Manual Mestre & Dossiê Editorial Completo: Ecossistema WOKDEX

Visão do Professor, Diretrizes Científicas, TP1, TP2, TP3 e Trilha de Iniciação Científica

Prof. Aléssio Miranda Júnior — Inteligência Artificial II (Engenharia de Computação)

2026-09-02

Índice

1 PARTE I — Dossiê Editorial, Diagnóstico Crítico & Bastidores Científicos **5**
1.1 1. Visão Editorial & Análise Crítica dos Trabalhos Práticos . . . 5
1.2 1. O Dilema Epistemológico & A Resposta Científica . . . 5
1.2.1 A Dúvida Inicial do Professor: . . . 5
1.2.2 A Resposta Racional e Metodológica: . . . 5
1.3 2. Diagnóstico Técnico por Trabalho Prático . . . 5
1.3.1 2.1. TP1 — O Classificador de Habilidades (Redes Neurais & MLOps) . . . 5
1.3.2 2.2. TP2 — O Oráculo RAG (Bancos Vetoriais & Spring AI) . . . 6
1.3.3 2.3. TP3 — O Agente Autônomo e Curador WOKDEX . . . 6
1.4 3. Estratégia de Sobrevivência do TP3: As APIs "Golden" de Fallback . . . 7
1.4.1 3.1. O Risco de Bloqueio em Cascata . . . 7
1.4.2 3.2. O Impacto no Ecossistema de Pesquisa . . . 7
1.4.3 3.3. As Tarefas e Infraestrutura do Professor (Golden Fallback) . . . 7
1.5 4. Guia de Mediação Pedagógica (Como Conduzir em Sala) . . . 7
1.5.1 4.2. Gestão de Gargalos e Avaliação por Pareamento (*Pair Programming*) . . . 7
1.6 5. A Estratégia de Publicação Científica (O Artigo Consolidado) . . . 8
1.6.1 5.1. Estrutura do Artigo Principal . . . 8
1.6.2 5.2. Questões de Pesquisa Formais (RQ_1 a RQ_4) . . . 8
1.6.3 5.3. Tabela de Engenharia Consolidada para o Artigo . . . 9
1.7 6. Roteiro de Consolidação Pelo Professor (Semana a Semana) . . . 9
1.8 2. Manual dos Bastidores: A Fábrica Invisível de Pesquisa WOKDEX . . . 10

2 Manual dos Bastidores: A Fábrica Invisível de Pesquisa WOKDEX **11**
2.1 1. A Estratégia da "Fábrica Invisível" (*Crowdsourcing Didático*) . . . 11
2.1.1 O Conceito . . . 11
2.2 2. As 5 Questões e Hipóteses de Pesquisa (RQ_1 a RQ_5) . . . 12
2.2.1 RQ_1 (NLP / Representação Textual em Português) . . . 12
2.2.2 RQ_2 (Modelagem Multi-Label & Desbalanceamento Extremo) . . . 12
2.2.3 RQ_3 (Recuperação Semântica / RAG Educacional) . . . 12
2.2.4 RQ_4 (Mitigação de Alucinação Sintática & Data Anchoring) . . . 12
2.2.5 RQ_5 (Validação Pedagógica via Simulated Students) . . . 12
2.3 3. O Portfólio de 3 Artigos Científicos . . . 13
2.4 4. O "Algoritmo de Fusão" do Professor (Semana a Semana) . . . 13
2.4.1 Fase 1: Pós-TP1 (Semana 3–4) — Compilação do Artigo 1 (NLP) . . . 13
2.4.2 Fase 2: Pós-TP2 (Semana 9–10) — Consolidação do RAG . . . 13
2.4.3 Fase 3: Demoday / Pós-TP3 (Semana 14–15) — O Gran Finale dos Artigos 2 e 3 . . . 14
2.5 5. Matriz de Hipóteses vs. Evidências Empíricas . . . 14

3 PARTE II — Visão Geral dos Trabalhos Práticos para os Alunos **15**

4 Bem-vindos à Missão **16**
4.1 O Problema: O Silêncio Pedagógico . . . 16
4.2 A Solução: O WOKDEX . . . 17
4.3 As 3 Camadas do YAML WOKDEX . . . 17
4.3.1 Camada 1: Descritiva (O que é este exercício?) . . . 17
4.3.2 Camada 2: Pedagógica (O que o aluno precisa saber?) . . . 17

4.3.3 Camada 3: Avaliativa (Como o aluno é testado?) . . . 18
4.4 Os 4 Tipos de Teste do WOKDEX . . . 18
4.4.1 O caso mais interessante: MISCONCEPTION . . . 18
4.5 Exemplo Real: O Exercício "Divisão" . . . 18
4.6 O Pipeline de 5 Fases (O Mapa do Semestre) . . . 19
4.7 Como as Equipes Funcionam? (Pareamento e Colaboração) . . . 20
4.8 Regras de Auditoria no GitLab (Como comprovar sua parte) . . . 21
4.9 Entregas e Avaliação . . . 21
4.9.1 A Balança da Nota (Como você será avaliado) . . . 21
4.10 Trilha IC: Equipes de Pesquisa (Voluntário) . . . 21
4.10.1 A Jornada Completa . . . 22
4.10.2 O Mapa de Integração do Semestre . . . 22
4.11 Acesse os Detalhes de Cada Entrega . . . 22
4.11.1 Trilha Regular (Todas as Equipes) . . . 22
4.11.2 Trilha IC (Equipes Voluntárias) . . . 22
4.11.3 Recursos e Documentação Integrada . . . 23

5 PARTE III — Especificações Técnicas das Entregas Regulares 24
5.1 1. TP1: O Classificador de Habilidades (Redes Neurais, NLP & MLOps) . . . 24
5.2 Por que este TP existe? . . . 24
5.3 Leitura Obrigatória (Fundamentação) . . . 24
5.4 O que é uma Skill no WOKDEX e no Dataset? . . . 25
5.5 O Dataset Oficial (`leetcode_problems_pt`) . . . 26
5.5.1 Snippet de Carga Rápida (Python / Pandas) . . . 27
5.6 A Divisão de Tarefas na Equipe (4 Alunos em 2 Duplas) . . . 28
5.6.1 Dupla 1: Ciência e Dados (O Cérebro) — ~10 a 15h por aluno . . . 28
5.6.2 Dupla 2: Engenharia de Software e MLOps (A Casca) — ~10 a 15h por aluno . . . 28
5.7 A API que você vai construir . . . 29
5.7.1 Requisição: `POST /predict` . . . 29
5.7.2 Resposta esperada: . . . 29
5.8 Relatório Técnico no `README.md` (Documentação do TP1) . . . 29
5.9 Sugestões de Testes Automatizados Obrigatórios (`pytest`) . . . 30
5.10 Rubrica de Avaliação Detalhada (8 Pontos) . . . 30
5.10.1 Nota Individual (4 pts) . . . 30
5.10.2 Nota do Grupo (4 pts) . . . 31
5.10.3 Escala Progressiva de Performance (F1-Score Macro) . . . 31
5.11 Cronograma Sugerido (4 Semanas) . . . 31
5.12 Checklist de Entrega — TP1 . . . 32
5.13 Adendo IC — Golden Benchmark e Estudo de Ablação Textual (Opcional) . . . 32
5.13.1 O que fazer . . . 32
5.13.2 Entregáveis IC-TP1 . . . 33
5.14 2. TP2: O Oráculo RAG (Bancos Vetoriais, Busca Semântica & Spring AI) . . . 34
5.15 Por que este TP existe? . . . 34
5.16 Start Pack e checkpoint técnico . . . 34
5.17 Corpus, proveniência e contrato de embeddings . . . 35
5.17.1 Contrato de embedding . . . 35
5.18 Metodologia honesta e sem vazamento . . . 35
5.19 API de recuperação e contrato de recomendação . . . 36
5.19.1 `GET /search` . . . 36
5.19.2 Recomendação de skills . . . 36
5.20 Golden Set e avaliação multi-label . . . 37
5.20.1 Análise de discordância . . . 37
5.21 Entregáveis de engenharia . . . 37
5.22 Papéis e entregas individuais . . . 38
5.23 Artigo SBC incremental . . . 38
5.24 Roteiro de desenvolvimento . . . 39
5.25 Rubrica (10,0 pontos) . . . 39
5.26 Desafios avançados opcionais . . . 40

5.27 Checklist de entrega . . . 40
5.28 3. TP3: O Agente Autônomo Curador WOKDEX & Demoday . . . 41
5.29 Proposito e escopo . . . 41
5.30 Objetivos de aprendizagem . . . 41
5.31 Produto a entregar . . . 41
5.32 Integracao por contratos . . . 42
5.33 Governanca de dados e provedor . . . 42
5.33.1 Testes de contrato e falhas . . . 42
5.34 Classificador, recuperacao e arquitetura hibrida . . . 42
5.35 Resposta RAG fundamentada . . . 43
5.35.1 Politica de abstencao . . . 43
5.35.2 Rubrica de fundamentacao por resposta . . . 43
5.36 Observabilidade . . . 44
5.37 Avaliacao experimental: seis casos obrigatorios . . . 44
5.38 Artigo SBC incremental . . . 44
5.39 Papeis e estimativa de esforco . . . 45
5.40 Sequência de desenvolvimento . . . 45
5.41 Apresentações em 30/11 e 02/12 . . . 45
5.42 Rubrica (20 pontos) . . . 45
5.43 Checklist de entrega final . . . 46
5.44 Desafios avancados opcionais . . . 46

6 PARTE IV — Trilha Especial de Iniciação Científica (Pesquisa Avançada) 47
6.1 1. Programa IC Especial: Visão Geral . . . 47
6.2 O que é o Programa IC Especial? . . . 47
6.3 Objetivo científico . . . 47
6.4 Quem pode participar? . . . 47
6.4.1 Critérios de seleção . . . 47
6.5 Método e entregáveis IC . . . 48
6.6 Papéis realistas para uma equipe de quatro . . . 48
6.7 Cronograma 2026-2 . . . 48
6.8 Mentoria e resultados esperados . . . 49
6.9 Riscos e mitigação . . . 49
6.10 2. TP3 Master: Simulated Students & Auditoria de Pipelines . . . 50
6.11 Natureza e elegibilidade . . . 50
6.12 Pergunta de pesquisa . . . 50
6.13 Dados, contratos e cuidados éticos . . . 50
6.14 Método fixo . . . 50
6.15 Entregáveis . . . 51
6.16 Papéis para quatro estudantes . . . 51
6.17 Artigo curto SBC (3-4 páginas) . . . 52
6.18 Cronograma . . . 52
6.19 Apresentações em 30/11 e 02/12 . . . 53
6.20 Rubrica (20 pontos) . . . 53
6.21 Checklist de entrega em 26/11 . . . 53

7 PARTE V — Recursos Oficiais, Schemas & Especificação do Dataset 54

8 Recursos Oficiais do WOKDEX 55
8.1 Dataset Oficial da Disciplina: LeetCode Problems Dataset Bilíngue (PT-BR) . . . 55
8.1.1 Composição e Estatísticas do Dataset . . . 55
8.1.2 Campos Disponíveis para os Modelos . . . 55
8.2 Artigo Científico Base . . . 56
8.2.1 As 4 Contribuições Científicas do Artigo . . . 56
8.3 JSON Schema Oficial (`wok-problem.json`) . . . 57
8.3.1 Campos Obrigatórios (`required`) . . . 57
8.4 Os 4 Tipos de Teste (TestType) . . . 57
8.4.1 Subtipos de MISCONCEPTION . . . 57
8.5 Heurística de Modulação de Dica (HMD) . . . 57

8.6 Arquivos Locais deste Diretório 58

Documento Confidencial do Professor (Dossiê Completo)

Este documento reúne, em um único compêndio integrado: 1. **Dossiê Editorial e Análise Crítica:** Diagnóstico do *Abismo Semântico*, limites do RAG e estratégia do Mega-Artigo. 2. **Manual dos Bastidores:** A Fábrica Invisível de Pesquisa e as hipóteses RQ_1 a RQ_5. 3. **Especificação Completa da Trilha Regular:** Visão Geral, TP1 (MLOps/Keras), TP2 (RAG/Spring AI) e TP3 (Agentes Autônomos). 4. **Trilha de Iniciação Científica (IC):** Programa IC Especial e TP3 Master (Simulated Students). 5. **Recursos & Especificações:** Schemas, métricas e base de dados canônica.

Capítulo 1

PARTE I — Dossiê Editorial, Diagnóstico Crítico & Bastidores Científicos

1.1 1. Visão Editorial & Análise Crítica dos Trabalhos Práticos

> **Documento Confidencial & Estratégico:** Este material é destinado ao uso exclusivo do professor da disciplina para balizar o planejamento didático, a mediação em sala de aula e a consolidação dos resultados científicos no artigo final.

1.2 1. O Dilema Epistemológico & A Resposta Científica

1.2.1 A Dúvida Inicial do Professor:

> *"A análise de enunciados de programação para achar as temáticas (TP1) e a recuperação via RAG (TP2) refletem a realidade ou correm risco alto de estarem longe dela? Será que é um experimento realmente bom para a disciplina e para publicação de um artigo?"*

1.2.2 A Resposta Racional e Metodológica:

Sim, o experimento corre riscos empíricos reais e conhecidos — e é EXATAMENTE por isso que ele é pedagogicamente extraordinário e altamente publicável.

Em Inteligência Artificial aplicada à Educação de Computação (*AIED / CS Education*), datasets lúdicos onde modelos ingênuos atingem 99% de acurácia são pedagogicamente estéreis e cientificamente irrelevantes. O que torna a tríade TP1 → TP2 → TP3 rica é o confronto direto dos alunos de Engenharia com as **fronteiras e limitações reais do estado da arte**.

1.3 2. Diagnóstico Técnico por Trabalho Prático

1.3.1 2.1. TP1 — O Classificador de Habilidades (Redes Neurais & MLOps)

- **O Fenômeno do Abismo Semântico (*Semantic Gap*):**
  - O enunciado de programação é uma narrativa contextual (*"Alice tem caixas de brinquedos…"*). O algoritmo ótimo (*Programação Dinâmica, Guloso, Backtracking*) é uma propriedade matemática da relação entre subproblemas e restrições temporais, **não uma palavra que aparece no texto**.

- **Comportamento Empírico Esperado:**
  - **Tags Sintáticas/Estruturais (`Tree`, `Graph`, `String`, `Matrix`):** $F_1 \geq 75\% - 85\%$. O texto cita explicitamente árvores ou matrizes.
  - **Tags Estratégicas/Conceituais (`Dynamic Programming`, `Greedy`, `Two Pointers`):** $F_1 \leq 35\% - 45\%$.
- **Diretriz de Avaliação do Professor:**
  - A régua de F_1-Macro foi calibrada realisticamente no material: $\geq 35\%$ **(Mínimo)**, $\geq 50\%$ **(Bom)**, $\geq 65\%$ **(Excelente)**.
  - O professor deve avaliar a **qualidade da análise de erro** no Draft 1 (os alunos explicarem por que a rede errou) em vez de apenas o número bruto.

1.3.2 2.2. TP2 — O Oráculo RAG (Bancos Vetoriais & Spring AI)

- **O Fenômeno da Similaridade Superficial (*Surface Similarity Trap*):**
  - Embeddings densos agrupam textos por semântica temática geral. Um problema de "dividir bombons" ficará próximo de outros problemas de "doces", mesmo que o primeiro seja Mochila 0/1 e o segundo seja Guloso.
- **O Fenômeno da Transferência Negativa (*Negative Transfer*):**
  - Se o RAG injetar um problema com historinha idêntica mas algoritmo oposto, ele pode atuar como um "distrator" para a LLM no TP3.
- **Por que o RAG ainda é indispensável?**
  1. Para queries conceituais diretas (*"inverter lista encadeada"*), a precisão é $> 85\%$.
  2. Ele fornece o **template estrutural** de como escrever um `testScenario` e uma `helpTip` sem spoiler.
  3. O LLM no TP3 possui raciocínio para filtrar a historinha e focar nas restrições.

> **i** Registro Editorial: Dilema da Stack do TP2 (Java Spring Boot vs. Python Unificado)
>
> - **Status Atual da Disciplina:** Mantida a stack em **Java (Spring Boot 3.x + Spring AI + PGVector)**. O professor já possui os materiais, starters de código e laboratórios estruturados em Java, garantindo controle pleno sobre a condução pedagógica da turma.
> - **Proposta Alternativa Analisada (Python Unificado):**
>   - *Conceito:* Unificar todo o pipeline da disciplina em Python (FastAPI + ChromaDB / LangChain + Pytest), mantendo o mesmo contrato REST (`GET /search?q=...&k=3`).
>   - *Vantagens Potenciais:* Elimina a troca de ecossistema tecnológico entre TPs, reduz o consumo de RAM dos contêineres para < 100 MB e aproveita a familiaridade dos alunos com o ecossistema do TP1 e TP3.
>   - *Motivo da Pausa:* Como o professor já possui todo o ecossistema de apoio e starter pronto em Java Spring Boot, a stack Java segue como padrão oficial neste momento.
> - **Diretriz de Decisão:** A proposta fica devidamente arquivada neste documento. Caso o professor observe atrito cognitivo desnecessário com o ecossistema JVM/Maven na execução do TP2 durante o semestre, a migração para Python poderá ser adotada como refinamento em ofertas futuras.

1.3.3 2.3. TP3 — O Agente Autônomo e Curador WOKDEX

- **O Papel do Agente:**
  - Integra o TP1 (`predict_skills`) para triagem inicial e o TP2 (`search_similar_cases`) para inspiração de cenários.
  - Utiliza raciocínio socrático (*Chain-of-Thought*) simulando o "Aluno Novato" que erra de propósito para gerar cenários do tipo `MISCONCEPTION`.
  - Garante 100% de conformidade sintática via **Data Anchoring** e laço fechado de **Self-Healing** contra o `wok-problem.json`.

1.4 3. Estratégia de Sobrevivência do TP3: As APIs "Golden" de Fallback

1.4.1 3.1. O Risco de Bloqueio em Cascata

Como o TP3 depende 100% da invocação remota às APIs REST do TP1 (`POST /predict`) e TP2 (`GET /search`), existe um risco sistêmico clássico de arquiteturas em microsserviços: **se uma equipe falhar na manutenção dos contêineres do TP1 ou TP2, ela ficaria completamente travada no TP3**.

1.4.2 3.2. O Impacto no Ecossistema de Pesquisa

Se equipes ficarem travadas sem conseguir rodar o Agente: 1. Elas não geram os arquivos `metadata.yaml` pedagógicos. 2. A rodada de **Validação Cruzada no Demoday** fica desbalanceada. 3. A equipe da **Trilha IC (Simulated Students)** perde massa de dados de teste para auditar com o *Agent Novice*, enfraquecendo a base empírica dos artigos.

1.4.3 3.3. As Tarefas e Infraestrutura do Professor (Golden Fallback)

Para eliminar esse ponto único de falha, o professor assume a seguinte tarefa estratégica: 1. **Subir e Manter os Contêineres de Referência (*Golden APIs*):** * `tp1-golden-api` (FastAPI / Keras): Endpoint canônico rodando localmente/nuvem para servir `POST /predict`. * `tp2-golden-api` (Spring Boot / PGVector): Endpoint canônico rodando para servir `GET /search`. 2. **Definição da Política Pedagógica de Contingência:** * Se o microsserviço de uma equipe quebrar, os alunos podem apontar a URL do `@tool` do Agente para a API Golden do professor. * **Regra de Penalidade:** O grupo sofre uma penalidade proporcional predefinida na pontuação de integração do TP1/TP2 anterior (ex.: −20% a −30% na fração de integração), mas **preserva 100% da pontuação do TP3** (Agente, Tool Calling, CoT, Data Anchoring e Demoday). 3. **Resultado:** Garantia de 100% de sobrevivência da turma até a fase final, com todos os grupos gerando YAMLs válidos e dados para a pesquisa.

1.5 4. Guia de Mediação Pedagógica (Como Conduzir em Sala)

Quando os alunos começarem a testar o TP1 e TP2 e relatarem: *"Professor, a minha rede neural só tirou F1 de 52%! Meu modelo está quebrado?"*, o professor deve mediar com a seguinte postura:

> 4.1. Roteiro de Mediação Socrática do Professor
>
> 1. **Validação e Provocação:** > *"Excelente observação! O seu modelo não está quebrado. Abram a matriz de confusão: em quais classes ele tirou 90% e em quais tirou 20%? Por que uma rede treinada em palavras acerta 'Árvore' e erra 'Programação Dinâmica'?"*
> 2. **Conexão com a Engenharia:** > *"No mundo real da Engenharia de Computação, dados textuais brutos têm ruído. Se o seu modelo de 2ms acerta 15 das 25 classes com custo zero, ele é uma ferramenta de triagem valiosa. No TP3, veremos como o LLM entra para resolver o que o TF-IDF não enxerga."*
> 3. **Foco na Escrita Científica:** > *"Documentem exatamente essa discrepância no README do TP1. Essa análise crítica sobre o Abismo Semântico será fundamental quando vocês forem redigir a seção de resultados do Artigo Final no TP3."*

1.5.1 4.2. Gestão de Gargalos e Avaliação por Pareamento (*Pair Programming*)

Um risco clássico em trabalhos práticos com equipes de 4 alunos é o **bloqueio sequencial** (ex.: a Dupla de Backend parar de trabalhar porque a Dupla de Machine Learning ainda está iterando hiperparâmetros).

O professor deve guiar a turma com as seguintes diretrizes operacionais:

1. **Desenvolvimento Baseado em Contratos (*API-First & Mocks*):**
   No primeiro dia de cada sprint, as duplas acordam os Schemas/DTOs. A Dupla 2 cria um modelo

mock/stub e constrói a API, o Dockerfile e os testes automatizados imediatamente, sem esperar pelo artefato final de IA.

2. **Avaliação Solidária e Justa na Dupla:**
   Em vez de isolar cada aluno em silos impermeáveis, o professor avalia as duplas em conjunto nas sprints de *Engenharia de Dados/ML* e *DevOps/Backend*. Commits conjuntos, *code reviews* e commits cruzados na branch da dupla evidenciam *Pair Programming* real e são altamente pontuados.
3. **Cultura de Ajuda Mútua:**
   Reforçar em sala que, embora existam papéis focais, a equipe é uma entidade única. Alunos que destravam parceiros e colaboram ativamente fortalecem a nota coletiva do grupo e a qualidade do Artigo Final.

1.6 5. A Estratégia de Publicação Científica (O Artigo Consolidado)

Para reduzir a sobrecarga burocrática de escrita durante o semestre, **não exigimos 3 artigos separados dos alunos**. A produção científica é estruturada de forma progressiva e centralizada:

1. **TP1:** Entrega de código, Docker, testes e relatório técnico objetivo no `README.md` (convergência, F_1-Macro e matriz de confusão).
2. **TP2:** Entrega de código, Docker Compose, testes e relatório técnico no `README.md` (Precision@3, MRR e latência P95).
3. **TP3 (O Capstone):** O grupo unifica todas as descobertas em um **Artigo Científico Consolidado (4 a 6 páginas em formato SBC)**.

A partir dos 10 artigos consolidados e dos dados de execução das equipes, o professor terá a base empírica completa para submeter o artigo final a congressos de primeiro escalão (**SBIE, RBIE, WEI, SIGCSE, AIED**).

1.6.1 5.1. Estrutura do Artigo Principal

- **Título Proposto:**
  Do Abismo Semântico aos Agentes com Auto-Cura: Um Estudo Empírico de Abordagens Clássicas, RAG e LLMs na Geração de Metadados Pedagógicos para Juízes Online
- **Veículo Alvo:** SBIE (Simpósio Brasileiro de Informática na Educação) / RBIE (Revista Brasileira de Informática na Educação).

1.6.2 5.2. Questões de Pesquisa Formais (RQ_1 a RQ_4)

Questão	Hipótese Investigada	Evidência nos TPs dos Alunos
RQ_1 **(NLP & Abismo Semântico)**	Modelos supervisionados clássicos sofrem degradação seletiva em classes conceituais ($F_1 < 40\%$), mas são eficientes em classes estruturais ($F_1 > 80\%$).	Tabela comparativa Baseline vs. MLP (README do TP1 de todas as equipes).
RQ_2 **(RAG & Transferência Negativa)**	A recuperação vetorial densa por linguagem natural sofre de viés temático, exigindo enriquecimento por restrições numéricas.	Precision@3 e análise de distratores do Golden Test Set (README do TP2).
RQ_3 **(Data Anchoring & Self-Healing)**	Prompting ingênuo de LLM gera $< 60\%$ de esquemas válidos; com Self-Healing e JSON Schema, a conformidade atinge 100% com ≤ 2 tentativas.	Logs de execução do validador Pydantic no TP3.

Questão	Hipótese Investigada	Evidência nos TPs dos Alunos
RQ_4 **(Arquitetura MLOps Híbrida)**	Pipelines híbridos (triagem rápida via MLP + contextualização RAG + síntese LLM) otimizam o trade-off entre latência, custo e qualidade pedagógica.	Tabela comparativa final e Artigo Consolidado do TP3.

1.6.3 5.3. Tabela de Engenharia Consolidada para o Artigo

Componente	Latência P95	Custo/Req	F_1 Sintático	F_1 Conceitual / Relevância
TP1 (MLP Keras)	$\approx$ 2 ms	R$ 0,00	$82,4\%$ (`Tree`, `String`)	$34,1\%$ (`DP`, `Greedy`)
TP2 (RAG Spring)	$\approx$ 42 ms	R$ 0,0001	$P@3 = 88,0\%$	$P@3 = 58,0\%$ (viés de metáfora)
TP3 (Agente CoT)	$\approx$ 3.200 ms	R$ 0,012	$94,5\%$	$86,2\%$ (dedução lógica)

1.7 6. Roteiro de Consolidação Pelo Professor (Semana a Semana)

1. **Semana 4 (Pós-TP1):** Compilar a tabela de F1 por classe das 10 equipes a partir dos `README.md` e da avaliação cega do Docker.
2. **Semana 9 (Pós-TP2):** Rodar o script de teste cego no endpoint `/search` dos grupos. Tabular os resultados de Precision@3 por modelo de embedding.
3. **Semana 15 (Demoday / Pós-TP3):** Coletar os Artigos Consolidados e os YAMLs gerados. Rodar o script de auditoria de regras pedagógicas (Anti-Spoiler e Triggering).
4. **Encerramento:** Convidar os alunos destaque (melhores métricas e textos mais maduros) para compor a equipe oficial de coautoria do artigo final consolidado.

1.8 2. Manual dos Bastidores: A Fábrica Invisível de Pesquisa WOKDEX

Capítulo 2

Manual dos Bastidores: A Fábrica Invisível de Pesquisa WOKDEX

Guia Estratégico de Pesquisa & Publicação Científica — Inteligência Artificial II
Autor: Prof. Aléssio Miranda Júnior
Documento Estritamente Confidencial (Uso Exclusivo do Professor / Lead Researcher)

2.1 1. A Estratégia da "Fábrica Invisível" (*Crowdsourcing Didático*)

2.1.1 O Conceito

Para os 40 alunos da turma, eles estão simplesmente cursando a disciplina de IA II e entregando 3 Trabalhos Práticos (TP1, TP2, TP3) com código em Python/Java e 3 relatórios parciais (Draft 1, 2 e 3) para compor a nota.

Para você (o **Lead Researcher & Editor-Chefe**), as 10 equipes funcionam como **10 laboratórios independentes e descentralizados** executando um experimento massivo de ablação, benchmark e validação cruzada do ecossistema WOKDEX.

```
                    TURMA REGULAR (10 Equipes de 4 Alunos)
                                    ...
   Equipe 1          Equipe 2          Equipe 3                    Equipe 10
(Gemini 2.0)     (GPT-4o-mini)     (Llama 3 8B)                (DeepSeek V3)


  Draft 1 (NLP)    Draft 2 (RAG)    Draft 3 (Agente)             Artefatos YAML


          COMITÊ DE IC (2 Equipes de 4 Alunos)
        • IC-Alpha: Curadoria Golden Benchmark (100 problemas)
        • IC-Beta: Simulated Students (Auditoria dos 10 pipelines)


          3 ARTIGOS CIENTÍFICOS PUBLICADOS
        1. Artigo NLP (SBIE/CBIE)
        2. Artigo Pipeline v2.0 WOKDEX (RBIE/IEEE TLT)
        3. Artigo Simulated Students (AIED/ACM TOCE)
```

2.2 2. As 5 Questões e Hipóteses de Pesquisa (RQ_1 a RQ_5)

Toda a arquitetura dos TPs foi desenhada para responder a 5 questões científicas fundamentais:

2.2.1 RQ_1 (NLP / Representação Textual em Português)

- **Pergunta de Pesquisa:** *Como diferentes técnicas de representação vetorial (TF-IDF esparso vs. Embeddings Multilíngues vs. Embeddings Densos de LLM) impactam a classificação multi-label de tópicos algorítmicos em língua portuguesa?*
- **Hipótese H_1:** Embeddings densos (`sentence-transformers` e `text-embedding-3-small`) atingem F1-Macro superior ao TF-IDF (ΔF1 $\geq 10\%$), mas o TF-IDF com bigramas apresenta menor latência e custo computacional equivalente para classes majoritárias (`Array`, `String`).
- **Como os TPs coletam os dados:** O Aluno A/B de cada uma das 10 equipes gera a tabela comparativa no Draft 1, enquanto a equipe IC-Alpha executa a ablação formal de 4 modelos.

2.2.2 RQ_2 (Modelagem Multi-Label & Desbalanceamento Extremo)

- **Pergunta de Pesquisa:** *Qual o impacto da otimização de limiares de decisão dinâmicos (τ_i) e ponderação de perda (*Class Weights*) na sensibilidade de tópicos de cauda longa (ex.: `Backtracking` com 87 problemas vs. `Array` com 1.626)?*
- **Hipótese H_2:** A calibração de $\tau_i \in [0.2, 0.4]$ para classes minoritárias aumenta o Recall em $\geq 25\%$ sem degradar a precisão global, superando a heurística fixa de $\tau = 0.5$.
- **Como os TPs coletam os dados:** Cada grupo documenta no Draft 1 a curva de F1 por classe e a matriz de confusão nas 25 skills.

2.2.3 RQ_3 (Recuperação Semântica / RAG Educacional)

- **Pergunta de Pesquisa:** *A ancoragem de busca vetorial densa em exercícios similares em português reduz a alucinação de precondições e cenários de teste mal-formados gerados por LLMs?*
- **Hipótese H_3:** A injeção de contexto via RAG ($k = 3$ análogos recuperados pelo TP2) reduz em $\geq 50\%$ a geração de casos de teste inconsistentes comparada à geração zero-shot sem contexto.
- **Como os TPs coletam os dados:** O TP2 gera as métricas de Precision@3, MRR e latência P95 no banco vetorial.

2.2.4 RQ_4 (Mitigação de Alucinação Sintática & Data Anchoring)

- **Pergunta de Pesquisa:** *O uso de loops de reflexão e validação estrita com JSON Schema (*Data Anchoring* via Pydantic/Instructor) garante conformidade estrutural em pipelines de geração curricular em larga escala?*
- **Hipótese H_4:** Pipelines com Self-Healing atingem $\geq 95\%$ de conformidade sintática contra o `wok-problem.json`, enquanto prompting direto atinge $< 60\%$.
- **Como os TPs coletam os dados:** Os testes automatizados do TP3 e a auditoria do Schema Validator catalogam a taxa de erro de formato de cada grupo.

2.2.5 RQ_5 (Validação Pedagógica via Simulated Students)

- **Pergunta de Pesquisa:** *Agentes LLM configurados como estudantes novatos (*Agent Novice*) conseguem auditar a eficácia de armadilhas pedagógicas (*test cases*) com concordância estatística frente a especialistas humanos?*
- **Hipótese H_5:** O *Scenario Trap Rate* medido pelo Agent Novice correlaciona-se com a avaliação de professores humanos (coeficiente de Fleiss' $\kappa \geq 0.70$ e teste de McNemar pareado $p < 0.05$).
- **Como os TPs coletam os dados:** O TP3 Master (IC-Beta) roda o Agent Novice contra os YAMLs gerados por todas as 10 equipes da turma.

2.3 3. O Portfólio de 3 Artigos Científicos

A partir das entregas dos alunos, você estruturará **três publicações de alto impacto**:

Artigo	Título Provisório	Foco Principal	Veículo Alvo	Coautores
Artigo 1	*Classificação Semântica Multi-Label de Problemas de Programação em Língua Portuguesa: Um Benchmark com 2.830 Enunciados*	NLP em PT-BR, TF-IDF vs. Embeddings, Desbalanceamento das 25 classes	**SBIE / CBIE / RBIE** (Qualis A)	Prof. Aléssio + Alunos com melhores classificadores + IC-Alpha
Artigo 2	*Ecossistema WOKDEX: Geração Automatizada de Metadados Pedagógicos para Juízes Online via Agentes Autônomos com Ancoragem de Dados e Auto-Cura*	MLOps híbrido (FastAPI + Spring AI + LangGraph), ReAct, Self-Healing contra JSON Schema	**RBIE / IEEE TLT**	Prof. Aléssio + Grupo destaque do TP3 + IC-Alpha
Artigo 3	*Simulating Novice Students with Multi-Agent LLMs for Validating Formative Feedback in Automated Grading Systems*	Simulated Students (Agent Novice + Instructor), Validação sem comitê de ética humana, Ablation Study	**AIED / ACM TOCE** (Internacional)	Prof. Aléssio + Equipe IC-Beta

2.4 4. O "Algoritmo de Fusão" do Professor (Semana a Semana)

Ao longo do semestre, você executará os seguintes passos discretos:

2.4.1 Fase 1: Pós-TP1 (Semana 3–4) — Compilação do Artigo 1 (NLP)

1. **Auditoria Cega:** No dia da entrega do TP1, execute seu script de teste cego (`eval_tp1_leaderboard.py`) nos 10 contêineres Docker usando os 100 problemas curados pela IC.
2. **Ranking:** Gere a tabela de F1-Macro, F1-Micro e tempo de inferência dos 10 grupos.
3. **Fusão dos Textos:**
   - Pegue a seção de *Trabalhos Relacionados* dos 2 grupos mais bem escritos.
   - Pegue a tabela de ablação da equipe IC-Alpha.
   - Cruze com o gráfico de desbalanceamento gerado pelo seu script.
   - **Resultado:** O Artigo 1 está com 80% do texto e 100% dos dados prontos!

2.4.2 Fase 2: Pós-TP2 (Semana 9–10) — Consolidação do RAG

1. **Auditoria de Retrieval:** Rode as queries do Golden Test Set nos 10 bancos vetoriais.

2. **Catalogação:** Registre qual modelo de embedding em português obteve o maior Precision@3 e MRR.
3. **Insumo:** Esses dados comporão a seção de *Information Retrieval* do Artigo 2.

2.4.3 Fase 3: Demoday / Pós-TP3 (Semana 14–15) — O Gran Finale dos Artigos 2 e 3

1. **Coleta dos Artefatos:** Baixe todos os arquivos JSON/YAML gerados pelos 10 agentes da sala.
2. **Execução do Simulated Students (IC-Beta):** A equipe IC-Beta roda o *Agent Novice* para tentar falhar propositalmente em cada um dos cenários de teste gerados pela turma.
3. **Matriz de Cruzamento Final:**

Equipe	LLM Base	Schema Compliance	Scenario Trap Rate	Hint Relevance	F1 TP1	Precision@3 TP2
G01	Gemini 2.0 Flash	100%	82%	88%	76.4%	80.0%
G02	GPT-4o-mini	98%	85%	92%	78.1%	83.3%
G03	Llama 3 8B	88%	68%	74%	71.0%	70.0%
...	...	...	...	...	...	...

4. **Operação "Editor-Chefe":**
   - Reúna os 4 a 6 alunos que tiveram as melhores métricas individuais.
   - Apresente a eles a oportunidade de unificar os rascunhos nos 3 artigos oficiais.
   - Como o professor lidera a escrita final, os alunos entram com revisão e validação, garantindo um processo transparente, ético e extremamente gratificante para todos.

2.5 5. Matriz de Hipóteses vs. Evidências Empíricas

Hipótese	Onde está a evidência empírica no código?	Métrica Auditável
H_1 **(NLP PT-BR)**	Comparação de `vectorizer.pkl` (TF-IDF vs Sentence-Transformers) no TP1	F1-Score Macro nas 25 classes
H_2 **(Desbalanceamento)**	Curva ROC/PR e variação de τ no TP1	Recall em `Backtracking` / `Stack`
H_3 **(RAG Grounding)**	Logs do VectorDB e Precision@3 no TP2	Precision@3 no Golden Test Set
H_4 **(Self-Healing)**	Contador de tentativas de validação contra `wok-problem.json` no TP3	% de JSONs 100% conformes
H_5 **(Simulated Students)**	Execução do `test_novice.py` (IC-Beta / TP3 Master)	Scenario Trap Rate + Fleiss' κ

Conclusão: Este desenho pedagógico transforma a disciplina em um ecossistema produtivo e sustentável. Os alunos aprendem MLOps e IA de ponta com métricas reais, e o seu programa de pesquisa avança com dados experimentais rigorosos e publicações garantidas.

Capítulo 3

PARTE II — Visão Geral dos Trabalhos Práticos para os Alunos

Capítulo 4

Bem-vindos à Missão

Nesta disciplina, os Trabalhos Práticos não são exercícios isolados nem projetinhos descartáveis. Vocês vão construir, do zero, um **sistema real de Inteligência Artificial** que resolve um problema de pesquisa aberto: ajudar alunos novatos de programação a entenderem seus próprios erros.

> **i** Atualização da oferta 2026-2
>
> O percurso obrigatório atual é **TP1: classificação supervisionada**, **TP2: retrieval e recomendação híbrida de skills** e **TP3: integração TP1–TP2 com assistente RAG fundamentado**. O TP2 é entregue antes do Módulo 3; encontros posteriores apenas reutilizam seus contratos, índice e evidências para preparar o TP3, sem acrescentar requisitos retroativos ao TP2. Java/Spring AI é uma opção de adaptador, não requisito do TP2. Agentes, Tool Calling e Fine-Tuning foram deslocados para a trilha opcional TP3.5. As seções desta página que descrevem geração autônoma de YAML registram o contexto de pesquisa WOKDEX e não constituem requisito avaliativo do núcleo da oferta.
> Datas, pesos, escopo obrigatório e avaliação da oferta são definidos exclusivamente no Plano de Ensino e Cronograma 2026-2.

Se o sistema que a turma construir funcionar, os melhores rascunhos e códigos serão consolidados em um **Artigo Científico Real**, submetido a congressos nacionais e internacionais (como o CBIE/SBIE), com os melhores alunos figurando como **coautores**.

Antes de falar sobre código, vocês precisam entender **o problema** que estão resolvendo.

4.1 O Problema: O Silêncio Pedagógico

Você já participou ou ouviu falar de competições como a Maratona de Programação? Ou talvez já tenha se divertido (e passado um pouco de raiva) em alguma disciplina onde o professor Aléssio utilizou o `maratona.alessiojr.com` (baseado no DOMjudge)? Além desses, é bem provável que já tenha esbarrado em juízes online clássicos como o Beecrowd (antigo URI), Codeforces ou LeetCode. Se a resposta for sim para qualquer uma dessas opções, então você definitivamente já passou por isso:

1. Você escreve seu código com carinho.
2. Submete a solução.
3. O juiz responde: `Wrong Answer`.
4. Você fica olhando para a tela sem saber **o quê** errou.

Isso é o que chamamos de **Silêncio Pedagógico**: o juiz online sabe que você errou, mas não te diz *por quê*. Ele não diz se o seu erro é de lógica, de formatação, ou se a sua solução está correta mas é lenta demais. Ele simplesmente diz "errado" e te abandona.

Esse problema é grave:

- Pesquisas científicas mostram que **mais de 30% dos alunos reprovam** em disciplinas introdutórias de programação, em parte porque o feedback que recebem é insuficiente.
- O professor não consegue analisar o raciocínio de 100 alunos individualmente.
- Sem orientação, o aluno recorre a **tentativas aleatórias** (*guess-checking*) ou, pior ainda hoje em dia, simplesmente joga o problema em uma **LLM (como o ChatGPT)** e copia a resposta pronta. Como ele não acompanha criticamente o raciocínio, o aprendizado real não acontece e ele acaba travando e errando novamente no próximo obstáculo ou na prova.

4.2 A Solução: O WOKDEX

O **WOKDEX** (*World of Kode: Open Didactic Exchange*) é um modelo de metadados pedagógicos criado pelo professor desta disciplina como parte de sua tese de doutorado.

A ideia é simples e poderosa: em vez de deixar a inteligência pedagógica presa dentro de uma plataforma (como o Moodle ou o CodeRunner), o WOKDEX **embutiu essa inteligência diretamente dentro do exercício**, na forma de um arquivo `metadata.yaml`.

Esse arquivo YAML acompanha cada exercício de programação e diz ao sistema:

- Quais **habilidades** (*skills*) o exercício exige do aluno (ex: `condicionais`, `lacos`, `vetores`).
- Quais **testes** existem e qual é a **intenção pedagógica** de cada um.
- Se um teste específico foi criado para **detectar um erro conceitual comum** (*misconception*) que o aluno provavelmente está cometendo.
- Qual **dica formativa** (*helpTip*) deve ser exibida caso o aluno falhe naquele cenário.

Dessa forma, o juiz online deixa de ser um robô binário ("certo/errado") e passa a ser um **tutor assíncrono** que diagnostica o tipo de erro e orienta o aluno.

4.3 As 3 Camadas do YAML WOKDEX

O arquivo `metadata.yaml` de cada exercício é organizado em três camadas:

4.3.1 Camada 1: Descritiva (O que é este exercício?)

Informações gerais como título, nível curricular (CS1, CS2 ou CS3), dificuldade e complexidade de tempo/espaço.

```yaml
name: "Divisão"
difficultyLevelId: "D"
timeComplexity: "O(1)"
skills:
  - "matematica"
  - "io"
```

4.3.2 Camada 2: Pedagógica (O que o aluno precisa saber?)

Lista de habilidades (*skills*) exigidas e dicas formativas (*helpTip*) por cenário de teste. Se o aluno errar, ele recebe uma dica contextualizada, não um genérico "Wrong Answer".

```yaml
helpTip: "Erro na declaração de tipo de variável.
          Suas variáveis são inteiras, mas a divisão
          pode gerar resultado decimal."
skills:
  - { skill: matematica, points: 1 }
```

4.3.3 Camada 3: Avaliativa (Como o aluno é testado?)

Os cenários de teste (`testScenarios`), cada um com um **tipo pedagógico** que define *por que* aquele teste existe.

4.4 Os 4 Tipos de Teste do WOKDEX

Esta é a inovação central do modelo. Em vez de tratar todos os testes como iguais (como fazem os juízes tradicionais), o WOKDEX classifica cada cenário de teste:

Tipo	Visível?	O que ele faz?
`SAMPLE`	Sim	Testes dos exemplos do enunciado. O aluno pode usá-los para depurar antes de submeter.
`FUNCTIONAL`	Não	Testes secretos que verificam comportamentos específicos não revelados ao aluno.
`MISCONCEPTION`	Não	A "armadilha pedagógica". Testes projetados para **capturar erros conceituais comuns**. Se o aluno cai nessa armadilha, o WOKDEX sabe *exatamente* qual é o erro mental dele.
`PERFORMANCE`	Não	Testes que verificam a **eficiência** do algoritmo (tempo/memória), separando a questão "seu código está certo?" de "seu código é rápido?".

4.4.1 O caso mais interessante: MISCONCEPTION

Imagine um exercício que pede para dividir dois números e exibir o resultado com duas casas decimais. Um aluno que declara variáveis como `int` em vez de `double` vai obter:

- `10 / 2 = 5.00` → **Parece correto!** (mas é um acidente: a divisão é exata)
- `19 / 6 = 3.00` → **Deveria ser 3.17** (a divisão inteira truncou o resultado)

O juiz tradicional daria apenas "Wrong Answer" no segundo caso. O WOKDEX, ao contrário, reconhece que a saída `3.00` é exatamente a saída que um aluno com o erro de `int` vs `double` produziria. Ele então emite a dica: *"Erro na declaração de tipo de variável. Verifique se está usando double/float."*

4.5 Exemplo Real: O Exercício "Divisão"

Abaixo está um trecho real do `metadata.yaml` do exercício "Divisão" do corpus WOKDEX. Este é um exemplo de evidência pedagógica que os sistemas dos trabalhos recuperam, analisam e apresentam com proveniência.

```yaml
name: "Divisão"
difficultyLevelId: "D"
timeComplexity: "O(1)"
skills:
  - "matematica"
```

```yaml
  - "io"

testScenarios:
  - id: "d-sample"
    name: "Exemplos"
    level: "D"
    testType: SAMPLE            # Testes visíveis do enunciado
    helpTip: "Verifique os testes básicos do enunciado."
    skills:
      - { skill: io, points: 1 }
      - { skill: mathematics, points: 1 }

  - id: "c-falso-positivo-inteiros"
    name: "Inteiros - Falso Positivo"
    level: "C"
    testType: MISCONCEPTION      # Armadilha pedagógica!
    helpTip: "Erro na declaração de tipo de variável."
    targetLanguages: ["c", "cpp", "java"]
    skills:
      - { skill: mathematics, points: 1 }
```

[!NOTE] Observe: o cenário `d-sample` é do tipo `SAMPLE` (testes visíveis do enunciado). Já o cenário `c-falso-positivo-inteiros` é do tipo `MISCONCEPTION`: ele foi construído **especificamente** para capturar o aluno que usou `int` em vez de `double`. O campo `targetLanguages` restringe este cenário às linguagens onde a misconception se manifesta.

4.6 O Pipeline de 5 Fases (O Mapa do Semestre)

O segundo artigo da pesquisa (submetido à RBIE) propõe um **pipeline de I.A. Generativa em 5 fases** para gerar o `metadata.yaml` automaticamente. Cada TP que vocês farão corresponde a uma ou mais fases deste pipeline:

Fase	O que acontece	Qual TP cobre isso
1	**Leitura do enunciado:** O sistema recebe o texto bruto de um exercício de programação.	TP1
2	**Inferência de Skills:** Uma Rede Neural ou um LLM analisa o texto e prevê quais habilidades são exigidas (`condicionais`, `vetores`, etc.).	TP1
3	**Busca de exercícios similares (RAG):** O sistema busca no banco vetorial exercícios parecidos para usar como contexto, evitando que a I.A. "invente" coisas.	TP2

Fase	O que acontece	Qual TP cobre isso
4	**Recomendação híbrida de skills:** O sistema compara o classificador do TP1, as skills derivadas dos vizinhos recuperados e uma combinação calibrada.	TP2
5	**Resposta RAG fundamentada:** O assistente apresenta evidências, citações, limites e abstenção quando a recuperação não sustenta uma resposta.	TP3

É por isso que os TPs não são trabalhos soltos: **cada TP constrói um pedaço deste pipeline**, e no final do semestre o sistema integra classificação, retrieval, recomendação híbrida e resposta RAG fundamentada. A emissão autônoma de YAML não faz parte do núcleo avaliativo de 2026-2.

> [!IMPORTANT] **Recursos Disponíveis Desde o Dia 1:** O professor disponibilizará no repositório base dois artefatos essenciais: (1) O **Corpus Starter WOKDEX** — um conjunto curado de 50–100 exercícios de programação já anotados com skills e cenários de teste, pronto para treino e indexação; e (2) O **JSON Schema oficial** (`wok-scheme.json`) — o contrato formal que define todos os campos, enums e restrições do `metadata.yaml`. Esses dois arquivos são o alicerce de **todos** os TPs.

4.7 Como as Equipes Funcionam? (Pareamento e Colaboração)

Para simular o ambiente de uma verdadeira *Start-up* de Inteligência Artificial, os grupos serão formados por **exatos 4 alunos**, organizados em **2 Duplas Operacionais**.

A metodologia de trabalho adota o conceito de **Programação em Par (*Pair Programming*) e Desenvolvimento Baseado em Contratos**, garantindo que ninguém fique bloqueado esperando o outro terminar:

- **Nível 1 - O Grupo (4 alunos):** É a entidade final. O grupo todo é responsável pela integração fim a fim dos microsserviços, pelo deploy no Docker e pelo **Artigo Científico Consolidado no TP3**.
- **Nível 2 - As Duplas Operacionais (2 alunos por pilar):**
  - **Dupla 1 (Dados & Inteligência Artificial):** Aluno A e Aluno B trabalham em regime de pareamento nos pipelines de NLP, vetorização e modelagem da Rede Neural.
  - **Dupla 2 (Engenharia de Software & MLOps):** Aluno C e Aluno D trabalham em regime de pareamento no desenvolvimento da API REST (FastAPI, Spring ou outra stack documentada), conteinerização Docker, testes automatizados e CI/CD.
- **Nível 3 - O Indivíduo (Você):** Dentro da sua dupla, cada aluno assume protagonismo em tarefas complementares, mas com **código compartilhado, revisado e construído em conjunto**.

> [!IMPORTANT] **Como Evitar Gargalos e Bloqueios (Trabalho em Paralelo via *Mocks*):**
> **Nenhum aluno deve ficar parado esperando o colega terminar!**
> A Dupla 2 (Backend/Docker) **não precisa** esperar a Dupla 1 treinar o modelo final para começar a programar. No primeiro dia, o grupo define o contrato Pydantic/DTO (`PredictRequest` e `PredictResponse`). A Dupla 2 utiliza um **modelo fictício (*mock/stub*)** que retorna dados fixos para construir toda a API, o `Dockerfile` e a suite de `pytest` imediatamente. Quando a Dupla 1 exporta o `modelo_skills.keras`, basta plugar o arquivo real.

[!TIP] **Pareamento (*Pair Programming*) e Ajuda Mútua:**
Apesar da distribuição de papéis, **a equipe é uma só e deve se ajudar mutuamente**. O pareamento na dupla é altamente incentivado: commits conjuntos, *code reviews* e coautoria em branches da sprint são práticas recomendadas da indústria. Se um colega tiver dificuldades, ajude-o a destravar.

4.8 Regras de Auditoria no GitLab (Como comprovar sua parte)

Para que a avaliação seja justa e transparente para todos os membros da equipe, o repositório deve seguir três boas práticas:

1. **O Manifesto de Autoria (`README.md`):** Na raiz do repositório, inclua a tabela indicando a composição das duplas, os papéis e os logins `@username` do GitLab.
2. **Branches de Feature e Pareamento:** Trabalhem em branches organizadas (ex.: `feat/dupla1-dados-nlp`, `feat/dupla2-fastapi-docker`). Commits de ambos os membros da dupla na branch são esperados e valorizados.
3. **Merge Requests (MRs) com Revisão:** Quando a funcionalidade da dupla estiver pronta, abram um *Merge Request* para a `main` com descrição clara e revisão do código pelo parceiro da dupla antes do merge.

4.9 Entregas e Avaliação

Consulte o Plano de Ensino e Cronograma 2026-2 para os prazos, pesos, escopo obrigatório e avaliação dos TPs. As páginas específicas dos trabalhos detalham somente a implementação e as evidências técnicas esperadas.

4.9.1 A Balança da Nota (Como você será avaliado)

A nota de **cada TP** é sempre dividida em duas metades exatas:

- **50% - Entrega em Grupo (Integração e Engenharia):**
  - No **TP1 e TP2:** O grupo entrega o microsserviço funcional em Docker, a suite de testes automatizados (`pytest` / integração) e o **Relatório Técnico Executivo no `README.md`** (com tabelas de métricas e gráficos). Não há exigência de formatar artigo científico nessas duas primeiras fases.
  - No **TP3 (Dossiê de Integração):** O grupo integra TP1 e TP2, defende a solução e conclui o **Artigo Científico SBC (4 a 6 páginas)**, reunindo método, métricas, evidências, limitações e resultados de classificação, retrieval e RAG.
- **50% - Entrega Individual (Engenharia):** A qualidade do *seu* código. Você usou as bibliotecas certas? O código está limpo? O seu módulo faz o que deveria fazer? A nota aqui é sua.

4.10 Trilha IC: Equipes de Pesquisa (Voluntário)

Além do caminho regular (TP1→TP2→TP3), esta disciplina oferece uma **trilha de Iniciação Científica** para equipes que querem ir além.

Como funciona?

- Todas as 10 equipes fazem o **mesmo TP1 e TP2** (pipeline WOKDEX focado em engenharia).
- No **TP3**, as equipes regulares integram TP1 e TP2 em um assistente RAG fundamentado e redigem o Artigo Científico Consolidado.
- As equipes IC fazem o **TP3 Master**: em vez do TP3 regular, auditam evidências, citações, abstenções e reprodutibilidade dos artefatos TP2–TP3.

O que muda nos TP1 e TP2?

Quase nada. As equipes IC fazem o **mesmo trabalho** que todas, com um pequeno **adendo opcional** (marcado com nos documentos de cada TP) que as prepara para o TP3 Master. Se a equipe desistir da trilha IC no meio do caminho, o adendo já feito **conta como bônus** na nota regular.

O que a equipe IC ganha?

- **Coautoria** em artigo científico real (SBIE 2027 / AIED 2027)
- Acesso a **GPU na nuvem** (AWS com crédito de pesquisa)
- Carta de recomendação do professor

Como se candidatar?

A candidatura é **voluntária e aberta** desde o primeiro dia. A equipe inteira (4 alunos) se candidata preenchendo um formulário rápido. O professor avalia:

1. A equipe tem **disponibilidade extra** (~4h/semana além da disciplina)?
2. Todos os membros concordam com o compromisso?
3. Há interesse genuíno em pesquisa?

   [!TIP] **Conselho do professor:** Se vocês estão empolgados, **façam um TP1 muito bem feito primeiro**. A excelência no TP1 é o melhor indicador de que a equipe está pronta para a trilha IC. Não adianta querer correr se o alicerce não está sólido. O TP1 é a fundação de tudo.

Se mais de 3 equipes se candidatarem (ótimo!), todas serão aceitas — quanto mais dados experimentais, melhor para a pesquisa. Os detalhes completos estão no **Programa IC Especial**.

4.10.1 A Jornada Completa

```
TP1 (todas)          TP2 (todas)          TP3
  Classificação        Retrieval +          Regular
   supervisionada       recomendação            Assistente RAG fundamentado
                        híbrida de skills        + Artigo SBC
  Artefato TP1         Artefatos TP2        Master (equipes IC aceitas)
   versionado           versionados              Auditoria de evidências,
                                                   citações e reprodução
```

4.10.2 O Mapa de Integração do Semestre

4.11 Acesse os Detalhes de Cada Entrega

Navegue abaixo para entender as regras, o escopo técnico e o papel individual de cada aluno dentro dos três grandes ciclos da disciplina:

4.11.1 Trilha Regular (Todas as Equipes)

- **Acompanhamento das Equipes e Repositórios (TP1)**
- **TP1: O Classificador de Habilidades (Redes Neurais e MLOps)**
- **TP2: Retrieval e Recomendação Híbrida de Skills**
- **TP3: Dossiê de Integração TP1–TP2 e Assistente RAG**
- **TP3.5: Trilha Avançada de Agentes e LLMs**

4.11.2 Trilha IC (Equipes Voluntárias)

- **Programa IC Especial — Visão Geral**
- **TP3 Master: Simulated Students**

4.11.3 Recursos e Documentação Integrada

- **Manual Integrado Consolidado (Visão Geral em Documento Único)**
- **Editorial Docente da Revisão dos TPs 2026-2**
- **Recursos WOKDEX (JSON Schema, Artigo SBIE e Especificações)**
- **Relatório Exploratório do Dataset (Gráficos e Estatísticas - HTML)**
- **Relatório Exploratório do Dataset (Versão PDF)**
- **Download Dataset CSV PT-BR (23.3 MB)**
- **Download Dataset JSON PT-BR (25.4 MB)**

Capítulo 5

PARTE III — Especificações Técnicas das Entregas Regulares

5.1 1. TP1: O Classificador de Habilidades (Redes Neurais, NLP & MLOps)

5.2 Por que este TP existe?

Volte ao `metadata.yaml` do exercício "Divisão" que você viu na página de Visão Geral. Repare no campo `skills`:

```yaml
skills:
  - "matematica"
  - "io"
```

Esses rótulos dizem quais habilidades cognitivas um aluno precisa dominar para resolver aquele exercício. Hoje, **quem classifica essas skills é o professor, manualmente**. Isso é lento, subjetivo e não escala para centenas de exercícios.

A missão do TP1 é construir uma **Rede Neural Clássica** que leia o texto bruto de um enunciado de programação e **preveja automaticamente** quais skills ele exige. Esse modelo será o motor das **Fases 1 e 2** do Pipeline WOKDEX.

> Repositórios e Composição das Equipes
>
> O acompanhamento dos grupos registrados no GitLab, links diretos dos repositórios e a listagem de alunos com sinalizadores de status estão disponíveis em **Grupos do TP1 (2026-2)**.

5.3 Leitura Obrigatória (Fundamentação)

Antes de programar, vocês precisam ler e entender estes dois artigos científicos. Eles serão a base do Draft 1 na seção de *Trabalhos Relacionados*.

Artigo 1: Kim, Y. (2014). ***"Convolutional Neural Networks for Sentence Classification".*** **arXiv:1408.588**
Por que ler: É exatamente o que vocês vão construir — uma rede neural que recebe uma frase e classifica em categorias. O paper tem apenas 6 páginas e mostra como vetorizações de texto (Word2Vec, TF-IDF) alimentam um classificador profundo. A arquitetura descrita é a referência mais citada do mundo para classificação de texto com Deep Learning.

Artigo 2: Feng, Z. et al. (2020). ***"CodeBERT: A Pre-Trained Model for Programming and Natural Lang***
Por que ler: Enquanto o artigo anterior trata de classificação de textos genéricos, este mostra que

existe um campo de pesquisa inteiro dedicado a aplicar NLP especificamente em **código-fonte e enunciados de programação**. Vocês não vão usar o CodeBERT em si (ele é pesado demais para o escopo do TP), mas entender que o problema que vocês estão atacando já é estudado por grandes labs (Microsoft Research) dá lastro científico ao Draft 1.

5.4 O que é uma Skill no WOKDEX e no Dataset?

As habilidades (*skills*) representam os conceitos algorítmicos e estruturas de dados necessários para resolver um exercício.

Para o TP1, trabalharemos com as **25 Skills/Tópicos mais representativos da amostra** (que cobrem 94% do volume de problemas e incluem desde estruturas básicas até paradigmas avançados como Grafos e Backtracking):

#	Skill / Tópico	Ocorrências no Dataset	Exemplo Típico
1	`Array` (Vetores)	1.626	"Encontre o par de elementos que soma o alvo"
2	`String` (Textos)	683	"Verifique se a palavra é um palíndromo"
3	`Hash Table` (Tabela Hash / Dicionário)	589	"Conte a frequência de cada caractere em $O(N)$"
4	`Dynamic Programming` (Programação Dinâmica)	508	"Calcule o número de formas de subir uma escada"
5	`Math` (Matemática)	503	"Converta números romanos para inteiros"
6	`Sorting` (Ordenação)	391	"Ordene o array por frequência decrescente"
7	`Greedy` (Algoritmos Gulosos)	366	"Distribua doces minimizando o total"
8	`Binary Search` (Busca Binária)	259	"Encontre a raiz quadrada inteira em $O(\log N)$"
9	`Depth-First Search` (Busca em Profundidade / DFS)	242	"Percorra todos os caminhos válidos da árvore"
10	`Matrix` (Matrizes / Arrays 2D)	216	"Rotacione uma imagem $N \times N$ em 90 graus"
11	`Bit Manipulation` (Manipulação de Bits)	212	"Conte o número de bits 1 na representação binária"
12	`Breadth-First Search` (Busca em Largura / BFS)	193	"Encontre o menor caminho no grafo"
13	`Prefix Sum` (Soma de Prefixos)	184	"Calcule a soma de um intervalo em $O(1)$"
14	`Tree` (Árvores)	182	"Calcule a profundidade máxima da árvore"

#	Skill / Tópico	Ocorrências no Dataset	Exemplo Típico
15	`Two Pointers` (Dois Ponteiros)	181	"Remova duplicatas de um vetor ordenado in-place"
16	`Simulation` (Simulação)	165	"Simule o movimento de um robô no grid"
17	`Heap (Priority Queue)` (Heap / Fila de Prioridade)	164	"Encontre o K-ésimo maior elemento"
18	`Counting` (Contagem de Elementos)	145	"Conte elementos com frequências específicas"
19	`Stack` (Pilhas)	138	"Valide se parênteses e colchetes estão balanceados"
20	`Graph` (Grafos)	133	"Detecte ciclos ou caminhos em grafos direcionados"
21	`Sliding Window` (Janela Deslizante)	130	"Maior substring contígua sem caracteres repetidos"
22	`Binary Tree` (Árvore Binária)	129	"Inverta ou reconstrua uma árvore binária"
23	`Enumeration` (Enumeração)	112	"Gere todas as combinações válidas de 3 dígitos"
24	`Design` (Projeto de Estruturas)	94	"Implemente um LRU Cache ou Fila usando Pilhas"
25	`Backtracking` (Retrocesso / Busca Exaustiva)	87	"Resolva o problema das N Rainhas ou Sudoku"

[!IMPORTANT] **Formulação do Problema: Classificação Multi-Label**
Um mesmo problema pode exigir **múltiplas skills simultâneas** (por exemplo, um exercício pode ter rótulos `['Array', 'Sorting', 'Two Pointers']`).
Portanto, sua Rede Neural **NÃO** deve usar Softmax na saída!
Utilize: 1. Camada de saída com **25 neurônios** e função de ativação `sigmoid`. 2. Função de perda (**loss**): `binary_crossentropy`. 3. Binarização dos alvos: `MultiLabelBinarizer` da biblioteca `scikit-learn`.

5.5 O Dataset Oficial (`leetcode_problems_pt`)

A base oficial de dados para o TP1 é o **LeetCode Problems Dataset Bilíngue**, disponibilizado em formato tabular (CSV) e estruturado (JSON), com 2.830 enunciados completos em Português (PT-BR).

Links para Acesso e Download

- **Visualizar Relatório Exploratório Interativo (HTML)** *(Distribuições de tamanho de texto, frequências e estatísticas)*
- **Download do Relatório Exploratório (PDF)** *(Versão consolidada para leitura e impressão)*
- **Download do Dataset em CSV (23.3 MB)**
- **Download do Dataset em JSON (25.4 MB)**

- **Documentação Técnica & Especificação de Schemas**

5.5.1 Snippet de Carga Rápida (Python / Pandas)

O Aluno A (Engenheiro de Dados) pode iniciar o pipeline com o seguinte fluxo:

```
import pandas as pd
import ast
from sklearn.preprocessing import MultiLabelBinarizer

# 1. Carregar dataset oficial
df = pd.read_csv("trabalho/database/leetcode_problems_pt.csv")

# 2. Selecionar features de entrada e target
# X: Texto do enunciado em Português (título + descrição completa contendo exemplos e restrições)
X_text = (df["title_pt"].fillna("").astype(str) + "\n\n" + df["description_pt"].fillna("").astype(st

# 3. Tratar a coluna de tópicos (que vem como string de lista)
def parse_topics(val):
    if pd.isna(val): return []
    try: return ast.literal_eval(val) if isinstance(val, str) else val
    except: return []

df["topics_list"] = df["topics"].apply(parse_topics)

# 4. Filtrar para as Top 25 Skills
TOP_25_SKILLS = [
    "Array", "String", "Hash Table", "Dynamic Programming", "Math",
    "Sorting", "Greedy", "Binary Search", "Depth-First Search", "Matrix",
    "Bit Manipulation", "Breadth-First Search", "Prefix Sum", "Tree", "Two Pointers",
    "Simulation", "Heap (Priority Queue)", "Counting", "Stack", "Graph",
    "Sliding Window", "Binary Tree", "Enumeration", "Design", "Backtracking"
]

df["target_skills"] = df["topics_list"].apply(
    lambda topics: [t for t in topics if t in TOP_25_SKILLS]
)

# 5. Binarizar alvos para Multi-Label (Y terá shape: [N, 25])
mlb = MultiLabelBinarizer(classes=TOP_25_SKILLS)
Y = mlb.fit_transform(df["target_skills"])
print("Total de classes mapeadas:", len(mlb.classes_))
print(f"Shape final de Y: {Y.shape}")
```

Datas, peso, escopo obrigatório e avaliação normativos constam do Plano de Ensino e Cronograma 2026-2; esta página descreve as evidências técnicas do projeto.

> [!IMPORTANT] **Sobre o timing do NLP:** A vetorização de texto (TF-IDF, Word2Vec) será coberta no módulo de Embeddings. Vocês podem começar a construir toda a infraestrutura (Keras, FastAPI, Docker, testes) usando o conhecimento dos materiais iniciais enquanto aguardam essa unidade para fechar o pipeline de dados do Aluno A. Consultem o plano para a sequência de encontros e prazos.

> [!TIP] **Baseline Obrigatório:** No Draft 1, o grupo **deve** comparar a performance da Rede Neural contra um baseline simples (ex: TF-IDF + Logistic Regression / LinearSVC com `OneVsRestClassifier`). A tabela comparativa (Precision, Recall, F1-Macro, tempo de treino) é item obrigatório da avaliação.

5.6 A Divisão de Tarefas na Equipe (4 Alunos em 2 Duplas)

Para garantir que o trabalho seja executado de forma fluida ao longo dos **20 dias**, a carga horária estimada é de **10 a 15 horas de dedicação individual por aluno** (~3 a 4 horas por semana). A equipe divide-se em duas duplas complementares:

```
                    EQUIPE DE 4 ALUNOS

  DUPLA 1: DADOS & ML        DUPLA 2: ENGENHARIA
(O Cérebro Matemático)      (A Casca MLOps / Docker)

Aluno A: Dados & NLP         Aluno C: Backend FastAPI
Aluno B: Redes Neurais       Aluno D: DevOps & Pytest
```

5.6.1 Dupla 1: Ciência e Dados (O Cérebro) — ~10 a 15h por aluno

Esta dupla é focada no pré-processamento de linguagem natural (NLP) em português e no treinamento da Rede Neural. O *output* final da dupla é o pipeline serializado (`vectorizer.pkl` / `mlb.pkl`), o modelo treinado (`modelo_skills.keras`) e as métricas de convergência.

- **Aluno A (Engenheiro de Dados):**
  - *Dedicação:* ~10–12 horas.
  - *Tarefas:* Carregar `leetcode_problems_pt.csv`, limpar os enunciados em português (`description_pt`), tratar as 25 skills com `MultiLabelBinarizer`, criar e serializar a vetorização de texto (ex: `TfidfVectorizer(max_features=5000, ngram_range=(1,2))`), e gerar a divisão estratificada Treino/Validação/Teste (80/10/10).
- **Aluno B (Engenheiro de Machine Learning):**
  - *Dedicação:* ~12–15 horas.
  - *Tarefas:* Construir e treinar o **Baseline** (ex: `LogisticRegression` / `LinearSVC` OvR), construir a **Rede Neural Keras** (camadas Densas, Dropout 0.3-0.5, saída 25 neurônios Sigmoid + `binary_crossentropy`), calibrar o limiar de decisão (τ), gerar gráficos de loss/épocas e calcular o **F1-Score Macro e Micro**.

5.6.2 Dupla 2: Engenharia de Software e MLOps (A Casca) — ~10 a 15h por aluno

Esta dupla pega o modelo exportado pela Dupla 1 e constrói um microsserviço moderno, conteinerizado e testado, pronto para a integração e comparação com o retrieval nos TP2 e TP3.

- **Aluno C (Dev Backend Python / FastAPI):**
  - *Dedicação:* ~10–12 horas.
  - *Tarefas:* Criar a API **FastAPI**, definir contratos estritos de entrada e saída com **Pydantic** (`PredictRequest`, `PredictResponse`), carregar os artefatos (`modelo_skills.keras` e `vectorizer.pkl`) na inicialização da aplicação (usando `lifespan`), e implementar os endpoints `POST /predict` e `GET /health`.
- **Aluno D (DevOps, QA e Docker):**
  - *Dedicação:* ~10–12 horas.
  - *Tarefas:* Escrever o `Dockerfile` otimizado e `docker-compose.yml`, configurar a suite de testes automatizados com `pytest` (mínimo 5 testes), configurar a pipeline de CI no GitLab (`.gitlab-ci.yml`), e garantir que a API suba limpa em qualquer máquina com um único comando `docker run`.

[!TIP] **Dinâmica de Pareamento (*Pair Programming*) e Desenvolvimento Não-Bloqueante:**
A Dupla 2 **não deve esperar** a Dupla 1 concluir o treinamento da rede neural para começar a trabalhar. Usem a abordagem de **Contrato Primeiro (*API-First / Contract-Driven*)**: 1. No primeiro dia, a equipe alinha os modelos Pydantic (`PredictRequest` e `PredictResponse`). 2. A Dupla 2 cria um modelo simulado (*mock*) que devolve predições estáticas e constrói toda a API FastAPI, o `Dockerfile` e a suite de `pytest` em paralelo. 3. Os membros de cada dupla

devem praticar pareamento contínuo, code reviews e ajuda mútua para manter o fluxo de entrega sem gargalos.

5.7 A API que você vai construir

O produto final de engenharia do TP1 é uma **API REST** conteinerizada. Abaixo está o contrato exato:

5.7.1 Requisição: `POST /predict`

```
{
  "enunciado": "Dada uma lista de inteiros nums e um inteiro target, retorne os índices dos dois núm
}
```

5.7.2 Resposta esperada:

```
{
  "skills": [
    { "skill": "Array", "confidence": 0.94 },
    { "skill": "Hash Table", "confidence": 0.88 },
    { "skill": "Two Pointers", "confidence": 0.72 }
  ],
  "model_version": "v1.0",
  "input_length": 132
}
```

[!IMPORTANT] No TP2 e no TP3, esta API fornece a evidência supervisionada que será comparada à recomendação por retrieval e exibida no assistente RAG. Se a API não estiver de pé e respondendo neste formato, a integração e os testes de contrato não poderão ser executados. É por isso que o Docker é obrigatório.

5.8 Relatório Técnico no `README.md` (Documentação do TP1)

O TP1 inicia o artigo SBC incremental: a equipe cria a estrutura do documento e registra problema, dataset, método supervisionado, métricas e limitações. O foco principal continua sendo Engenharia, Código e Métricas Técnicas; as seções serão ampliadas no TP2 e consolidadas no TP3.

O grupo deve documentar de forma clara e estruturada no próprio `README.md` do repositório GitLab (que servirá de base direta para o Artigo Final no TP3):

1. **Trabalhos Relacionados e Fundamentação:**
   - Contextualizar brevemente os artigos base (Kim 2014 para CNN/NLP e CodeBERT para representação de código).
2. **Metodologia Preditiva (Dupla 1):**
   - **Justificativa da Formulação Multi-Label:** Explicar matematicamente por que a saída usa **Sigmoid independente por neurônio + `binary_crossentropy`** em vez de Softmax.
   - **Arquitetura da Rede:** Tabela com as camadas da rede neural, número de neurônios, funções de ativação, taxa de Dropout e otimizador.
3. **Resultados e Tabela Comparativa Obrigatória (Dupla 1 e 2):**
   - Tabela comparando o **Baseline** (TF-IDF + Logistic Regression / SVM) contra a **Rede Neural (Keras)**:

Modelo	Precision (Macro)	Recall (Macro)	F1-Score (Macro)	F1-Score (Micro)	Tempo de Treino
Baseline (TF-IDF + Regressão Logística OvR)	...	...	...	...	~2s
Rede Neural Keras (Dense + Dropout)	...	...	...	...	~45s

4. **Análise de Desbalanceamento, Threshold τ e o Abismo Semântico (Dupla 1):**
   - Gráfico de convergência (*Loss* de Treino vs. Validação).
   - **Discussão do Abismo Semântico (*Semantic Gap*):** Analisar criticamente quais das 25 classes foram mais fáceis/difíceis. Por que classes sintáticas como `String` ou `Tree` apresentam F1 mais elevado do que paradigmas abstratos como `Dynamic Programming` ou `Greedy`?
   - Efeito do limiar de decisão τ dinâmico no F1-Score das classes minoritárias (ex: `Backtracking`).
5. **Engenharia e Arquitetura MLOps (Dupla 2):**
   - Diagrama ou descrição da conteinerização Docker, latência média da rota `POST /predict` (em milissegundos) e testes unitários.

5.9 Sugestões de Testes Automatizados Obrigatórios (`pytest`)

O Aluno D (DevOps) deve implementar, no mínimo, a seguinte suite de testes com `pytest` (arquivo `tests/test_api.py`):

#	Função de Teste	O que deve verificar
1	`test_api_health`	A rota `GET /health` responde HTTP 200 e status `"ok"`
2	`test_predict_valid_input`	`POST /predict` com enunciado válido retorna HTTP 200 e campos `skills`, `model_version`, `input_length`
3	`test_skills_in_top25`	Todas as skills retornadas na lista pertencem estritamente às 25 classes válidas
4	`test_confidence_range`	O valor de `confidence` de cada skill prevista está rigorosamente no intervalo $[0.0, 1.0]$
5	`test_empty_input_validation`	Enunciado vazio (`""` ou whitespace) retorna HTTP 422 (validação Pydantic)
6	`test_predict_deterministic`	Duas chamadas com o mesmo texto produzem as mesmas predições

---

5.10 Rubrica de Avaliação Detalhada (8 Pontos)

5.10.1 Nota Individual (4 pts)

Critério	Pontos	Detalhes
Commits individuais na branch correta	1	Avaliado via Git Blame (evidência de autoria)
Entrega técnica individual funcionando	2	Pipeline do Aluno A OU Rede do Aluno B OU FastAPI do Aluno C OU Docker/Testes do Aluno D
Qualidade e organização do código	1	Boas práticas, docstrings, código modular e limpo

5.10.2 Nota do Grupo (4 pts)

Critério	Pontos	Detalhes
API integrada responde no formato correto	1	`docker run` + `curl POST /predict` funcional
Performance da Rede Neural (F1-Score Macro)	1	Veja escala progressiva calibrada abaixo
Relatório Técnico no README.md completo	2	Tabela comparativa baseline vs. rede, matriz de confusão, discussão do abismo semântico

5.10.3 Escala Progressiva de Performance (F1-Score Macro)

Nível	F1-Score Macro	Pontuação	Critério de Engenharia
Mínimo	$\geq 35\%$	0.3 pt	Supera o baseline ingênuo / classe majoritária
Bom	$\geq 50\%$	0.7 pt	Rede Keras regularizada (Dropout, limiar τ calibrado)
Excelente	$\geq 65\%$	1.0 pt	Otimização avançada de hiperparâmetros ou análise exemplar de erro

> [!NOTE] **Por que a régua de F1-Macro é calibrada nesta faixa?**
> Lembre-se do **Abismo Semântico (*Semantic Gap*)**: os enunciados usam metáforas narrativas (*"Alice divide doces"*) e a técnica algorítmica (*DP, Guloso*) é uma abstração matemática implícita. Um F1-Macro de 50–65% em 25 classes multi-label desbalanceadas é um resultado de engenharia sólido e realista. A profundidade da sua análise crítica no `README.md` tem peso decisivo na pontuação do grupo.

5.11 Cronograma Sugerido (4 Semanas)

Para não acumular trabalho e perder a data de entrega, sugerimos o seguinte ritmo para a equipe:

Período	Foco da Equipe	Meta da Semana
Semana 1	Pesquisa e Setup	Leitura do artigo (Kim 2014), divisão de papéis, setup do repositório GitLab e script base de TF-IDF.
Semana 2	Modelagem	Treinamento do baseline (SVM/LR) e primeira versão da Rede Neural (Keras).
Semana 3	Engenharia MLOps	Otimização do F1-Score, criação da API em FastAPI e criação do `Dockerfile`.
Semana 4	Qualidade e README	Escrita dos testes `pytest`, validação final, preenchimento do relatório no `README.md` e Merge Request final.

5.12 Checklist de Entrega — TP1

- ☐ Repositório GitLab com branches individuais (`feat/aluno-a-*`, `feat/aluno-b-*`, etc.)
- ☐ `README.md` na raiz com tabela de autoria (nome, papel, `@username`)
- ☐ `docker build` e `docker run` funcionais (sem erros)
- ☐ `POST /predict` retorna JSON no formato especificado
- ☐ 6 testes `pytest` passando (ver seção de Testes Obrigatórios)
- ☐ F1-Score macro 35% no dataset de teste do professor
- ☐ Relatório Técnico no `README.md` contendo:
  - ☐ Trabalhos Relacionados (citando Kim 2014 e CodeBERT)
  - ☐ Tabela comparativa: baseline (TF-IDF + SVM/LR) vs Rede Neural
  - ☐ Matriz de confusão / F1 por classe nas 25 skills
  - ☐ Curva de loss (treino vs validação)
  - ☐ Justificativa da formulação Multi-Label com Sigmoid
  - ☐ Discussão crítica sobre o Abismo Semântico nas classes conceituais

---

5.13 Adendo IC — Golden Benchmark e Estudo de Ablação Textual (Opcional)

Esta seção é exclusiva para equipes da Trilha de Iniciação Científica (IC). As equipes regulares podem ignorá-la. Se uma equipe IC desistir da trilha, o adendo já executado conta como **+1 ponto bônus** na nota regular do TP1.

Enquanto as equipes regulares treinam seus classificadores na divisão padrão, as equipes IC atuam como o **Comitê Científico de Validação** do ecossistema WOKDEX, conduzindo experimentos aprofundados que fundamentarão os artigos da disciplina.

5.13.1 O que fazer

1. **Curadoria do Golden Benchmark Set (100 Problemas):**
   - Selecionar e auditar minuciosamente uma amostra estratificada de **100 problemas** do dataset `leetcode_problems_pt.csv`, cobrindo as 25 skills e os níveis `Easy`, `Medium` e `Hard`.
   - Verificar a fidelidade do enunciado em português e validar a consistência das skills anotadas.
   - Este conjunto será o **teste cego do professor** para avaliar as APIs de todas as equipes no Demoday.

2. **Estudo de Ablação Textual (TF-IDF vs. Embeddings em Português):**
   - Comparar o desempenho da Rede Neural sob diferentes representações vetoriais de texto em PT-BR:
     1. `TF-IDF` esparso (unigramas + bigramas).
     2. `paraphrase-multilingual-MiniLM-L12-v2` (Sentence-Transformers).
     3. `BERTimbau` (`neuralmind/bert-base-portuguese-cased`).
     4. `text-embedding-3-small` (OpenAI).
   - Tabular o F1-Macro e a latência de inferência de cada abordagem.
3. **Análise de Sensibilidade ao Desbalanceamento:**
   - Testar o impacto de *Class Weights* e limiares de decisão dinâmicos (τ_i) nas classes com menor suporte estatístico (ex: `Backtracking` com 87 amostras vs. `Array` com 1.626).

5.13.2 Entregáveis IC-TP1

☐ Arquivo `golden_benchmark_100.csv` catalogado e revisado.
☐ Draft 1 Estendido (3–4 páginas formato SBC) contendo a tabela comparativa de ablação vetorial e o estudo de desbalanceamento.

[!TIP] Os dados gerados neste adendo comporão diretamente a seção experimental do **Artigo 1 da Disciplina** (Classificação Semântica de Enunciados em Língua Portuguesa para Educação em Computação).

5.14 2. TP2: O Oráculo RAG (Bancos Vetoriais, Busca Semântica & Spring AI)

5.15 Por que este TP existe?

No TP1, cada equipe treinou um preditor supervisionado que recebe o enunciado de um exercício e estima suas *skills*. Essa é uma fonte útil de recomendação, mas não é a única: também é possível recuperar exercícios semanticamente próximos e aproveitar as *skills* anotadas nesses vizinhos. O TP2 investiga, com uma infraestrutura única e avaliável, quando cada estratégia ajuda, falha ou discorda das demais.

O produto não é um sistema que promete ser superior em todos os cenários. É uma comparação empírica, honesta e reproduzível entre três recomendadores de *skills* para o WOKDEX:

1. **Supervisionado (TP1):** probabilidades multi-label produzidas pelo modelo do TP1.
2. **Retrieval por embeddings:** *skills* inferidas a partir dos exercícios recuperados por similaridade semântica.
3. **Híbrido ajustado na validação:** combinação dos dois escores, com pesos e limiares escolhidos sem consultar o conjunto de teste.

Cada equipe tem quatro estudantes, cerca de **15 horas por estudante**, totalizando **60 horas de equipe**. O prazo de entrega do TP2 é **19/10/2026**. Datas, peso, escopo obrigatório e avaliação normativos constam do Plano de Ensino e Cronograma 2026-2; esta página descreve as evidências técnicas do projeto.

5.16 Start Pack e checkpoint técnico

Os marcos abaixo são de acompanhamento técnico e não têm nota própria:

Data	Marco	Expectativa
30/09/2026	Start Pack	Registrar papéis, corpus e proveniência, divisão inicial em `splits.json`, proposta de `embedding-contract.json` e plano do Golden Set.
05/10/2026	Checkpoint técnico não avaliativo	Apresentar a planilha de seleção do embedding, uma tabela inicial de benchmark/custo e o estado da recuperação ou seus impedimentos.

O prazo público de entrega continua sendo **19/10/2026**. A governança de qualquer eventual alteração oficial de prazo está exclusivamente no Plano de Ensino e Cronograma.

Pergunta de pesquisa

RQ: Como os recomendadores supervisionado, baseado em recuperação e híbrido diferem na recomendação multi-label de *skills* para enunciados WOKDEX mantidos fora do índice?
H1: O recomendador supervisionado e o de recuperação apresentarão perfis de erro distintos, especialmente para *skills* frequentes e raras.
H2: Um híbrido ajustado somente na validação pode alterar o compromisso entre precisão e revocação em relação aos componentes isolados.
Estas são hipóteses a investigar, não garantias de superioridade. Resultados equivalentes, inferiores ou inconclusivos são resultados válidos se forem corretamente medidos e discutidos.

5.17 Corpus, proveniência e contrato de embeddings

O corpus oficial é o **LeetCode Problems Dataset Bilíngue (PT-BR)**, disponibilizado em `database/leetcode_problems_pt.json` ou `database/leetcode_problems_pt.csv`. A documentação dos campos e do processo de curadoria está em `database/README.md`. Registrem no repositório a versão, URL/arquivo de origem, data de obtenção, número de registros lidos, número descartado e o SHA-256 do arquivo efetivamente usado. Não misturem dados externos sem documentar licença, origem e finalidade.

Os campos mínimos são `id`, `title_pt`, `description_pt`, `difficulty` e `topics`. `topics` é o conjunto de rótulos verdadeiros para avaliação; ele não pode ser enviado ao modelo supervisionado como entrada nem incluído no texto vetorizado. `hints_pt` pode ser usado apenas se a equipe declarar a decisão e aplicar o mesmo contrato em todos os subconjuntos.

5.17.1 Contrato de embedding

Antes de indexar, criem `embedding-contract.json` versionado. Ele deve registrar, no mínimo:

```
{
  "contract_version": "1.0",
  "model_id": "nome-e-versao-exata-do-modelo",
  "dimensions": 384,
  "distance": "cosine",
  "normalization": "L2",
  "language": "pt-BR",
  "input_template": "{title_pt}\n\n{description_pt}",
  "chunking": {"strategy": "one-document-per-problem", "max_chunks_per_problem": 1},
  "corpus_sha256": "..."
}
```

O modelo de embedding pode ser local ou hospedado, desde que sua versão, dimensionalidade, normalização, custo/limitação de execução e licença sejam declarados. O contrato existe para impedir comparações entre vetores incompatíveis e para permitir reindexação. Título, descrição, limpeza de HTML/Markdown e qualquer truncamento precisam ser aplicados de forma determinística e descritos no README.

> [!IMPORTANT] Para evitar vazamento de rótulos, `topics`, dificuldade e taxa de aceitação não entram no texto do embedding nem no escore de similaridade. Eles podem aparecer apenas como metadados de resposta e, no caso de `topics`, como anotação dos vizinhos para derivar recomendações após a recuperação.

5.18 Metodologia honesta e sem vazamento

1. Fixem uma semente e dividam os **IDs de problemas antes de qualquer indexação** em treino, validação e teste. Recomendação: 70%/15%/15%, com estratificação multi-label quando viável. Publiquem `splits.json` com os IDs e a semente.
2. Treinem ou reutilizem o artefato congelado do TP1 apenas com treino. Se o TP1 tiver usado outra divisão, reexecutem a avaliação com esta divisão ou documentem claramente a limitação; não reportem resultado como teste comparável sem isso.
3. Construam o índice de retrieval **somente com documentos de treino**. Os textos de validação e teste são consultas, nunca documentos recuperáveis. Uma consulta não pode recuperar a si mesma, uma duplicata identificada, nem um item do mesmo grupo de duplicatas quando esse agrupamento estiver disponível.
4. Usem validação para escolher `K`, a regra de agregação dos vizinhos, limiares por *skill* e o peso do híbrido. Fixem esses valores antes de abrir os rótulos do teste.
5. Executem uma única avaliação final no teste. Não alterem parâmetros após examiná-la; correções de defeito exigem novo registro de experimento e justificativa.

Para a recomendação por retrieval, uma regra simples e auditável é somar, para cada *skill* s, as similaridades normalizadas dos K vizinhos de treino que possuem s. Para o híbrido, combinem escores comparáveis, por exemplo:

$$score_{hibrido}(s \mid q) = \alpha\, p_{TP1}(s \mid q) + (1 - \alpha)\, score_{retrieval}(s \mid q),$$

com α, normalização e limiar definidos na validação. Outras regras são aceitas se forem especificadas, implementadas e testadas. Não é permitido escolher manualmente a melhor estratégia para cada consulta de teste.

5.19 API de recuperação e contrato de recomendação

O serviço pode ser implementado em Java/Spring Boot, Python/FastAPI ou outra stack HTTP documentada. A avaliação valoriza o contrato e a reprodutibilidade, não uma biblioteca específica. O ambiente deve subir com Docker Compose.

5.19.1 `GET /search`

Exemplo: `GET /search?q=inverter+lista+encadeada&k=3`

Parâmetros obrigatórios: `q` não vazio e `k` inteiro entre 1 e 20. A resposta deve informar o contrato e os vizinhos usados:

```
{
  "query": "inverter lista encadeada",
  "k": 3,
  "embedding_contract_version": "1.0",
  "index_split": "train",
  "results": [
    {
      "exercise_id": 206,
      "title_pt": "Inverter Lista Encadeada",
      "similarity_score": 0.94,
      "difficulty": "Easy",
      "skills": ["Linked List", "Recursion"],
      "snippet": "Dada a cabeça de uma lista encadeada..."
    }
  ],
  "search_time_ms": 42
}
```

`similarity_score` deve ter semântica documentada (por exemplo, cosseno) e os resultados devem vir em ordem não crescente de similaridade. Retornem `400` para parâmetros inválidos e uma resposta JSON de erro com `code` e `message`.

5.19.2 Recomendação de skills

Implementem `POST /skills/recommend` ou forneçam o campo equivalente na resposta de `/search`. A equipe deve publicar um único contrato, com recomendação dos três métodos e evidência de proveniência. Exemplo para `POST /skills/recommend`:

```
{
  "query": "encontre o menor caminho em um grafo sem pesos",
  "k": 5,
  "methods": {
    "supervised_tp1": {
      "skills": [{"name": "Breadth-First Search", "score": 0.81}]
    },
    "retrieval": {
      "skills": [{"name": "Breadth-First Search", "score": 0.76, "supporting_ids": [127, 433]}]
    },
```

```
    "hybrid": {
      "alpha": 0.60,
      "threshold": 0.50,
      "skills": [{"name": "Breadth-First Search", "score": 0.79}]
    }
  }
}
```

O endpoint não precisa expor os rótulos verdadeiros da consulta. `supporting_ids` torna explícito quais documentos indexados sustentam uma recomendação por retrieval.

5.20 Golden Set e avaliação multi-label

Criem e versionem `evaluation/golden-set.yaml` com **pelo menos 15 consultas**, todas derivadas de itens do conjunto de teste, mas sem expor os rótulos ao serviço. Cada entrada deve conter `query_id`, texto da consulta, `source_exercise_id`, `expected_skills`, e uma breve justificativa de anotação. O Golden Set deve cobrir ao menos cinco famílias de *skills*, consultas com uma e com múltiplas *skills*, e casos potencialmente ambíguos. Ele é um recorte legível para auditoria humana; as métricas principais devem ser calculadas também sobre todo o teste.

Para cada um dos três métodos, reportem no README e em arquivo tabular versionado:

Métrica	Como reportar
Precision@K	Média da proporção de *skills* relevantes nas primeiras K recomendações, para pelo menos `K=1, 3, 5` quando aplicável.
Recall@K	Média da fração das *skills* verdadeiras presente nas primeiras K recomendações.
F1 micro	Calculado sobre todas as decisões rótulo-consulta.
F1 macro	Média do F1 por *skill*, incluindo a política para classes sem suporte.
Exact match	Fração de consultas cujo conjunto previsto é exatamente o conjunto verdadeiro.

Definam como desempates, *skills* fora do vocabulário e consultas sem recomendação são tratados. Reportem suporte por *skill* para tornar o F1 macro interpretável. Latência e tamanho do índice são métricas operacionais recomendadas, mas não substituem as métricas de qualidade acima.

5.20.1 Análise de discordância

Selecionem pelo menos **cinco consultas** em que dois métodos discordam e classifiquem cada caso como acerto exclusivo, erro compartilhado, diferença de granularidade, ambiguidade de anotação, similaridade superficial ou outro motivo justificado. Mostrem consulta, rótulos esperados, previsões, vizinhos relevantes e interpretação. Discordância não é defeito por si só: ela é evidência para responder à pergunta de pesquisa.

5.21 Entregáveis de engenharia

O repositório deve conter:

- `docker-compose.yml` e instruções para subir serviço, dependências e carga inicial sem segredo versionado;
- código de ingestão, índice persistente ou procedimento reprodutível de construção, contrato de embeddings e manifesto de proveniência;

- API `GET /search` e contrato de recomendação de *skills*;
- testes unitários e de integração para validação de entrada, ordenação/forma de `/search`, exclusão de documentos fora do treino, agregação de *skills* e cálculo de métricas;
- pipeline de CI que execute testes e, quando o ambiente permitir, valide build/Compose sem depender de credenciais privadas;
- `README.md` com arquitetura, como executar, papéis, decisões experimentais, limitações, tabela das métricas, análise de discordância e links para os artefatos;
- `evaluation/golden-set.yaml`, `splits.json`, resultados reproduzíveis e script/comando que os gere.

Uma demonstração manual não substitui testes. Um serviço que exija chave privada deve disponibilizar modo local, *fixture* determinística ou instruções claras para que a suíte básica possa ser executada sem a chave.

5.22 Papéis e entregas individuais

Os papéis organizam o trabalho, mas toda pessoa deve revisar uma contribuição de colega e contribuir com commits identificáveis. Ajustes de divisão são possíveis se forem declarados no README e preservarem carga e responsabilidade comparáveis.

Papel	Estimativa	Entregas verificáveis
Pessoa A – Dados e protocolo experimental	~15h	Proveniência, limpeza determinística, `splits.json`, Golden Set, verificação anti-vazamento e script de avaliação.
Pessoa B – Retrieval e índice vetorial	~15h	Contrato de embeddings, ingestão/indexação apenas do treino, busca KNN, `GET /search` e testes de recuperação.
Pessoa C – Integração TP1 e recomendador híbrido	~15h	Adaptador do artefato/API congelado do TP1, recomendador por vizinhos, ajuste na validação, endpoint/contrato de *skills* e testes.
Pessoa D – Qualidade, reprodução e comunicação científica	~15h	Docker Compose, CI, testes de integração, README de execução, tabelas/gráficos e consolidação das seções SBC.

5.23 Artigo SBC incremental

Escrevam agora, no repositório, `artigo/` ou `docs/artigo/` com as seções abaixo em Markdown ou LaTeX. Elas serão insumo para a versão consolidada posterior, mas já precisam refletir o experimento real deste TP:

1. **Introdução e RQ:** problema, objetivo comparativo, pergunta e hipóteses sem alegação antecipada de ganho.
2. **Trabalhos relacionados:** ao menos duas fontes sobre recuperação densa, recomendação baseada em vizinhos ou classificação multi-label, com relação explícita ao projeto.
3. **Método:** corpus e proveniência, divisão, contrato de embeddings, três recomendadores, tuning e salvaguardas anti-vazamento.

4. **Protocolo de avaliação:** Golden Set, métricas, ambiente e política de reprodutibilidade.
5. **Resultados e discussão inicial:** tabela dos três métodos, discordâncias, ameaças à validade e limitações. Não preencham resultados inexistentes; usem `pendente` até executar o experimento.

5.24 Roteiro de desenvolvimento

Período	Marco	Evidência
Etapa	Marco	Evidência
:—	:—	:—
1	Start Pack: protocolo, papéis, corpus e divisão	`splits.json`, proveniência, plano do Golden Set e esqueleto do artigo.
2	Checkpoint: contrato e retrieval mínimo	`embedding-contract.json`, planilha de seleção do embedding, tabela de benchmark/custo, índice de treino e `/search` testado.
3	Três recomendadores	Adaptador TP1, agregação dos vizinhos e híbrido com tuning somente na validação.
4	Avaliação	Golden Set com >=15 consultas, métricas completas e análise de discordância.
5	Qualidade e entrega	Docker, CI, testes de integração, README, seções SBC e resultados versionados.

5.25 Rubrica (10,0 pontos)

Critério	Pontos	Evidência
Protocolo, corpus, proveniência e proteção contra vazamento	1,5	Divisão versionada, contrato, índice só de treino e justificativas.
Retrieval e API	1,5	`GET /search`, índice funcional, erros tratados e contrato cumprido.
Três recomendadores e tuning válido	2,0	TP1, retrieval e híbrido reproduzíveis; parâmetros escolhidos na validação.
Golden Set, métricas e análise de discordância	2,0	>=15 consultas, todas as métricas requeridas e cinco discordâncias analisadas.
Engenharia reprodutível	1,5	Docker, testes, CI, README e execução documentada.
Contribuições individuais e seções SBC incrementais	1,5	Papéis verificáveis, revisão entre pares, commits e texto científico consistente.
Total	**10,0**	

5.26 Desafios avançados opcionais

Estes desafios não são pré-requisitos nem exigem agentes ou LLMs. Podem enriquecer a discussão, sem substituir os entregáveis obrigatórios:

- comparar dois modelos de embedding sob o mesmo protocolo e orçamento declarado;
- testar busca híbrida léxica+densa, com pesos escolhidos na validação;
- medir estabilidade por diferentes sementes de divisão ou *bootstrap* com intervalos de confiança;
- criar uma interface simples para inspecionar consulta, vizinhos e origem de cada *skill* recomendada;
- avaliar efeito de remover título, exemplos ou restrições do texto de consulta.

5.27 Checklist de entrega

☐ Repositório GitLab com autoria, papéis e revisão entre pares documentados.
☐ Corpus, SHA-256, divisão e contrato de embedding versionados.
☐ Índice contém somente documentos de treino; validação e teste são apenas consultas.
☐ `GET /search` e o contrato de recomendação de *skills* implementados e testados.
☐ Comparação supervisionado, retrieval e híbrido, sem tuning no teste.
☐ Golden Set com >=15 consultas e resultados completos no teste.
☐ Precision@K, Recall@K, F1 micro, F1 macro e exact match para os três métodos.
☐ Cinco ou mais discordâncias analisadas.
☐ Docker, testes, CI e README reproduzíveis.
☐ Seções incrementais do artigo SBC escritas agora.

Documentação complementar

A visão geral da oferta e as relações entre os três trabalhos foram atualizadas em `index.qmd` e no editorial docente. Manuais de pesquisa e a trilha IC podem preservar propostas históricas de agentes e YAML; eles não alteram os requisitos deste enunciado.

5.28 3. TP3: O Agente Autônomo Curador WOKDEX & Demoday

5.29 Proposito e escopo

O TP3 consolida o que a equipe construiu no **TP1** e no **TP2**. Em vez de criar um novo sistema isolado, a equipe deve integrar os contratos dos dois servicos e entregar um **assistente RAG fundamentado**: para uma pergunta ou enunciado de programacao, o sistema apresenta a classificacao de skills, recupera evidencias do corpus e produz uma resposta rastreavel, com citacoes e possibilidade explicita de abstencao.

O produto central nao e um agente autonomo. **Nao sao requisitos do TP3:** LangChain, LlamaIndex, tool calling, cadeia de raciocinio, auto-correcao, geracao de YAML ou loop autonomo. Uma LLM pode ser usada apenas na etapa de geracao da resposta RAG, desde que a resposta seja fundamentada exclusivamente nas evidencias recuperadas e que seu uso seja mensurado. A implementacao pode ser inteiramente deterministica quando a equipe justificar a decisao.

Esforco planejado: 60 horas por equipe, aproximadamente 15 horas por estudante.

Equipe: 4 estudantes.

Prazo de entrega: 26/11/2026. **Apresentação:** 30/11 e 02/12/2026.

Datas, peso, escopo obrigatório e avaliação normativos constam do Plano de Ensino e Cronograma 2026-2; esta página descreve as evidências técnicas do projeto.

5.30 Objetivos de aprendizagem

Ao concluir o TP3, a equipe devera demonstrar que consegue:

1. Integrar sistemas independentes por contratos HTTP/documentados, sem acoplamento ao codigo interno do TP1 ou TP2.
2. Explicar, com exemplos e limites, quando usar classificacao, recuperacao e uma arquitetura hibrida.
3. Construir uma resposta RAG cujas afirmacoes sejam apoiadas por fontes recuperadas identificaveis.
4. Projetar uma abstencao segura para entradas ambiguas, sem evidencia suficiente ou fora do dominio.
5. Avaliar o sistema com casos definidos antes da demonstracao e analisar falhas de integracao.
6. Consolidar em artigo SBC o material, as metricas e as decisoes dos TPs 1 e 2.

5.31 Produto a entregar

Entreguem um repositorio executavel e um dossie tecnico-cientifico. A interface pode ser uma **CLI** ou uma **UI web** simples; escolha uma e documente como executa-la. A demonstracao deve aceitar uma entrada textual e apresentar, no minimo:

1. A classificacao de skills fornecida pelo contrato do TP1, com escore ou indicacao de confianca quando disponivel.
2. Os documentos ou exercicios recuperados pelo TP2, com identificador, titulo e escore/ranking quando disponivel.
3. Uma resposta ao usuario que diferencie claramente fato recuperado, inferencia e recomendacao.
4. Citacoes na propria resposta, por exemplo `[E1]` e `[E2]`, que apontem para os itens recuperados exibidos pela interface.
5. A decisao de responder ou abster-se e sua justificativa objetiva.
6. Um identificador de requisicao e os eventos minimos de observabilidade descritos adiante.
7. Um diagrama de fluxo de dados, em Mermaid textual ou formato equivalente, mostrando entrada, TP1, TP2, integrador, evidencias/citacoes, resposta e destinos de logs ou dados persistidos.

O assistente deve tratar perguntas sobre exercicios, skills e estrategias de resolucao presentes no escopo do corpus. Ele nao deve apresentar uma resposta inventada como se fosse evidenciada. Quando a base nao sustentar a resposta, deve declarar a limitacao, indicar a ausencia de evidencia e, se pertinente, pedir esclarecimento.

5.32 Integracao por contratos

Antes de integrar, a equipe deve registrar no `README.md` ou no dossie uma tabela de contratos. Ela deve conter endpoint, metodo, entrada, saida, codigos de erro, timeout e responsavel pelo contrato. Adaptem os nomes reais definidos pela equipe no TP1 e TP2; nao e permitido presumir uma biblioteca especifica.

Componente	Responsabilidade minima	Evidencia exigida
TP1	Receber texto e retornar skills previstas	Exemplo de requisicao/resposta e teste de contrato
TP2	Receber consulta e retornar itens recuperados	Exemplo de requisicao/resposta e teste de contrato
Integrador TP3	Compor as respostas e aplicar a politica de evidencia/abstencao	Fluxo executavel, logs e testes de integracao

O integrador deve usar configuracao externa para URLs e timeouts. Durante o desenvolvimento, mocks que respeitem o contrato sao permitidos; a entrega deve incluir ao menos uma execucao com as implementacoes reais da equipe ou uma justificativa objetiva para indisponibilidade temporaria.

5.33 Governanca de dados e provedor

Documentem uma decisao de dados e provedor no README ou dossie. Ela deve declarar se ha PII nos textos de entrada, logs ou corpus; como a equipe a evita, minimiza ou anonimiza; o prazo e local de retencao; e se cada componente e local ou hospedado. Para cada provedor hospedado, registrem finalidade, dados enviados, configuracao relevante, custo/limite conhecido e alternativa de mock ou execucao local. Nao enviem PII, segredos ou logs completos a provedores sem uma decisao documentada e justificavel.

5.33.1 Testes de contrato e falhas

Implementem testes automatizados para, no minimo, os seguintes casos:

Falha ou contrato	Comportamento esperado
Resposta valida do TP1	Skills sao interpretadas sem alterar seu significado
Resposta valida do TP2	Evidencias citadas correspondem aos itens retornados
Campo obrigatorio ausente ou tipo invalido	Falha clara, sem resposta aparentemente confiavel
Timeout ou indisponibilidade de um servico	Mensagem degradada; nao inventar resultado do componente ausente
Lista vazia de recuperacao	Abstencao ou pedido de refinamento, com registro no log

Documentem ao menos dois defeitos reais encontrados na integracao, sua causa, como foram reproduzidos e a correcao ou limitacao remanescente.

5.34 Classificador, recuperacao e arquitetura hibrida

O dossie e o artigo devem comparar as tres estrategias, usando o proprio sistema e nao apenas definicoes teoricas.

Estrategia	Entrada e saida	Vantagem	Limitacao a investigar
Classificador (TP1)	Enunciado para skills	Baixa latencia e rotulos consistentes	Pode errar classes proximas ou desconhecidas

Estrategia	Entrada e saida	Vantagem	Limitacao a investigar
Recuperacao (TP2)	Consulta para itens semelhantes	Traz evidencias observaveis do corpus	Similaridade nao garante relevancia pedagogica
Hibrida (TP3)	Skills + recuperacao para resposta fundamentada	Combina sinal preditivo e evidencia	Exige contratos, politica de citacao e controle de falhas

Na arquitetura hibrida, explicitem a regra de composicao. Por exemplo: as skills do TP1 podem orientar a consulta do TP2, mas uma recomendacao especifica so pode aparecer se houver evidencia recuperada que a apoie. A equipe deve declarar o valor de `k`, o limiar ou criterio de qualidade usado e como esses parametros foram escolhidos.

5.35 Resposta RAG fundamentada

Uma resposta e considerada fundamentada quando cada afirmacao relevante pode ser ligada a uma ou mais evidencias recuperadas. Para cada resposta avaliada, guardem:

Item	Registro necessario
Entrada	Pergunta/enunciado exatamente como recebido
Predicao	Skills e confianca, quando existir
Recuperacao	IDs, titulos, ranking/escores e trecho utilizado
Resposta	Texto mostrado ao usuario com marcadores de citacao
Decisao	`RESPONDER`, `ABSTER` ou `PEDIR_ESCLARECIMENTO`
Justificativa	Regra, limiar ou ausencia de evidencia que motivou a decisao

Se houver componente gerativo, o prompt ou a instrucao deve exigir que ele use somente os trechos recuperados e cite os identificadores recebidos. A equipe deve verificar programaticamente que toda citacao exibida referencia uma evidencia da requisicao corrente. Nao exibam raciocinio interno do modelo; registrem apenas entradas, saidas, parametros relevantes e evidencias.

5.35.1 Politica de abstencao

Definam uma politica simples, reproduzivel e testavel. Ela pode combinar, por exemplo, pontuacao baixa de recuperacao, divergencia entre skills e evidencias, consulta curta/ambigua, ou ausencia de resultados. A politica deve produzir uma mensagem util, como: “Nao encontrei evidencia suficiente no corpus para recomendar uma estrategia. Informe as restricoes, o formato de entrada e um exemplo.”

Abstencao correta vale mais que uma resposta confiante sem suporte.

5.35.2 Rubrica de fundamentacao por resposta

Avaliem cada resposta dos seis casos nesta rubrica e incluam o resultado na tabela de casos:

Nivel	Criterio
`fundamentada`	Todas as afirmacoes relevantes possuem citacao valida para evidencia retornada na requisicao; limites e recomendacoes sao explicitados.
`parcial`	A resposta tem evidencias validas, mas ao menos uma afirmacao relevante, inferencia ou recomendacao carece de suporte suficiente ou de delimitacao.

Nivel	Criterio
`nao fundamentada`	Ha afirmacao relevante sem suporte, citacao invalida/ausente ou resposta apresentada como evidenciada sem que a recuperacao a sustente.
`abstencao adequada`	O sistema nao responde de modo afirmativo quando falta evidencia e explica objetivamente a limitacao ou o esclarecimento necessario.

5.36 Observabilidade

Cada requisicao deve gerar logs estruturados ou registros equivalentes que permitam reconstruir o fluxo. Nao registrem chaves, tokens, dados sensiveis ou texto de raciocinio interno de LLMs.

Registrem ao menos: `request_id`, horario, rota/acao, versao dos servicos, latencia do TP1, latencia do TP2, numero de itens recuperados, decisao final, codigos de erro e identificadores das evidencias citadas. Incluam no dossie uma amostra legivel de dois fluxos: uma resposta fundamentada e uma abstencao/degradacao.

5.37 Avaliacao experimental: seis casos obrigatorios

Construam uma colecao fixa de **seis casos** antes da rodada final. Cada caso deve ter entrada, expectativa, criterio de julgamento e resultado observado. Pelo menos um membro que nao implementou diretamente aquele modulo deve revisar cada expectativa.

Caso	Tipo	Expectativa minima
1	No dominio, direto	Skills e evidencias coerentes; resposta com citacoes validas
2	No dominio, estrategia distinta	Comparacao entre classificador, recuperacao e hibrido
3	Consulta com vocabulário diferente/parafrase	Recuperacao ou abstencao explicada; sem citacao falsa
4	Ambiguo	Pedido de esclarecimento ou resposta condicionada e citada
5	Fora do dominio	Abstencao clara, sem inventar fontes ou skills
6	Falha controlada de contrato/servico	Degradacao segura, log e mensagem compreensivel

Para os seis casos, apresentem uma tabela com saida do classificador, evidencias recuperadas, decisao final, nivel da rubrica de fundamentacao, latencia e julgamento humano breve (`adequado`, `parcial`, `inadequado`). A analise deve explicar pelo menos uma situacao em que a arquitetura hibrida nao foi melhor que um componente isolado.

5.38 Artigo SBC incremental

Completem o artigo no formato SBC, com **4 a 6 paginas** (sem contar referencias e apendices, se o modelo adotado os separar). O texto deve evoluir os materiais iniciados nos TPs 1 e 2: nao recomecem a narrativa nem descartem metricas ja produzidas.

Estrutura minima:

1. Introducao e problema pedagogico.
2. Fundamentacao e trabalhos relacionados, incluindo classificacao e RAG.

3. Metodo: contratos TP1/TP2, arquitetura hibrida e politica de abstencao.
4. Avaliacao: protocolo dos seis casos, metricas herdadas do TP1/TP2 e resultados do TP3.
5. Discussao: classificacao versus recuperacao versus hibrido, falhas, custo/latencia e ameacas a validade.
6. Conclusao e proximo passo justificavel.

O artigo deve conter pelo menos um diagrama de arquitetura, uma tabela comparativa das tres estrategias e a tabela resumida dos seis casos. Citem as fontes efetivamente usadas; a leitura recomendada do TP2, Lewis et al. (2020), e um ponto de partida apropriado para contextualizar RAG.

5.39 Papeis e estimativa de esforco

Cada estudante assume um papel principal de aproximadamente 15 horas, sem impedir revisao cruzada e pareamento. Registrem autoria, tarefas e evidencias no `README.md`.

Papel	Responsabilidades principais	Evidencia individual
Integrador de contratos	Clientes TP1/TP2, configuracao, mocks e testes de contrato	MR/commits e testes executaveis
Responsavel por recuperacao e evidencia	Consulta hibrida, formatacao de fontes, verificacao de citacoes	Casos com trilha de evidencia
Responsavel por experiencia e abstencao	CLI/UI, mensagens de degradacao, politica de abstencao	Fluxos demonstraveis e testes de decisao
Responsavel por avaliacao e artigo	Protocolo de seis casos, observabilidade, tabelas e consolidacao SBC	Resultados reproduziveis e secao do artigo

5.40 Sequência de desenvolvimento

Organizem contratos, integração, avaliação, síntese e ensaio de modo a cumprir o prazo normativo do TP3. As apresentações e defesa técnica ocorrem em **30/11 e 02/12/2026**.

Até **16/11/2026**, entreguem um alerta técnico de risco **não avaliativo**: contratos e mocks verificados, um caso fundamentado e um caso de abstenção executados, e um risco registrado com responsável e próximo passo. O alerta antecipa bloqueios de integração; não é uma nova aula nem substitui a entrega final.

5.41 Apresentações em 30/11 e 02/12

Apresentem em ate 6 minutos, com participacao de todos os integrantes:

1. Problema, contratos e arquitetura hibrida (1 minuto).
2. Demo da CLI/UI em um caso no dominio, mostrando evidencias e citacoes (2 minutos).
3. Demo de abstencao, ambiguidade ou falha degradada (1 minuto).
4. Resultados dos seis casos e comparacao das tres estrategias (1 minuto).
5. Uma falha real, sua correcao e uma limitacao remanescente (1 minuto).

Preparem uma captura de tela ou video curto como contingencia. O material de contingencia nao substitui a defesa tecnica nem os testes no repositorio.

5.42 Rubrica (20 pontos)

Criterio	Pontos	Evidencia de avaliacao
Integracao dos contratos TP1 e TP2	3,0	Clientes configuraveis, execucao integrada e testes de contrato

Criterio	Pontos	Evidencia de avaliacao
Resposta RAG com evidencia e citacoes validas	3,5	Trilha de evidencia, rubrica de fundamentacao por resposta e verificacao de citacoes por requisicao
Politica de abstencao e tratamento seguro de degradacao	2,0	Casos ambiguo, fora do dominio e falha controlada
CLI/UI e observabilidade	2,0	Fluxo utilizavel, logs/rastreabilidade e mensagens claras
Avaliacao dos seis casos e analise critica	3,0	Protocolo, resultados, comparacao e limites identificados
Falhas reais e testes de contrato/integracao	2,0	Dois defeitos documentados e suite automatizada relevante
Artigo SBC incremental TP1-TP2-TP3	2,5	Estrutura, evidencias, metricas e discussao cientifica
Apresentacao e defesa tecnica	1,0	Clareza, participacao e demonstracao consistente
Contribuicao individual comprovavel	1,0	Papel, commits/MRs, revisao e evidencia de autoria
Total	**20,0**	

5.43 Checklist de entrega final

- ☐ Repositorio com instrucoes de execucao, configuracao e autoria dos quatro papeis.
- ☐ CLI ou UI funcional com classificacao, recuperacao, resposta citada e abstencao.
- ☐ Diagrama de fluxo de dados e decisao de PII, retencao e provedor documentados.
- ☐ Tabela de contratos TP1/TP2 e testes automatizados de contrato/integracao.
- ☐ Logs ou registros de observabilidade sem segredos.
- ☐ Seis casos avaliados, incluindo ambiguo, fora do dominio e falha controlada.
- ☐ Dois defeitos de integracao documentados com causa e resolucao/limitacao.
- ☐ Artigo SBC incremental completo, com diagrama e tabelas exigidas.
- ☐ Slides ou roteiro da apresentacao e material de contingencia.
- ☐ Desafios opcionais, se realizados, claramente separados do nucleo avaliado.

5.44 Desafios avancados opcionais

Equipes que desejarem explorar LLMs, tool calling ou agentes podem seguir a TP3.5: Trilha Avancada Opcional. Essa trilha nao substitui nenhum item desta rubrica, nao e pre-requisito para a nota e deve permanecer isolada do caminho principal de execucao.

Capítulo 6

PARTE IV — Trilha Especial de Iniciação Científica (Pesquisa Avançada)

6.1 1. Programa IC Especial: Visão Geral

6.2 O que é o Programa IC Especial?

A disciplina oferece uma trilha opcional de Iniciação Científica para equipes que demonstrem interesse e condições de conduzir uma investigação experimental curta. Em 2026-2, a proposta é auditar os artefatos que a própria turma produz no **TP2** e no **TP3**: retrieval, proveniência, contratos, evidências, citações, abstinção e reprodução.

Regra de avaliação

O TP3 Master substitui o TP3 regular **somente para equipes explicitamente aceitas na Trilha IC pelo professor**. Para elas, a entrega vale os mesmos **20 pontos**, tem prazo em **26/11** e apresentação em **30/11 e 02/12**. Não há acúmulo de TP3 regular e Master. Todas as demais equipes seguem o TP3 regular.

A seleção ocorre após o TP1 e a substituição precisa ser confirmada antes do início do TP3. A participação na IC é voluntária; quem não aceitar o convite, ou não for formalmente aceito, segue o percurso regular sem penalidade.

6.3 Objetivo científico

A investigação pergunta se as respostas RAG integradas no TP3 são sustentadas pelas evidências recuperadas no TP2, se citam corretamente suas fontes, se se abstêm quando não há suporte e se outra pessoa consegue reproduzir os resultados a partir dos artefatos versionados.

O trabalho não tenta criar agentes ou gerar `metadata.yaml`. Em 2026-2, o TP3 regular é um assistente RAG fundamentado, e os dados da auditoria são seus contratos, logs, casos de avaliação, citações e artefatos de retrieval. Agentes, LLMs com tool calling e orquestração pertencem à TP3.5, que é opcional, isolada e nunca fonte de dados obrigatória para a IC.

6.4 Quem pode participar?

6.4.1 Critérios de seleção

Critério	Como é observado
Trabalho consistente no TP1	Métricas, organização e correção do experimento
Contribuições identificáveis	Commits/MRs, revisão de colega e responsabilidade assumida
Qualidade de engenharia	Testes, documentação, controle de versão e comunicação de limites
Interesse em pesquisa	Formulário e conversa breve com o professor
Disponibilidade	Cerca de 15 horas por estudante no TP3, sem promessa de trabalho extra ilimitado

O professor pode aceitar uma ou mais equipes de quatro estudantes conforme a capacidade de mentoria e a disponibilidade de artefatos. A aceitação será documentada no repositório/canal da disciplina.

6.5 Método e entregáveis IC

As equipes aceitas seguem o método fixo do TP3 Master:

1. Inventariar e congelar os artefatos do TP2 até 19/10: corpus, hashes, `splits.json`, contrato de embeddings, Golden Set, resultados e comandos.
2. Definir antes da rodada final ao menos 12 casos: os seis casos exigidos no TP3 e seis derivados do Golden Set ou das discordâncias do TP2.
3. Auditar, por caso, o suporte das evidências, a validade das citações e a decisão de responder, abster-se ou pedir esclarecimento.
4. Reexecutar a auditoria com uma pessoa diferente da autora principal em ambiente limpo ou perfil separado.
5. Consolidar resultados, divergências, limitações e um artigo curto SBC de 3-4 páginas.

Os entregáveis detalhados, papéis, rubrica e checklist estão no TP3 Master. Dados de terceiros só podem ser usados com origem, permissão/licença e anonimização documentadas. Não versionem segredos, dados pessoais ou raciocínio interno de modelos.

6.6 Papéis realistas para uma equipe de quatro

Papel	Foco
Pessoa A - Protocolo e dados	Proveniência, inventário, hashes, anonimização e casos
Pessoa B - Evidência e citações	Vínculo entre retrieval, trechos, IDs e citações RAG
Pessoa C - Abstenção e integração	Decisões, falhas de contrato, degradação segura e logs
Pessoa D - Reprodução e artigo	Ambiente, scripts, segunda execução, tabelas e texto SBC

Toda pessoa revisa uma entrega de colega e mantém autoria verificável. Os papéis podem ser ajustados quando documentados no README, desde que preservem responsabilidades comparáveis.

6.7 Cronograma 2026-2

Marco	Data/período	Resultado esperado
Seleção e confirmação IC	Após TP1, antes do TP3	Equipe aceita e escopo de substituição confirmado
Fechamento dos artefatos TP2	Até **19/10**	Inventário, protocolo e casos candidatos preparados

Marco	Data/período	Resultado esperado
Congelamento e piloto	20/10-10/11	Contratos, hashes, dois casos auditados e política de abstinção verificável
Alerta técnico de risco (não avaliativo)	**16/11**	Contratos e mocks verificados, um caso fundamentado, um caso de abstenção e risco com responsável
Auditoria e reprodução	11/11-25/11	12+ casos, revisão cruzada, segunda execução e tabelas
Entrega TP3 Master	**26/11**	Repositório e artigo SBC de 3-4 páginas
Apresentação	**30/11 e 02/12**	Defesa dos resultados, limitações e recomendação

6.8 Mentoria e resultados esperados

- Reunião curta semanal, em horário combinado, para revisar protocolo, impedimentos e qualidade dos artefatos.
- Revisão de código/documentação por meio de MRs ou mecanismo equivalente.
- Resultados podem ser positivos, negativos ou inconclusivos; a qualidade do método e a transparência dos limites são o critério central.
- Uma evolução para projeto de IC, relatório técnico ou artigo posterior depende dos resultados, disponibilidade e avaliação do professor. Coautoria não é garantida pela participação na disciplina.

6.9 Riscos e mitigação

Risco	Mitigação
Serviço TP1/TP2 indisponível	Usar mock explicitamente identificado para testar contrato, sem confundi-lo com execução real
Artefato sem versão ou hash	Registrar a lacuna como achado; não reconstruir silenciosamente o experimento
Poucos dados de outras equipes	Priorizar os artefatos da própria equipe e dados autorizados; não depender de coleta ampla
Resultado sem melhora aparente	Reportar falhas, abstinções inadequadas e limites como evidência válida
Experimento com agentes	Mantê-lo em TP3.5, separado da auditoria e da nota-base

6.10 2. TP3 Master: Simulated Students & Auditoria de Pipelines

6.11 Natureza e elegibilidade

Esta é uma alternativa de pesquisa ao TP3 regular, destinada **somente a equipes da Trilha IC explicitamente aceitas pelo professor**. Para essas equipes, o TP3 Master **substitui o TP3 regular**, vale os mesmos **20 pontos** e mantém as apresentações em 30/11 e 02/12. Equipes não aceitas na Trilha IC devem realizar integralmente o TP3 regular.

O TP3 Master não cria uma obrigação extra e não acumula duas notas de TP3. A aceitação será confirmada pelo professor após o TP1, com confirmação do escopo e da equipe antes do início do TP3. Caso uma equipe IC não seja formalmente aceita para esta substituição, permanece no TP3 regular.

Escopo revisado de 2026-2

O TP3 regular não exige agentes, LangChain, LangGraph, tool calling, geração de YAML ou qualquer sistema multiagente. Este TP3 Master também não usa YAMLs ou agentes como fonte de dados. Seus objetos de estudo são os artefatos reais do **TP2** (índice, contrato de embedding, `splits.json`, Golden Set, resultados e proveniência) e do **TP3** (contratos, evidências recuperadas, citações, decisões de abstinção, logs e seis casos avaliados).

Agentes pertencem exclusivamente à TP3.5, trilha opcional e experimental. Se uma equipe a explorar, seus resultados devem ficar separados e não podem ser a fonte dos dados nem condição para a auditoria.

6.12 Pergunta de pesquisa

RQ: Em casos de avaliação previamente definidos ou coletados sob protocolo, as respostas RAG do TP3 apresentam evidência rastreável, citações válidas, abstinção apropriada e resultados reproduzíveis a partir dos artefatos do TP2?

O objetivo não é provar que uma estratégia é sempre melhor. A equipe deve identificar evidências suficientes, falhas de suporte, decisões de abstinção adequadas ou inadequadas e lacunas que impeçam a reprodução. Resultado negativo ou inconclusivo é válido quando o protocolo, os dados e as limitações estiverem claros.

6.13 Dados, contratos e cuidados éticos

Usem apenas artefatos versionados da própria equipe, fornecidos voluntariamente por equipes participantes ou disponibilizados pelo professor. Não coletem dados pessoais, conversas privadas, chaves, tokens ou raciocínio interno de LLM. Casos coletados devem ser anonimizados e registrados com origem, data, licença/permissão e finalidade.

O conjunto auditado deve preservar o vínculo entre:

- o manifesto e SHA-256 do corpus, o `splits.json` e o `embedding-contract.json` do TP2;
- a consulta, o resultado de retrieval e os `supporting_ids`/trechos que sustentam a recomendação;
- o contrato TP1/TP2, a resposta RAG, as citações e a decisão `RESPONDER`, `ABSTER` ou `PEDIR_ESCLARECIMENTO` do TP3;
- versão de código, ambiente, semente, comando de execução, data e resultados produzidos.

Não alterem rótulos, citações, logs ou resultados após a auditoria sem criar nova execução identificada. Dados brutos eventualmente sensíveis ficam fora do repositório público; publiquem somente versões anonimizadas, agregadas ou instruções de acesso autorizadas.

6.14 Método fixo

O método abaixo é obrigatório para tornar os resultados comparáveis e viáveis no tempo disponível.

1. **Congelamento e inventário.** Até 19/10, registrem versões, hashes, responsáveis, comandos e disponibilidade dos artefatos TP2 já produzidos. A falta de um artefato é um achado de reprodutibilidade, não algo a ser preenchido retroativamente sem registro.
2. **Casos auditáveis.** Definam antes da rodada final pelo menos **12 casos**: os seis tipos obrigatórios do TP3 (direto, estratégia distinta, paráfrase, ambíguo, fora do domínio e falha controlada) mais seis casos derivados do Golden Set ou de discordâncias do TP2. Cada caso deve conter entrada, expectativa, critério de julgamento, origem e revisor que não o implementou.
3. **Auditoria de evidência e citação.** Para cada caso, verifiquem se cada afirmação relevante está apoiada por item retornado na mesma requisição, se a citação aponta para o ID/título correto e se a evidência foi recuperada do índice de treino conforme o contrato TP2. Classifiquem como `suportada`, `parcial`, `não suportada` ou `não verificável`.
4. **Auditoria de abstinção.** Apliquem a política documentada do TP3 aos casos ambíguos, fora do domínio, sem recuperação e com falha controlada. Julguem se a decisão e a mensagem evitam inventar fonte, skill ou recomendação. Registrem falsos `RESPONDER` e abstinções excessivas.
5. **Reprodução independente.** Uma pessoa diferente da autora principal executa os comandos em ambiente limpo ou perfil separado, usando as versões congeladas. Comparem decisões, citações, IDs recuperados e métricas; expliquem qualquer divergência.
6. **Síntese.** Produzam tabelas de cobertura de evidência, validade de citações, adequação de abstinção e sucesso de reprodução, além de uma análise qualitativa de pelo menos quatro falhas ou limitações.

Não é exigido teste estatístico. Se a equipe usar intervalos de confiança ou outro método adicional, deve justificá-lo e manter o protocolo principal intacto.

Até **16/11/2026**, registrem um alerta técnico de risco **não avaliativo**: contratos e mocks verificados, um caso fundamentado e um caso de abstenção executados, e um risco com responsável. O marco não cria aula adicional nem substitui a entrega final.

6.15 Entregáveis

O repositório da equipe deve conter:

- `audit/protocol.md` com pergunta, método fixo, critérios de julgamento, papéis e ameaças à validade;
- `audit/artifact-inventory.csv` ou equivalente com origem, versão/hash, licença/permissão, responsável e comando de reprodução de cada artefato TP2/TP3;
- `audit/cases.yaml` com os 12 ou mais casos, expectativas congeladas, origem e revisão cruzada;
- decisão documentada de PII, retenção e provedor para os dados e serviços usados na auditoria ou herdados do TP3;
- registros de auditoria por caso, incluindo consulta, evidências, resposta, citações, decisão, julgamento e justificativa;
- scripts/comandos para gerar as tabelas e repetir a auditoria, com ambiente, dependências e sementes documentados;
- resultado da reprodução independente e relatório de divergências;
- `artigo/` ou `docs/artigo/` com o artigo curto no formato SBC;
- `README.md` com instruções de execução, autoria, limitações, dados que não podem ser publicados e localização dos artefatos.

É aceitável usar mocks compatíveis quando um serviço estiver indisponível, desde que o caso seja marcado como tal. Mocks não contam como reprodução da execução real e não podem substituir silenciosamente os resultados reais.

6.16 Papéis para quatro estudantes

Cada estudante assume aproximadamente 15 horas, faz revisão de uma entrega de colega e mantém commits/MRs identificáveis.

Papel	Responsabilidade principal	Evidência individual
Pessoa A - Protocolo e dados	Inventário, proveniência, anonimização, congelamento e `cases.yaml`	Protocolo, hashes, permissões e casos revisados
Pessoa B - Evidência e citações	Auditoria de retrieval, trechos, `supporting_ids` e correspondência das citações	Planilhas/registros de cobertura e testes de validação
Pessoa C - Abstenção e integração	Auditoria das decisões, falhas de contrato e mensagens degradadas	Casos de abstinção, logs e análise de decisões
Pessoa D - Reprodução e artigo	Ambiente limpo, scripts, comparação de execuções, tabelas e redação SBC	Relatório de reprodução, tabelas geradas e seção do artigo

6.17 Artigo curto SBC (3-4 páginas)

Entreguem um artigo de **3 a 4 páginas**, sem contar referências e apêndices quando o modelo SBC os separar. O texto deve ser baseado nos casos predefinidos/coletados e nos registros reais da auditoria, não em resultados ilustrativos.

Estrutura mínima:

1. Problema e pergunta de pesquisa.
2. Artefatos TP2/TP3, contratos e proveniência dos dados.
3. Método: casos, critérios para evidência/citação, política de abstinção e reprodução independente.
4. Resultados: tabelas de cobertura de evidência, validade de citações, abstinção e reprodução.
5. Discussão: falhas, limitações, ameaças à validade e próximo passo justificável.

Incluam pelo menos uma tabela de casos, uma tabela de resultados e as citações acadêmicas efetivamente usadas. A seção de método deve permitir que outra equipe refaça a auditoria sem depender de conhecimento oral.

6.18 Cronograma

Período	Marco	Evidência
Até **19/10**	Fechamento do TP2 e preparação IC	Inventário inicial, papéis, protocolo e casos candidatos do Golden Set/discordâncias
20/10-31/10	Congelamento de artefatos	Hashes, contratos TP2, comandos e casos auditáveis definidos antes da rodada final
01/11-10/11	Integração TP3 e piloto	Registros de evidência/citação, política de abstinção e dois casos piloto revisados
11/11-17/11	Auditoria principal	Doze ou mais casos executados, julgamentos cruzados e falhas registradas
18/11-25/11	Reprodução e síntese	Execução independente, comparação, tabelas e artigo SBC revisado
26/11	Entrega do TP3 Master	Repositório, auditoria e artigo SBC de 3-4 páginas
30/11 e 02/12	Apresentações	Demonstração dos achados, defesa do método e limitações

6.19 Apresentações em 30/11 e 02/12

Apresentem em até 6 minutos, com participação de todos:

1. Pergunta, artefatos e protocolo (1 minuto).
2. Um caso com resposta fundamentada e citações verificadas (1 minuto).
3. Um caso de abstinção ou falha degradada (1 minuto).
4. Resultados da auditoria e da reprodução independente (2 minutos).
5. Limitação relevante e recomendação concreta para o TP3 (1 minuto).

Preparem captura de tela ou vídeo curto como contingência, sem substituir os registros e scripts versionados.

6.20 Rubrica (20 pontos)

Critério	Pontos	Evidência
Protocolo, casos e governança de dados	3,0	Método fixo, 12+ casos, origem, revisão cruzada e tratamento de dados
Auditoria de evidências e citações	4,0	Registros por caso, rastreabilidade TP2 e classificação justificada
Auditoria de abstinção e falhas seguras	3,0	Casos ambíguos, fora do domínio e falha controlada, sem fonte inventada
Reprodutibilidade e engenharia	3,0	Inventário, hashes, scripts, ambiente e execução independente
Análise crítica e resultados	2,5	Tabelas, divergências, quatro ou mais falhas/limitações e conclusões proporcionais
Artigo curto SBC	2,5	3-4 páginas, método, casos, contratos, citações e resultados reais
Apresentação e defesa técnica	1,0	Clareza, participação e limites explicitados
Contribuição individual comprovável	1,0	Papel, commits/MRs e revisão de colega
Total	**20,0**	

6.21 Checklist de entrega em 26/11

- ☐ Aceitação explícita da equipe na Trilha IC registrada pelo professor.
- ☐ Protocolo, inventário e 12 ou mais casos versionados antes da rodada final.
- ☐ Artefatos TP2 e TP3 rastreáveis por versão/hash, contrato e comando.
- ☐ Auditoria de evidência, citação e abstinção registrada para todos os casos.
- ☐ Reprodução independente executada e divergências explicadas.
- ☐ Scripts, ambiente e instruções suficientes para repetir as tabelas.
- ☐ Artigo SBC de 3-4 páginas baseado nos resultados reais.
- ☐ Material da apresentação de 30/11 e 02/12 e contingência preparados.
- ☐ Qualquer experimento com agentes isolado como TP3.5 opcional, sem compor a fonte de dados.

Capítulo 7

PARTE V — Recursos Oficiais, Schemas & Especificação do Dataset

Capítulo 8

Recursos Oficiais do WOKDEX

Este documento centraliza todos os **artefatos canônicos, bases de dados e especificações formais** do ecossistema WOKDEX utilizados na disciplina de **Inteligência Artificial II (IA II)**.

8.1 Dataset Oficial da Disciplina: LeetCode Problems Dataset Bilíngue (PT-BR)

Para viabilizar o treinamento de redes neurais profundas (TP1), a indexação semântica em banco vetorial (TP2) e a curadoria autônoma com agentes (TP3), a disciplina adota o **LeetCode Problems Dataset Bilíngue**, uma base curada com **2.830 problemas públicos de programação traduzidos e auditados em Português Brasileiro (PT-BR)**.

> **Links Oficiais para Acesso e Download**
>
> - **Relatório Exploratório Interativo (HTML do Jupyter Notebook)** *(Gráficos de distribuição, vocabulário e contagem de palavras)*
> - **Relatório Exploratório Consolidado (PDF)** *(Versão para leitura e impressão)*
> - **Download do Dataset em CSV (23.3 MB)**
> - **Download do Dataset em JSON Estruturado (25.4 MB)**
> - **Documentação Técnica & Especificação de Schemas**

8.1.1 Composição e Estatísticas do Dataset

- **Total de Registros:** 2.830 problemas gratuitos com enunciado completo.
- **Distribuição por Dificuldade:**
  - **Easy (CS1/CS2):** 752 problemas (26,6%) — ideal para conceitos introdutórios e estruturas básicas.
  - **Medium (CS2/CS3):** 1.417 problemas (50,1%) — estruturas de dados avançadas e algoritmos clássicos.
  - **Hard (CS3/Maratona):** 661 problemas (23,3%) — benchmarks avançados e otimização complexa.
- **Metodologia de Curadoria:** Tradução técnica assistida por LLM (`gpt-5.4-mini` via OpenAI Batch API com *Structured Outputs*), garantindo **98,9% de fidelidade estrutural perfeita** na preservação de tags HTML, expressões matemáticas, tabelas, código inline e variáveis originais.

8.1.2 Campos Disponíveis para os Modelos

Campo	Tipo	Aplicação nos TPs
`id`	Inteiro	Identificador unívoco do problema (`1` a `3549`).
`title_pt`	Texto	Título do problema em português.
`description_pt`	Texto (HTML/Markdown)	**Feature Textual** (X_{text}) para NLP no TP1 e **Documento de Chunk** no TP2 (RAG).
`difficulty`	Categórico (`Easy`, `Medium`, `Hard`)	Feature tabular de entrada ou baseline de classificação.
`topics`	Lista de Strings (JSON)	**Rótulos Alvo** (y) — skills e tópicos algorítmicos em português.
`hints_pt`	Lista de Textos	Dicas pedagógicas oficiais traduzidas para enriquecimento no TP2/TP3.
`acceptance_rate`	Float (0.0 a 100.0)	Métrica de taxa de aceitação (feature tabular para modelos Multi-Input).
`likes` / `dislikes`	Inteiro	Métricas de engajamento da comunidade de programadores.
`solution_code_python`	Código Python	Implementação de referência para validação de testes.

8.2 Artigo Científico Base

Miranda Junior, A.; Santos, V. F. (2026). *"Quebrando o Silêncio Pedagógico: O Modelo WOKDEX para Feedback Formativo em Juízes Online"*. SBIE 2026 — Trilha TPIE (Trabalhos e Pesquisas em Informática na Educação). CBIE 2026.

- **PDF Local do Artigo:** `artigo-sbie-wokdex.pdf`
- **Contribuições Formalizadas:** `contribuicoes-artigo-sbie.md`
- **Afiliação:** CEFET-MG (DECOM-TM) / UFMG (DCC)

8.2.1 As 4 Contribuições Científicas do Artigo

ID	Contribuição	Descrição
C1	**Schema WOKDEX**	Modelo de metadados estruturado (YAML validado por JSON Schema) que formaliza atributos pedagógicos para exercícios de programação.
C2	**Tipologia de 4 Testes**	Formalização das categorias `SAMPLE`, `FUNCTIONAL`, `MISCONCEPTION` e `PERFORMANCE`.
C3	**Corpus de Referência Golden**	Catálogo com exercícios anotados manualmente com cenários pedagógicos ricos.
C4	**Heurística de Modulação de Dica (HMD)**	Algoritmo que modula a emissão e a assertividade do feedback formativo com base na taxa de falha dos testes.

8.3 JSON Schema Oficial (`wok-problem.json`)

O arquivo `wok-problem.json` define a especificação sintática estrita (JSON Schema Draft-07) que todo artefato pedagógico do WOKDEX deve respeitar.

- **Arquivo Local:** `wok-problem.json` (28 KB)
- **URL de Produção:** `https://api.mundodocodigo.com.br/api/public/json/wok-problem.json`

8.3.1 Campos Obrigatórios (`required`)

```
id, name, slug, version, origin, description, editorial,
difficultyLevelId, timeComplexity, skills, solutions,
statements, successmsg, testScenarios
```

8.4 Os 4 Tipos de Teste (TestType)

Tipo	Visível ao Aluno?	Função Pedagógica no WOKDEX
`SAMPLE`	Sim	Testes dos exemplos do enunciado para depuração inicial do aluno.
`FUNCTIONAL`	Não	Testes que verificam a corretude funcional da solução em casos de borda normais.
`MISCONCEPTION`	Não	**Armadilha pedagógica** — testes projetados para capturar erros conceituais específicos (ex.: divisão inteira vs. ponto flutuante).
`PERFORMANCE`	Não	Testes com entradas volumosas para avaliar a complexidade assintótica de tempo e memória.

8.4.1 Subtipos de MISCONCEPTION

Subtipo	Alvo Pedagógico	Exemplo Clássico
`TYPE`	Tipo de dado inadequado (CS1)	Uso de `int` no lugar de `double` na divisão.
`FORMAT`	Formatação de saída incorreta (CS1)	Omissão de quebra de linha `\n` ou espaço sobressalente.
`STRATEGY`	Abordagem algorítmica insuficiente (CS2/CS3)	Resolução por força bruta onde se exige Programação Dinâmica.

8.5 Heurística de Modulação de Dica (HMD)

A HMD (Contribuição C4 do artigo) define **quando** o sistema deve emitir a dica pedagógica (`helpTip`):

Nível de Confiança	Condição do Teste	Ação do Juiz Online
Alta	Todos os testes do cenário falharam	Emite `helpTip` com máxima certeza do diagnóstico.
Moderada	Maioria dos testes falhou	Emite `helpTip` acompanhado de ressalva investigativa.
Baixa	Apenas 1 teste isolado falhou	Não emite dica (evita falso diagnóstico por erro de digitação pontual).

8.6 Arquivos Locais deste Diretório

Arquivo	Descrição
`wok-problem.json`	JSON Schema oficial do modelo WOKDEX (Draft-07) — 28 KB
`artigo-sbie-wokdex.pdf`	Artigo científico completo publicado no SBIE 2026 — 250 KB
`contribuicoes-artigo-sbie.md`	Detalhamento das 4 contribuições acadêmicas formais
`index.qmd`	Esta página de referência técnica e documentação