



# Software Developer Intern - Technical Assessment

## Overview

The goal of this assessment is to evaluate your ability to:

1. **Ingest and manipulate data** from multiple formats.
2. Develop a **local application** to visualize the data and expose it as REST API endpoints.
3. Write clean, maintainable code using **Object-Oriented Programming (OOP)** principles and rigorous testing.

You are free to use **any programming language or frameworks** you are most comfortable with.

---



## Task Description

### 1. Data Ingestion and Manipulation

- You are provided with data in the following formats:
    - **CSV** (e.g., company revenue data)
    - **PPTX** (.pptx format)
    - **JSON** (nested data)
    - **PDF** (contains tabular data).
  - **Requirements:**
    - Write a script or application that:
      - Reads and ingests data from **all the provided formats**.
      - Cleans the data to handle missing values and inconsistencies.
      - Merges the data into a **unified structure** for further use (e.g., a table, array, or object).
- 

### 2. Data Visualization and REST API Development

- Develop a **local application** with the following features:
    - **Data Visualization:**
      - Display the processed data using at least **two visualizations** (e.g., bar chart, table, or line graph).
      - Allow basic filtering or sorting of the data.
    - **REST API Endpoints:**
      - **GET /api/data**: Returns the full unified dataset in JSON format.
      - **GET /api/data/{file\_type}**: Returns data specific to a file type (e.g., "csv", "excel").
  - **Guidelines:**
    - Use any web framework or tool of your choice (e.g., Flask, FastAPI, Express, Spring Boot, Django, etc.).
    - The local app should be easy to run and interact with (a simple frontend is fine).
-



### 3. Code Quality: OOP and Testing

- **Code Structure:**
    - Follow **Object-Oriented Programming (OOP)** principles to organize your solution into classes and methods (e.g., DataIngestion, DataProcessor, Visualization, APIHandler).
  - **Testing:**
    - Implement **unit tests** and **integration tests** to ensure code reliability.
    - Use any testing framework of your choice (e.g., JUnit, PyTest, Mocha, Jest).
-



## Submission Guidelines

### 1. Code Repository:

- Submit your work as a **GitHub repository** containing:
  - All code files.
  - Dependencies file (e.g., `requirements.txt`, `package.json`, `pom.xml`, etc.).
  - Instructions to set up and run the project.

### 2. README File:

Include the following:

- **Overview:** A description of your solution and approach.
  - **Setup Instructions:** Steps to run the app locally and access the API.
  - **Testing Instructions:** Steps to execute unit and integration tests.
  - **Assumptions or Challenges:** Any assumptions or challenges you faced.
-



## Evaluation Criteria

Criteria	Weight
<b>Data Ingestion</b>	30%
Ability to process and merge data from all provided formats. Handle inconsistencies and missing values.	
<b>Visualization and API</b>	20%
Quality of visualizations and API functionality (endpoints and data correctness).	
<b>Code Quality and Structure</b>	25%
OOP principles, modular design, and readability.	
<b>Testing (Unit and Integration)</b>	20%
Coverage, reliability, and clarity of tests.	
<b>Documentation</b>	5%
Completeness and clarity of the README and setup instructions.	

---

## Timeline

- **Deadline:** You have **3 days** to complete this assessment.
  - **Estimated Effort:** The task is designed to be completable within **1 day of focused work**.
-



## Tools and Resources

- Use any programming language or framework of your choice.
  - Sample data files (CSV, PPT, JSON, and PDF) will be provided.
  - Optional: Use any libraries/tools for data visualization and testing.
- 

## Notes

- Ensure the solution is modular and easy to understand.
- Focus on delivering a clean, functional application rather than perfecting every feature.
- If you face any challenges with file formats (e.g., parsing PDFs), note it in your README and handle what you can.