Health Monitoring System (HMS)

Project Overview

The Health Monitoring System (HMS) is a Java-based application designed to manage patient data, including vitals, medical history, appointments, and doctor information. It allows users to register new patients, schedule appointments, update patient vitals, manage doctor appointments, and retrieve sorted patient data. The project uses object-oriented programming (OOP) principles and file I/O for data storage and manipulation.

Features

Patient Management:

- Register new patients with personal and medical details.
- Update patient vitals.
- Retrieve sorted patient data by age**.**

Appointment Management:

- Schedule appointments between patients and doctors.
- View and manage appointments for each doctor.

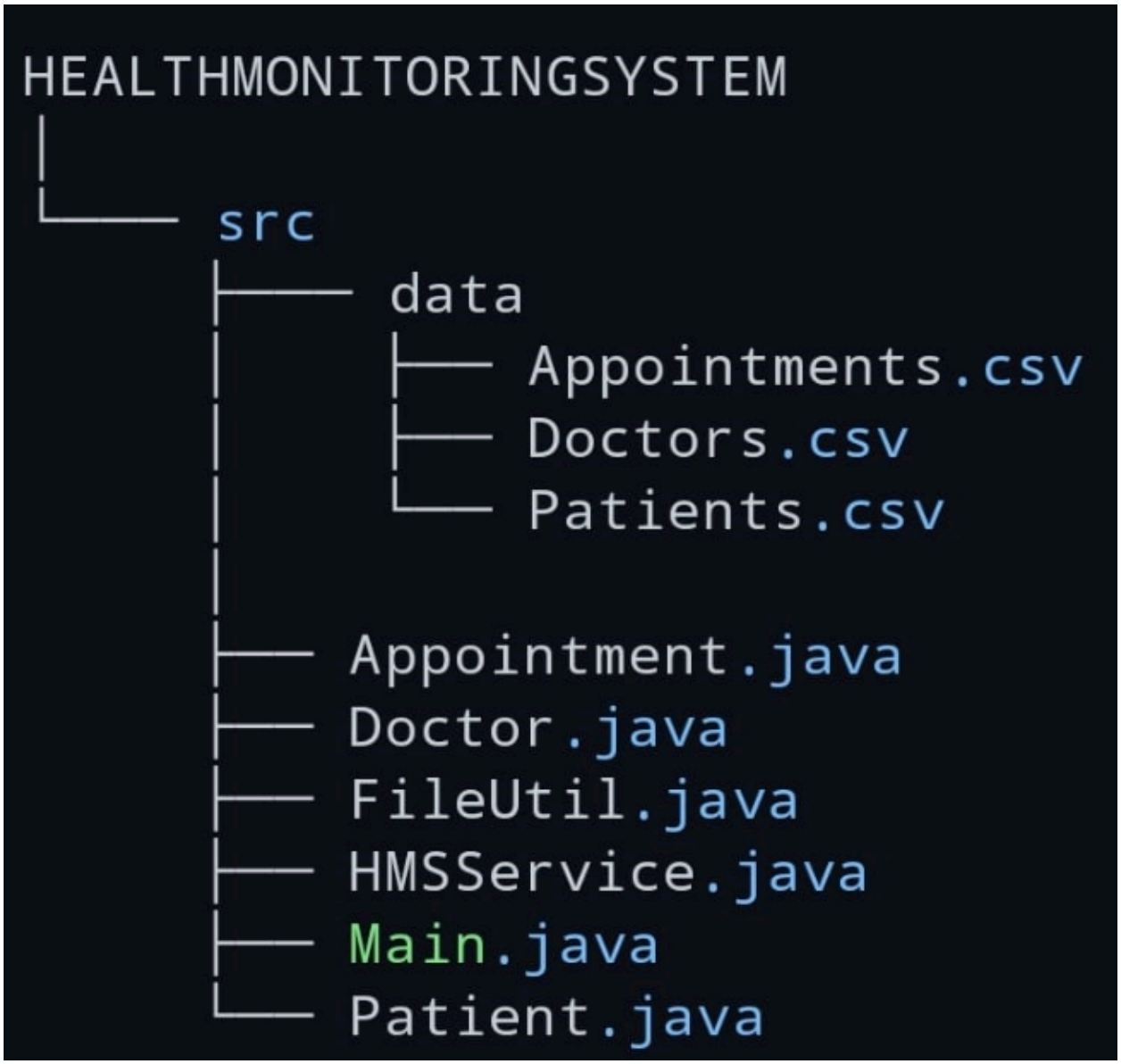
Doctor Management:

- Maintain a list of doctors and their specializations.
- Retrieve all appointments assigned to a specific doctor.

Medical History:

- Retrieve and update patient medical history.

Project Structure



Key Classes

`Patient.java`: Manages patient details such as name, age, gender, medical history, and vitals.
`Doctor.java`: Manages doctor details including name and specialization.
`Appointment.java`: Manages appointments between patients and doctors.

`FileUtil.java`: Handles reading and writing data to CSV files for patients, doctors, and appointments.
`HMSService.java`: The main service class that provides an interactive console interface for managing the system's operations.

Installation and Setup

1.Clone the repository:
git clone https://github.com/ujjwal10122000/HealthMonitoringSystem.git

2. Navigate to the project directory:

cd HealthMonitoringSystem

3.Compile the project:

javac src/*.java

4.Run the project:

java src.HMSService

Usage

Once the program is running, you'll see a menu offering various options. Here's how to interact with the system:

1. **Register New Patient**: Enter details like ID, name, age, gender, medical history, and vitals.
2. **Schedule Appointment**: Provide appointment details such as appointment ID, patient ID, doctor ID, and appointment date.
3. **Update Patient Vitals**: Update the patient's vitals by entering their ID and new vitals.
4. **Retrieve Sorted Patient Data**: View all patients sorted by age.
5. **Manage Doctor's Appointments**: View all appointments associated with a specific doctor.
6. **Retrieve Medical History**: View a patient’s medical history based on their ID.

Data Storage

Patient, doctor, and appointment data are stored in CSV files located in the `data` folder:

- `Patients.csv`: Stores patient information.
- `Doctors.csv`: Stores doctor information.
- `Appointments.csv`: Stores appointment details.

Contact

For any inquiries, feel free to contact the project maintainer:

- **Name**: Ujjwal dhake
- **Contact no. :** 7389795087
- **Email**: ujjwadhake777l@gmail.com