Project: Docling for Document Processing (Backend & API Focus)

Project Title: Docling Document Processing API Development
Objective: To assess the ability to integrate and utilize Docling's backend functionalities for OCR and structured output generation, demonstrating proficiency in API interaction, error handling, and data processing.
Scenario: Your team needs a robust backend service to process various document types (PDFs, images) and extract their content in a structured, machine-readable format. This service should leverage Docling's capabilities for OCR on scanned documents and convert all documents into a standardized output.
Deliverables:

1. **Python Script/Service:**
    - A Python script or a small Flask/FastAPI service that exposes an endpoint (e.g., /process_document).
    - This endpoint should accept a document (e.g., a file upload or a URL to a document) as input.
    - The service should use Docling to:
        - Perform OCR if the document is a scanned image (e.g., JPG, PNG) or a scanned PDF.
        - Process digital PDFs, DOCX, or other supported formats.
        - Convert the processed document into **Markdown** and **JSON** formats.
        - Return both the Markdown and JSON outputs as part of the API response.
    - Implement robust error handling for invalid inputs, Docling processing failures, and other potential issues.
    - Consider implementing asynchronous processing if dealing with large documents.
2. **API Documentation (Markdown/Postman Collection):**
    - Provide clear documentation for the API endpoint, including:
        - Endpoint URL and HTTP method (e.g., POST /process_document).
        - Request body structure (e.g., how to send the document).
        - Response body structure for successful and error cases (including Markdown and JSON outputs).
        - Example requests and responses.
3. **Setup and Run Instructions:**
    - A README.md file with clear instructions on how to set up the project (e.g., pip install commands, any environment variables).
    - Instructions on how to run the script/service.
    - How to test the API endpoint.
4. **Sample Documents:**
    - Provide at least one sample scanned PDF/image for OCR testing.
    - Provide at least one sample digital PDF or DOCX for structured extraction.

Evaluation Criteria:

- **Docling Usage:** Correct and effective utilization of Docling's core functionalities for OCR and document conversion.
- **API Design:** Clarity, consistency, and RESTfulness of the API endpoint.
- **Code Quality:** Readability, maintainability, modularity, and adherence to Python best

practices.
- **Error Handling:** Robustness and clarity of error messages.
- **Output Formatting:** Accuracy and completeness of the generated Markdown and JSON outputs.
- **Documentation:** Clarity and completeness of the API documentation and setup instructions.
- **Efficiency (Bonus):** Consideration of performance for larger documents (e.g., streaming, asynchronous processing).

Pre-Requisite Knowledge / Information for Pre-Reading:

To successfully complete this project, the fresher should familiarize themselves with the following:

1. Docling Documentation:
- **Docling PyPI Page:** https://pypi.org/project/docling-google-ocr/ - This is a good starting point for installation and basic usage.
- **Docling GitHub Repository:** https://github.com/docling-project/docling - Explore the docs directory and examples for deeper insights into its capabilities, especially the docling-serve component for API usage and docling-parse for backend parsing.
- **Docling for AI Workflows (Red Hat Article):** https://developers.redhat.com/videos/docling-efficient-document-processing-ai-workflows - Provides a good overview of Docling's purpose and how it fits into AI pipelines.

2. Docling OCR Capabilities:
- **Unstract Blog on Docling OCR:** https://unstract.com/blog/docling-alternative/ - Discusses Docling's OCR in comparison to other tools, highlighting its strengths and limitations.
- **Docling OCR Integration (GitHub):** https://github.com/felixdittrich92/docling-OCR-OnnxTR - Shows how OCR engines can be integrated.

3. Docling Output Formats:
- **Docling Documentation (various sections):** The official Docling documentation and examples often show how to export to Markdown, JSON, and other formats. Look for export_to_markdown() and similar functions.
- **IBM Tutorial on Document Question Answering:** https://www.ibm.com/think/tutorials/build-document-question-answering-system-with-docling-and-granite - Mentions output formats like Markdown and JSON.

4. Python Web Frameworks (Optional but Recommended):
- **Flask Quickstart:** https://flask.palletsprojects.com/en/latest/quickstart/ - For building a simple API.
- **FastAPI Tutorial:** https://fastapi.tiangolo.com/tutorial/ - Another excellent choice for building performant APIs.

5. General API Concepts:
- Understanding of REST principles (GET, POST, request/response bodies, status codes).
- Handling file uploads in Python web frameworks.