

API Task: Document360

Hi there 🙌 ,

Congratulations on making it to the assessment round!

This task is designed to evaluate how you think through a real-world scenario, implement solutions effectively, and communicate your approach. We are excited to see how you explore, execute, and present your solution.

Please carefully review the task requirements and expectations listed below before getting started. If you have any questions, feel free to reach out.

Task : **API Task: Document360 - Customer API (CRUD Operations)**

Duration : 1 hour 30 minutes

Objective:

To assess the candidate's ability to interact with REST APIs using standard HTTP methods and implement basic CRUD operations using a console application. This task will help us evaluate your technical approach, code structure, error handling, and understanding of API integration.

Prerequisite:

Ensure your development environment is ready with one of the following:

- Visual Studio / Visual Studio Code
- IntelliJ / Eclipse / PyCharm
- Or any IDE relevant to your chosen language

◆ **Requirements:**

The task consists of **4 main API operations**. You are expected to:

- Build a **console application** in a language of your choice (Java, C#, Python, etc.)
- Implement **4 individual methods/classes** for each CRUD operation:
 - GET – Fetch all drive folders
 - POST – Create a new drive folder
 - PUT – Update the folder name
 - DELETE – Remove the folder
- Use **standard naming conventions** for variables, classes, and methods
- Include request headers (esp. `api_token`) correctly in each request
Print/log:
- The **request URL, headers, and body** (for POST/PUT)

- The **response status, body**, and any error messages
- Store and pass the folder ID dynamically across methods
- Keep the code modular and readable with comments where necessary




Expected Timeline (1 hour 30 Mins)

Time	Task
0–15 mins	Task #1 – Fetch all drive folders using GET method
15–35 mins	Task #2 – Create a new drive folder using POST
35–55 mins	Task #3 – Rename the created folder using PUT
55–70 mins	Task #4 – Delete the created folder using DELETE
70–90 mins	Bonus – Implement response validation, error handling, and documentation

Expectations:

- ✓ Proper request and response logging
- ✓ Use of modular methods or functions
- ✓ Clean and readable code structure
- ✓ Include error handling (e.g., invalid folder ID, timeouts)
- ✓ Bonus: Validate HTTP status codes and expected response structure (e.g., JSON schema)

Constraints:

-  Avoid using in-built methods for response parsing unless necessary - prefer manual inspection where possible
-  Avoid hardcoding folder IDs - store and reuse dynamically
-  No third-party API libraries unless pre-approved (e.g., use requests in Python, HttpClient in Java/C#)

Task Descriptions:

Task #1 – GET All Drive Folders

- Objective: Retrieve the list of all existing drive folders.
- Endpoint: GET <https://apihub.document360.io/v2/Drive/Folders>
- Expected Output: Log the full response showing folder names, IDs, and metadata.

Task #2 – POST Create a New Folder

- Objective: Create a folder in the drive (nested or root level).
- Endpoint: POST <https://apihub.document360.io/v2/Drive/Folders>
- Expected Output: Log success message with newly created folder name and ID.
- **Note:** Store the folder ID for use in the next tasks.

Task #3 – PUT Update Folder Name

- Objective: Rename the folder created in Task #2.
- Endpoint: PUT <https://apihub.document360.io/v2/Drive/Folders/{folderId}>
- Expected Output: Log updated folder name and confirmation message.

Task #4 – DELETE the Folder

- Objective: Delete the folder created in Task #2.
- Endpoint: DELETE <https://apihub.document360.io/v2/Drive/Folders/{folderId}>
- Expected Output: Log a confirmation of successful deletion.

Authentication

All API requests must include the following header: **api_token** : <your_token_here>

Submission Format:

- Upload your completed code to a public or private GitHub repository.
- Make sure the repository contains:
 - README.md with:
 - Brief overview of the task
 - Language and tools used
 - Steps to run the project
 - Sample output/log screenshots (optional)
- Clearly organized folders/files for each API task (GET, POST, PUT, DELETE)
- Any configuration or token files should be mocked or excluded using .gitignore
- Once done, **share the GitHub link with us** by replying to the assessment email or directly via the hiring platform.

The above context is framed to give you a detailed guideline on how you have to approach the task and to help you stay equipped well in advance to complete the task successfully. Please feel free to reach out to us if you have any questions.

Wishing you good luck 