Study Case Submission Template

Please use this template to document your solution. Submit it as a **PDF file** along with your project repository.

---

1. Title

AI CV Evaluator

---

2. Candidate Information

- **Full Name: Muhammad Irfan Noor Wahid**
- **Email Address: irfannoor681@gmail.com**

---

3. Repository Link

- **Repository: https://github.com/irfan-wahid/ai-cv-evaluator**

---

4. Approach & Design (Main Section)

Program yang telah saya buat ini bernama AI CV Evaluator, yaitu sebuah sistem yang dapat mengetahui seberapa tepat CV dan project yang telah dibuat oleh kandidat dengan informasi lowongan pekerjaan serta project submission yang dibutuhkan. Sistem ini memberikan hasil berupa angka tingkat kecocokan beserta penjelasannya, seperti feedback CV, feedback project, serta rangkuman keseluruhan mengenai kemiripan, dengan memanfaatkan teknologi AI.

Pada tahap awal perencanaan sebelum melakukan implementasi sistem, saya memulai dengan melakukan analisis terhadap requirements yang dibutuhkan. Dari yang saya tangkap, tujuan dibangunnya sistem ini adalah untuk melakukan evaluasi CV dan project submission yang dibuat oleh kandidat terhadap informasi yang ada pada suatu lowongan pekerjaan. Untuk membangun sistem tersebut, saya memecah proses menjadi beberapa bagian seperti mencari tahu istilah-istilah teknis yang belum pernah saya ketahui sebelumnya, menyusun daftar endpoint yang diperlukan, memahami rancangan proses bisnis pada setiap endpoint tersebut, hingga merancang database yang dibutuhkan untuk mengelola data seperti lowongan pekerjaan, kandidat, dan hasil evaluasi CV maupun project.

Dalam membangun sistem ini, terdapat beberapa asumsi utama yang saya tetapkan. Pertama, data lowongan pekerjaan dan project submission dikirim dalam bentuk plain text, sedangkan data CV dan project yang dikirim oleh kandidat berupa file PDF. Kedua, kriteria penilaian untuk CV dan project ditentukan oleh penyedia lowongan pekerjaan. Selain itu, setiap kandidat wajib dan hanya dapat mengirim satu CV dan satu project dalam proses pelamaran.

Adapun ruang lingkup sistem juga dibatasi agar fokus pada tujuan utama. Lingkup yang termasuk adalah kemampuan mengunggah CV dan project, melakukan evaluasi otomatis berbasis AI, serta menghasilkan skor kecocokan, feedback CV, feedback project, dan rangkuman evaluasi. Hasil evaluasi akan tersimpan dalam database dan dapat diakses melalui endpoint API. Sementara itu, hal-hal yang tidak termasuk adalah fitur untuk memperbarui maupun menghapus data lowongan pekerjaan, serta memperbarui maupun menghapus data lamaran kandidat.

Kemudian lanjut ke bagian teknis dari sistem. Sistem backend ini dibangun dengan menggunakan Node.JS dengan framework Express.JS untuk membangun endpoint yang dibutuhkan. Untuk list endpoint yang saya buat adalah sebagai berikut:

1. POST /job-vacancy

   Digunakan untuk recruiter menambahkan data lowongan pekerjaan seperti deskripsi pekerjaan, project submission, hingga rubrik penilaian.

2. POST /job-application/:jobVacancyId

   Digunakan untuk kandidat mengirim file CV dan project submission ke data lamaran yang dituju.

3. POST /job-application/evaluate

   Digunakan untuk mengevaluasi tingkat kecocokan CV dan project submission pada lamaran kandidat yang telah dikirim sebelumnya. Jadi sebelum menjalankan endpoint ini, kandidat perlu mengirim data CV dan project submission terlebih dahulu.

4. GET /job-application/result/:id

   Digunakan untuk mengetahui status dari evaluate yang terbagi menjadi 4 status yaitu, queued yang berarti masih

pada antrian, processing ketika sedang diproses, completed ketika evaluasi selesai, dan failed jika terjadi kegagalan pada saat evaluasi.

Untuk database yang saya bangun menggunakan PostgreSQL dengan bantuan extension PgVector untuk menyimpan data beberapa data yang di-embedding menggunakan teknologi AI. Berikut merupakan skema database yang saya rancang.

Berdasarkan gambar tersebut terdapat 2 tabel, yaitu job_vacancies yang berfungsi untuk menyimpan data lowongan pekerjaan dan job_applications untuk menyimpan data lamaran pekerjaan. Terdapat data embedding bertipe vector yang digunakan untuk menyimpan array hasil embedding untuk melakukan perbandingan kemiripan antara lamaran dengan data lowongan pekerjaan.

Proses evaluasi pada sistem ini berjalan pada latar belakang dengan memanfaatkan library BullMQ untuk redis queue worker sehingga tidak mengganggu proses lain ketika proses evaluasi.

Pada sistem ini saya memanfaatkan service dari CohereAI untuk proses generate embedding data dan generate text. Alasan saya menggunakan CohereAI ini karena gratis dengan tertentu dan hasil yang dihasilkan cukup baik. Service yang saya gunakan dari CohereAI antara lain:

1. Cohere.embed

   Digunakan untuk melakukan generate embed data dari text menjadi dalam bentuk vector yang nantinya akan digunakan untuk mencari tingkat kecocokan. Disini saya memanfaatkan model ‘embed-english-v2.0’.

2. Cohere.chat

   Digunakan untuk melakukan generate text. Sebagai contoh pada sistem ini saya memakai prompt

```
You are evaluating a candidate. Return JSON only, no explanation.

Job description: ${jobVacancy.description}

Study case: ${jobVacancy.studyCaseBrief}

Candidate CV: ${cvText}

Candidate Project: ${projectText}


Scoring rubric:

CV: ${rubricCv.join(", ")}

Project: ${rubricProject.join(", ")}


Instructions:

- For each CV rubric item, give a numeric score (1–5) and feedback.

- For each Project rubric item, give a numeric score (1–5) and feedback.

- Also provide overall summary.
```

JSON format:

```
{
   "cvFeedback": "string",
   "projectFeedback": "string",
   "overallSummary": "string",
   "rubricScore": {
      "cv": { "<rubric_name>": number },
      "project": { "<rubric_name>": number }
   }
}
```

Tujuan dibuatnya prompt tersebut adalah agar AI bisa melakukan generate text dengan JSON format untuk mengembalikan data seperti feedback, summary, dan penilaian terhadap rubrik sesuai dengan data yang telah disimpan pada database. Disini saya menggunakan model ‘command-a-03-2025’ dengan temperature 0 supaya hasil yang dihasilkan tidak terlalu acak.

Dalam integrasi dengan AI tersebut pastinya tidak terhindarkan dari error baik dari sisi program ini maupun dari pihak AI tersebut. Namun, hal tersebut dapat diusahakan untuk dihindarkan. Seperti pada sistem ini karena saya memanfaatkan queue, saya menambahkan retry supaya ketika ada kegagalan ketika proses evaluasi sistem akan berusaha untuk mengulanginya sebanyak 3 kali sebelum dinyatakan gagal. Ketika sudah 3 kali namun masih saja gagal, data akan berstatus ‘FAILED’.

Terdapat pula error handling pada setiap endpoint API yang saya lakukan. Hal ini dilakukan untuk mencegah sistem mati ketika terjadi error atau kesalahan dari sisi code, sehingga akan memunculkan pesan error yang akan memberi tahu error apa yang terjadi dan pada bagian mana error tersebut yang dapat dilihat pada log.

Dalam pengembangan sistem ini, saya juga mempertimbangkan beberapa case atau skenario tidak biasa yang mungkin terjadi, seperti kurangnya input pada endpoint, ataupun kegagalan respon dari layanan AI eksternal. Walaupun pengujian untuk seluruh skenario tersebut belum dilakukan secara menyeluruh, saya sudah menambahkan validasi dasar pada sisi backend serta mekanisme retry pada proses evaluasi untuk mengurangi risiko kegagalan. Ke depan, pengujian lebih detail terhadap kasus-kasus ini perlu dilakukan agar sistem benar-benar tangguh dan mampu memberikan respon yang sesuai ketika menghadapi kondisi di luar skenario normal.

> ✍ This is your chance to be a storyteller. Imagine you’re presenting to a CTO, clarity and reasoning matter more than buzzwords.

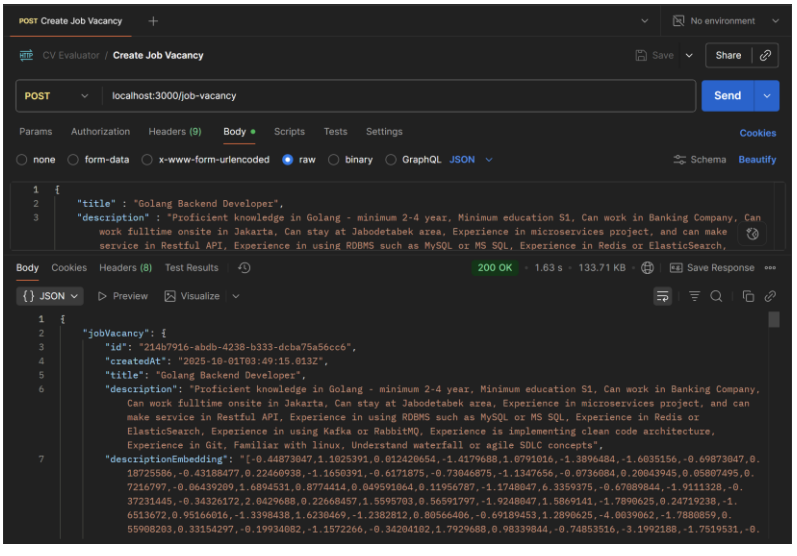
---

5. Results & Reflection

- **Outcome**

Pada sistem yang saya bangun ini, semua endpoint sudah berjalan cukup baik. Berikut hasil dari setiap endpoint:

1. POST /job-vacancy

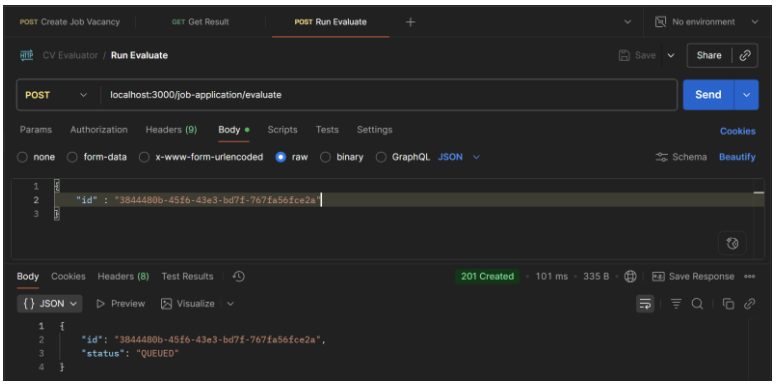


Pada endpoint tersebut saya sudah berhasil menambahkan data lowongan pekerjaan, rectuiter dapat menambahkan data seperti judul lowongan, deskripsi, penjelasan project yang perlu dikerjakan oleh kandidat, hingga rubrik skor.

2. POST /job-application/:jobVacancyId

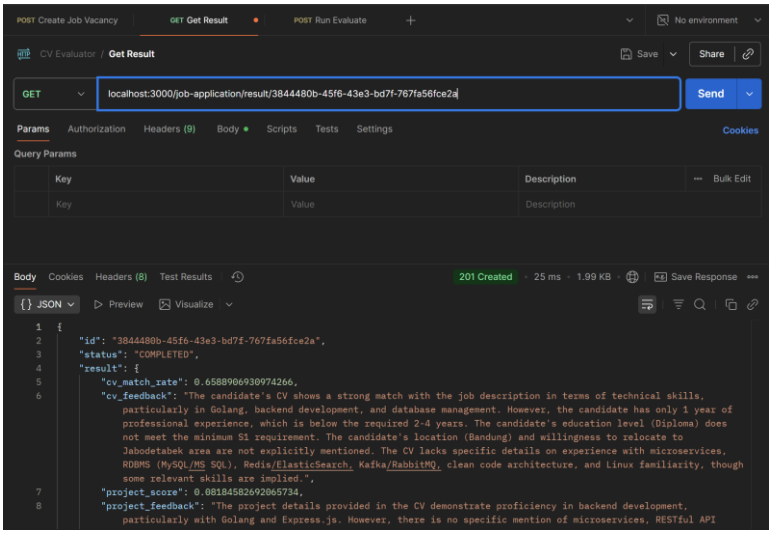
   Pada endpoint tersebut kandidat sudah bisa menambahkan file pdf CV dan project yang telah dikerjakan sesuai dengan ketentuan pada lowongan pekerjaan.

3. POST /job-application/evaluate



Pada endpoint tersebut recruiter sudah bisa menjalankan proses evaluasi CV dan project berdasarkan lamaran yang telah dikirim oleh kandidat. Tetapi masih ada kekurangan pada endpoint ini yaitu belum dipakainya rubrik skor untuk mengetahui final_score seperti yang terdapat pada requirements, sehingga perhitungan rubrik saya buat untuk menampilkan skor per-point saja. Untuk rubrik skor sudah masuk ke data vector pada database hanya saja belum terpakai, sehingga hanya plain text yang dimasukkan ke prompt AI langsung untuk mendapatkan nilai per-point tersebut.

4. GET /job-application/result/:id



Pada endpoint tersebut recruiter sudah bisa melakukan pengecekan terhadap status evaluasi, ketika masih proses hanya akan memunculkan status, dan ketika sudah selesai akan memunculkan output berupa status ‘COMPLETED’ beserta data hasil evaluasi.

- **Evaluation of Results**

  Hasil keluaran dari AI sudah cukup baik dan konsisten, karena pada code saya atur temperature menjadi 0 sehingga hasil konsisten dan tidak terlalu acak. Dapat dilihat pada hasil endpoint cek result berikut.

```
{
  "id": "3844480b-45f6-43e3-bd7f-767fa56fce2a",
  "status": "COMPLETED",
  "result": {
    "cv_match_rate": 0.6588906930974266,
    "cv_feedback": "The candidate's CV shows a strong match with the job description in terms of technical skills,
      particularly in Golang, backend development, and database management. However, the candidate has only 1 year of
      professional experience, which is below the required 2-4 years. The candidate's education level (Diploma) does
      not meet the minimum S1 requirement. The candidate's location (Bandung) and willingness to relocate to
      Jabodetabek area are not explicitly mentioned. The CV lacks specific details on experience with microservices,
      RDBMS (MySQL/MS SQL), Redis/ElasticSearch, Kafka/RabbitMQ, clean code architecture, and Linux familiarity, though
      some relevant skills are implied.",
    "project_score": 0.08184582692065734,
    "project_feedback": "The project details provided in the CV demonstrate proficiency in backend development,
      particularly with Golang and Express.js. However, there is no specific mention of microservices, RESTful API
      implementation, or experience with Kafka/RabbitMQ, Redis, or ElasticSearch. The projects show creativity and
      technical competence but lack detailed documentation and resilience testing.",
    "overallSummary": "The candidate demonstrates strong technical skills in Golang and backend development, aligning
      well with the job requirements. However, the candidate falls short in terms of professional experience (1 year
      vs. 2-4 years), education level (Diploma vs. S1), and explicit experience with required technologies like
      microservices, Kafka/RabbitMQ, Redis, and ElasticSearch. The candidate's potential and skills are promising, but
      further experience and education are needed to fully meet the job criteria."
  }
}
```


- **Future Improvements**
  Jika memiliki waktu lebih banyak, saya ingin memperbaiki kekurangan menambahkan beberapa fitur untuk meningkatkan kualitas sistem. Misalnya, penentuan final score berdasarkan scoring rubric yang sudah disimpan pada database vector, memperluas dukungan format file, menambahkan fitur untuk mengupdate dan menghapus data lowongan maupun lamaran, serta meningkatkan sistem feedback agar lebih kaya dengan visualisasi skor dan analisis mendalam.
