**Study Case Submission**

1. **Title**

AI Powered CV Evaluator

1. **Candidate Information**

* **Full Name :** Rachmat Hidayat Abduh
* **Email Address :** rachmath.059@gmail.com

1. **Repository Link**

<https://github.com/Rchmt-cmd/ai-powered-cv-evaluator>

1. **Approach & design (Main Section)**

* **Initial Plan**

My initial plan for this study case submission was to analyse and thoroughly understand the content of the case study brief before moving on to the development phase. I identified the main components of the CV Evaluator system, the data input and output flow, as well as the tools and technologies to be used. Based on this analysis, I defined several key assumptions for the project:

1. A simple web service is required to handle user requests.
2. External RAG and LLM services can be useful for rapid and effective development.
3. A database is needed to store document data and track the progress of each job pipeline.

* **System & Database Design**

A screenshot of a phone

Description automatically generated

1. Web Service (Express JS)

This project uses the Express.js framework as its web service. Express.js is a lightweight, unopinionated, and fully customizable web development framework, making it highly suitable for fast and simple project development.

1. RAG Service (Ragie.AI)

Ragie.AI is a fully managed *Retrieval-Augmented Generation (RAG)* platform, also known as *RAG-as-a-Service*. Ragie.AI is well-suited for this project as it simplifies AI integration with data without the need to build a custom retrieval system, while also improving the accuracy and relevance of model-generated analyses or evaluations. In this project, internal document files are uploaded to the Ragie.AI platform and processed there, allowing the web service to simply retrieve relevant data chunks based on the query.

1. LLM Service (GroqCloud)

GroqCloud is a high-performance AI inference platform that enables the execution of large language models (LLMs) with extremely low latency and high efficiency. In this project, GroqCloud is used to accelerate the inference process and generate real-time AI responses, making it ideal for automated evaluation and AI-driven analysis.

1. Database (MySQL)

The MySQL database is used to store CV and project report documents for evaluation, as well as the progress of each job processed within the evaluation pipeline. This design ensures that there are no blocking requests on the web service. The following is the database schema design.

A screenshot of a computer program

Description automatically generated

1. Evaluation Pipeline

A white rectangular cards with black text

Description automatically generated

There are two pipelines in this project: the evaluator pipeline and the summarizerpipeline.  
The evaluator pipeline handles the evaluation process of the candidate’s CV and project report, while the summarizer pipeline generates a summary based on the output of the evaluator pipeline.  
The following is the workflow of the evaluation pipeline:

1. The document path is retrieved from the database and loaded from object storage.
2. The document is then processed into raw text.
3. Context retrieval is performed through the RAG service (Ragie.ai) based on the type of document being processed.
4. The prompt is sent to the LLM service, and the output is parsed into JSON format.
5. Finally, the evaluated result from the LLM is stored back into the MySQL database.

* **LLM Integration**
* **Prompting Strategy**
* **Resilience & Error Handling**
* **Edge Cases Considered**

1. **Results & Reflection**

* **Outcome**
* **Evaluation of Results**
* **Future Improvements**

1. **Screenshots of Real Responses**

* POST /upload → returns documents id

A screenshot of a computer

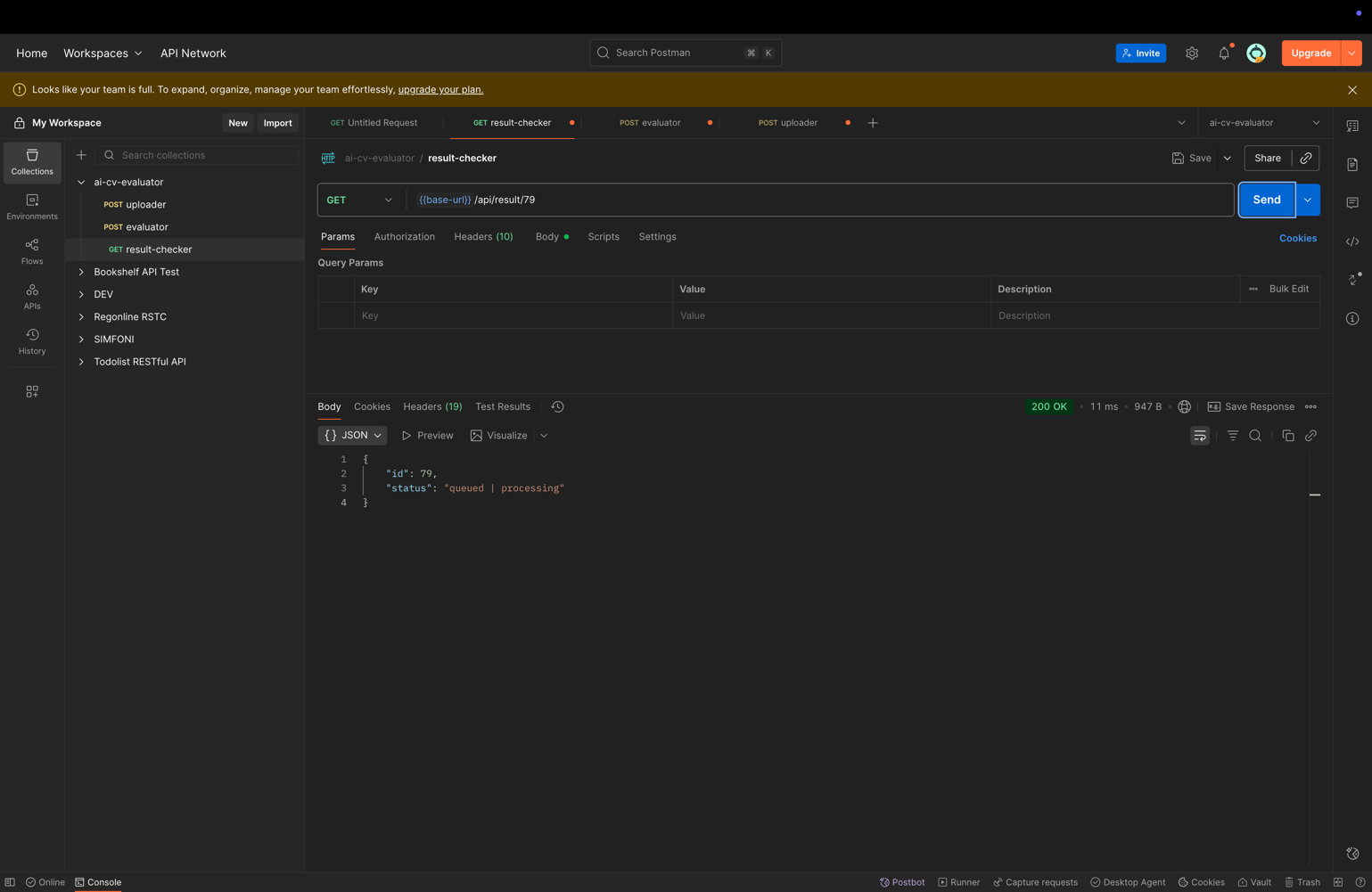
Description automatically generated

* POST /evaluate → returns job\_id + status

A screenshot of a computer

Description automatically generated

* GET /result/:id → returns final evaluation (scores + feedback)



A screenshot of a computer

Description automatically generated

1. **(Optional) Bonus Work**