

# Roteiro de Apresentação Oral

1	<p>Introdução</p> <p>Apresentação do app SkillSeeds e do objetivo da entrega (adicionar features dinâmicas).</p>
2	<p>O Desafio</p> <p>Tínhamos 4 novas entidades (Lesson, Step, Progress, Achievement) e 2 novas features. Como implementar isso rápido e mantendo o padrão de arquitetura (DTO/Mapper)?</p>
3	<p>A Solução (Estudo Guiado com IA)</p> <p>Decidimos usar a IA do VS Code como uma assistente de "pair programming". Nós, como desenvolvedores, focamos na <b>arquitetura</b> (definindo o fluxo de dados, os contratos das entidades e as regras do Supabase). A IA focou na <b>implementação</b> (gerando o código repetitivo dos repositórios, providers e telas).</p>
4	<p>Demo - Feature 1 (Tela de Lições)</p> <p><b>Mostrar o Prompt 1 (Repositório):</b> "Este foi o prompt que demos à IA. Pedimos um repositório que filtrasse por track_id e usasse nosso Mapper."</p> <p><b>Mostrar o Código Gerado (LessonRepository):</b> "Este é o resultado. A IA gerou o try/catch, a chamada .select().eq() e o loop de mapeamento, economizando tempo."</p> <p><b>Explicar o Código:</b> "O importante é que a IA seguiu nosso padrão: (data as List).map((map) =&gt; LessonMapper.toEntity(LessonDTO.fromMap(map)))."</p> <p><b>Mostrar o Prompt 3 (UI):</b> "Também pedimos a tela, especificando o uso do <b>ref.watch</b> e do .when()."</p> <p><b>Rodar o App (Demo):</b> Executar o app, clicar na Trilha "Design" e mostrar a tela de lições funcionando.</p>
5	<p>Decisões de Design</p> <p>"Usamos um <b>FutureProvider.family</b> para a tela de lições, pois ele é ideal para buscar dados que <i>dependem</i> de um parâmetro (o trackId). Para as Conquistas, um FutureProvider simples foi o suficiente."</p>
6	<p>Segurança e Ética</p> <p>"A IA não teve acesso a nenhuma chave de API (que estão no .env, ignorado pelo Git). Os prompts só continham nomes de tabelas e lógica de UI. A segurança dos dados é garantida pelas Políticas de RLS que criamos no Supabase, permitindo apenas leitura pública."</p>
7	<p>Conclusão</p> <p>"A IA foi uma ferramenta poderosa de produtividade. Ela não substituiu a necessidade de arquitetura, mas acelerou drasticamente a implementação, permitindo focar em <b>o quê</b> fazer, e não em <b>como</b> digitar o código."</p>