Study Case Submission

1. AI System for Calculate CV and Project Report

2. Candidate Information

- **Full Name:** Adhimukti Surya Anindyajati
- **Email Address:** adhisurya05@gmail.com

3. Repository Link

- https://github.com/suryadragn/ai-cv-evaluator

4. Approach & Design (Main Section)

- **Initial Plan**

After getting the Case Study Task, I choose my well-known languanges and I got just node.js, I don't know about what is vector DB etc. Then I Analyze the step by step for doing the task. First of all, I make base node.js express and make route /upload, /evaluate, then /result, and I choose Chroma DB for the DB, study for a bit. Then I manage to use it using default_tenants because this is new for me. After that I choose Gemini for AI side of checking.

Break down the requirements :

1. Storage file PDF from client then save it on my local then save it on metadata to the database.
2. PDF parsing with npm library to extract the PDF to text.
3. Asynchronous Processing to Evaluate LLM for processing duty to checking the extracted PDF text. Then endpoint have to respone the first thing then still doing the job to processing Evalute PDF data to AI.
4. Stuctured Output to LLM fix the output for result Evaluated value that format to JSON.

Key Assumption :

1. PDF is always contains Texts because it is CV and Project Report.
2. Scalibility LLM basic is I assume that Gemini AI API's latency is enough low to do job for processing PDF extracted Text.
3. Chroma DB is new for me, but I think it good and fast to save the data for id and name file, then another collections for the evaluate value.

Scope Boundaris :

1. Just for PDF, that /upload API endpoint is just for PDF with handler.
2. Using Metadata storage because using standard database like (PostgreSQL/MySQL) is need time for setup.

- **System & Database Design**

API Endpoints Design :

Endpoints	Method	Form-data	Function
/upload	POST	- cv - project_report	Get the uploaded File CV and Project Report (PDF). Save it on file disk, extract the PDF file using pdfjs-dist to get the text , and document & metadata (JSON) to ChromaDB
/evaluate	POST	- id - job_title	Get the id and job_title, then search on metadata from ChromaDB . Save the first metadata evaluate. Send json reply, and still do the job to evaluate the PDF text using Gemini AI after done, Save the result to metadata evaluate.
/status/:id	GET		Get the result from Metadata Evaluate result on ChromaDB

Database Schema :

1. Collections : candidates

```
{
    "ids": [
        "1"
    ],
    "embeddings": null,
    "documents": [
        "\n        --- CV FILE: 1761018205479.pdf ---\n        CURRICULUM VITAE  ADHIMUKTI SURYA
ANINDYAJATI  Backend Engineer | Spesialis Integrasi & Pengembangan Berbasis Data   | Kabupaten Karanganyar,
Jawa Tengah (Siap Bekerja Remote)  Kontak:   081329844484 | adhisurya05@gmail.com | LinkedIn:
[https://www.linkedin.com/in/adhisurya]  .... {{ it's too long i cut here, contains cv file, cv text , report
file, report text, and upload at }}
       ] ,
          "uris": null,
          "metadatas": [
              {
                  "uploaded_at": "2025-10-21T03:43:26.114Z",
                  "report_name": "1761018205484.pdf",
                  "cv_name": "1761018205479.pdf"
              }
          ],
          "include": [
              "documents",
              "metadatas"
          ]
      }
```

2. Collections : eval_candidates

```
{
    "ids": [
        "1"
    ],
    "embeddings": null,
    "documents": [
        "On Evaluate Request for Record ID: 1 , and status set to completed"
    ],
    "uris": null,
    "metadatas": [
        {
            "cv_match_rate": 4.75,
            "project_score": 3.25,
            "status": "completed",
            "overall_summary": "Adhimukti adalah kandidat yang sangat menjanjikan untuk posisi backend
programmer, dengan latar belakang teknis yang kuat, pengalaman kepemimpinan dalam proyek-proyek kompleks, dan
rekam jejak dalam memberikan solusi berkinerja tinggi. Ditambah dengan minat dan inisiatifnya dalam
mengintegrasikan AI/LLM ke dalam sistem backend, ia memiliki potensi besar untuk menjadi aset berharga yang
siap menghadapi tantangan teknis modern dan belajar dengan cepat.",
            "cv_feedback": "Adhimukti menunjukkan kecocokan yang sangat kuat sebagai backend programmer dengan
pengalaman lebih dari 4 tahun dalam peran kepemimpinan. Keterampilan teknisnya dalam PHP, JavaScript, MySQL,
RESTful API, dan server-side logic sangat solid, didukung oleh pencapaian signifikan dalam optimasi performa
dan integrasi sistem enterprise kompleks.",
            "project_feedback": "Proyek ini menunjukkan pemahaman yang baik tentang arsitektur sistem AI
dengan asynchronous processing untuk evaluasi LLM dan desain API yang terstruktur. Meskipun deskripsi
implementasi detail untuk prompt design dan error handling spesifik belum sepenuhnya mendalam, kemampuan
adaptasi dan pembelajaran teknologi baru (Chroma DB, Gemini AI) sangat menonjol."
        }
    ],
    "include": [
        "documents",
        "metadatas"
    ]
}
```

Job Qeueue / Long-running task handling :

Endpoint /evaluate is calling the fungsion to process the evalute job_title and extracted cv and project report from Chroma DB and sending json respon to client without waiting the proces of Gemini AI. Client get fast respone after send /evalute. Client can send access to /result without wait /evalute to be done.

- **LLM Integration**

  I use Gemini 2.5 Flash because I already use it on my hobby project Bot Telegram n8n. It is free after all. My Promp design using role and input so I make the AI is impersonate to role “Anallisis Perekrutasn AI” and giving 2 paramater job_title that input during access the API endpoint and combinedText that contains documen on ChromeDB using id send on endpoint. RAG strategy is combaining CV and Report extracted text and storage on ‘candidates’ Collections.

Prompting Strategy

My prompt is like this using parameter that send and my document from ChromaDB,

````
“Anda adalah seorang **Analisis Perekrutan AI** yang sangat teliti. Tugas Anda adalah mengevaluasi seorang kandidat berdasarkan **CV** dan **Laporan Proyek** mereka terhadap suatu posisi pekerjaan.

    ### I. INPUT: TEKS KANDIDAT & POSISI PEKERJAAN

    1.  **POSISI PEKERJAAN:** "${job_title}"

    2.  **TEKS GABUNGAN KANDIDAT:** Teks berikut adalah hasil ekstraksi dari CV dan Laporan Proyek:

      ---

      ${combinedText}

      ---

    ### II. METODE EVALUASI (1-5 SKALA)

    Lakukan evaluasi dua bagian berdasarkan panduan di bawah ini, berikan skor pada skala 1 hingga 5 (hanya angka).

    #### A. CV Match Evaluation (Bobot Total: 100%)

    | Parameter | Deskripsi | Bobot | Scoring Guide (Skala 1-5) |

    | :--- | :--- | :--- | :--- |

    | **Technical Skills Match** | Kesesuaian keterampilan teknis (backend, database, APIs, cloud, AI/LLM) dengan persyaratan pekerjaan. | 40% | 1=Irrelevant, 5=Excellent Match |

    | **Experience Level** | Jumlah tahun pengalaman dan kompleksitas proyek sebelumnya. | 25% | 1=<1 yr/trivial, 5=>5 yrs/high-impact |

    | **Relevant Achievements** | Dampak hasil kerja masa lalu (scaling, performance, adoption). | 20% | 1=No measurable impact, 5=Major measurable impact |

    | **Cultural / Collaboration Fit** | Komunikasi, pembelajaran, mindset teamwork/leadership. | 15% | 1=Not demonstrated, 5=Excellent & well-demonstrated |

    #### B. Project Deliverable Evaluation (Skala 1-5)

    | Parameter | Deskripsi | Bobot | Scoring Guide (Skala 1-5) |

    | :--- | :--- | :--- | :--- |

    | **Correctness (Prompt & Chaining)** | Implementasi prompt design, LLM chaining, RAG context injection. | 30% | 1=Not implemented, 5=Fully correct & thoughtful |

    | **Code Quality & Structure** | Bersih, modular, reusable, diuji. | 25% | 1=Poor, 5=Excellent quality + strong tests |

    | **Resilience & Error Handling** | Penanganan *long jobs*, *retries*, *randomness*, kegagalan API. | 20% | 1=Missing, 5=Robust, production-ready |

    | **Documentation & Explanation** | Kejelasan README, instruksi setup, *trade-off* explanation. | 15% | 1=Missing, 5=Excellent, 5=Excellent & insightful |

    | **Creativity / Bonus** | Fitur ekstra di luar persyaratan. | 10% | 1=None, 5=Outstanding creativity |

    ### III. OUTPUT: JSON TERSTRUKTUR

    Berikan seluruh analisis dan skor dalam satu objek JSON.

    \`\`\`json

    {

      "cv_match_rate": 0.0,

      "cv_feedback": "Ringkasan 2-3 kalimat mengenai kecocokan CV, mencakup kekuatan dan kelemahan dalam Technical Match dan Experience Level.",
````

"project_score": 0.0,

"project_feedback": "Ringkasan 2-3 kalimat mengenai kualitas Proyek, mencakup poin positif (Code Quality, Correctness) dan kekurangan (Resilience, Documentation).",

"overall_summary": "Ringkasan menyeluruh 3-5 kalimat mengenai kecocokan kandidat secara total (gabungan CV dan Proyek) dan rekomendasi rekrutmen."

}

\`\`\`

Berikan hanya objek JSON, tidak ada teks pendahuluan atau penutup lainnya."

I use Gemini to make this prompt too.

- **Resilience & Error Handling**

API Failure handling with try and catch for handling the error and failure. Status handling is if LLM proseccing is fail my function "saveChromaEvaluate" with catch error make the client not waiting forever. I didn't make any retries after all.

- **Edge Cases Considered**

The scenario for unusual input is file not pdf and no paramter send. If on endpoint /upload that send not pdf and return "Hanya file PDF yang diizinkan" on fileFilter function, then if pdf not send 1 or both it would respone like "Kirimkan kedua persyaratan: 'cv' dan 'project_report' (PDF)." and if /evaluate no parameter would return "Permintaan harus menyertakan 'id' dan 'job_title'." or on result would return 404 not found and if id is false is become "Record dengan ID ${id} tidak ditemukan."

How I test, I do some test on API endpoint like what I said on above. Testing all the posibble and as expected the return would come.

5. Results & Reflection

- **Outcome**

First let talk about what is didn't work as expected. Ok, first of all. When I choose node.js and study a bit for API implementation because usually I use PHP framework like CI or Yii2. I think would be same with node.js because there's a npm while on PHP have composer but when I finish doing some endpoint I wanna get pdf parse, it getting hard. because some library has commonJS and old js configuration. So I use OpenAI first I use it, because I get free pro version from subscribe shopee VIP. but it didn't help much. so I change to Gemini AI then after pdf-parse then pdf-to-text still didn't work, and the final choice is pdfjs-dist still hard to accomplise after make polyfills.js get global.DOMMatrix become class {} and import the main app.js is finally done. After that is Auto-Increment ID, for the first i using uploads.json for storage before save it to chromaDB. but in the end I fully use chromaDB for ID if 0 length on ids so currentId is 0 and if has array value getting the max id and plus 1 for the next ID. Then after that I manage to make /upload to parse the PDF then save it on ChromaDB, on my first try is on /evalute endpoint is work perfectly, after I change and move it to /upload my pdfjs-dist send warning TT i don't know why. but Gemini said it's Ok, it's just a warn.

Then we talk about what is work well and perfectly fine is Saving Vektor and Metadata on ChromaDB even though is my first time using it, Alhamdulillah, perfectly done. Then my Asynchronize /evaluate is fine too. because I send json respone first to client, it didn't have to wait loong to respone, and didn't fail ether.

- **Evaluation of Results**

The Evaluation Results is consider stable because Prompt is consistent and clear. Using Instruction Role first on first statement of the prompt make the result is stable, and you giving the clear rubrics to evaluate. So LLM have target and blueprint what to do. And Json Output that structure to match the result we want.

- **Future Improvements**

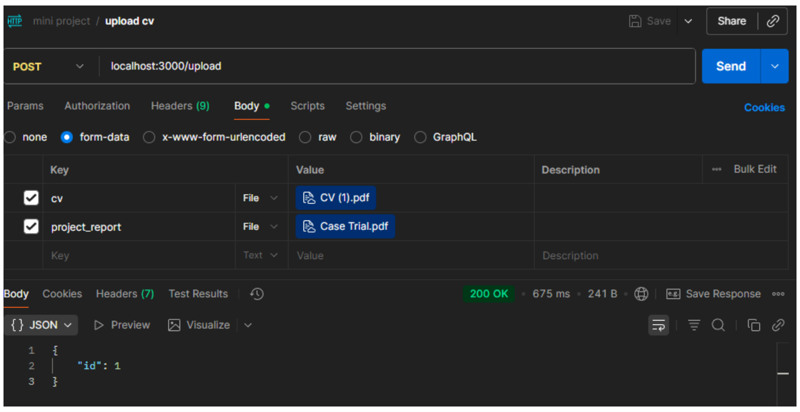
For the future improvement I wanna study about queueing task first, I imagine If I would be job street / job seeker web that wanna check CV and Project Report someone, then many people access it and not have queue it would be desaster. Then I wanna make over the ID thing maybe using uuid or combine key from extracted text. Then I wanna make it after parsing PDF save not the whole text but I wanna get the focus point what is suitable for the job_title.

Let's talk the constraints, the biggest problem was time. I focused only on the main steps: getting the text from the PDFs, saving it in the Chroma database, and asking the Gemini AI for the score. Because of this, we could not build better features like RAG (Retrieval-Augmented Generation). RAG would have helped the AI check the documents faster.Another issue at the start was saving the ChromaDB data. We had to be very careful to make sure the database saved the files forever (persistent mode). If we did not, all the CV data would be gone when the computer program stopped.Lastly, the AI is slow (LLM Latency). Asking the Gemini API for a score takes the longest time. We made the /upload page give an ID right away, so the user doesn't wait. But the final result on the /status/:id page still takes some time because the AI needs to finish its slow work in the background.

6. Screenshots of Real Responses

1. API Endpoint and Screenshot :

→ /upload



POST localhost:3000/upload

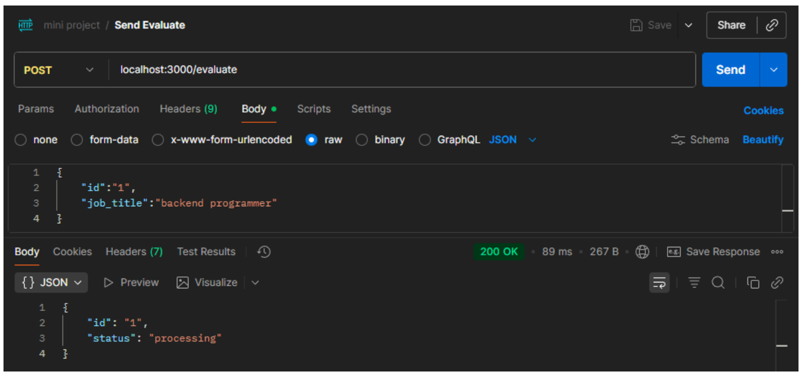
○ none ● form-data ○ x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	Key		Value	Description
☑	cv	File	CV (1).pdf	
☑	project_report	File	Case Trial.pdf	
	Key	Text	Value	Description

Body — 200 OK · 675 ms · 241 B

```
{
    "id": 1
}
```

→ /evaluate



POST localhost:3000/evaluate

○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

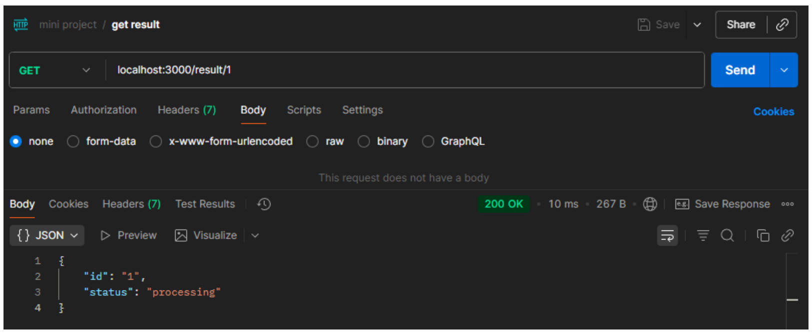
```
{
    "id":"1",
    "job_title":"backend programmer"
}
```

Body — 200 OK · 89 ms · 267 B

```
{
    "id": "1",
    "status": "processing"
}
```

→ /result/:id [/result/1]

Status : Processing



GET localhost:3000/result/1

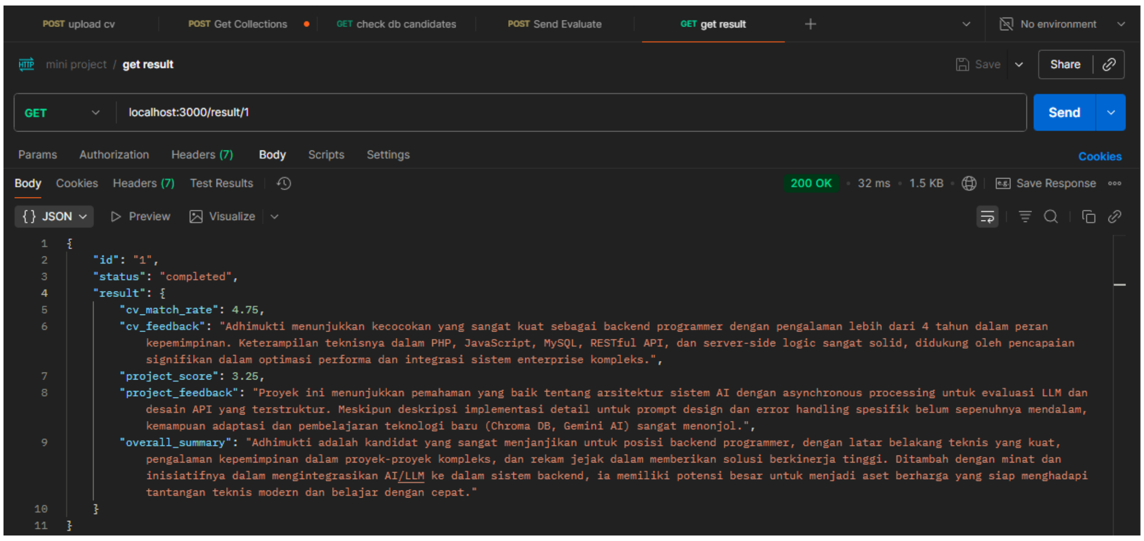
● none ○ form-data ○ x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

This request does not have a body

Body — 200 OK · 10 ms · 267 B

```
{
    "id": "1",
    "status": "processing"
}
```

Status : Completed



GET localhost:3000/result/1

Body — 200 OK · 32 ms · 1.5 KB

```
{
    "id": "1",
    "status": "completed",
    "result": {
        "cv_match_rate": 4.75,
        "cv_feedback": "Adhimukti menunjukkan kecocokan yang sangat kuat sebagai backend programmer dengan pengalaman lebih dari 4 tahun dalam peran kepemimpinan. Keterampilan teknisnya dalam PHP, JavaScript, MySQL, RESTful API, dan server-side logic sangat solid, didukung oleh pencapaian signifikan dalam optimasi performa dan integrasi sistem enterprise kompleks.",
        "project_score": 3.25,
        "project_feedback": "Proyek ini menunjukkan pemahaman yang baik tentang arsitektur sistem AI dengan asynchronous processing untuk evaluasi LLM dan desain API yang terstruktur. Meskipun deskripsi implementasi detail untuk prompt design dan error handling spesifik belum sepenuhnya mendalam, kemampuan adaptasi dan pembelajaran teknologi baru (Chroma DB, Gemini AI) sangat menonjol.",
        "overall_summary": "Adhimukti adalah kandidat yang sangat menjanjikan untuk posisi backend programmer, dengan latar belakang teknis yang kuat, pengalaman kepemimpinan dalam proyek-proyek kompleks, dan rekam jejak dalam memberikan solusi berkinerja tinggi. Ditambah dengan minat dan inisiatifnya dalam mengintegrasikan AI/LLM ke dalam sistem backend, ia memiliki potensi besar untuk menjadi aset berharga yang siap menghadapi tantangan teknis modern dan belajar dengan cepat."
    }
}
```

7. (Optional) Bonus Work

I just added /health endpoint and /chroma-check/:id endpoint, to make sure my chromaDB is good and can storage the data. Any file that come with my git repository is the fact I'm still beginner of this task. After all, this is the best opportunity you gave to me. Thanks, It mean a lot for me to study some modern tech for web development.