Design Document & Approach Strategy

Project: AI-Powered Excel Mock Interviewer
Prepared by: Vanshika Mishra

1. Executive Summary

The organization is currently experiencing rapid expansion across Finance, Operations, and Data Analytics divisions. Proficiency in Microsoft Excel has been identified as a critical skill for new hires, as it directly impacts productivity and analytical decision-making. Traditional manual interviews for Excel are time-consuming, inconsistent, and resource-intensive, causing bottlenecks in the hiring pipeline. These limitations create delays in candidate evaluation and lead to variability in assessment quality.

The objective of this project is to design and implement an automated, interactive Excel interview system. The system evaluates candidate responses in real-time, provides constructive inline feedback, and persistently stores evaluation results for audit and future analytics. By standardizing the interview process, the solution ensures fairness, efficiency, and a consistent evaluation standard for all candidates.

2. Project Goals

1. **Structured Interview Flow:** Provide a coherent, multi-turn interview experience that mimics a real human interview, complete with instructions, one-question-at-a-time presentation, and motivational prompts.

2. **Intelligent Answer Evaluation:** Each candidate response is evaluated in real-time, generating a numeric score and constructive remarks for every answer.

3. **Interactive Feedback:** Inline motivational and constructive feedback is displayed dynamically during the interview to enhance candidate engagement and maintain focus.

4. **Data Management:** Candidate answers, scores, and remarks are stored in a structured JSON database to allow audit, analytics, and potential future integration with enterprise systems.

5. **Scalability:** The system supports multiple concurrent candidates through session management and asynchronous backend evaluation.

---

3. System Architecture

3.1 Overview

The system consists of three main components, fully integrated to provide a smooth, real-time interactive interview experience.

Component	Purpose	Technology
Frontend (UI)	Collects candidate information, presents questions, displays timer and feedback	Streamlit
Backend (API Layer)	Handles question retrieval, per-answer evaluation, and summary aggregation	FastAPI
Answer Evaluation	Evaluates candidate responses with structured scoring and remarks	Ollama LLM (Phi3:mini)
Async Handling	Ensures non-blocking evaluation and manages timeout for each response	Python asyncio
Data Layer	Stores candidate responses, scores, and evaluation remarks persistently	JSON files
HTTP Requests	Facilitates communication between frontend and backend	Python requests
Utility Functions	Random selection of questions, motivational feedback, timing operations	Python random, time

Architecture Diagram:

```
+-------------------+      +----------------+
+-------------------+
|   Streamlit UI    | <--> |   FastAPI API  | <--> |   JSON
Database  |
+-------------------+      +----------------+
+-------------------+
```

3.2 Frontend Design (Streamlit)

The frontend is designed to provide a **responsive and interactive user experience** while maintaining simplicity. Key features include:

- **Welcome Page:** Captures candidate name, institute, and consent for rules acknowledgment.
- **Instructions Panel:** Displays interview guidelines in a clear, interactive manner.
- **Question Flow:**
  - One question is displayed per page.
  - Supports both MCQ and text-based answers.
  - Motivational messages appear intermittently to keep candidates engaged.
- **Timer Management:** A 30-minute countdown begins only when the candidate clicks the "Start Interview" button.
- **Results Page:** Displays final scores, per-question remarks, and provides a feedback text area for candidates.

Rationale for Streamlit: Streamlit enables rapid prototyping, seamless integration with Python backend code, and minimal deployment overhead. It provides a highly interactive, visually clean interface compatible with both cloud hosting and local deployment.

3.3 Backend Design (FastAPI)

The backend is responsible for all **data operations and answer evaluation**. It exposes asynchronous endpoints to ensure **robust session handling** for multiple candidates simultaneously.

Endpoints:

1. **/questions** – Retrieves 25 random questions from the `questions.json` dataset. Questions are preprocessed to remove duplicates and irrelevant content, and are uniquely identified for evaluation.
2. **/evaluate** – Accepts a candidate's response for a given question, sends a structured prompt to **Ollama LLM (Phi3:mini)**, and returns a JSON containing a numeric score

(0–4) and short constructive remarks.

3. **/summary** – Aggregates candidate answers for display in the frontend, enabling a final review of all scores and remarks.

Answer Evaluation Workflow:

- When a candidate submits an answer, the frontend sends it to the `/evaluate` endpoint.
- The backend constructs a structured prompt including the **question text**, **candidate response**, and **evaluation instructions**.
- The LLM evaluates the response and returns a JSON containing:
    - **score:** 0–4 (capped for consistency)
    - **remarks:** Short, constructive comments on the answer
- The backend stores each response, score, and remark in the JSON database.
- Inline, human-like feedback messages are randomly displayed on the frontend to simulate strengths and weaknesses in a natural conversational tone.

Technologies and Justifications:

- **FastAPI:** Lightweight, asynchronous, and highly performant for handling multiple concurrent evaluation requests.
- **Python asyncio:** Ensures LLM evaluation does not block the system; timeouts prevent stalled evaluations.
- **Ollama LLM (Phi3:mini):** Provides structured, consistent scoring while allowing rich, human-readable remarks.
- **JSON Database:** Simple, auditable, and easily extendable storage solution for candidate responses.

3.4 Database & Data Storage

Candidate responses, scores, and remarks are stored in **structured JSON files**, containing:

- Candidate name and institute
- Question ID
- Candidate response
- Score
- Remarks
- Timestamp

This approach ensures **easy retrieval, analytics, auditing**, and potential migration to relational databases like PostgreSQL or MySQL in the future.

---

4. Question Bank Strategy

Source: Kaggle Excel datasets and top-ranked Excel interview questions.

Preprocessing Steps:

- Remove duplicates and non-technical content
- Standardize formatting (lowercase conversion, remove special characters)
- Assign unique IDs to each question
- Include metadata: question type (MCQ/text), difficulty, and expected answer

Storage: All questions are stored in `questions.json`. Random sampling ensures each candidate receives a unique set of 25 questions, maintaining fairness and coverage across topics.

---

5. Evaluation Workflow

- Candidate submits an answer through Streamlit interface.

- Frontend sends request to `/evaluate` endpoint.
- Backend constructs a structured prompt for LLM evaluation:
  - Includes question text, candidate response, and evaluation instructions.
  - Instructs the model to respond only in JSON format.
- **Structured Evaluation Methodology:** For experience- or scenario-based questions, the LLM evaluates answers using the **STAR technique** (Situation, Task, Action, Result) to ensure comprehensive assessment of the candidate's approach and reasoning. For both MCQ and descriptive answers, the system first analyzes the candidate's response and then assigns a score out of 4, providing constructive remarks for feedback.
- LLM evaluates the answer:
  - Assigns a score (0–4)
  - Provides short, constructive remarks
- Backend stores the result in the database and sends it back to frontend.
- Upon interview completion, `/summary` aggregates all answers and generates a performance summary with inline strengths, weaknesses, and improvement suggestions delivered naturally during feedback flow.

---

6. Session Management & State Handling

- Streamlit Session State is used to maintain:
  - Current question index
  - Candidate responses
  - Timer and elapsed time
  - Motivational feedback messages

- Skipped questions are handled gracefully; unattempted questions trigger a gentle warning without halting progress.

- The system maintains state even if the candidate interacts slowly or refreshes the page.

---

7. Deployment Strategy

- **Frontend & Backend:** Streamlit deployed on Streamlit Cloud
- **Database:** JSON files for now, with a clear path to migrate to relational databases

- **Scalability:** Async endpoints allow multiple concurrent candidates; timeout handling ensures evaluations do not block the system

- **Security:** HTTPS communication, input sanitization, and session isolation for each candidate

---

8. Advantages of the Approach

- **Efficiency:** Automates evaluation, significantly reducing senior analyst time

- **Consistency:** Standardized scoring eliminates human bias

- **Interactive Experience:** Motivational prompts and inline feedback mimic human interview interactions

- **Data-Driven Insights:** Evaluation results stored for analytics and potential improvements

- **Scalable & Flexible:** Technology stack allows expansion to multiple candidates and additional question sets

---

9. Future Enhancements

- Dynamic question generation using NLP techniques for greater variability

- Integration with candidate dashboards for long-term progress tracking
- Advanced analytics on candidate performance over multiple sessions
- Migration to a relational database for enterprise-level scalability
- Addition of more complex evaluation rules and inline feedback customization

---

10. Conclusion

The project delivers a **comprehensive, automated Excel interview system** featuring:

- Interactive, structured interview flow
- Real-time evaluation using structured LLM prompts
- Persistent storage of candidate responses, scores, and remarks
- Inline, human-like motivational feedback

The **chosen technology stack** (Streamlit, FastAPI, Ollama LLM, asyncio, JSON) ensures efficiency, scalability, and maintainability. The system provides a reliable, auditable, and interactive solution for assessing Excel skills consistently across multiple candidates, supporting the organization's growing hiring needs.