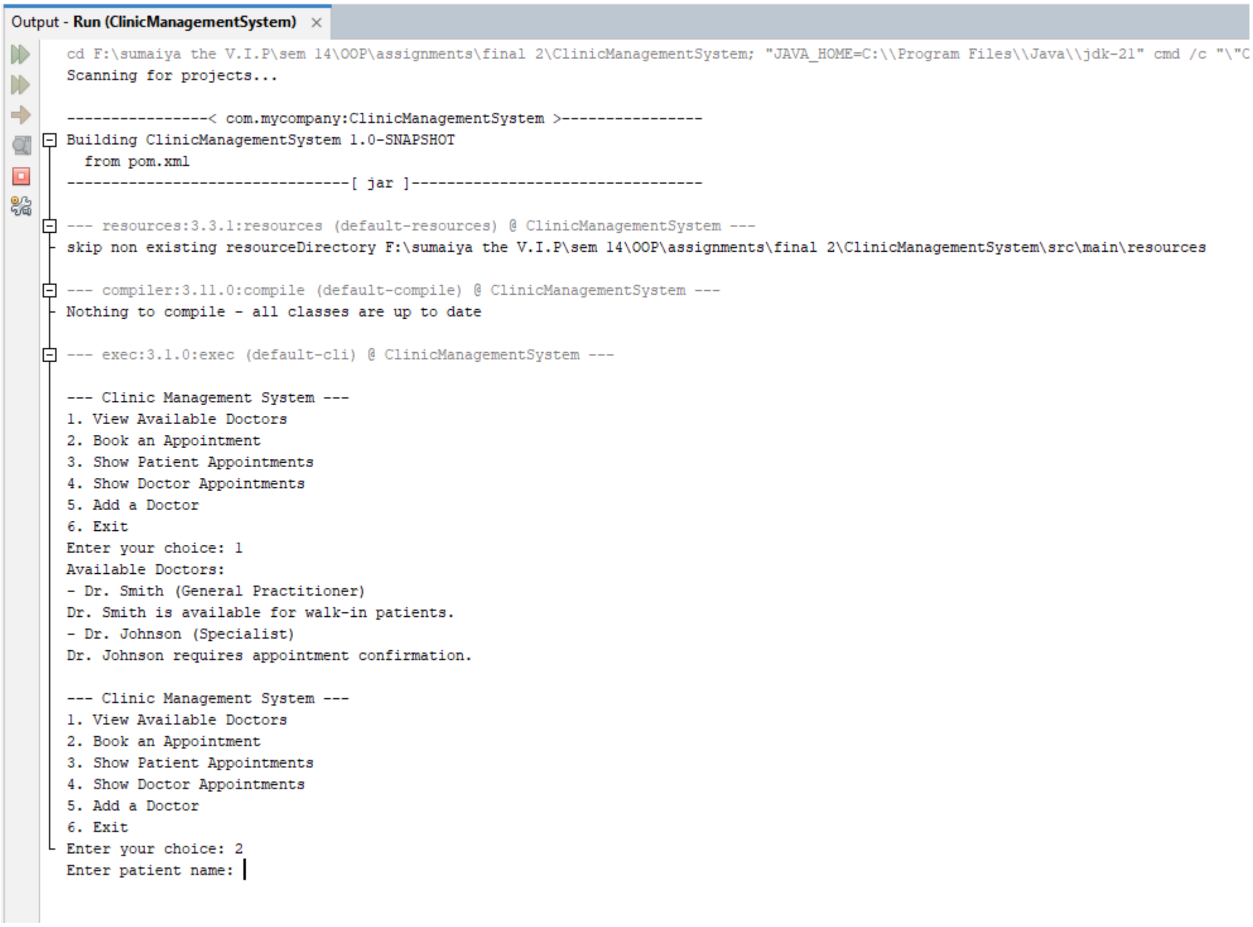
Output:



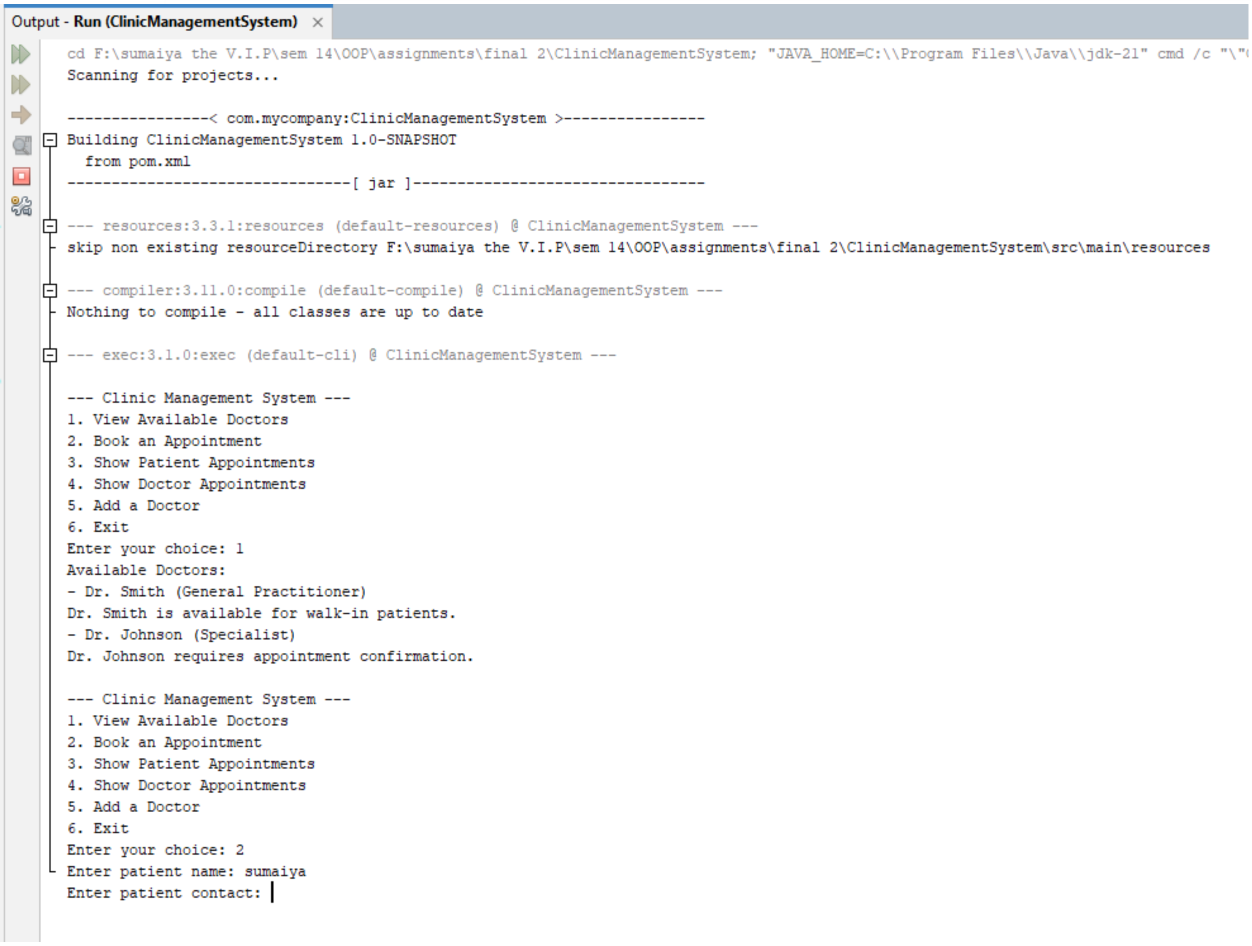
```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem; "JAVA_HOME=C:\\Program Files\\Java\\jdk-21" cmd /c "\"C:\
Scanning for projects...

----------------< com.mycompany:ClinicManagementSystem >----------------
Building ClinicManagementSystem 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ ClinicManagementSystem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ ClinicManagementSystem ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ ClinicManagementSystem ---

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 
```



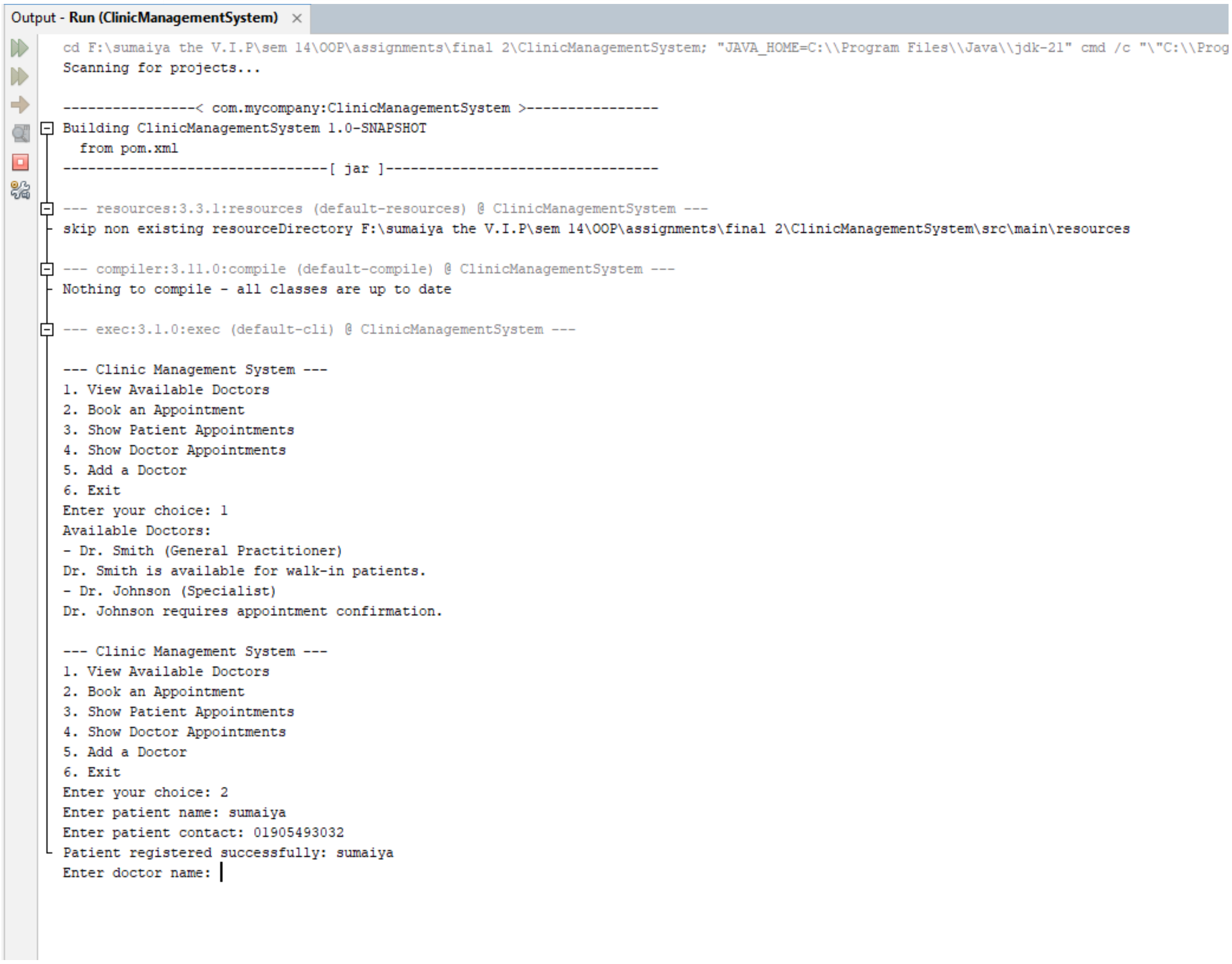
```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem; "JAVA_HOME=C:\\Program Files\\Java\\jdk-21" cmd /c "\"C:\\
Scanning for projects...

----------------< com.mycompany:ClinicManagementSystem >----------------
Building ClinicManagementSystem 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ ClinicManagementSystem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ ClinicManagementSystem ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ ClinicManagementSystem ---

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 1
Available Doctors:
- Dr. Smith (General Practitioner)
Dr. Smith is available for walk-in patients.
- Dr. Johnson (Specialist)
Dr. Johnson requires appointment confirmation.

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 
```



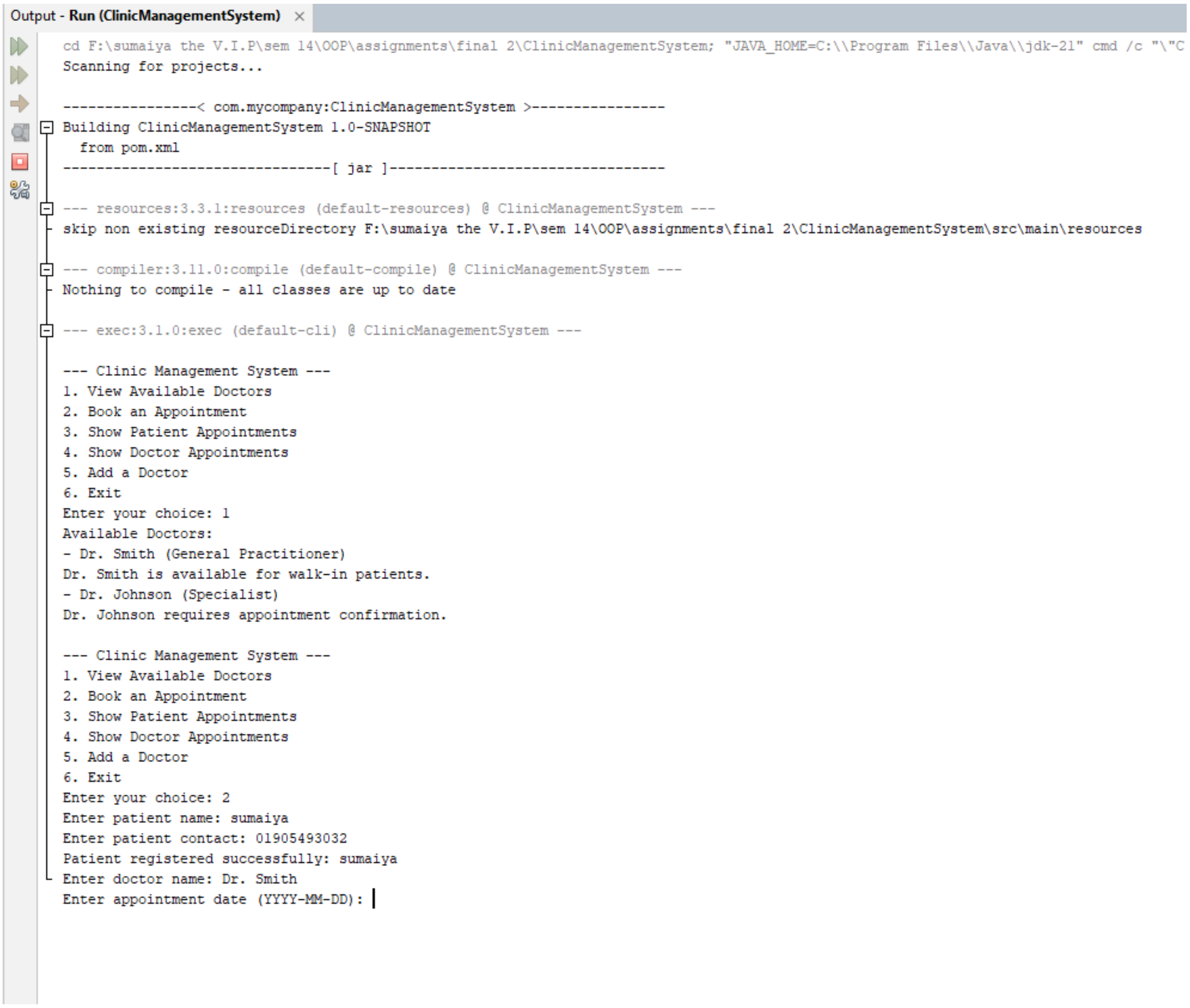
```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem; "JAVA_HOME=C:\\Program Files\\Java\\jdk-21" cmd /c "\"C
Scanning for projects...

----------------< com.mycompany:ClinicManagementSystem >----------------
Building ClinicManagementSystem 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ ClinicManagementSystem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ ClinicManagementSystem ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ ClinicManagementSystem ---

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 1
Available Doctors:
- Dr. Smith (General Practitioner)
Dr. Smith is available for walk-in patients.
- Dr. Johnson (Specialist)
Dr. Johnson requires appointment confirmation.

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 2
Enter patient name: 
```



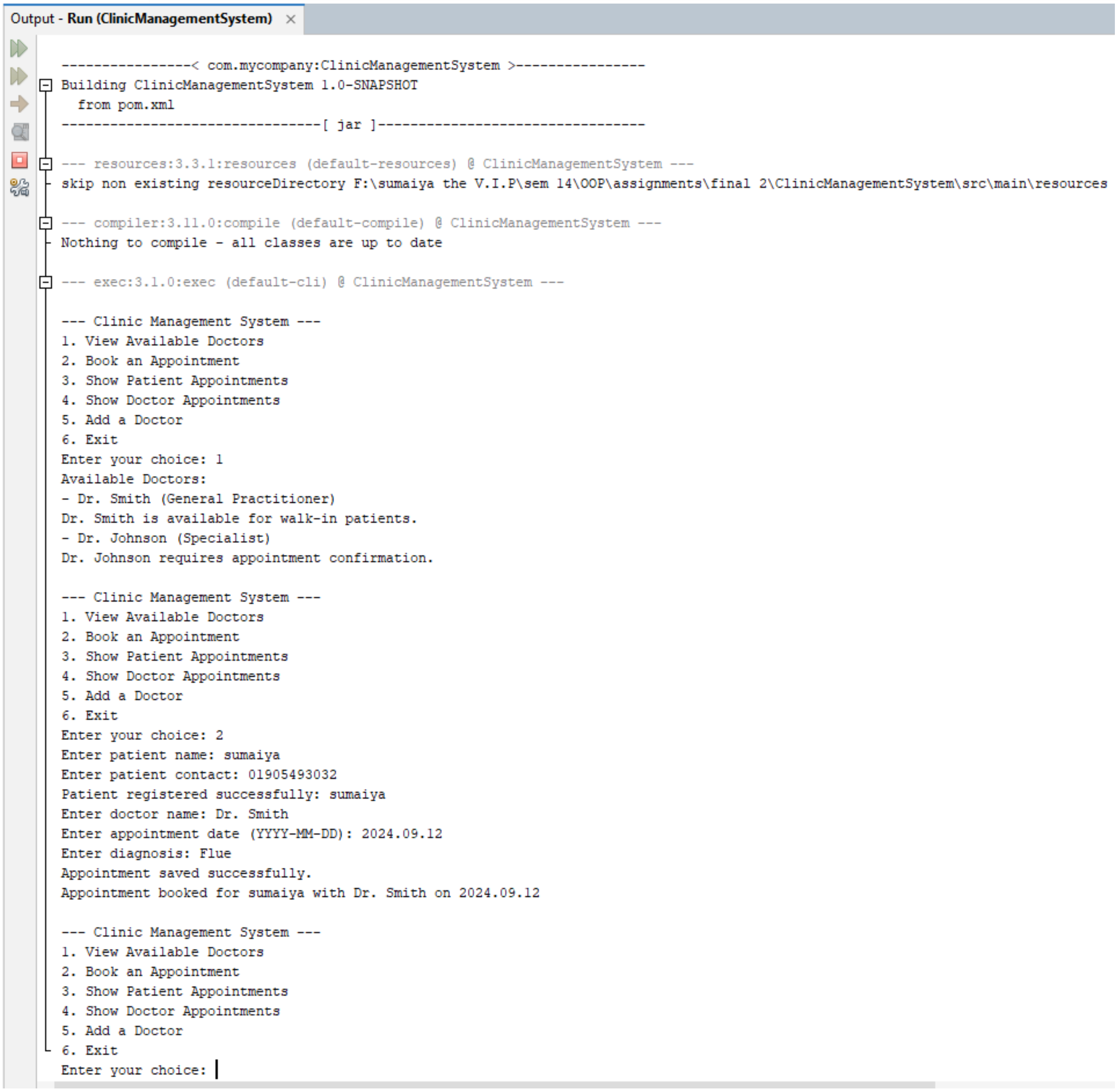
```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem; "JAVA_HOME=C:\\Program Files\\Java\\jdk-21" cmd /c "\"
Scanning for projects...

----------------< com.mycompany:ClinicManagementSystem >----------------
Building ClinicManagementSystem 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ ClinicManagementSystem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ ClinicManagementSystem ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ ClinicManagementSystem ---

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 1
Available Doctors:
- Dr. Smith (General Practitioner)
Dr. Smith is available for walk-in patients.
- Dr. Johnson (Specialist)
Dr. Johnson requires appointment confirmation.

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 2
Enter patient name: sumaiya
Enter patient contact: 
```



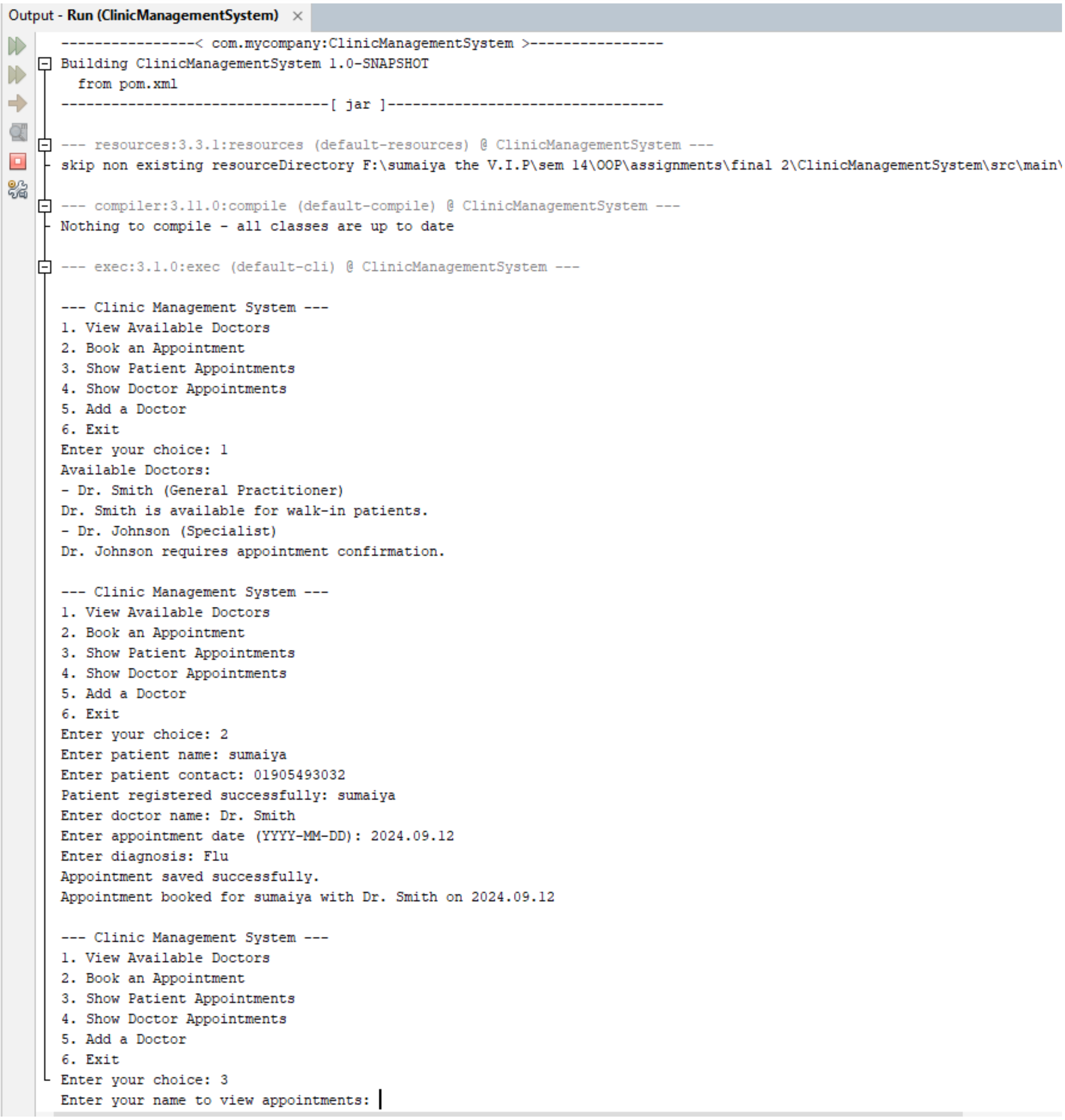
```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem; "JAVA_HOME=C:\\Program Files\\Java\\jdk-21" cmd /c "\"C:\\Prog
Scanning for projects...

----------------< com.mycompany:ClinicManagementSystem >----------------
Building ClinicManagementSystem 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ ClinicManagementSystem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ ClinicManagementSystem ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ ClinicManagementSystem ---

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 1
Available Doctors:
- Dr. Smith (General Practitioner)
Dr. Smith is available for walk-in patients.
- Dr. Johnson (Specialist)
Dr. Johnson requires appointment confirmation.

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 2
Enter patient name: sumaiya
Enter patient contact: 01905493032
Patient registered successfully: sumaiya
Enter doctor name:
```



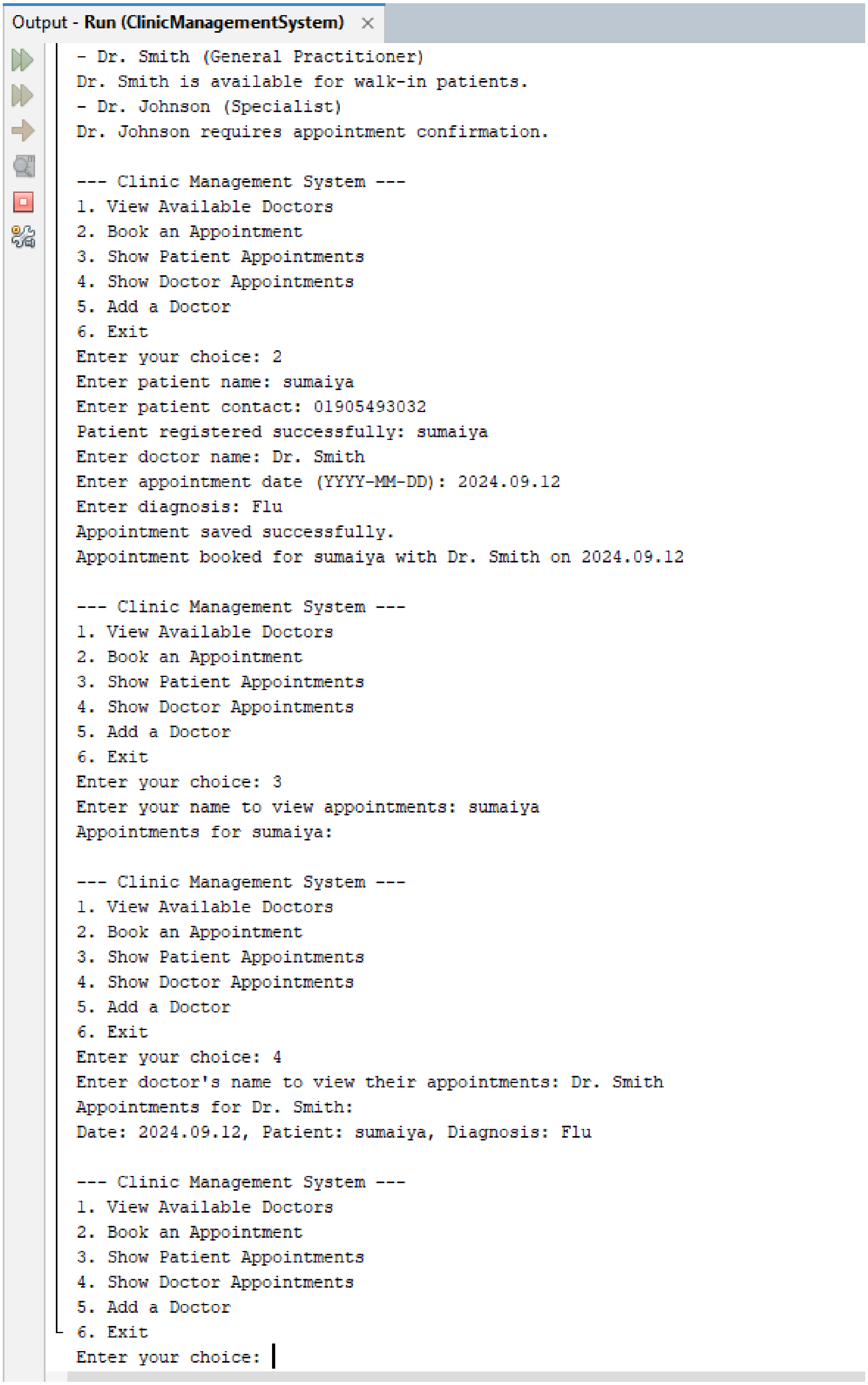
```
cd F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem; "JAVA_HOME=C:\\Program Files\\Java\\jdk-21" cmd /c "\"C
Scanning for projects...

----------------< com.mycompany:ClinicManagementSystem >----------------
Building ClinicManagementSystem 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ ClinicManagementSystem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ ClinicManagementSystem ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ ClinicManagementSystem ---

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 1
Available Doctors:
- Dr. Smith (General Practitioner)
Dr. Smith is available for walk-in patients.
- Dr. Johnson (Specialist)
Dr. Johnson requires appointment confirmation.

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 2
Enter patient name: sumaiya
Enter patient contact: 01905493032
Patient registered successfully: sumaiya
Enter doctor name: Dr. Smith
Enter appointment date (YYYY-MM-DD):
```



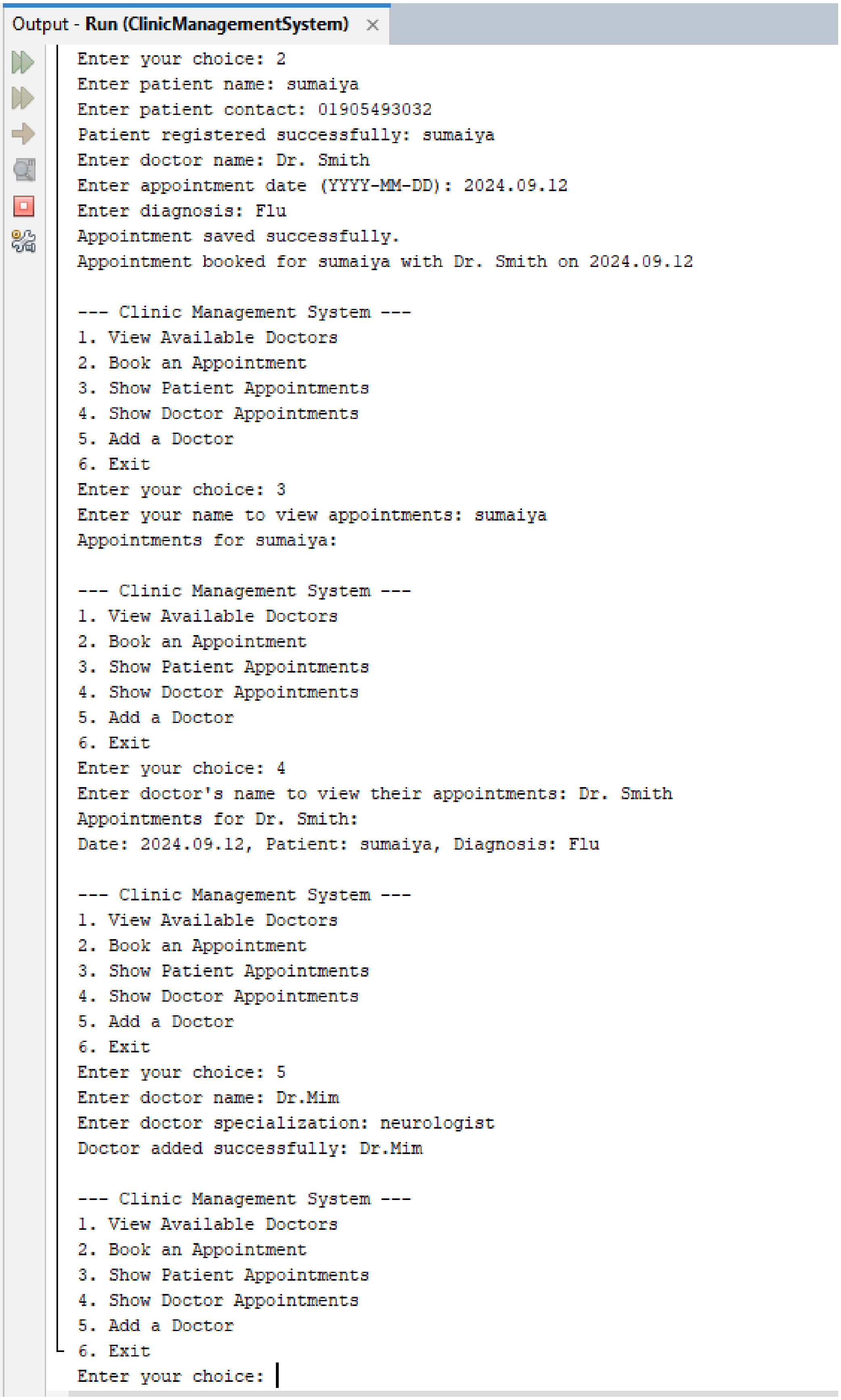
```
----------------< com.mycompany:ClinicManagementSystem >----------------
Building ClinicManagementSystem 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ ClinicManagementSystem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ ClinicManagementSystem ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ ClinicManagementSystem ---

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 1
Available Doctors:
- Dr. Smith (General Practitioner)
Dr. Smith is available for walk-in patients.
- Dr. Johnson (Specialist)
Dr. Johnson requires appointment confirmation.

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 2
Enter patient name: sumaiya
Enter patient contact: 01905493032
Patient registered successfully: sumaiya
Enter doctor name: Dr. Smith
Enter appointment date (YYYY-MM-DD): 2024.09.12
Enter diagnosis: Flue
Appointment saved successfully.
Appointment booked for sumaiya with Dr. Smith on 2024.09.12

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 
```



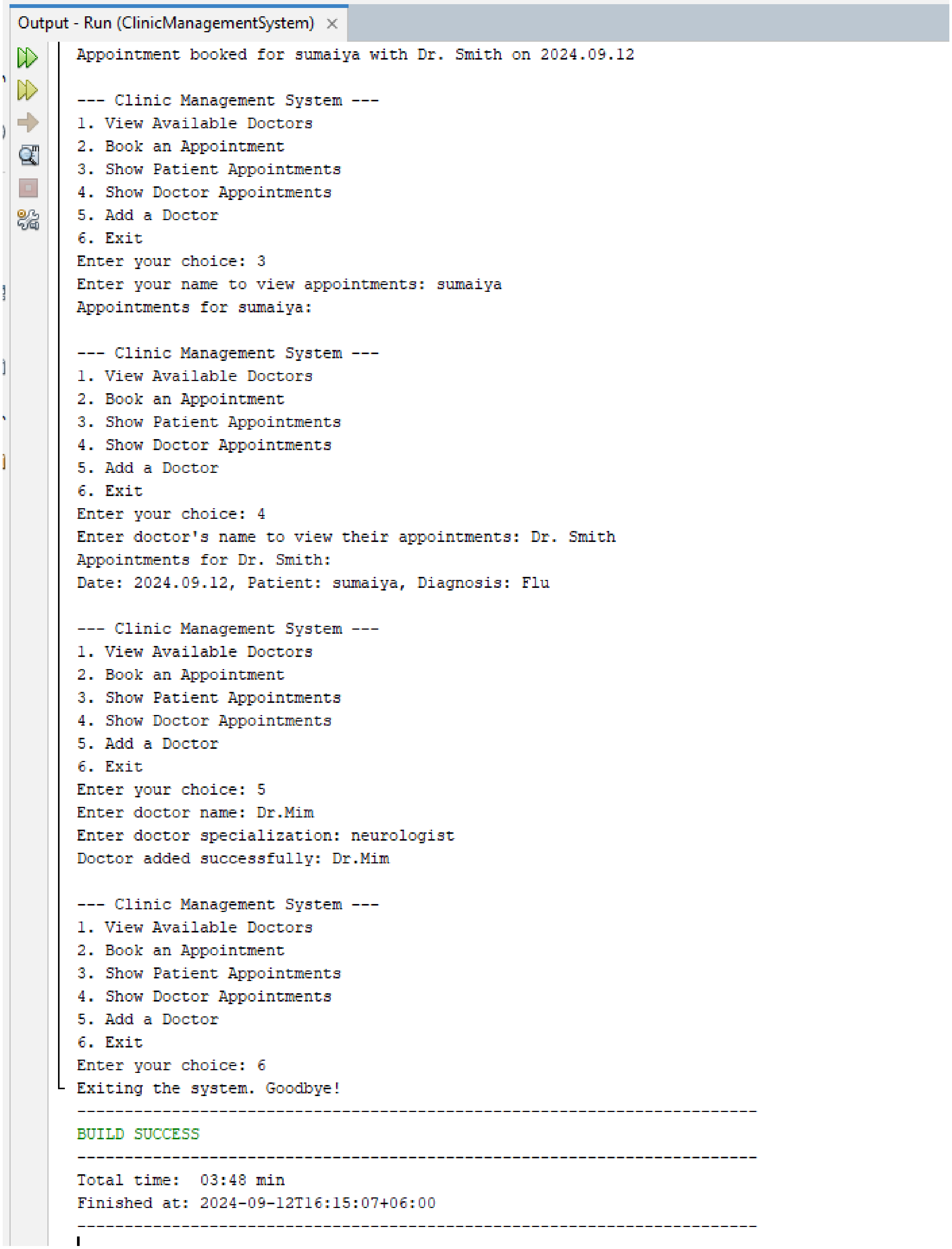
```
----------------< com.mycompany:ClinicManagementSystem >----------------
Building ClinicManagementSystem 1.0-SNAPSHOT
  from pom.xml
--------------------------------[ jar ]---------------------------------

--- resources:3.3.1:resources (default-resources) @ ClinicManagementSystem ---
skip non existing resourceDirectory F:\sumaiya the V.I.P\sem 14\OOP\assignments\final 2\ClinicManagementSystem\src\main\

--- compiler:3.11.0:compile (default-compile) @ ClinicManagementSystem ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ ClinicManagementSystem ---

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 1
Available Doctors:
- Dr. Smith (General Practitioner)
Dr. Smith is available for walk-in patients.
- Dr. Johnson (Specialist)
Dr. Johnson requires appointment confirmation.

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 2
Enter patient name: sumaiya
Enter patient contact: 01905493032
Patient registered successfully: sumaiya
Enter doctor name: Dr. Smith
Enter appointment date (YYYY-MM-DD): 2024.09.12
Enter diagnosis: Flu
Appointment saved successfully.
Appointment booked for sumaiya with Dr. Smith on 2024.09.12

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 3
Enter your name to view appointments:
```



```
- Dr. Smith (General Practitioner)
Dr. Smith is available for walk-in patients.
- Dr. Johnson (Specialist)
Dr. Johnson requires appointment confirmation.

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 2
Enter patient name: sumaiya
Enter patient contact: 01905493032
Patient registered successfully: sumaiya
Enter doctor name: Dr. Smith
Enter appointment date (YYYY-MM-DD): 2024.09.12
Enter diagnosis: Flu
Appointment saved successfully.
Appointment booked for sumaiya with Dr. Smith on 2024.09.12

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 3
Enter your name to view appointments: sumaiya
Appointments for sumaiya:

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 4
Enter doctor's name to view their appointments: Dr. Smith
Appointments for Dr. Smith:
Date: 2024.09.12, Patient: sumaiya, Diagnosis: Flu

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice:
```

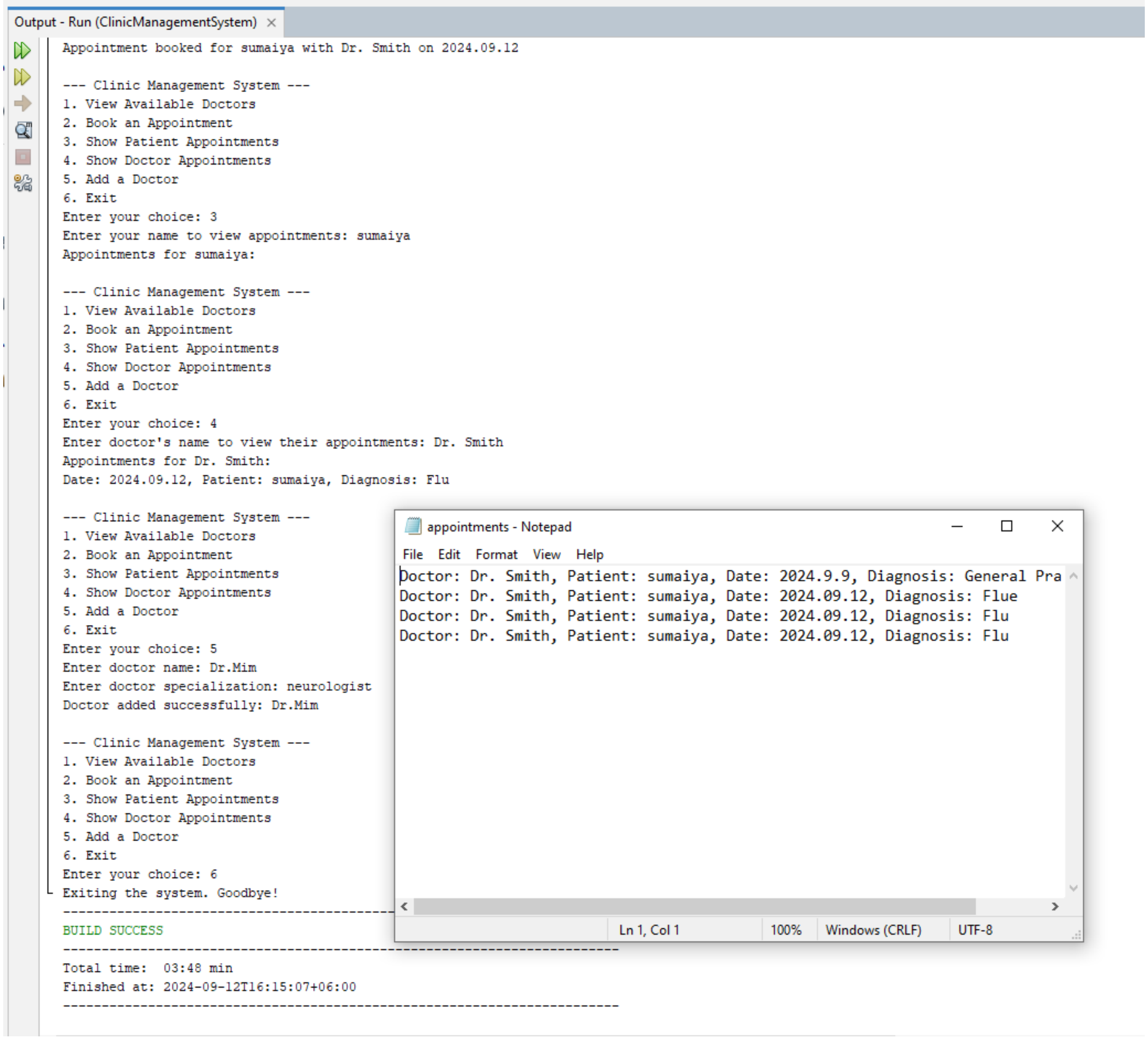


```
Enter your choice: 2
Enter patient name: sumaiya
Enter patient contact: 01905493032
Patient registered successfully: sumaiya
Enter doctor name: Dr. Smith
Enter appointment date (YYYY-MM-DD): 2024.09.12
Enter diagnosis: Flu
Appointment saved successfully.
Appointment booked for sumaiya with Dr. Smith on 2024.09.12

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 3
Enter your name to view appointments: sumaiya
Appointments for sumaiya:

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 4
Enter doctor's name to view their appointments: Dr. Smith
Appointments for Dr. Smith:
Date: 2024.09.12, Patient: sumaiya, Diagnosis: Flu

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 5
Enter doctor name: Dr.Mim
Enter doctor specialization: neurologist
Doctor added successfully: Dr.Mim

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice:
```



```
Appointment booked for sumaiya with Dr. Smith on 2024.09.12

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 3
Enter your name to view appointments: sumaiya
Appointments for sumaiya:

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 4
Enter doctor's name to view their appointments: Dr. Smith
Appointments for Dr. Smith:
Date: 2024.09.12, Patient: sumaiya, Diagnosis: Flu

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 5
Enter doctor name: Dr.Mim
Enter doctor specialization: neurologist
Doctor added successfully: Dr.Mim

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 6
Exiting the system. Goodbye!
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  03:48 min
Finished at: 2024-09-12T16:15:07+06:00
------------------------------------------------------------------------
```

Appointments.txt


Output - Run (ClinicManagementSystem)
appointments - Notepad
File Edit Format View Help
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8

```
Appointment booked for sumaiya with Dr. Smith on 2024.09.12

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 3
Enter your name to view appointments: sumaiya
Appointments for sumaiya:

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 4
Enter doctor's name to view their appointments: Dr. Smith
Appointments for Dr. Smith:
Date: 2024.09.12, Patient: sumaiya, Diagnosis: Flu

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 5
Enter doctor name: Dr.Mim
Enter doctor specialization: neurologist
Doctor added successfully: Dr.Mim

--- Clinic Management System ---
1. View Available Doctors
2. Book an Appointment
3. Show Patient Appointments
4. Show Doctor Appointments
5. Add a Doctor
6. Exit
Enter your choice: 6
Exiting the system. Goodbye!
------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  03:48 min
Finished at: 2024-09-12T16:15:07+06:00
------------------------------------------------------------------------
```

```
Doctor: Dr. Smith, Patient: sumaiya, Date: 2024.9.9, Diagnosis: General Pra
Doctor: Dr. Smith, Patient: sumaiya, Date: 2024.09.12, Diagnosis: Flue
Doctor: Dr. Smith, Patient: sumaiya, Date: 2024.09.12, Diagnosis: Flu
Doctor: Dr. Smith, Patient: sumaiya, Date: 2024.09.12, Diagnosis: Flu
```