

Ficha de Acompanhamento

Acompanhamento Semanal

- **Git:** Commits feitos regularmente (sem “mega-commits” no final).
- **Pull Requests:** Código fundido na **main** via PR com revisão de pares.
- **CI:** Pipeline (GitHub Actions) a passar (testes + linter **ruff/flake8**).
- **Gestão (ADM):** Backlog atualizado e tarefas distribuídas (Jira).
- **ADRs:** Decisões técnicas importantes documentadas (ex: escolha do DB, modelo LLM).
- **Reunião Técnica (RT):** Apresentação de progresso/dúvidas ao docente (Product Owner).

Ativ.	Semanas														
	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
Git															
PR															
CI															
ADM															
ADR															
RT															

Observações

FASE 1: Dados e Infraestrutura (Semanas 2-4)

Foco: Engenharia de Dados, Setup DevOps e Persistência.

Semana 2: Definição e Setup

- ☐ Definir **Domínio do Problema** (ex: Saúde, Finanças, E-commerce).
- ☐ Atribuir Papéis: Data Eng, AI Eng, Backend, Project Master.
- ☐ Criar repositório Git e definir estratégia de branch (Flow ou Trunk).
- ☐ Configurar Docker básico (`Dockerfile` + `docker-compose.yml`).
- ☐ Configurar Linter e CI Básico.

Semana 3: Ingestão de Dados

- ☐ Identificar fontes de dados (APIs, Datasets, Crawlers).
- ☐ Escrever scripts de ingestão para recolher **>100k registros**.
- ☐ Pelo menos **3 fontes** de dado diferentes.
- ☐ Validar qualidade dos dados “brutos”.
- ☐ Garantir que todos os containers sobem e comunicam entre si.

Semana 4: Persistência

- ☐ Levantar Bancos de Dados no Docker.
- ☐ Implementar pipeline ETL para SQL (dados estruturados).
- ☐ Implementar armazenamento NoSQL (documentos/logs).
- ☐ Escolher modelo de *Embedding* e configurar DB Vetorial (ex: Chroma).
- ☐ Teste de Integração: Script Python conecta aos 3 tipos de DB.

FASE 2: Lógica e Inteligência (Semanas 5-9)

Foco: NeSy (Neuro-bólico), RAG e Backend.

Semana 5: RAG Básico (Exploração Semântica)

- ☐ Criar pipeline de indexação (chunks -> embeddings -> vector store).
- ☐ Implementar função de busca vetorial na API.
- ☐ Testar recuperação de documentos com perguntas simples.
- ☐ Criar estrutura inicial da API Backend.

Semana 6: Ferramentas e Function Calling

- ☐ Implementar *Function Calling / Tools* no LLM.
- ☐ Criar ferramentas para o LLM consultar o Banco SQL (ex: “Calcular total de vendas”).
- ☐ Garantir que o LLM sabe escolher entre RAG (texto) e SQL (cálculo).

Semana 7: Arquitetura NeSy e Prompts

- ☐ Refinar Prompts (Mover para `prompts.yaml`).

- ☐ Implementar lógica determinística de suporte no Backend.
- ☐ Implementar Sistema de Logging (Registrar input/output do LLM).

Semana 8: MVP End-to-End (Checkpoint Crítico)

- ☐ O sistema consegue ir do Dado -> Resposta simples no Terminal/API?
- ☐ Verificar se o sistema usa pelo menos 2 ferramentas distintas.
- ☐ Ajustar performance (Opcional: testar PEFT/LoRA ou modelos menores se estiver lento).

Semana 9: (Páscoa - acumula com a semana 10)

- ☐ Criar script `eval.py` com 5-10 perguntas de controlo.
- ☐ Validar raciocínio do Agente.
- ☐ Congelar lógica complexa de Backend.

FASE 3: Integração e Interface (Semanas 10-13)

Foco: Frontend (Data App) e Experiência de Decisão.

Semana 10: Interface de Chat

- ☐ **ALERTA:** A BD deve estar pronta. Se não estiver, há penalização.
- ☐ Implementar Chat UI básico (Streamlit/Chainlit/Gradio).
- ☐ Conectar Chat UI ao Backend (API).

Semana 11: Dashboard Analítico

- ☐ Criar visualizações de dados (Gráficos, KPIs) no Streamlit/Gradio.
- ☐ Alimentar gráficos diretamente via SQL (bypass ao LLM para dados puros).

Semana 12: Interatividade Contextual

- ☐ Ligar Chat ao Dashboard: O chat altera os gráficos? (ex: Filtros aplicados pelo agente).
- ☐ Implementar feedback visual (ex: “Pensando...”, “Consultando SQL...”).

Semana 13: Deploy Simulado e Testes Finais

- ☐ Atualizar `docker-compose.yml` para versão “Produção”.
- ☐ Testar deploy numa máquina limpa (não na “máquina do aluno”).
- ☐ Gerir segredos via `.env` (Remover chaves do código).

FASE 4: Polimento e Entrega (Semana 14-15)

Foco: Documentação, Avaliação e Defesa.

Semana 14: Polimento

- ☐ Rever “Integridade Académica” (Garantir compreensão do código gerado por AI).
- ☐ Corrigir bugs detetados nos logs.
- ☐ Melhorar UX (mensagens de erro claras).
- ☐ Preparar **Demo ao Vivo** (Cenários de sucesso garantidos).

Semana 15: Entregáveis Finais

- ☐ **Relatório Técnico:** Problema, Solução, ADRs, Análise Crítica.
- ☐ **Vídeo/Demo:** Preparar setup para apresentação (20 min).
- ☐ **Code Freeze:** Nenhuma alteração funcional, apenas documentação.

Semana 16: Apresentações.