# RELATÓRIO DE SPRINTS

## Documento de Desenvolvimento do Projeto de Extensão 3/2025.2
## Projeto: EstudaAI (Equipe 2)

Alunos: Ingrid Lima Vieira, Henrique Chuvas de Sousa Donato, João Pedro Moreira de Abreu, Letícia Carneiro de Araújo, Gabriel dos Santos Rios Silveira

## 01. Histórico das Sprints:

### a. Sprint 1:

- Nessa etapa foram decididas as principais ferramentas que viriam a ser utilizadas pela equipe. Nessa Sprint, a principal entrega foi um mockup do projeto, já com as telas que implementariam as principais funcionalidades do sistema. A participação foi homogênea entre os integrantes do grupo.
- A ideia inicial do sistema era de um assistente geral de estudos que ofereceria resumos de aulas e materiais para auxiliar no acompanhamento das disciplinas. As principais funcionalidades incluíam a Geração de Resumos e Lembretes de Aulas. Foram também pensadas funcionalidades para dois perfis de usuário: Aluno e Professor, com o principal diferencial sendo que o professor poderia usar o serviço para avaliação da sua didática, assim como teria um espaço dedicado para cálculo de notas dos alunos. Esses últimos dois aspectos seriam deixados de lado após o primeiro Fim de Sprint.

### b. Sprint 2

A partir dessa Sprint, as atividades foram separadas em frentes, determinadas pela especialização de cada grupo.

## i. FRONT END

- Nessa fase, foram feitas alterações no protótipo, sobretudo no tocante à identidade visual, que foi reformulada por completo. Depois, iniciou-se a implementação das telas no Next.js (login, cadastro, transcrição/resumo, lembretes, etc.), já com alguma validação de login/cadastro. Um dos principais desafios dessa fase foi a integração do frontend com o backend, que não conseguimos finalizar até o fim da sprint. Assim, nessa fase fizemos muitos testes com dados “hardcoded” para verificar as funcionalidades implementadas (ex: criar cards de Lembretes com dados definidos no próprio código, ao invés de puxados do banco de dados).

## ii. BACK END

- Na sprint 2, foi feita a estrutura principal do back (pastas, rotas e tabelas), além da implementação dos primeiros endpoints. Também foram vistas maneiras de implementar autenticação de usuário, de integrar o backend com o serviço de IA para realizar as funcionalidades principais e de como salvar arquivos no banco. Como foco principal de entrega para o back foi estabelecido que seria a geração de resumos e lembretes a partir dos materiais enviados pelo usuário. A forma como definimos a saída da resposta da LLM foi simples, mas ainda foi um desafio fazer uma lógica para processar a saída no back e salvar tudo corretamente no banco de dados.

## iii. IA

- Do serviço de IA foram implementadas as primeiras versões do que viria a ser o pipeline de processamento dos dados enviados e geração do resumos. Os principais avanços a nível de IA generativa foram a padronização do formato de saída, já pensando-se na forma como seria recebido pelo Back End. O formato de saída estava

sendo definido como contexto para a LLM, o que resultou em um prompt mais volumoso, e resultou em algumas inconsistências, como o aparecimento de aspas nas extremidades da saída, que comprometem a saída. A capacidade de conversar com a LLM sobre o conteúdo do material ainda estava caminhando, sendo baseada puramente no contexto prévio do modelo, e não no material.

- Nessa etapa, o foco foi organizar a estrutura dos prompts a fim de gerar saídas que trouxessem um sumário claro do conteúdo passado. O que, felizmente, dado as capacidades dos modelos da família Gemini para questões educacionais, se mostrou pouco trabalhoso. Portanto, o maior trabalho dentro dessa demanda foi refinar a saída.

## c. Sprint 3

### i. FRONT END

- Mediante feedback da Sprint anterior, realizamos mais algumas alterações no protótipo (ex: mudança do ícone de Lembretes, que antes remetia muito à “notificação”), e finalizamos a implementação das telas que faltavam, como a tela de disciplinas. Além disso, conseguimos realizar a integração entre o front e o back. Uma das principais dificuldades foi integração com a tela de transcrição/resumo, que era a tela que continha o chat com a LLM. Durante toda a sprint fizemos alguns ajustes nas telas à medida que víamos necessidade, como a adição de um botão de deletar disciplina, ou a mudança do tamanho dos cards de tópicos.

### ii. BACK END

- Após implementar a geração de resumos e lembretes, foram finalizados os endpoints necessários para mostrar as informações no frontend, e foram ajustados detalhes de implementação de alguns endpoints já feitos. A partir disso, o foco foi colocado na geração do plano de resumos (ponto que foi discutido na apresentação) e no

ajuste das tabelas do banco, para acomodar a formatação do plano no frontend. No entanto, para isso, era necessário enviar os arquivos para o serviço de IA processá-los e salvar as transcrições no banco. Após integrar corretamente a saída das transcrições com o back, foi feita a lógica para integrar o plano de estudos criado pela LLM e salvar no banco. Da forma que foi estruturado o plano, é necessário acessar múltiplas tabelas, mas com a vantagem de conseguir formatar melhor na hora de enviar para o front. Feito o plano de estudos, iniciou-se a implementação da geração de embeddings e chunking dos materiais. Inicialmente, toda a geração e armazenamento de embeddings ia ficar no serviço de IA, mas foi decidido que seria mais adequado que as embeddings fossem salvas no próprio back. Após as embeddings funcionarem, foi implementada a lógica de integrar as mensagens do usuário mandadas pelo front, enviá-las juntamente com os chunks relevantes (achados pela busca vetorial com as embeddings) para a resposta e a mensagem, representando assim uma forma rudimentar de RAG. Os últimos passos do back foram ajustar esta lógica para que a LLM conseguisse responder às perguntas do usuário com base no material enviado, e a finalização de algumas rotas de edição e deleção de itens.

### iii. IA

- Durante a etapa final, foram testadas as recomendações feitas a respeito dos prompts durante o último feedback para avaliar como estes melhorariam a saída, em especial, foi testado a geração do prompt em inglês e analisada a possibilidade de usar um “JSON Mode”, no modelo, tal qual é visto na API da OpenAI. Os resultados não se mostraram significativos para a aplicação.
- A geração de prompt em inglês, com comando específico para gerar a resposta em português ofereceu resultados pouco relevantes para considerar uma mudança total dos demais prompts, se tornando apenas um artefato no código. A aplicação dessa técnica foi um achado interessante, porém, para o

escopo do projeto e os custos por cada saída, o ganho foi pouco perceptível.

- Por outro lado, a geração por JSON Mode se mostrou como uma característica somente dos Modelos GPT. Porém, uma alternativa que foi implementada foi o uso de um modelo de validação de saída por meio de Pydantics, uma biblioteca de validação de dados, que, quando anexada a chain, promoveu resultados mais consistentes. No entanto, como o objetivo era reduzir o tamanho do prompt, e, ao mesmo tempo, oferecer uma saída limpa, infelizmente o modelo deixou a desejar bastante nesse aspecto.
- Adicionalmente, o prompt do plano de estudos foi refinado, para atender a sua função de orientação. A criação de regras específicas para a geração da checklist e decisão do que são “tópicos relevantes” dentro do contexto da aplicação foi um desafio menor durante essa questão, visto que o Gemini insistia em oferecer metas bastante redundantes e até mesmo ignorantes do contexto do aluno como “Estudar tópicos dados”, o que no final, gerava um plano extenso e redundante. Uma técnica que foi utilizada para condicionar uma melhor saída, foi utilizar exemplos oferecidos nos mockups como exemplos positivos, e as saídas pouco inspiradas, geradas pela LLM como exemplos negativos, o que gerou resultados muito mais satisfatórios durante os testes.
- Como o foco principal dessa etapa foi a integração dos módulos, as principais dificuldades vieram da comunicação entre o Back e o serviço de IA, que constantemente precisavam trocar informações, o que se intensificou durante a implementação da funcionalidade de chat, a qual exigia algum contexto para responder o usuário adequadamente. Coisas como armazenamento e geração de embeddings, busca dos chunks relevantes, e adição de contexto de mensagens prévias, foram projetadas nessa etapa também.
- Para a conversa com a LLM, como comentado pela seção do back, foi aplicado um RAG rudimentar com uma memória pequena das conversas. Com as informações relevantes sendo puxadas do back, a

função de chat receberia 3 entradas: a própria mensagem, que serviria para a busca dos termos relevantes, os chunks relevantes à pergunta, e uma memória pequena que consistia dos 3 últimos turnos de conversa com a LLM, para contextualizá-la do assunto durante cada chamada, como uma versão hard coded da memória de curto prazo. Infelizmente a implementação de memória não pode ser melhor explorada, visto que esse foi um aspecto visto como uma prioridade menor.

- Nessa etapa também, devido aos desafios na comunicação, foi decidido por descontinuar o suporte ao processamento de áudio, dadas as limitações das ferramentas escolhidas para esse propósito.
- Atualmente, para o entregável final, a fim de não perder nenhuma informação, ambos os resumos e planos de estudos são gerados com a junção dos materiais oferecidos pelo usuário. A escolha de tal foi feita, como dito, com a intenção de não haver perda das informações durante a transcrição, especialmente dado que algumas informações seriam melhor aproveitadas em uma saída que na outra (Ex.: Fazer as listas recomendadas possui mais valor como uma meta dentro do Plano de Estudos, do que como um comentário avulso dentro do resumo).