SMALL CLINIC MANAGEMENT SYSTEM - REPORT

Name: Huỳnh Vũ Minh Hiền **Student's ID:** 24110091

Subject: Object - Oriented Programming

I. Object-Oriented Analysis (OOA) Model

Objects and Attributes

Human: name, id, gender

Patient: name, id, age, medicalHistory

ChronicPatient (inherits from Patient): conditionType, lastCheckupDate, riskLevel, medications, doctorNotes, treatmentPlan, nextAppointmentDate

Doctor: name, id, specialty, certifications, appointments

Appointment: date, time, reason, status, location, notes, patient, doctorId

Methods

Human: getName, setName, getId, setId, getGender, setGender, displayInfo

Patient: getName, setName, getId, setId, getAge, setAge, getMedicalHistory, setMedicalHistory, displayInfo

ChronicPatient: getConditionType, setConditionType, getLastCheckupDate, setLastCheckupDate, getRiskLevel, setRiskLevel, getMedications, setMedications, getDoctorNotes, setDoctorNotes, getTreatmentPlan, setTreatmentPlan, getNextAppointmentDate, setNextAppointmentDate, displayInfo

Doctor: getName, setName, getId, setId, getSpecialty, setSpecialty, getCertifications, addCertification, getAppointments, addAppointment, displayInfo

Appointment: getDate, setDate, getTime, setTime, getReason, setReason, getStatus, setStatus, getLocation, setLocation, getNotes, setNotes, getPatient, setPatient, getDoctorId, setDoctorId, displayInfo

Human: This is the most general abstraction that represents any individual in the system. It defines shared attributes such as name, ID, and age, along with basic behaviors. Both patients and doctors inherit from this class, ensuring consistency and reusability of common features.

Patient:
A patient represents an individual who receives medical care at the clinic. Each patient has a unique ID, name, and age, as well as a **medical history** (a record of past diagnoses, treatments, and visits). Patients can **schedule appointments**, update their medical history, and have their **number of visits** tracked by the system.

ChronicPatient (Specialized Patient):
A chronic patient is a specialized type of patient who suffers from long-term or ongoing conditions. In addition to the standard patient attributes, chronic patients include: Condition type, Last check-up date, Prescribed medications and treatment plans, Risk level and doctor notes, Scheduled follow-up appointments for continuous monitoring. Unlike regular patients who visit as needed, chronic patients usually require more frequent and structured care, often with regular check-ups (every 3 months). This class overrides some patient methods to enforce these additional requirements.

Doctor:
A doctor is a healthcare professional responsible for diagnosing and treating patients. Each doctor has: A unique ID, name, and specialty, professional background (years of experience, certifications), a list of assigned appointments. Doctors can schedule, update, or cancel appointments, as well as maintain professional notes about patients. They act as the central link between patients and the medical services provided by the clinic.

Appointment: An appointment represents a scheduled meeting between a patient and a doctor. It includes: Date and time, reason for the visit, status (scheduled, completed, or canceled), location, additional notes from the doctor. Appointments ensure that patients receive care in an organized manner and that doctors manage their workload efficiently.

II.

Class Human: General abstraction representing an individual (base for Patient).

Class Patient: Represents a person receiving medical care, with age and medical history.

Class ChronicPatient: Specialized Patient for long-term conditions, adds medications, doctor notes, risk levels, and overrides methods.

Class Doctor: Independent professional, manages appointments and certifications.

Class Appointment: Connects patient and doctor, stores visit details.

System Operations:

The system supports the following functionalities:

- Creating and managing **patients** (both regular and chronic).

- Creating and managing **doctors.**

- Scheduling, updating, and canceling **appointments.**

- Displaying detailed information about **patients, doctors, and appointments.**

Inheritance relationship:

- Patient inherits from Human (adds age, medicalHistory).
- ChronicPatient inherits from Patient (adds conditionType, lastCheckupDate, riskLevel, treatmentPlan, nextAppointmentDate). Shows “is-a” relationship.
- Aggregation: Doctor manages a list of Appointments (vector<Appointment>). Doctor is independent, not inheriting from Human.
- Association: Appointment links a Patient and a Doctor (stores patient info and doctorId).

III. Testing the System

Test Case 1

- Create a **Patient** and add medical history.

- Create a **ChronicPatient**, add history, medications, treatment plan, and doctor notes.

- Create a **Doctor**, set phone, email, certifications, and years of experience.

- Create **two appointments** (for Patient and ChronicPatient).

- Add appointments to Doctor.

- Display all patient info, doctor info, and appointments.

- Output: Shows details of patients, doctor certifications, and appointment schedules.

Test Case 2

- Update status of one appointment to complete.

- Attempt to update another appointment with invalid status waiting.

- Display updated results.

- Output: Status updated correctly for valid input, error message shown for invalid status.

IV. Documentation & LLM Usage

OOA, design, code, and tests: Already included above.

LLM Usage: LLM is a support tool. It has helped me: Suggest property and method names and ensure class design follows encapsulation principles. It also review and improve the report structure to be more organized.

Organization: This report is structured into OOA model, class overview, and testing section for clarity.