Assignment 3: Meeting Transcript Summarizer & Action-Item Extractor

1. Background & Context

Teams often record meetings and distribute raw transcripts, but reading through hundreds of lines to find key takeaways and next steps is time-consuming. Automating this process with a Generative AI model can save hours and ensure no action item is missed.

2. Problem Statement

"Given a plain-text meeting transcript, automatically produce (a) a concise two-sentence summary of the meeting's main discussion points, and (b) a list of all action items mentioned, each prefaced with a dash ('–'), outputting the result as a JSON object."

3. Objectives & Learning Outcomes

• Prompt Design: Craft clear instructions to elicit structured responses (summary + action list) from an LLM.

• Structured Output: Enforce a JSON schema so downstream tools can reliably ingest the result.

• Error Handling: Detect and recover from malformed LLM outputs (e.g. missing JSON).

• Minimal UI/CLI: (Optional) Wrap the extractor in a simple script or notebook to demo end-to-end.

4. Technical Requirements

1. Input:
   - A single string containing the full meeting transcript (100–600 words).
   - Speaker labels (e.g. “Alice: …”, “Bob: …”) may be present but are optional.

2. Output:
   - A JSON object with exactly two keys:

```
{
  "summary": "…",
  "action_items": [
    "- …",
    "- …"
  ]
}
```

3. Model:
   - Use GPT-4o-mini (or GPT-4.1-mini) with temperature=0 for consistency.
   - Wrap your prompt in a function, e.g.
     def extract_meeting_notes(transcript: str) -> dict:
       # call OpenAI API with prompt template…
       return parsed_json

4. Validation:
   - Define a Pydantic model to enforce:
     class MeetingNotes(BaseModel):
       summary: str
       action_items: List[str]
   - On parser errors, retry once with a “Please output valid JSON only” system prompt.

5. Prompt Template (Example)

```
You are a meeting assistant.

1. Summarize the meeting transcript below in exactly two sentences.

2. Then list all action items mentioned, each as a separate bullet
beginning with a dash.

Return the result strictly as JSON with keys "summary" and
"action_items".


Transcript:

{transcript}
```

6. Deliverables

1. Code: Jupyter Notebook OR zip of .py files.

7. Sample Inputs for Testing

Sample Input 1

```
Alice: Welcome everyone. Today we need to finalize the Q3
roadmap.

Bob: I've emailed the updated feature list—please review by
Friday.

Carol: I'll set up the user-testing sessions next week.

Dan: Let's push the new UI mockups to staging on Wednesday.

Alice: Great. Also, can someone compile the stakeholder
feedback into a slide deck?

Bob: I can handle the slide deck by Monday.

Alice: Thanks, team. Meeting adjourned.
```

Sample Input 2

```
Host: Let's kick off our marketing sync.

Emma: The social campaign draft is 80% done; I'll share it
today.

Frank: I spoke with the design team—they'll deliver assets by
Tuesday.

Emma: Once we have assets, I'll schedule the ads for next week.

George: Reminder: submit your budget requests before end of
day.

Host: Noted. I'll send out the final budget spreadsheet.
```

Sample Input 3

```
John: Sales numbers for April are in—up 12% over target.

Lia: I'll distribute the detailed report to leadership this afternoon.

Mike: We need to train two new reps to handle increased volume.

Lia: I'll coordinate recruiting with HR by end of week.

John: Excellent. Let's aim to onboard them by May 15th.
```

Sample Input 4

Sample Input 5

```
HR: We need to update the travel-expense policy.

Alex: I'll draft the new policy doc and share with legal.

HR: Please include guidelines on per diem limits.

Legal: I'll review and provide feedback within three days.

HR: Once finalized, schedule a companywide announcement.
```