Assignment: Patient Health Records Management and Consultation Booking Application

Objective

Develop a secure web-based application where patients can upload and manage their health records, while hospitals and doctors can access these records, book appointments/consultations, and view patient medical histories. The application must ensure data privacy, role-based access, and seamless interaction between patients and healthcare providers.

Requirements

Patient Health Records Upload:

Allow patients to upload health records (e.g., PDFs, images, text) with metadata (patient ID, date, record type).

Validate file formats and extract basic metadata automatically.

Patient-Doctor Matching & Appointment System:

Enable doctors/hospitals to search/filter patients based on medical conditions, demographics, or records.

Implement an appointment booking system where doctors can schedule consultations and patients receive notifications.

Health Record Analysis & Reporting:

Allow doctors to view trends in patient data (e.g., lab results over time) and generate summary reports.

Add alerts for critical health values (e.g., abnormal blood pressure).

Access Control & Security:

Implement role-based access (patient, doctor, admin). Patients must approve access to their records.

Encrypt sensitive data and ensure compliance with healthcare privacy standards (e.g., HIPAA/GDPR).

User Interface:

Build a web interface using Streamlit or Gradio with separate dashboards:

Patients: Upload records, view appointments, manage access.

Doctors/Hospitals: Search patients, view records, book appointments, analyze data.

API Development:

Design APIs to handle communication between frontend and backend (e.g., record upload, appointment scheduling).

Deployment:

Deploy the application on Hugging Face Spaces or a similar platform for testing.

Documentation:

Submit a detailed README covering setup, dependencies, security measures, and usage instructions.

Submission Guidelines

GitHub Repository: Include:

app.py (main script)

requirements.txt (dependencies)

README.md (setup/usage guide)

utils.py (helper functions for encryption, data processing)

api.py (API endpoints)

Deployment: Share the Hugging Face Spaces deployment link.

Code Quality: Follow PEP8 guidelines, add comments, and ensure modularity.

Input Format

Patients: Upload health records via the interface.

Doctors/Hospitals: Search using patient IDs, medical conditions, or dates.

Expected Output Format

A structured system including:

json


```
{
  "Patient Profile": {
    "ID": "P123",
    "Records": [
      {
        "Type": "Lab Report",
        "Date": "2023-10-01",
        "Summary": "HDL: 60 mg/dL, LDL: 100 mg/dL",
        "Access Granted To": ["Dr. Smith"]
      }
    ],
```

```
  "Appointments": [
    {
      "Doctor": "Dr. Smith",
      "Date": "2023-11-15",
      "Status": "Confirmed"
    }
  ]
 },
 "Access Logs": ["Dr. Smith viewed record on 2023-10-05"],
 "Health Trends": {"Blood Pressure": "120/80 mmHg (Avg)"}
}
```


Documentation Requirements

Project Setup: Steps to install dependencies and run the app.

Security Measures: Explain encryption, access control, and compliance.

API Usage: Describe endpoints and how to integrate them (e.g., using Postman).

Assumptions & Limitations: Clarify constraints (e.g., no real-time patient-doctor chat).

Evaluation Criteria

Correctness: Accurate data handling, secure access control.

Efficiency: Fast search/upload and minimal latency.

Robustness: Handle invalid files, unauthorized access attempts, and edge cases.

Deployment: Functional deployment on Hugging Face Spaces.

Code Quality: Readability, documentation, and adherence to best practices.

Deadline: 1 week from the start date.

Submission: GitHub repo link + Deployment link + Video Demo (5 mins max).