AI AND DATA SCIENCE

Hospital Queue Management System

Date: 3-12-2025

Instructor: Ahmed Diab

Project 2

- Hospital Queue Management System

HQMS

- This project implements a complete hospital patient queue management system using Object-Oriented Programming (OOP) principles and a modular package structure.
- The system simulates how a hospital manages patients across multiple medical specializations, handling urgent cases and maintaining queues with priority rules.

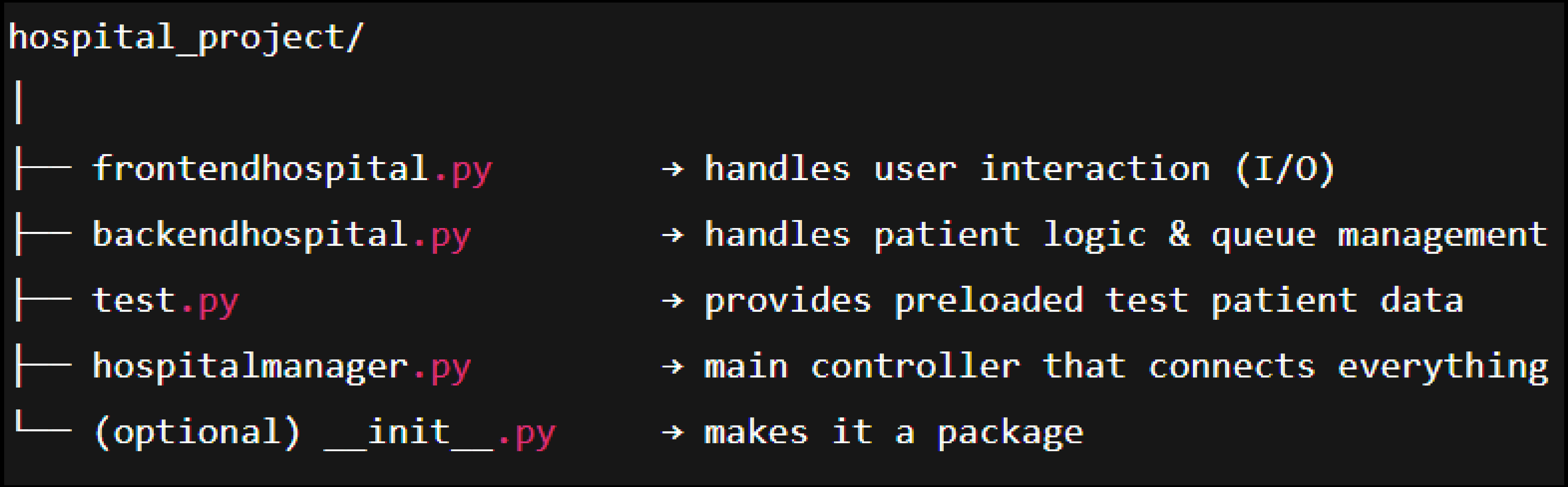
HQMS

The application is divided into Front-End, Back-End, Testing Module, and Main Manager, each in separate Python files to demonstrate clean architecture, code organization, and real software engineering structure.

Core Features

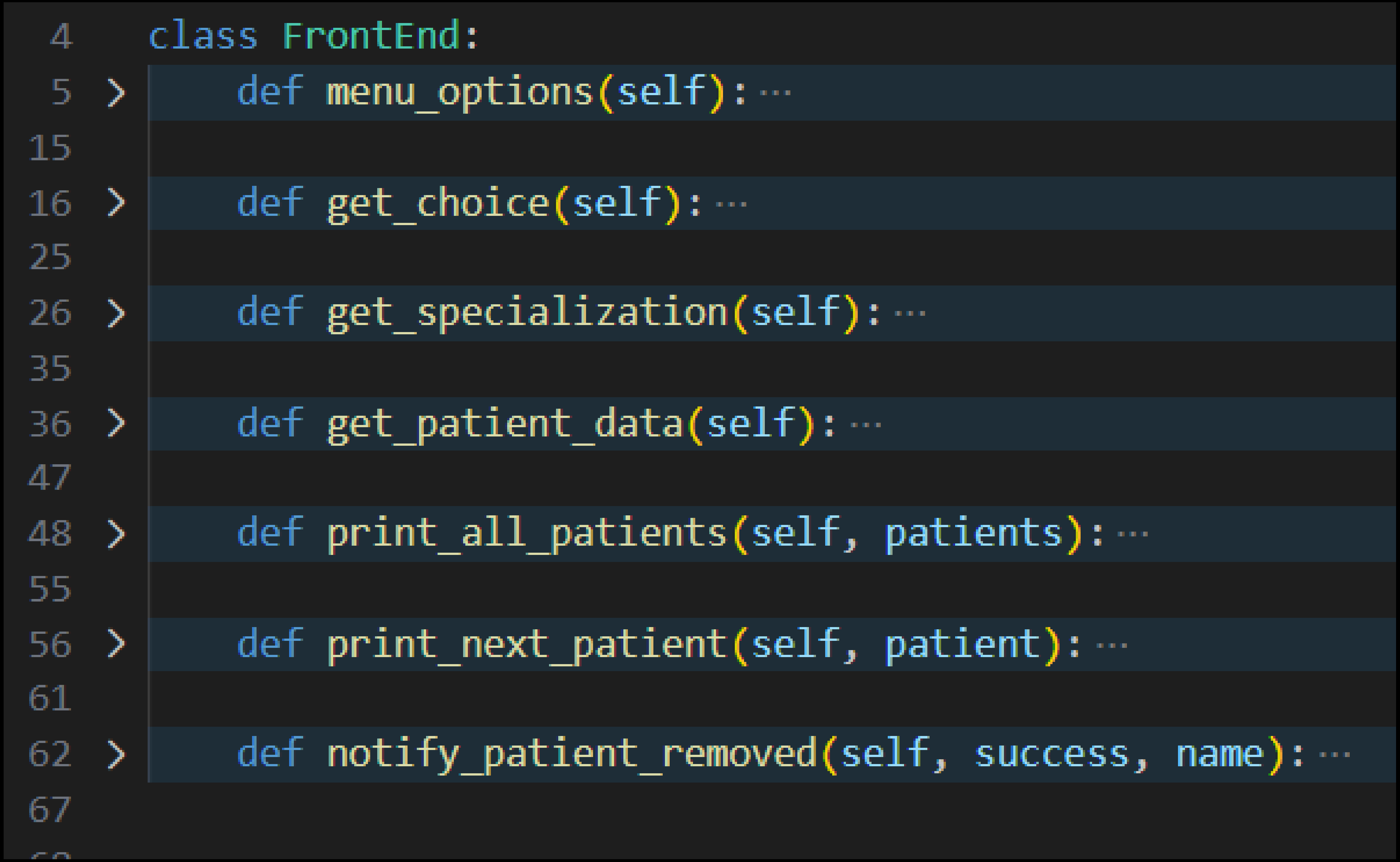
- Add a new patient (with priority: Normal / Urgent / Super Urgent)
- Print all patients organized by specialization
- Get the next patient based on urgency rules
- Remove a patient leaving the queue
- Separation of logic and user interaction
- Pre-loaded test data to simulate a running hospital

Project Architecture



```
hospital_project/
│
├── frontendhospital.py       → handles user interaction (I/O)
├── backendhospital.py        → handles patient logic & queue management
├── test.py                   → provides preloaded test patient data
├── hospitalmanager.py        → main controller that connects everything
└── (optional) __init__.py    → makes it a package
```

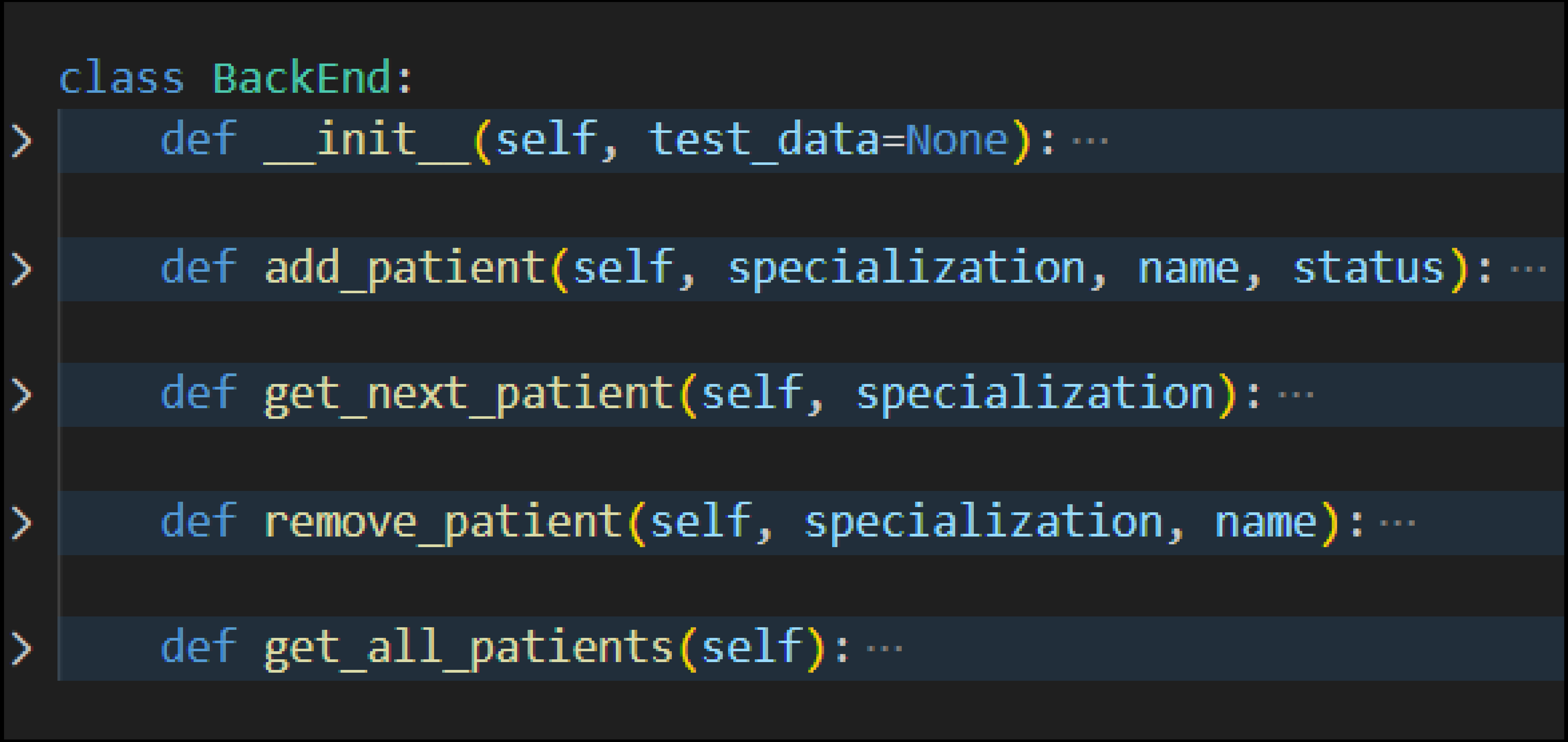
- FrontEndHospital.py



```
class FrontEnd:
    def menu_options(self):⋯

    def get_choice(self):⋯

    def get_specialization(self):⋯

    def get_patient_data(self):⋯

    def print_all_patients(self, patients):⋯

    def print_next_patient(self, patient):⋯

    def notify_patient_removed(self, success, name):⋯

```

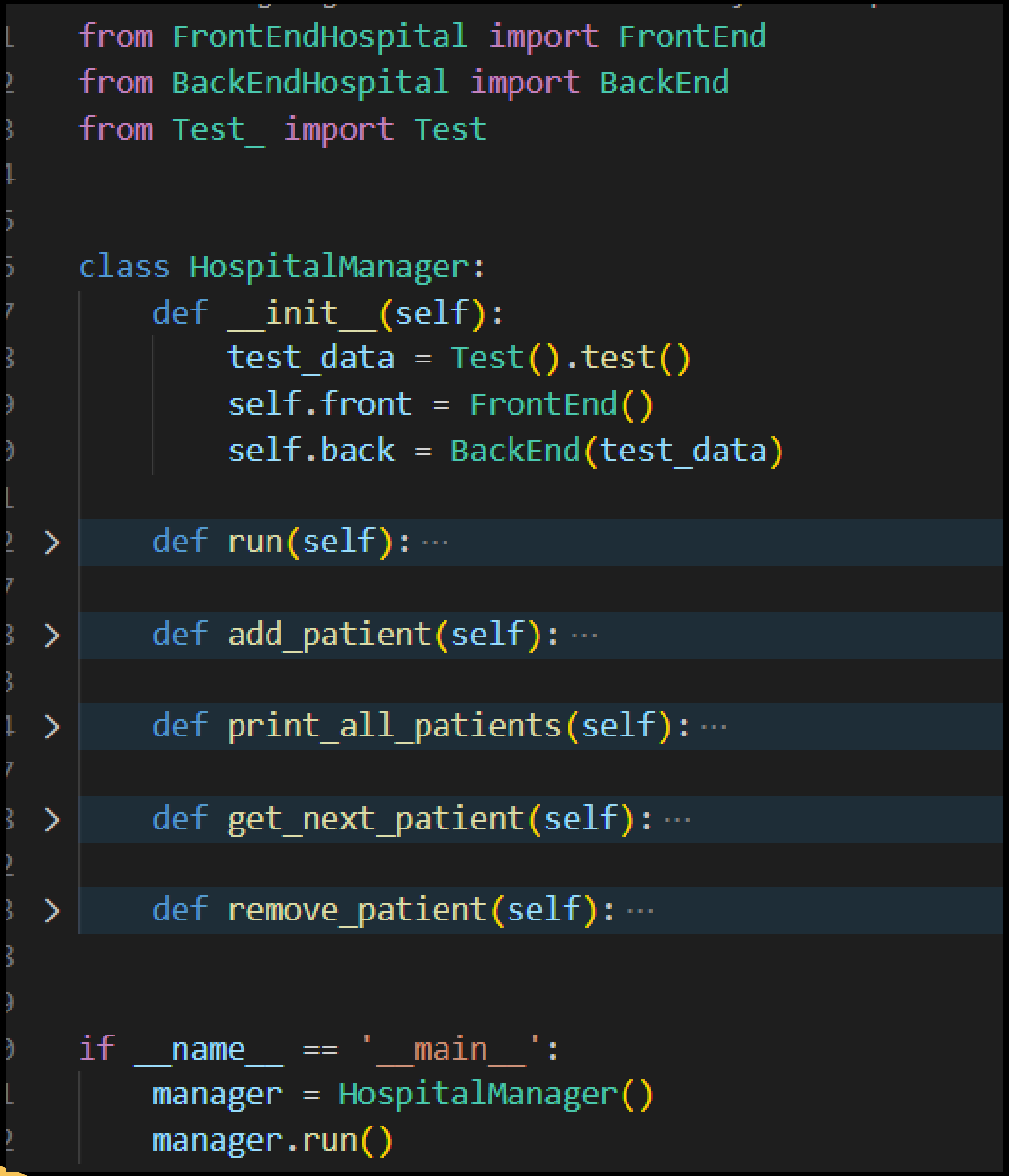
HQMS

- BackEndHospital.py



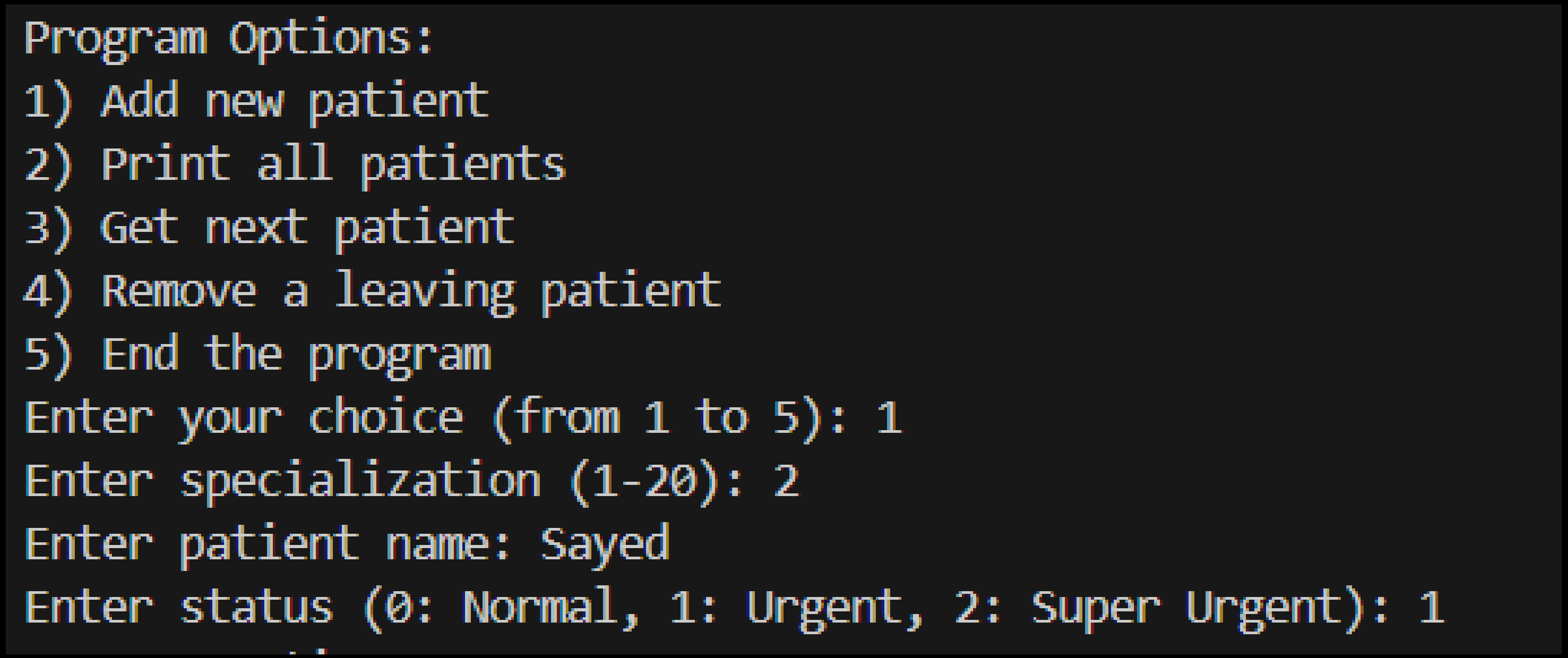
```python
class BackEnd:
    def __init__(self, test_data=None):···

    def add_patient(self, specialization, name, status):···

    def get_next_patient(self, specialization):···

    def remove_patient(self, specialization, name):···

    def get_all_patients(self):···
```

- HospitalManager.py



```python
from FrontEndHospital import FrontEnd
from BackEndHospital import BackEnd
from Test_ import Test


class HospitalManager:
    def __init__(self):
        test_data = Test().test()
        self.front = FrontEnd()
        self.back = BackEnd(test_data)

    def run(self): ...

    def add_patient(self): ...

    def print_all_patients(self): ...

    def get_next_patient(self): ...

    def remove_patient(self): ...


if __name__ == '__main__':
    manager = HospitalManager()
    manager.run()
```

- output example



```
Program Options:
1) Add new patient
2) Print all patients
3) Get next patient
4) Remove a leaving patient
5) End the program
Enter your choice (from 1 to 5): 1
Enter specialization (1-20): 2
Enter patient name: Sayed
Enter status (0: Normal, 1: Urgent, 2: Super Urgent): 1
```

دمتم
سالمين