Maximum allocated time: 3 working days. Please confirm your start time to HC.

Case 1: Python - Basic Data Processing
Objective: Test data wrangling, logic, and language fluency.

Task:
You are need to create a CSV file `patients.csv` with the following structure:

```
id,name,age,symptoms
1,Andi,45,"demam, batuk, sesak napas"
2,Budi,29,"mual, sakit perut"
3,Citra,62,"pusing, kehilangan keseimbangan"
4,Dita,30,”susah tidur”
5,Eka,18,”gusi berdarah”
6,Fitra,49,”pusing, sakit perut”
7,Gio,49,”menggigil, batuk, sakit kepala”
8,Harianto,33,”memar di tangan”
9,Idul,88,”susah tidur”
10,Jaka,54,”sesak nafas”
```

Write a Python script that:

- Loads the CSV
- Tokenizes the symptoms into lists
- Filters patients older than 40 with more than 2 symptoms

Deliverable: A Python file `filter_patients.ipynb`

Case 2: Querying – SQL for Patient Visit Insights
Objective: Test data logic.

Schema Description:

Table: patients

Column Name	Type	Description
id	INT (PK)	Unique ID of the patient
name	TEXT	Full Name of the patient
age	INT	Age in years

Table: visits

Column Name	Type	Description
id	INT (PK)	Unique ID of the visit
patient_id	INT (FK)	References patients(id)
department	TEXT	Department visited (e.g, Neurology
visit_date	DATE	Date of the visit

Table: symptoms

Column Name	Type	Description
id	INT (PK)	Unique ID of the visit
visit_id	INT (FK)	References visits(id)
symptom	TEXT	Symptom description (e.g, “mual”)

Task:

- Find the top 5 most recent visits to the Neurology department
- Where the patient had at least 3 recorded symptoms
- And the patient is older than 50 years

Return the following columns:

- `patients.name`
- `patients.age`
- `visits.visit_date`
- `symptom_count`

Case 3: End-to-End Mini Project
Objective: Test integration and real-world thinking.
Task:

Background / Purpose:

In a hospital triage system, patients often describe their symptoms at reception. To assist non-medical staff and reduce wait times, we want to **automatically suggest the most relevant specialist department** based on patient input. This system will:

- Speed up the triage process
- Reduce human error in initial referrals
- Improve patient experience and routing efficiency

Create a minimal FastAPI service with:

- Endpoint: `POST /recommend`
- Accepts: JSON body with patient info (gender, age, symptoms)

```
{
  "gender": "female",
  "age": 62,
  "symptoms": ["pusing", "mual", "sulit berjalan"]
}
```

- Uses:
  - LLM (e.g: https://aistudio.google.com) to generate recommendation base on patient info
  - Langchain or llamaindex
- Returns: JSON with recommended department

```
{
  "recommended_department": "Neurology"
}
```

Deliverables:

- Python script or repo with the FastAPI app
- Brief README on how to run the app