Technical Test – Part 2: Full-Stack Interface, Authentication & Cloud Deployment

Objective

Extend **Technical Test – Part 1** by building a secure, cloud-hosted web interface that allows users to interact with all existing backend features.

This part is designed to evaluate your ability to design and implement a **production-oriented full-stack system**, including frontend, authentication, background processing, persistence, and deployment.

Functional Requirements

1. Authentication & Access Control

- Implement a **sign-in system** to secure access to the platform.
- Authentication must be handled using:
  - **Firebase Authentication**, or
  - **GCP Identity Platform / custom auth on GCP**.
- Only authenticated users may access workflows, inputs, and outputs.
- You may assume a single user role (authenticated user). No RBAC is required.

---

2. Workflow Selection

- As a user, I want to:
  - View a list of available workflows (as defined in Technical Test – Part 1).
  - Select a workflow to execute.
- Each workflow should expose:
  - Name or identifier
  - Description (if available)
  - Required inputs

---

3. Dynamic Input Handling

- Once a workflow is selected, the interface must:
    - Display the required inputs for that workflow.
    - Allow the user to upload **one or many inputs**.
- In this test, inputs represent **progress notes** (file or text format, consistent with Part 1).
- The UI should clearly indicate:
    - Which inputs have been uploaded
    - Their current status (e.g. pending, submitted)

---

4. Background Execution

- Upon submission:
    - Workflow execution must run **asynchronously** (background processing).
    - The UI must not block while processing is taking place.
- Acceptable execution mechanisms include:
    - Cloud Functions
    - Cloud Run
    - Firebase background jobs
    - Any equivalent GCP-native solution
- Each uploaded input must be processed independently under the selected workflow.

---

5. Output Collection & Persistence

- For each submitted input, the system must persist:
    - Input reference
    - Workflow used
    - Processing timestamp
    - Final output payload
- Outputs must be stored in a durable, queryable way.
- Acceptable storage options include:
    - Firestore
    - Cloud Storage with metadata persistence
    - Any reasonable GCP-hosted storage solution

---

6. Results Visualization

- The interface must include an **Outputs section** where the user can:
    - View historical workflow executions

    - Filter or group results by:
        - Date of processing
        - Workflow
- For each output entry, the user should be able to:
    - See the original input reference
    - See the corresponding output produced by the workflow

Non-Functional Requirements

Hosting & Infrastructure

- The full platform must be hosted on:
    - **Firebase**, or
    - **Google Cloud Platform (GCP)** directly.
- Both frontend and backend must be deployed and accessible via a public URL.
- Local-only or non-deployed solutions are not sufficient.

Architecture Expectations

- Clear separation between:
    - Frontend
    - Backend / execution layer
    - Storage
- Clean API boundaries between frontend and backend.
- Background execution must be explicit and intentional (not simulated with UI delays).

Code Quality

- Code should be well-structured and readable.
- Key design decisions should be explained via:
    - Code comments, and/or
    - A short README
- We are particularly interested in:
    - Trade-offs made
    - Simplicity and clarity
    - Correctness over over-engineering

Deliverables

1. **Source code repository**
2. **Live deployed URL**
3. **README** including:
    - Architecture overview
    - Authentication approach
    - How workflows from Part 1 are integrated
    - How background processing is implemented
    - Assumptions and limitations

Evaluation Criteria

- Correct integration with Technical Test – Part 1
- Frontend UX quality and data flow clarity
- Cloud architecture design
- Asynchronous processing correctness
- Security basics (auth-gated access)
- Code clarity and reasoning