Documento de Projeto: Velo Edu Challenge

Construindo um Sistema Multiagente de Análise Acadêmica

Olá, equipe! Bem-vindos à versão final do Velo Edu Challenge.

Este documento foi reestruturado para ser um guia completo e claro. Nosso objetivo continua o mesmo: construir uma ferramenta com um propósito real de identificar alunos que precisam de apoio. A grande novidade é como vamos fazer isso: usando uma equipe de "agentes" de software especializados que colaboram para chegar à melhor conclusão.

1. O Desafio em Linguagem Simples: O Que Vamos Construir?

Na prática, vocês vão construir um programa em Python que é executado pelo terminal.

Este programa funcionará como um "comitê de especialistas virtuais". Ele fará três coisas:

Ler e Entender: Lerá um arquivo CSV com o histórico de vários alunos. Diagnosticar em Equipe: Para um aluno específico, múltiplos agentes de software trabalharão em conjunto. Um analisará as notas, outro a frequência. Um terceiro formulará uma hipótese, que será desafiada por um "advogado do diabo" para garantir a precisão. Sugerir uma Ação: Com base no diagnóstico validado pela equipe, o programa sugerirá uma ação concreta e inteligente.

O Entregável Principal: Um script Python que você pode rodar assim: python analisar_aluno.py --arquivo historico_academico.csv --id alu_101

E a saída no terminal será um objeto JSON detalhado com a análise completa, mostrando o trabalho de cada agente.

2. Como Funciona por Dentro: Arquitetura Multiagente Colaborativa

Abandonamos um modelo linear simples para adotar um modelo de "mesa redonda" de especialistas, orquestrado por um agente central.

A Equipe de Agentes:

1. Coordenador de Análise (O Maestro): Gerencia todo o fluxo. Recebe o ID do aluno, convoca os especialistas, coleta os resultados, orquestra a validação e monta o relatório final. 2. Analisador de Desempenho (O Especialista em Notas): Focado exclusivamente em notas. Busca tendências de queda, inconsistências e desempenho abaixo da média. 3. Analisador de Engajamento (O Especialista em Frequência): Focado exclusivamente em presença. Busca padrões de ausência e desengajamento. 4. Agente de Diagnóstico (O Médico): Recebe os relatórios dos dois analistas e formula uma hipótese inicial (ex: "Queda de rendimento associada a baixo engajamento"). 5. Validador de Hipóteses (O Advogado do Diabo): Recebe a hipótese e os dados brutos para tentar refutá-la ou confirmá-la, adicionando nuances importantes. 6. Conselheiro Acadêmico (O Estrategista): Recebe o diagnóstico final e validado para escolher a ação mais precisa em um "playbook" de intervenções.

Fluxo de Trabalho:

(Início: id_aluno)
|
v

1. Coordenador de Análise

```
|--> 2. Analisador de Desempenho (Paralelo)
|--> 3. Analisador de Engajamento (Paralelo)
|
v
```

(Coordenador coleta e consolida os relatórios)
|
v

4. Agente de Diagnóstico (cria a hipótese)

|
v

5. Validador de Hipóteses (confirma ou refina)

|
v
(Coordenador recebe o diagnóstico validado)
|
v

6. Conselheiro Acadêmico (sugere a ação)

|
v
(Coordenador monta o JSON final)

3. A Fonte da Verdade: Entendendo o Dataset

Vocês criarão um arquivo `historico_academico.csv` simulado com a seguinte estrutura:

id_aluno	**nome_aluno**	**curso**	**periodo*do*curso**	**id_disciplina**	**semestre_letivo**	**data_evento**	**tipo_evento**	**presenca**	**nota**
alu_101	Ana Silva	Eng. de Software	3	CS101	2024.2	2024-09-20	prova	1	6.5
alu_101	Ana Silva	Eng. de Software	3	CS101	2025.1	2025-02-10	aula	0	null

Esta segunda linha significa que a aluna Ana Silva, no seu 3º período, faltou à aula de CS101 no dia 10/02/2025.

4. A Saída: O JSON Final Detalhado

O resultado final deve ser um JSON que não apenas dá a resposta, mas também mostra o processo de raciocínio da equipe de agentes.


```
{
  "idAluno": "alu_101",
  "dataAnalise": "2025-09-23T02:47:00Z",
  "scoreRiscoEvasao": 82,
  "diagnosticoChave": "DIFICULDADE_PONTUAL_EM_DISCIPLINA_CRITICA",
   "justificativa": "O sistema identificou uma queda de rendimento e engajamento fortemente concentrada na disciplina CS101, sugerindo uma dificuldade específica em vez de um desinteresse geral pelo curso.",
  "processoDeAnalise": {
    "relatorioDesempenho": "Detectada queda de 2.1 pontos na média geral, com a nota em CS101 sendo 40% inferior à média das outras disciplinas.",
    "relatorioEngajamento": "Padrão de ausências identificado nas últimas 4 semanas, ocorrendo exclusivamente nas aulas de CS101.",
    "hipoteseInicial": "Queda de rendimento geral por desengajamento.",
    "validacaoDaHipotese": "Hipótese inicial refutada. A análise cruzada mostra que o problema não é geral. A causa raiz é uma dificuldade específica em CS101, que está impactando a motivação."
  },
  "metricasAluno": {
    "mediaGeralSemestreAtual": 6.1,
    "mediaGeralSemestreAnterior": 8.2,
    "frequenciaPresencaAtual": "65%",
    "disciplinaCritica": "CS101"
  },
  "acaoRecomendada": {
```

"playbookId": "PB_PEDAG_02",
"canal": "Sistema Acadêmico / E-mail do Coordenador",
"titulo": "Agendar Reunião de Apoio Pedagógico Focado",
"templateMensagem": "ALERTA: Aluno [Nome do Aluno] (ID: alu_101) apresentou sinais de dificuldade pontual. Sugestão: Coordenador do curso deve convidá-lo para uma conversa e oferecer tutoria específica para a disciplina CS101."
}
}

5. As Ferramentas (Stack Técnica) e Pré-requisitos

- **Base Obrigatória:** Python 3.9+, Pandas, Pydantic, python-dotenv.

- **IA e Orquestração:**
  - **Opção A (Recomendado):** Usar **LangChain** para modelar os agentes e orquestrar o fluxo. Ferramentas como `AgentExecutor` ou `Router Chains` são ideais para implementar o "Coordenador".
  - **Opção B (Avançado):** Usar a biblioteca `openai` diretamente e criar sua própria lógica de orquestração em Python.

- **Desafio Central:** A implementação do **Coordenador de Análise** é uma parte crucial do projeto. Vocês precisarão pensar em como gerenciar o estado e o fluxo de dados entre os agentes.

6. Esclarecendo Dúvidas Comuns (FAQ do Projeto)

- **"Isso é Machine Learning? Preciso treinar um modelo?"**
  - Não. Este projeto **não envolve o treinamento de um modelo.** Usamos um modelo de linguagem (LLM) já treinado como uma ferramenta para interpretar dados e gerar texto.

- **"Por que analisar um aluno por vez?"**
  - Este é o **primeiro passo**. Fazer a análise individual funcionar perfeitamente é o foco. Um "scanner" que varre a base toda seria a Fase 2 deste produto.

- **"O dataset não é muito simples?"**
  - Sim, e isso é **intencional**. O objetivo é focar na arquitetura dos agentes e na lógica de colaboração dentro de um prazo viável (2 meses).

7. Entregáveis e Critérios de Aceitação

Entregáveis:

1. Documento de Cenário (PDF).
2. Dataset Sintético (`.csv`).
3. Código-Fonte Completo (Repositório Git).
4. Documentação Técnica (`README.md`).
5. Apresentação Final (Pitch de 15 min).

Critérios de Aceitação:

- **Funcional:** O script executa sem erros e o JSON de saída é válido.

- **Técnico:** O código é bem organizado, modularizado por agente, e o repositório Git é bem mantido.

- **Qualidade da Inteligência:**
  - [ ] A `justificativa` gerada é clara, baseada em dados e coerente.
  - [ ] As ações recomendadas são pedagogicamente relevantes.
  - [ ] A estrutura do JSON de saída reflete claramente a colaboração entre os agentes, preenchendo a seção `processoDeAnalise`.