Faculdade XPe

RELATÓRIO

PROJETO APLICADO

XP Educação

Relatório do Projeto Aplicado

Plataforma de Engenharia de Dados para Análise de Performance e Retenção no Setor Fitness

Murilo Silva Felipe

Orientador(a): Daniella Pimenta Brito Alves Alves

28 de Junho de 2025


XPe

Murilo Silva Felipe

XP EDUCAÇÃO

RELATÓRIO DO PROJETO APLICADO

Plataforma de Engenharia de Dados para Análise de Performance e Retenção no Setor Fitness

Relatório de Projeto Aplicado desenvolvido para fins de conclusão do curso de Engenharia e Arquitetura de Dados.

Orientador (a): Daniella Pimenta Brito Alves Alves

Anastácio - MS

28 de Junho de 2025

Sumário

1. CANVAS do Projeto Aplicado

1.1 Desafio

O Brasil, apesar de figurar como o segundo maior mercado de academias do mundo (Fonte: IHRSA), enfrenta um paradoxo crítico: uma taxa de evasão que se aproxima de 60% nos primeiros três meses (Fonte: ACAD Brasil). Este cenário expõe uma lacuna fundamental entre a oferta de serviços de fitness e a capacidade de engajar e reter clientes a longo prazo. A crescente adoção de tecnologias, como aplicativos de exercício e wearables (Fonte: ACSM; Statista), gerou uma abundância de dados de saúde, mas de forma desordenada e fragmentada.

O desafio central deste projeto, portanto, não é apenas tecnológico, mas estratégico e de dados. Ele consiste em transformar a grande quantidade de dados de saúde gerados de forma isolada em um ativo unificado, portátil e acionável para o usuário. Atualmente, as informações de treino, nutrição e sono — pilares interdependentes da saúde — residem em silos, impedindo uma visão holística tanto para o aluno quanto para os profissionais que o acompanham.

Este projeto se propõe a resolver o paradoxo da hiperconectividade e da desintegração de dados, desenvolvendo uma plataforma que capacite o indivíduo com o controle de sua jornada de bem-estar. O objetivo é criar uma solução que não só ataque as causas da evasão, como a falta de motivação e de resultados perceptíveis, mas que também crie um novo modelo de colaboração de dados no ecossistema de saúde e fitness. A especificação completa deste desafio é detalhada nas seções a seguir.

1.1.1 Análise de Contexto

O Brasil se posiciona como um dos maiores mercados de fitness do mundo, ocupando a segunda posição em número de academias, com mais de 31 mil unidades (Fonte: IHRSA Global Report 2023). Apesar da magnitude do mercado, o setor enfrenta um desafio crônico e bem documentado: as altas taxas de evasão. Estudos indicam que cerca de 60% dos alunos de academias no Brasil desistem nos primeiros três meses (Fonte: Pesquisa ACAD Brasil). A falta de motivação e a dificuldade em visualizar resultados concretos são consistentemente citadas como causas principais para este abandono, superando até mesmo a questão do custo.

Neste cenário, a tecnologia emerge como o principal vetor de transformação. A pesquisa "Worldwide Survey of Fitness Trends" do American College of Sports Medicine (ACSM) aponta a "Tecnologia Vestível" (Wearable Technology) como a tendência número 1 para o setor por vários anos consecutivos, seguida de perto por "Aplicativos de Exercício para Dispositivos Móveis". Isso demonstra uma clara demanda do consumidor por monitoramento de dados e feedback em tempo real. O mercado de aplicativos de fitness no Brasil reflete essa tendência, com projeção de receita superior a US$ 400 milhões anuais e uma base de mais de 45 milhões de usuários (Fonte: Statista Digital Market Outlook).

Contudo, o ecossistema digital atual é marcado pela fragmentação. As soluções existentes, sejam aplicativos proprietários de grandes redes de academias ou plataformas para personal trainers, criam silos de informação. O histórico de progresso do aluno fica restrito a um único serviço, dificultando a portabilidade e a continuidade do acompanhamento. Adicionalmente, essas plataformas raramente integram os três pilares da saúde preconizados pela Organização Mundial da Saúde (OMS): atividade física, nutrição e qualidade do sono. A literatura científica é vasta ao correlacionar a qualidade do sono com a recuperação muscular e o desempenho cognitivo, e a nutrição com a composição corporal e a performance atlética, tornando a visão isolada de apenas um desses fatores fundamentalmente incompleta.

Esta lacuna representa uma oportunidade estratégica. O setor de healthtechs no Brasil está em plena expansão, tendo movimentado mais de US$ 530 milhões em investimentos nos últimos anos (Fonte: Relatórios Distrito). No entanto, poucas soluções se dedicam a resolver o problema da integração de dados de bem-estar de forma agnóstica e centrada no usuário. O desafio, portanto, reside em desenvolver uma plataforma que unifique esses dados, capacitando o aluno com a posse e a portabilidade de seu histórico e permitindo que os profissionais de saúde colaborem de forma mais eficaz, atacando diretamente as causas da evasão e promovendo resultados mais sustentáveis.

1.1.2 Personas

Para este projeto, foram desenvolvidas duas personas principais que representam os públicos-alvo da solução. A criação destes perfis se baseia nos requisitos do Mapa de Empatia e na construção de um biótipo detalhado para uma compreensão aprofundada dos usuários.

Persona 1: O Aluno Focado em Dados

- Biótipo
    - **Nome:** Bruno Alves
    - **Idade:** 29 anos
    - **Profissão:** Analista de Marketing Digital
    - **Características Comportamentais:** Bruno é movido por dados e métricas em sua vida profissional, utilizando gráficos e relatórios para tomar decisões. Ele busca aplicar essa mentalidade analítica à sua vida pessoal, mas encontra dificuldades no universo fitness. É usuário proficiente de tecnologia e espera que as ferramentas digitais sejam eficientes e integradas.
- Mapa de Empatia
    - **O que ele pensa e sente?** Ele pensa: "Se não consigo medir meu progresso, como saberei que estou no caminho certo?". Sente-se frustrado ao perceber que sua ficha de treino é a mesma há meses e ao perder todo o seu histórico de evolução ao considerar uma troca de academia.
    - **O que ele vê?** Ele vê amigos em redes sociais como o Strava compartilhando mapas de corrida, recordes pessoais e evolução de performance. Na academia, ele vê instrutores sobrecarregados, sem tempo para um acompanhamento individualizado.
    - **O que ele fala e faz?** Ele fala com amigos sobre a dificuldade de se manter motivado sem ver progresso claro. Tenta anotar suas cargas no bloco de notas do celular, mas o processo é desorganizado e ele acaba desistindo. Busca vídeos de execução de exercícios no YouTube durante o treino.
    - **O que ele escuta?** Ele escuta em podcasts sobre alta performance que "o que não é medido, não é gerenciado". Escuta dos instrutores a frase padrão "qualquer dúvida, é só chamar", mas hesita em perguntar por vê-los sempre ocupados.
    - **Suas Dores:** A perda de todo o histórico de dados ao trocar de serviço; a falta de uma visão unificada que conecte seu treino, sua nutrição e seu sono; a ausência de feedback visual sobre sua evolução de performance.

- **Suas Necessidades:** Uma plataforma única e portátil para consolidar seu histórico de saúde; visualizar seu progresso de forma gráfica e intuitiva; ter acesso a instruções claras (vídeos) para executar os treinos com segurança.

Persona 2: A Profissional de Educação Física sobrecarregada

- Biótipo
    - **Nome:** Carla Medeiros
    - **Idade:** 36 anos
    - **Profissão:** Educadora Física (Personal Trainer)
    - **Características Comportamentais:** Carla é apaixonada por ajudar seus alunos a atingirem seus objetivos. É extremamente dedicada, mas sente-se constantemente sobrecarregada por tarefas administrativas que consomem o tempo que ela gostaria de dedicar à estratégia de treino e ao acompanhamento próximo de seus clientes de consultoria online.
- Mapa de Empatia
    - **O que ela pensa e sente?** Ela pensa: "Eu poderia entregar um resultado muito melhor se passasse menos tempo montando planilhas e mais tempo analisando o desempenho dos meus alunos". Sente-se ansiosa por não saber se seus alunos online estão, de fato, cumprindo o plano ou executando os exercícios corretamente.
    - **O que ela vê?** Ela vê seu WhatsApp lotado com dúvidas de alunos, feedbacks e cobranças, tudo de forma desorganizada. Vê concorrentes utilizando plataformas digitais para escalar seus negócios e oferecer um serviço mais profissional.
    - **O que ela fala e faz?** Ela passa horas nos finais de semana criando e ajustando treinos em planilhas de Excel e enviando por PDF. Utiliza o WhatsApp para se comunicar, o que gera um fluxo constante de interrupções e dificulta o resgate de informações importantes.
    - **O que ele escuta?** Ela escuta de seus alunos pedidos por "um aplicativo para facilitar o acompanhamento". Escuta de colegas sobre as vantagens e custos das plataformas de gestão existentes no mercado.
    - **Suas Dores:** O tempo excessivo gasto com trabalho manual e repetitivo (criação de planilhas); a falta de visibilidade sobre a adesão e o progresso real dos alunos à distância; a dificuldade em gerenciar a comunicação com dezenas de clientes de forma eficiente.
    - **Suas Necessidades:** Uma ferramenta centralizada para otimizar a prescrição e o ajuste de treinos; um canal de comunicação estruturado com seus alunos; acesso rápido ao histórico de cargas e feedbacks de cada aluno para tomar decisões mais rápidas e informadas.

Persona 3: A Nutricionista Colaborativa

- **Biótipo**
    - **Nome:** Dra. Fernanda Lima
    - **Idade:** 33 anos
    - **Profissão:** Nutricionista Clínica e Esportiva
    - **Características Comportamentais:** Fernanda é extremamente analítica e acredita em uma abordagem baseada em evidências. Ela se esforça para criar planos alimentares que se alinhem perfeitamente com a rotina e o gasto calórico de seus pacientes, mas se frustra com a qualidade dos dados que recebe.
- **Mapa de Empatia**
    - **O que ela pensa e sente?** Ela pensa: "Como posso calcular o gasto calórico para a dieta se o paciente me diz que 'treinou pesado', mas não sabe os detalhes?". Ela se sente insegura ao basear suas recomendações em informações vagas ou imprecisas relatadas pelos pacientes.
    - **O que ela vê?** Ela vê seus pacientes trazendo diários alimentares preenchidos de forma inconsistente. Vê o sucesso de colegas que atuam em equipes multidisciplinares com comunicação fluida.
    - **O que ela fala e faz?** Ela passa grande parte da consulta tentando extrair do paciente detalhes sobre sua rotina de treinos. Pede que eles "tentem anotar tudo direitinho" para o próximo retorno.
    - **O que ela escuta?** Ela escuta seus pacientes dizerem: "Acho que treinei umas 3 ou 4 vezes na semana, não tenho certeza". Escuta em congressos sobre a importância da análise de dados integrados para a nutrição de precisão.
    - **Suas Dores:** A dependência de dados de atividade física auto-relatados, que são frequentemente imprecisos; a dificuldade em ajustar planos alimentares em tempo real com base nas variações de treino do paciente; a falta de um canal de comunicação direto e profissional com o educador físico do paciente.
    - **Suas Necessidades:** Acesso a dados de treino confiáveis e atualizados de seus pacientes (com consentimento); uma plataforma que permita a colaboração com outros profissionais da saúde; otimizar seu tempo de consulta, focando em estratégia nutricional em vez de investigação de dados.

Persona 4: O Dono de Academia Focado em Retenção

- **Biótipo**
    - **Nome:** Ricardo Mendes
    - **Idade:** 48 anos
    - **Profissão:** Empresário, proprietário de uma academia de médio porte
    - **Características Comportamentais:** Ricardo é focado nos resultados do negócio. Ele analisa constantemente as métricas de aquisição e, principalmente, de evasão (churn) de clientes. Ele entende que reter um cliente é mais barato e lucrativo do que adquirir um novo.
- **Mapa de Empatia**
    - **O que ele pensa e sente?** Ele pensa: "Todo mês eu gasto uma fortuna com marketing para trazer gente nova, mas a porta dos fundos parece uma porteira aberta". Ele se sente frustrado ao ver seu esforço de aquisição ser minado por uma alta taxa de cancelamento.
    - **O que ele vê?** Ele vê o relatório financeiro e o número de matrículas canceladas todo fim de mês. Vê os alunos novatos andando perdidos pela academia e, semanas depois, não os vê mais. Vê academias concorrentes oferecendo "experiências" e "tecnologia" como diferenciais.
    - **O que ele fala e faz?** Ele conversa com sua equipe sobre a importância de "dar atenção aos novatos". Cria promoções e pacotes de longo prazo para tentar "prender" o cliente.
    - **O que ele escuta?** Ele escuta de sua equipe que "os instrutores não dão conta, é muita gente pra pouco professor". Escuta de clientes que saem que "não estavam vendo resultado" ou "estavam desmotivados".
    - **Suas Dores:** Alta taxa de evasão de clientes, especialmente nos primeiros 90 dias; o alto custo de aquisição de novos clientes; a dificuldade em criar um diferencial competitivo que não seja apenas o preço.
    - **Suas Necessidades:** Ferramentas que aumentem o engajamento e a percepção de valor por parte do aluno; uma forma de identificar alunos em risco de evasão antes que eles cancelem o plano; oferecer um serviço que fortaleça o relacionamento do aluno com a academia.

Persona 5: A Aluna Iniciante e Desmotivada

- **Biótipo**
    - **Nome:** Mariana Costa
    - **Idade:** 24 anos
    - **Profissão:** Assistente Administrativa
    - **Características Comportamentais:** Mariana decidiu começar a academia por recomendação médica e para melhorar a autoestima. Ela se sente intimidada pelo ambiente, não tem familiaridade com os exercícios e se desmotiva facilmente quando não vê resultados imediatos ou quando se sente perdida.
- **Mapa de Empatia**
    - **O que ela pensa e sente?** Ela pensa: "Todo mundo aqui parece saber o que está fazendo, menos eu". Ela se sente ansiosa, julgada e inadequada. A ideia de ir para a academia gera mais estresse do que bem-estar.
    - **O que ela vê?** Ela vê pessoas levantando cargas pesadas, usando máquinas complexas e com físicos que parecem inalcançáveis para ela. Vê a ficha de treino como um papel cheio de nomes que ela não entende.
    - **O que ela fala e faz?** Ela fala para as amigas que "começou a academia", mas evita dar detalhes. Quando vai, muitas vezes acaba fazendo apenas esteira e transport porque tem vergonha de tentar usar os aparelhos de musculação.
    - **O que ela escuta?** Ela escuta o instrutor passar o treino rapidamente e dizer "é fácil, você pega o jeito". Escuta o barulho de pesos caindo e pessoas conversando em grupos, o que a faz se sentir ainda mais deslocada.
    - **Suas Dores:** O sentimento de intimidação e "não pertencimento" ao ambiente da academia; a falta de um guia claro e simples sobre o que fazer; a ausência de pequenas vitórias que a mantenham motivada no início.
    - **Suas Necessidades:** Um plano de treino "à prova de erros", com instruções visuais e passo a passo; um senso de progresso focado no hábito (ex: "Parabéns, você completou sua segunda semana!") e não apenas na performance; sentir-se acolhida e guiada, mesmo que de forma digital.

1.1.3 Justificativas

A seção a seguir apresenta os fatores que justificam o desenvolvimento do projeto, detalhando os benefícios futuros esperados e a proposta de valor da solução. A justificativa para o investimento neste projeto se baseia na sua capacidade de gerar valor para todos os atores do ecossistema de saúde e bem-estar.

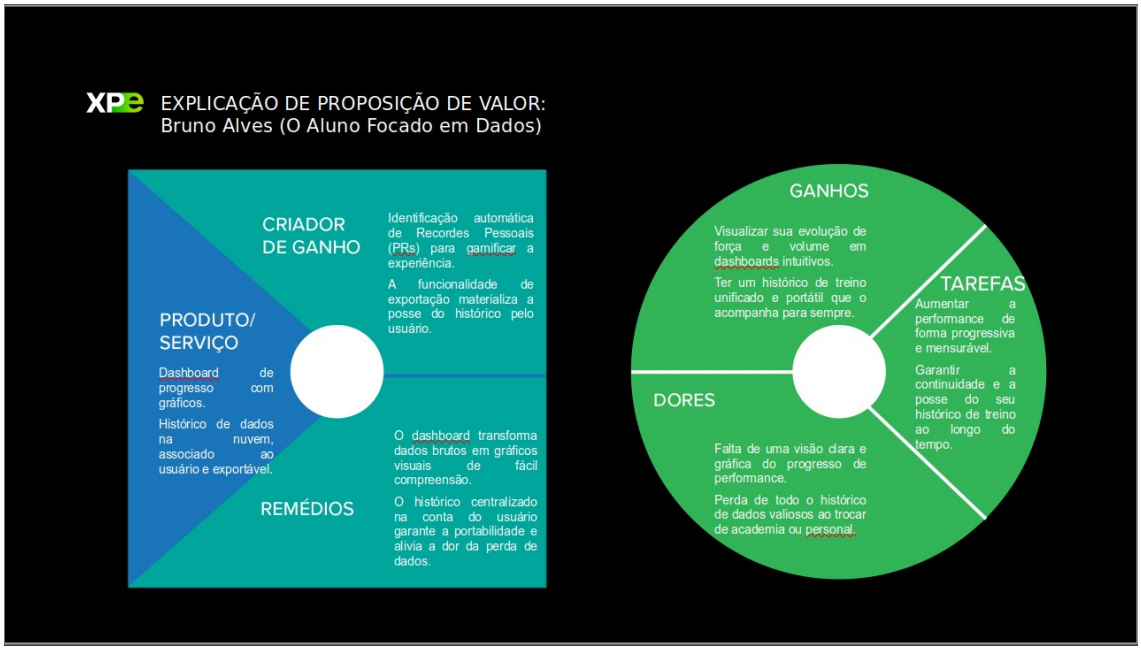
- **Proposta de Valor:** Entregar uma plataforma de saúde integrada e portátil que centraliza os dados de treino, nutrição e sono, empoderando o aluno com o controle de seu histórico e promovendo uma colaboração de dados eficaz entre os diferentes profissionais que o acompanham.

Fatores de Justificativa:

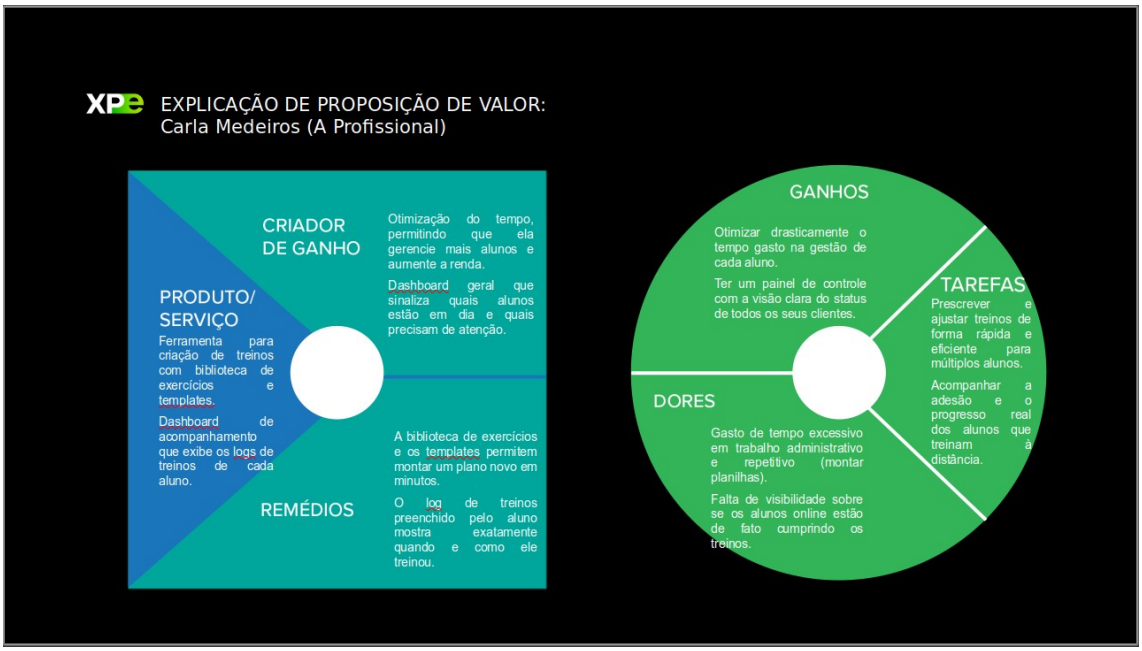
- **Redução da Evasão e Aumento da Retenção de Alunos:**
    - **Cenário Atual:** A alta taxa de evasão, especialmente de alunos iniciantes como a persona Mariana, é a maior dor do gestor de academia, Ricardo. A falta de motivação, a intimidação e a ausência de resultados perceptíveis são as principais causas.
    - **Benefício Futuro Esperado:** A plataforma atuará como uma ferramenta de engajamento, oferecendo um guia claro para a iniciante Mariana e métricas de progresso visuais para o aluno focado em dados, Bruno. Ao aumentar a percepção de valor e o senso de progresso, espera-se diminuir significativamente a evasão, impactando positivamente a receita recorrente das academias.
- **Otimização do Trabalho e Escalabilidade para Profissionais:**
    - **Cenário Atual:** A instrutora Carla e a nutricionista Fernanda perdem um tempo precioso com tarefas administrativas e repetitivas, como montar planilhas e tentar coletar dados imprecisos de seus clientes. Isso limita sua capacidade de atendimento e a qualidade do serviço prestado.
    - **Benefício Futuro Esperado:** A solução automatizará a coleta e a apresentação de dados, liberando os profissionais para focarem na análise e na estratégia. Isso não representa apenas uma redução de custos operacionais em tempo de trabalho, mas também possibilita que eles escalem seus negócios, atendendo mais clientes com maior qualidade e criando novas formas de gerar receitas.
- **Melhora na Eficácia do Acompanhamento de Saúde:**
    - **Cenário Atual:** A falta de comunicação e de dados integrados entre os profissionais (Carla e Fernanda) impede um acompanhamento verdadeiramente holístico. As decisões são tomadas com base em informações parciais ou auto-relatadas, o que pode comprometer os resultados do aluno Bruno.

- **Benefício Futuro Esperado:** Ao permitir o compartilhamento consentido de dados, a plataforma possibilita que a nutricionista crie planos mais precisos com base no gasto calórico real, e que a instrutora ajuste o treino ao perceber, por exemplo, uma queda na qualidade do sono do aluno. Esse impacto social se traduz em melhores resultados de saúde para o usuário final.

- **Empoderamento e Portabilidade de Dados para o Aluno:**
  - **Cenário Atual:** O histórico de saúde e esforço do aluno pertence à academia ou à plataforma do personal, não ao próprio aluno. Ao mudar de serviço, Bruno perde todo o seu valioso histórico.
  - **Benefício Futuro Esperado:** O projeto posiciona o aluno como o verdadeiro dono de seus dados, garantindo portabilidade e continuidade. Isso representa um diferencial competitivo significativo e alinha a solução a tendências de privacidade e empoderamento do consumidor.



EXPLICAÇÃO DE PROPOSIÇÃO DE VALOR: Carla Medeiros (A Profissional)



EXPLICAÇÃO DE PROPOSIÇÃO DE VALOR: Mariana Costa (A Aluna Iniciante)

EXPLICAÇÃO DE PROPOSIÇÃO DE VALOR: Dra. Fernanda Lima (A Nutricionista Colaborativa)

EXPLICAÇÃO DE PROPOSIÇÃO DE VALOR: Ricardo Mendes (O Dono de Academia)

1.1.4 Hipóteses

Esta seção apresenta as hipóteses centrais que servem como premissa para o desenvolvimento da solução. Cada hipótese foi formulada a partir da análise de contexto e das necessidades das personas , transformando as observações em sentenças afirmativas que serão validadas durante a fase de experimentação do projeto.

- **Hipótese 1: Hipótese de Engajamento e Retenção**
    - **Observação:** A aluna iniciante Mariana se sente perdida e desmotivada, enquanto o dono da academia, Ricardo, sofre com a alta evasão.
    - **Afirmação (Hipótese):** Acreditamos que, ao fornecer uma experiência de onboarding guiada e um sistema de acompanhamento de hábitos (frequência) em vez de performance (cargas), a retenção de alunos iniciantes nos primeiros 90 dias aumentará em 25%.
    - **Problema:** Aumento do Engajamento e Retenção de Alunos Iniciantes
        - **Problema 1.1:** Alunos iniciantes não sabem como manusear os equipamentos de musculação e sentem vergonha de perguntar.
        - **Problema 1.2:** A ficha de treino em papel ou PDF é confusa, pouco intuitiva e não gera um senso de progresso visual que motive o aluno.
        - **Problema 1.3:** O ambiente da academia é percebido como intimidante por novos membros, que se sentem deslocados e julgados.
- **Hipótese 2: Hipótese de Valor para o Profissional**
    - **Observação:** A instrutora Carla gasta horas em trabalho administrativo com planilhas, e a nutricionista Fernanda depende de dados imprecisos.
    - **Afirmação (Hipótese):** Acreditamos que, ao utilizar uma plataforma integrada para prescrição de treinos e visualização de dados dos alunos, os profissionais de saúde (educadores físicos e nutricionistas) conseguirão reduzir em 4 horas semanais o tempo gasto com tarefas administrativas, permitindo focar em atividades de maior valor.
    - **Problema:** Otimização do Tempo e Valor para Profissionais
        - **Problema 2.1:** Profissionais de educação física gastam tempo excessivo em trabalho administrativo, digitando os mesmos exercícios repetidamente ao montar planos de treino.

- **Problema 2.2:** A comunicação com dezenas de alunos via aplicativos de mensagem (como o WhatsApp) é desorganizada, ineficiente e mistura vida pessoal com profissional.
- **Problema 2.3:** Nutricionistas baseiam seus planos em dados de atividade física auto-relatados pelos pacientes, que são frequentemente imprecisos ou incompletos.

- **Hipótese 3: Hipótese de Colaboração e Eficácia**
  - **Observação:** O aluno Bruno não consegue compartilhar seus dados de forma eficiente, e a nutricionista Fernanda não tem acesso ao histórico de treino real para otimizar a dieta.
  - **Afirmação (Hipótese):** Acreditamos que, ao permitir o compartilhamento de dados consentido entre diferentes profissionais da saúde dentro da plataforma, a eficácia do acompanhamento multidisciplinar aumentará, resultando em uma melhoria de 15% nos indicadores de performance e adesão do aluno em um período de 6 meses.
  - **Problema:** Eficácia do Acompanhamento Multidisciplinar
    - **Problema 3.1:** Não existe um canal de comunicação formal e eficiente para que o instrutor físico e o nutricionista de um mesmo aluno possam trocar informações e alinhar estratégias.
    - **Problema 3.2:** Decisões importantes sobre o plano do aluno (aumento de carga ou ajuste de dieta) são tomadas com base em dados isolados, sem considerar a visão completa de sua saúde (treino, sono, nutrição).
- **Hipótese 4: Hipótese de Portabilidade e Valor Percebido**
  - **Observação:** O aluno focado em dados, **Bruno**, frustra-se ao perder todo o seu histórico de progresso ao trocar de academia ou profissional.
  - **Afirmação (Hipótese):** Acreditamos que, ao oferecer ao aluno a posse e a portabilidade total de seu histórico de treino, a percepção de valor da plataforma aumentará, resultando em 30% dos usuários considerando este o principal diferencial para escolher ou permanecer em um serviço de acompanhamento.
  - **Problema:** Valorização da Portabilidade e Posse dos Dados
    - **Problema 4.1:** Alunos com meses ou anos de histórico de progresso perdem todos os seus dados ao trocar de personal trainer ou de academia, gerando grande frustração e descontinuidade.
    - **Problema 4.2:** O aluno não se sente o verdadeiro "dono" de seus dados de saúde, percebendo que seu esforço está "preso" a um serviço.

- **Hipótese 5: Hipótese de Gestão Proativa**
    - **Observação:** O dono de academia, Ricardo, só percebe a evasão de um cliente quando ele já cancelou o plano, sem ter ferramentas para agir proativamente.
    - **Afirmação (Hipótese):** Acreditamos que, ao analisar os dados de frequência e progresso dos alunos, é possível criar um "índice de engajamento" que identifique, com 70% de precisão, os alunos com alto risco de evasão no próximo mês, permitindo ações de retenção direcionadas.
    - **Problema: Capacidade de Gestão Proativa para Academias**
        - **Problema 5.1:** Gestores de academia identificam a evasão de um cliente apenas de forma reativa, ou seja, depois que o cancelamento do plano já ocorreu.
        - **Problema 5.2:** Não há indicadores claros que sinalizem quais alunos estão com baixo engajamento e, portanto, em alto risco de abandonar o serviço no curto prazo.

Problema	Criterização			Somatório (G x U x T)	Priorização (º)	Grau de Risco (Baixo, Médio, Alto)
	Gravidade (G) 1 a 5	Urgência (U) 1 a 5	Tendência (T) 1 a 5			
1.1	5	5	4	100	1	Alto
1.2	5	5	3	75	2	Médio
4.1	4	4	4	64	3	Médio
2.1	4	3	4	48	4	Médio
5.2	4	2	4	32	5	Médio
1.3	5	2	3	30	6	Médio
4.2	3	3	3	27	7	Médio
2.3	3	2	4	24	8	Baixo
2.2	3	2	3	18	9	Baixo
3.1	3	2	2	12	10	Baixo
3.2	4	1	3	12	11	Baixo
5.1	4	1	3	12	12	Baixo

Justificativa (Foco no MVP)

Problema 1.1:

G(5): Causa direta de evasão e risco de lesão.

U(5): É um problema do Dia 1 do cliente. Se não for resolvido, a chance de ele não voltar é altíssima.

T(4): A tendência é o aluno evitar os equipamentos e ter um treino ineficaz, levando ao abandono.

Problema 1.2:

G(5): Principal "concorrente" do app e fonte de frustração.

U(5): É o problema central a ser resolvido pelo app no dia a dia.

T(3): O problema se mantém, mas não necessariamente piora sozinho; apenas continua sendo ruim.

Problema 4.1:

G(4): Dor muito alta para o aluno engajado (Bruno), mas menos impactante para a sobrevivência inicial do negócio do que a evasão dos novatos.

U(4): Urgente para provar o diferencial do produto.

T(4): A frustração aumenta a cada novo treino registrado que o aluno sabe que pode perder.

Problema 2.1:

G(4): Grande ineficiência para a persona Carla.

U(3): É uma dor real, mas os profissionais já têm um "jeitinho" de fazer (copiar/colar), então não impede o trabalho de continuar.

T(4): Piora com o aumento de clientes.

Problema 5.2:

G(4): Grande dor para o gestor Ricardo.

U(2): Feature de valor para o negócio, mas só se torna útil após ter uma base de usuários e dados coletados. Não é uma prioridade para o MVP inicial focado no usuário final.

T(4): A tendência é que a falta desses indicadores se torne cada vez mais prejudicial. À medida que o mercado se torna mais competitivo e a gestão mais orientada a dados, não ter essa ferramenta representará uma desvantagem estratégica crescente para o gestor Ricardo.

Problema 1.3:

G(5): Problema gravíssimo, mas

U(2): a capacidade de um app resolver um problema ambiental/cultural é mais limitada e de longo prazo. A urgência de uma solução pelo app é menor.

T(3): A tendência é que a percepção de intimidação do aluno iniciante Mariana se consolide com o tempo se nada for feito, levando ao abandono. O problema não se resolve sozinho; ele se repete a cada novo membro inseguro que entra, mantendo a evasão constante.

Problema 4.2:

G(3): Conceito mais abstrato que o problema 4.1.

U(3): Importante para o marketing do produto.

T(3): A consciência sobre posse de dados tende a crescer.

Problema 2.3:

G(3): Dor real para a nutricionista Fernanda.

U(2): Validação de uma persona secundária, pode ser feita após o core do produto estar validado.

T(4): A imprecisão dos dados é um problema crescente.

Problema 2.2:

G(3): Ineficiência para o profissional.

U(2): Existem alternativas (e-mail, etc.).

T(3): O problema piora com mais clientes.

Problema 3.1:

G(3): Dor de um cenário de alta maturidade.

U(2): Feature avançada.

T(2): O problema se mantém, mas não é a maior das dores.

Problema 3.2:

G(4): O cerne do problema holístico.

U(1): A urgência é baixa, pois primeiro é preciso coletar os dados de forma eficiente para depois integrá-los.

T(3): A tendência é que este problema se agrave em consequência. Com a proliferação de mais apps e wearables, mais dados serão gerados de forma isolada, tornando a "visão 360º" da saúde do aluno cada vez mais difícil e fragmentada, o que aumenta a chance de erros ou acompanhamentos ineficazes.

Problema 5.1:

G(4): Dor de negócio.

U(1): Similar ao 5.2, depende de uma base de dados robusta para ser resolvido.

T(3): A tendência é a manutenção do prejuízo. O problema se repetirá mês após mês. Se a concorrência se tornar mais eficiente na retenção, o impacto negativo desta gestão reativa tende a piorar, tornando a academia menos competitiva.

A análise de priorização via Matriz GUT evidencia que os problemas mais críticos a serem resolvidos para garantir a viabilidade e o valor inicial do produto estão centrados na experiência do aluno, especialmente na usabilidade e na percepção de progresso. Portanto, o MVP (Produto Mínimo Viável) e a primeira Sprint de experimentação se concentrarão em validar soluções para os problemas de maior pontuação (1.1, 1.2 e 4.1), que abordam diretamente a dificuldade de uso dos equipamentos, a clareza da ficha de treino e a portabilidade do histórico.

1.2 Solução

1.2.1 Objetivo SMART

Com base na análise de desafios e na priorização de problemas, e considerando o cronograma de desenvolvimento de três semanas, o objetivo central do projeto é desenvolver um protótipo de engenharia de dados ponta-a-ponta, desde a geração de dados até a sua visualização.

S (Específico):

Desenvolver um protótipo funcional completo que demonstre o ciclo de vida do dado, composto por cinco componentes principais:

Arquitetura de Dados: Implementar em PostgreSQL os schemas para um banco de dados transacional (OLTP) e um Data Warehouse (OLAP) com modelo em estrela.

Fonte de Dados Simulada: Criar um script Python (Data Seeder) para popular o banco OLTP com uma massa de dados de treino realista, simulando a utilização da plataforma por múltiplos usuários ao longo do tempo.

API de Dados (Backend): Desenvolver uma API em Python com FastAPI contendo endpoints para a inserção de dados (a serem usados pelo seeder) e para a consulta de dados analíticos (a serem consumidos pelo dashboard).

Pipeline de Dados (ETL): Construir um script em Python com Pandas que extrai os dados do OLTP, aplica transformações e os carrega no Data Warehouse.

Camada de Visualização (Frontend): Desenvolver um dashboard de página única em Vue.js que consome os dados da API e exibe os principais indicadores de performance e engajamento.

M (Mensurável):

O sucesso será medido pela entrega funcional de cada componente:

Dados: O script Data Seeder deverá gerar e inserir com sucesso pelo menos 500 registros de treino no banco OLTP.

Pipeline: O pipeline ETL deverá processar 100% dos dados gerados, populando corretamente o Data Warehouse.

API: Todos os endpoints de inserção e consulta deverão estar funcionais e testados via Swagger UI ou Postman.

Dashboard: A página em Vue.js deverá renderizar com sucesso pelo menos 2 visualizações de dados (gráficos ou tabelas) a partir dos dados consumidos da API.

A (Atingível):

O escopo foi estrategicamente limitado para ser alcançável no prazo estipulado. A criação de uma interface de usuário complexa foi substituída por um script de geração de dados e a camada de visualização foi focada em um dashboard simples, permitindo concentrar os esforços no fluxo de dados, que é o cerne do projeto. As tecnologias escolhidas (Python, FastAPI, Pandas, Vue.js, PostgreSQL) foram selecionadas por serem padrão de mercado e adequadas para a execução do escopo definido.

R (Relevante):

O projeto é altamente relevante, pois demonstra competência no ciclo de vida completo da engenharia de dados: da modelagem e geração de dados, passando pelo processamento via pipeline ETL, até a exposição através de uma API e a visualização em um frontend moderno. A abordagem valida a capacidade de transformar dados brutos em insights acionáveis.

T (Temporal):

O desenvolvimento completo do protótipo será realizado em um prazo total de 3 semanas, dividido em três Sprints de uma semana cada, com a apresentação para a banca ocorrendo ao final da Sprint 3.

1.2.2 Escopo do Projeto

Esta seção apresenta as condições, suposições e limitações que definem o escopo e norteiam o desenvolvimento eficiente do projeto, garantindo que os requisitos de custo, prazo e qualidade sejam cumpridos dentro do estabelecido.

Premissas do Projeto

As premissas são as condições que consideramos como verdadeiras para o sucesso do projeto, mesmo sem controle direto sobre elas.

Premissa 1: Viabilidade da Stack Tecnológica Open-Source.

Assume-se que o ecossistema de ferramentas de código aberto escolhido (PostgreSQL, Python/FastAPI, Pandas, Vue.js) será estável, performático e suficiente para atender a todos os requisitos do protótipo funcional.

Consequência se não for verdadeira: A descoberta de um bug crítico ou uma limitação de performance em alguma das ferramentas poderia exigir uma mudança de arquitetura, impactando severamente o cronograma de 3 semanas.

Premissa 2: Representatividade dos Dados Simulados.

Assume-se que a massa de dados gerada pelo script "Data Seeder" será uma representação fiel o suficiente dos dados de usuários reais, permitindo validar a lógica do pipeline ETL e a eficácia do modelo de dados do Data Warehouse.

Consequência se não for verdadeira: O pipeline pode funcionar para os dados simulados, mas falhar ao ser exposto a complexidades do mundo real. As análises e os insights gerados no dashboard podem não refletir a realidade.

Premissa 3: Disponibilidade do Desenvolvedor.

Assume-se que o autor do projeto terá disponibilidade de tempo e foco para se dedicar integralmente às atividades planejadas em cada uma das três Sprints de uma semana.

Consequência se não for verdadeira: Qualquer imprevisto pessoal ou profissional que reduza a dedicação ao projeto impactará diretamente a capacidade de cumprir o cronograma e entregar o escopo definido.

Restrições do Projeto

As restrições são as limitações conhecidas, internas ou externas, que foram impostas ao projeto.

Restrição 1: Tempo (Cronograma).

O projeto possui um prazo fixo e não negociável de 3 semanas, dividido em três Sprints de uma semana cada. Esta é a principal restrição e dita todo o escopo do que pode ser realizado.

Restrição 2: Recursos Humanos.

A equipe é composta por um único membro, que assume todos os papéis do projeto: Arquiteto de Dados, Engenheiro de Dados, Desenvolvedor Backend e Desenvolvedor Frontend.

Restrição 3: Custo.

O projeto possui um orçamento de R$ 0,00. Todas as ferramentas, softwares e plataformas utilizadas devem ser gratuitas ou de código aberto.

Restrição 4: Escopo do MVP.

O escopo está estritamente limitado às funcionalidades definidas no Objetivo SMART (seção 1.2.1). Funcionalidades para personas secundárias ou que não são essenciais para validar o fluxo de dados ponta-a-ponta estão explicitamente fora do escopo desta entrega.

Recursos e Conhecimentos Necessários

Recursos de Software:

Linguagem de Programação: Python

Framework de API: FastAPI

Biblioteca de Transformação de Dados: Pandas

Banco de Dados: PostgreSQL

Framework Frontend: Vue.js

Sistema de Controle de Versão: Git / GitHub

Ferramenta de Teste de API: Postman ou Swagger UI

Recursos de Hardware:

Computador pessoal com capacidade para executar o ambiente de desenvolvimento (servidor da API, banco de dados, ambiente de frontend).

Habilidades e Conhecimentos:

Programação em Python.

Desenvolvimento de APIs RESTful.

Linguagem SQL e modelagem de dados (Relacional e Dimensional/Estrela).

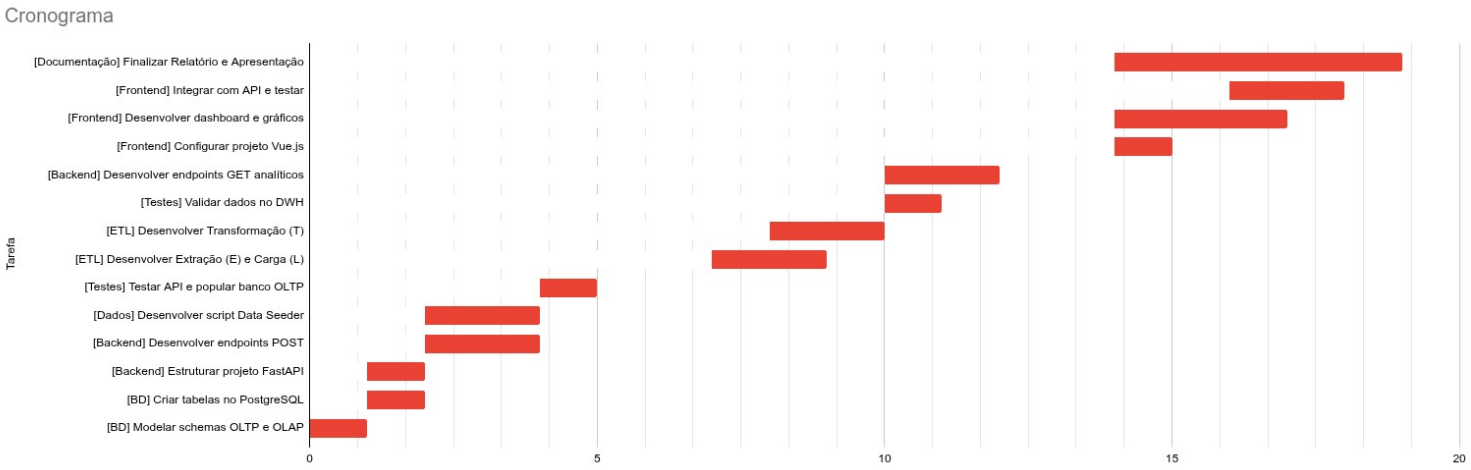
Princípios de ETL (Extração, Transformação e Carga).

Fundamentos de Vue.js para desenvolvimento de interfaces.

Uso de Git para controle de versão.

1.2.3 Cronograma de Ações Planejadas

O desenvolvimento do projeto será executado em um ciclo de 3 Sprints, cada uma com duração de uma semana. O objetivo é ter entregas de valor claras ao final de cada ciclo, culminando no protótipo funcional completo para a apresentação final. O detalhamento das ações para cada Sprint está descrito nas tabelas abaixo.



Sprint 1: Fundação e Geração de Dados (07/07/2025 a 11/07/2025)

Meta: Ter a arquitetura de banco de dados implementada e um método para popular o ambiente transacional com dados realistas.

Ação Principal	Responsável	Prazo
Modelagem dos schemas OLTP e OLAP	Autor do Projeto	Dia 1
Criação das tabelas no PostgreSQL	Autor do Projeto	Dia 2
Desenvolvimento dos endpoints de inserção na API	Autor do Projeto	Dias 3-4
Desenvolvimento do script "Data Seeder"	Autor do Projeto	Dias 3-4
Teste da API e povoamento do banco de dados OLTP	Autor do Projeto	Dia 5

Sprint 2: Pipeline ETL e API de Leitura (14/07/2025 a 18/07/2025)

Meta: Ter um pipeline de dados funcional que mova e transforme os dados do OLTP para o DWH, e uma API que exponha esses dados já processados.

Ação Principal	Responsável	Prazo
Desenvolvimento do pipeline ETL (Extração, Transformação, Carga)	Autor do Projeto	Dias 1-3
Validação da integridade dos dados no Data Warehouse	Autor do Projeto	Dia 4
Desenvolvimento dos endpoints de leitura analítica na API	Autor do Projeto	Dias 4-5
Teste completo do fluxo de dados ponta-a-ponta	Autor do Projeto	Dia 5

Sprint 3: Visualização e Entrega Final (21/07/2025 a 25/07/2025)

Meta: Ter uma camada de visualização funcional que demonstre o valor dos dados processados e preparar todos os materiais para a apresentação final.

Ação Principal	Responsável	Prazo
Desenvolvimento do dashboard em Vue.js (componentes e gráficos)	Autor do Projeto	Dias 1-3
Integração do dashboard com a API e testes	Autor do Projeto	Dia 4
Finalização do Relatório do Projeto Aplicado	Autor do Projeto	Dias 1-5
Criação da apresentação e vídeo para a banca	Autor do Projeto	Dias 4-5
Entrega e apresentação final do projeto	Autor do Projeto	Dia 5

2. Área de Experimentação

Após a definição do desafio e o planejamento da solução detalhados no Capítulo 1, esta seção documenta a fase de execução e validação prática do projeto. O objetivo é apresentar as evidências do trabalho realizado, demonstrando como os requisitos do backlog foram desenvolvidos e quais foram os resultados alcançados em cada ciclo.

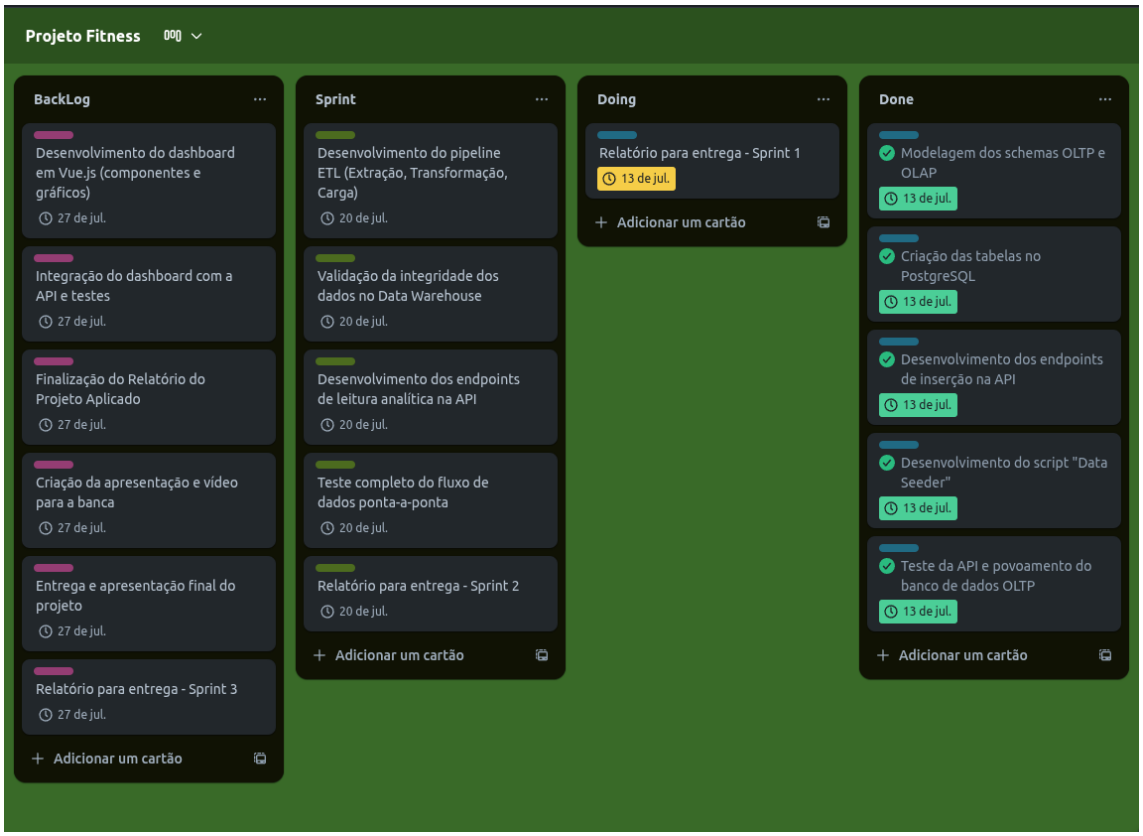
Seguindo o cronograma de três sprints de uma semana, cada subseção a seguir corresponde a um ciclo de desenvolvimento completo. Para cada sprint, serão apresentados os artefatos de planejamento, as evidências de execução dos requisitos — como diagramas de arquitetura, trechos de código e scripts — e as provas dos resultados obtidos através de testes automatizados e da operação da aplicação funcional.

Este capítulo, portanto, expõe de forma transparente a jornada de construção do protótipo, validando as hipóteses levantadas e registrando as lições aprendidas ao longo do processo, conforme a metodologia proposta.

2.1 Sprint 1

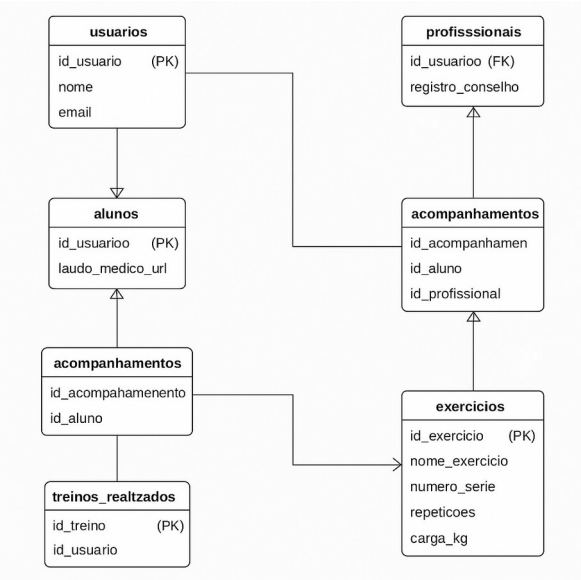
2.1.1 Solução

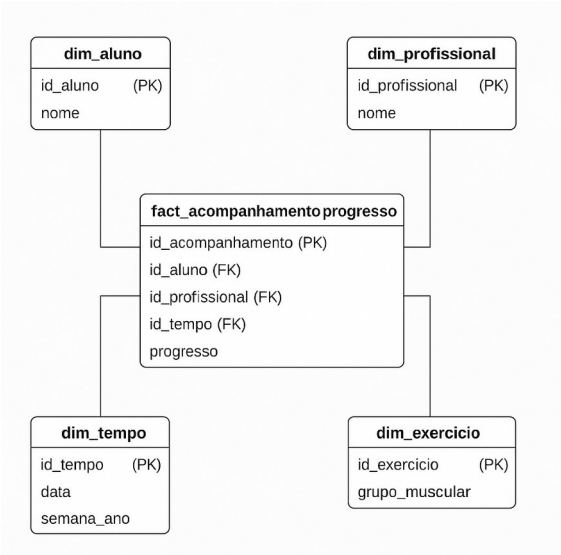
Evidência do planejamento:



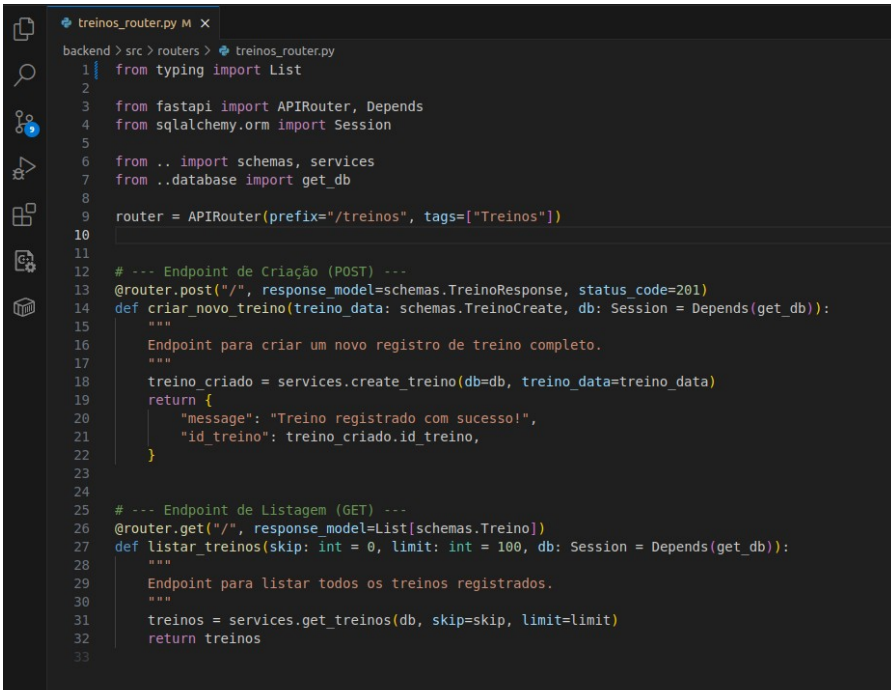
Evidência da execução de cada requisito:

[BD] Modelagem e Criação das Tabelas:

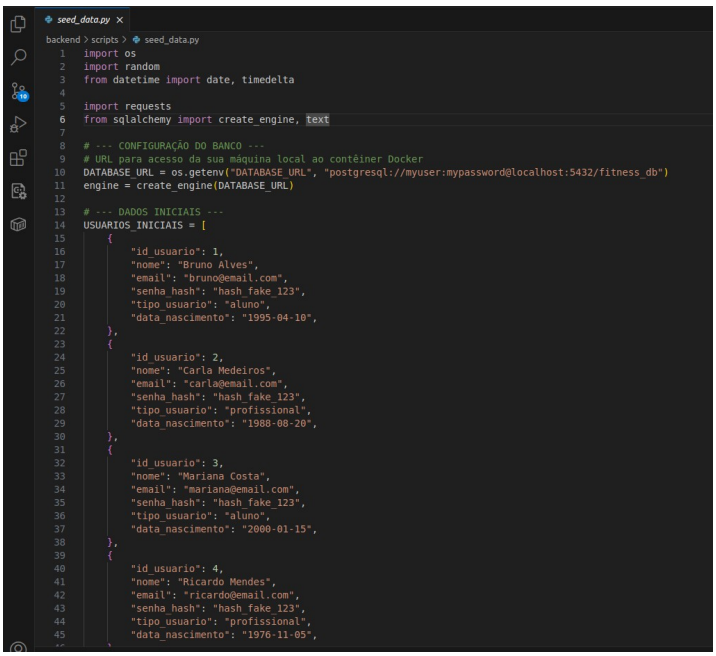


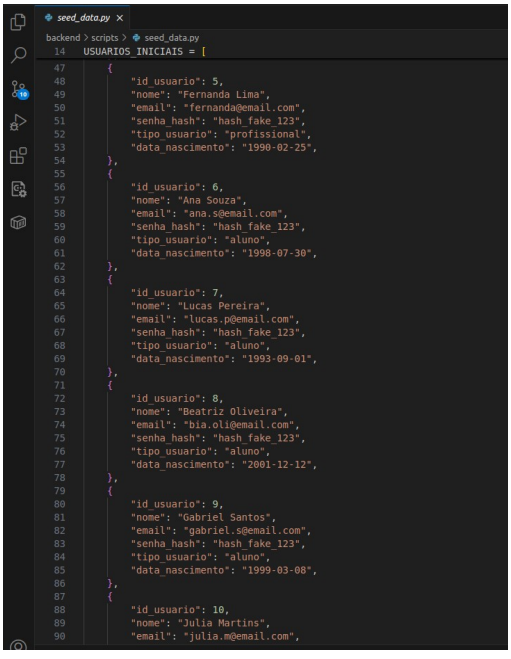


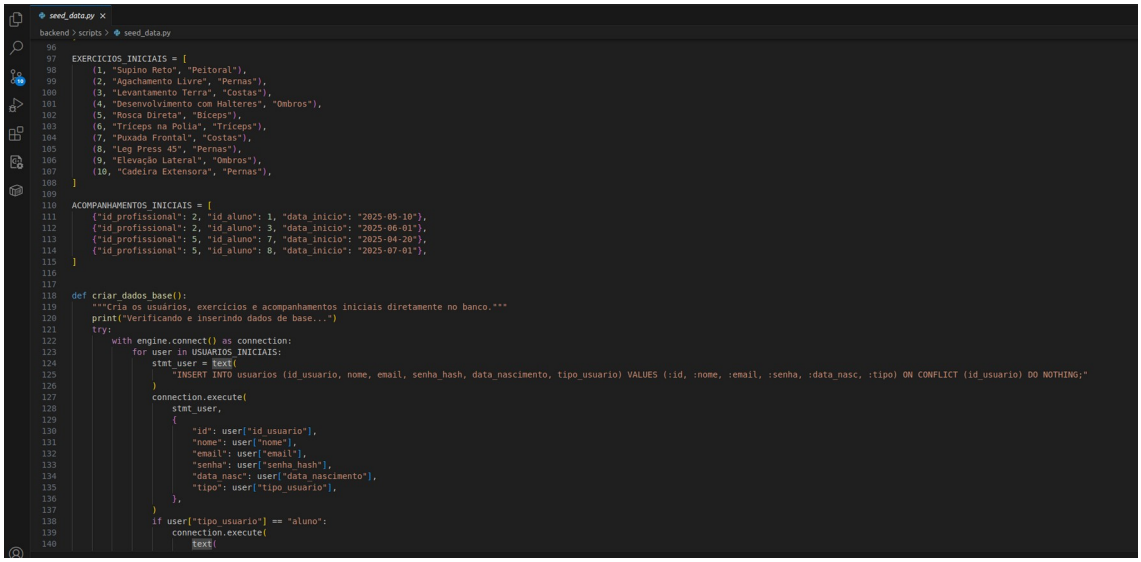
[Backend] Desenvolvimento da API:



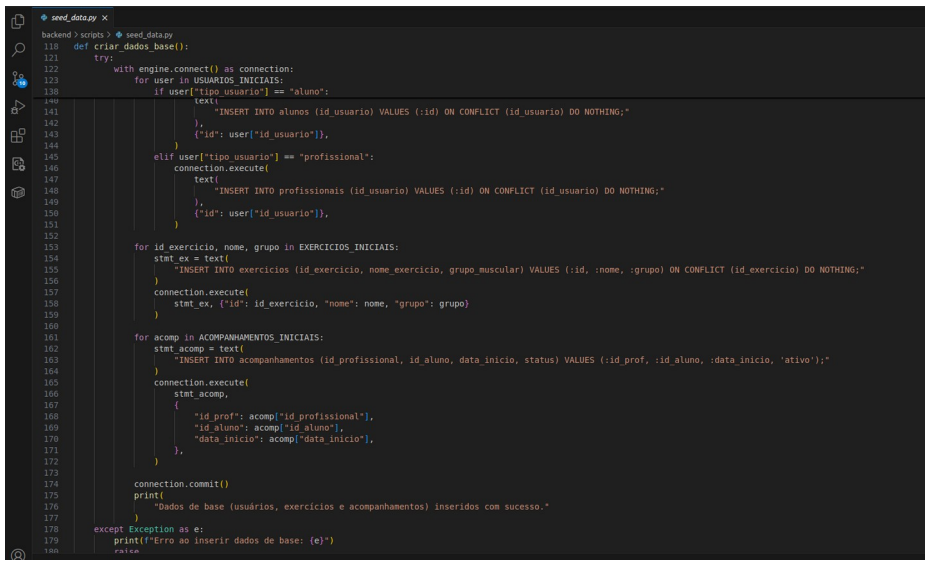
[Dados] Criação do Seeder:



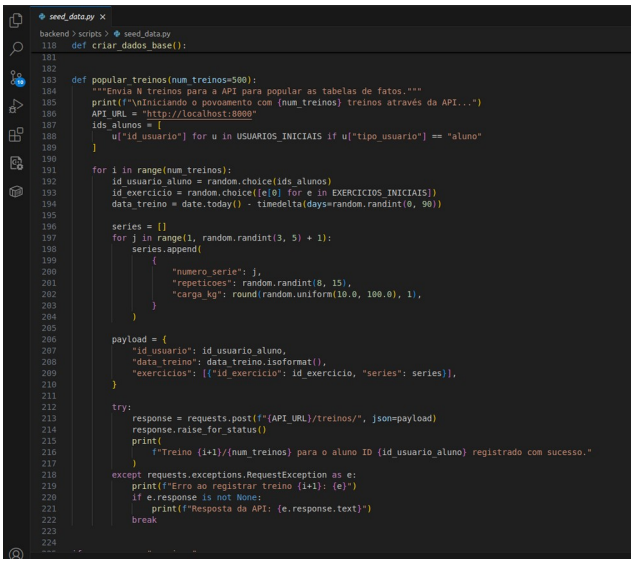




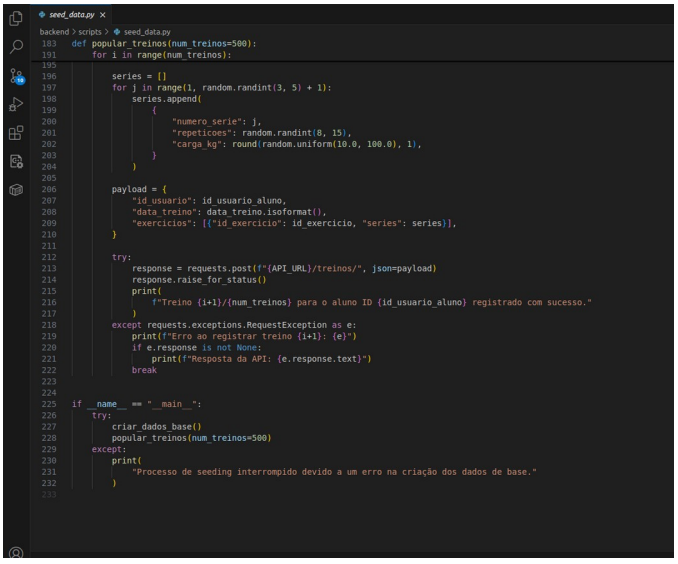
```python
EXERCICIOS_INICIAIS = [
    (1, "Supino Reto", "Peitoral"),
    (2, "Agachamento Livre", "Pernas"),
    (3, "Levantamento Terra", "Costas"),
    (4, "Desenvolvimento com Halteres", "Ombros"),
    (5, "Rosca Direta", "Biceps"),
    (6, "Triceps na Polia", "Triceps"),
    (7, "Puxada Frontal", "Costas"),
    (8, "Leg Press 45°", "Pernas"),
    (9, "Elevação Lateral", "Ombros"),
    (10, "Cadeira Extensora", "Pernas"),
]

ACOMPANHAMENTOS_INICIAIS = [
    {"id_profissional": 2, "id_aluno": 1, "data_inicio": "2025-05-10"},
    {"id_profissional": 2, "id_aluno": 3, "data_inicio": "2025-06-01"},
    {"id_profissional": 5, "id_aluno": 7, "data_inicio": "2025-04-20"},
    {"id_profissional": 5, "id_aluno": 8, "data_inicio": "2025-07-01"},
]


def criar_dados_base():
    """Cria os usuários, exercícios e acompanhamentos iniciais diretamente no banco."""
    print("Verificando e inserindo dados de base...")
    try:
        with engine.connect() as connection:
            for user in USUARIOS_INICIAIS:
                stmt_user = text(
                    "INSERT INTO usuarios (id_usuario, nome, email, senha_hash, data_nascimento, tipo_usuario) VALUES (:id, :nome, :email, :senha, :data_nasc, :tipo) ON CONFLICT (id_usuario) DO NOTHING;"
                )
                connection.execute(
                    stmt_user,
                    {
                        "id": user["id_usuario"],
                        "nome": user["nome"],
                        "email": user["email"],
                        "senha": user["senha_hash"],
                        "data_nasc": user["data_nascimento"],
                        "tipo": user["tipo_usuario"],
                    },
                )
                if user["tipo_usuario"] == "aluno":
                    connection.execute(
                        text(
```



```python
def criar_dados_base():
    try:
        with engine.connect() as connection:
            for user in USUARIOS_INICIAIS:
                if user["tipo_usuario"] == "aluno":
                    connection.execute(
                        text(
                            "INSERT INTO alunos (id_usuario) VALUES (:id) ON CONFLICT (id_usuario) DO NOTHING;"
                        ),
                        {"id": user["id_usuario"]},
                    )
                elif user["tipo_usuario"] == "profissional":
                    connection.execute(
                        text(
                            "INSERT INTO profissionais (id_usuario) VALUES (:id) ON CONFLICT (id_usuario) DO NOTHING;"
                        ),
                        {"id": user["id_usuario"]},
                    )

            for id_exercicio, nome, grupo in EXERCICIOS_INICIAIS:
                stmt_ex = text(
                    "INSERT INTO exercicios (id_exercicio, nome_exercicio, grupo_muscular) VALUES (:id, :nome, :grupo) ON CONFLICT (id_exercicio) DO NOTHING;"
                )
                connection.execute(
                    stmt_ex, {"id": id_exercicio, "nome": nome, "grupo": grupo}
                )

            for acomp in ACOMPANHAMENTOS_INICIAIS:
                stmt_acomp = text(
                    "INSERT INTO acompanhamentos (id_profissional, id_aluno, data_inicio, status) VALUES (:id_prof, :id_aluno, :data_inicio, 'ativo');"
                )
                connection.execute(
                    stmt_acomp,
                    {
                        "id_prof": acomp["id_profissional"],
                        "id_aluno": acomp["id_aluno"],
                        "data_inicio": acomp["data_inicio"],
                    },
                )

            connection.commit()
            print(
                "Dados de base (usuários, exercícios e acompanhamentos) inseridos com sucesso."
            )
    except Exception as e:
        print(f"Erro ao inserir dados de base: {e}")
        raise
```

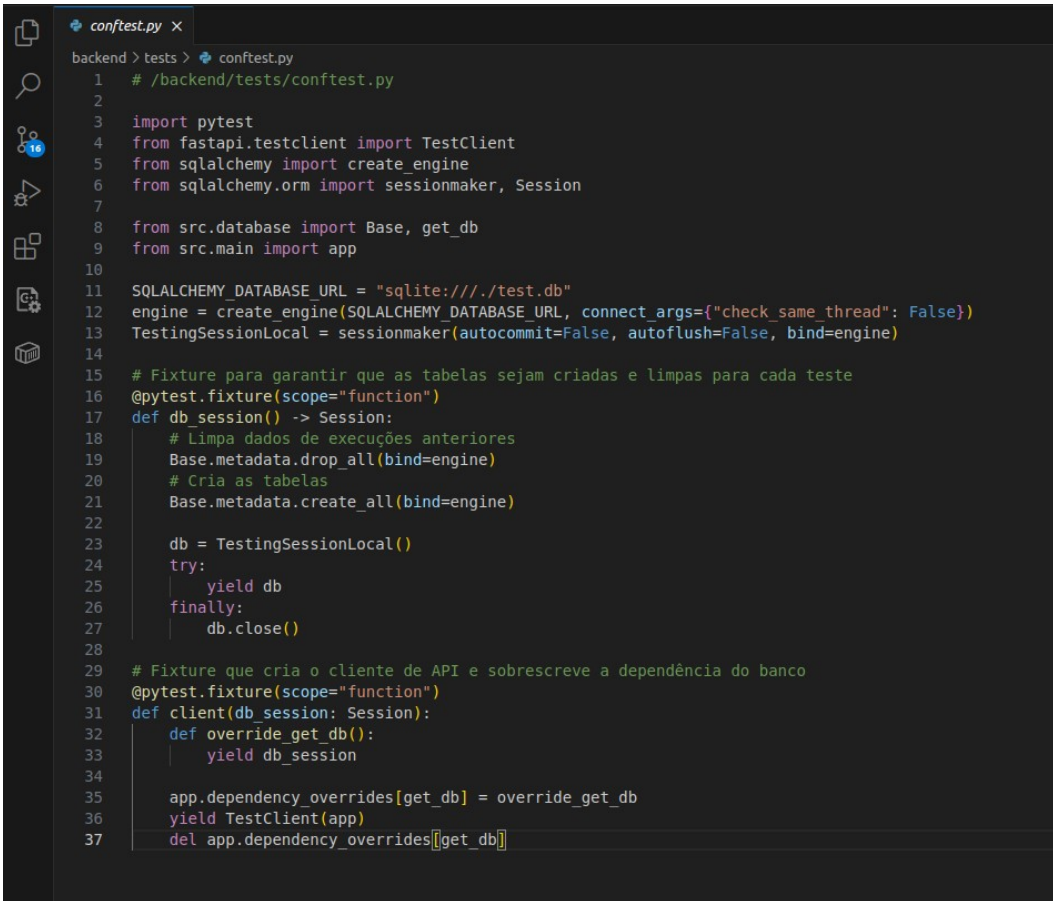


```python
def criar_dados_base():


def popular_treinos(num_treinos=500):
    """Envia N treinos para a API para popular as tabelas de fatos."""
    print(f"\nIniciando o povoamento com {num_treinos} treinos através da API...")
    API_URL = "http://localhost:8000"
    ids_alunos = [
        u["id_usuario"] for u in USUARIOS_INICIAIS if u["tipo_usuario"] == "aluno"
    ]

    for i in range(num_treinos):
        id_usuario_aluno = random.choice(ids_alunos)
        id_exercicio = random.choice([e[0] for e in EXERCICIOS_INICIAIS])
        data_treino = date.today() - timedelta(days=random.randint(0, 90))

        series = []
        for j in range(1, random.randint(3, 5) + 1):
            series.append(
                {
                    "numero_serie": j,
                    "repeticoes": random.randint(8, 15),
                    "carga_kg": round(random.uniform(10.0, 100.0), 1),
                }
            )

        payload = {
            "id_usuario": id_usuario_aluno,
            "data_treino": data_treino.isoformat(),
            "exercicios": [{"id_exercicio": id_exercicio, "series": series}],
        }

        try:
            response = requests.post(f"{API_URL}/treinos/", json=payload)
            response.raise_for_status()
            print(
                f"Treino {i+1}/{num_treinos} para o aluno ID {id_usuario_aluno} registrado com sucesso."
            )
        except requests.exceptions.RequestException as e:
            print(f"Erro ao registrar treino {i+1}: {e}")
            if e.response is not None:
                print(f"Resposta da API: {e.response.text}")
            break


```



```python
def popular_treinos(num_treinos=500):
    for i in range(num_treinos):

        series = []
        for j in range(1, random.randint(3, 5) + 1):
            series.append(
                {
                    "numero_serie": j,
                    "repeticoes": random.randint(8, 15),
                    "carga_kg": round(random.uniform(10.0, 100.0), 1),
                }
            )

        payload = {
            "id_usuario": id_usuario_aluno,
            "data_treino": data_treino.isoformat(),
            "exercicios": [{"id_exercicio": id_exercicio, "series": series}],
        }

        try:
            response = requests.post(f"{API_URL}/treinos/", json=payload)
            response.raise_for_status()
            print(
                f"Treino {i+1}/{num_treinos} para o aluno ID {id_usuario_aluno} registrado com sucesso."
            )
        except requests.exceptions.RequestException as e:
            print(f"Erro ao registrar treino {i+1}: {e}")
            if e.response is not None:
                print(f"Resposta da API: {e.response.text}")
            break


if __name__ == "__main__":
    try:
        criar_dados_base()
        popular_treinos(num_treinos=500)
    except:
        print(
            "Processo de seeding interrompido devido a um erro na criação dos dados de base."
        )
```

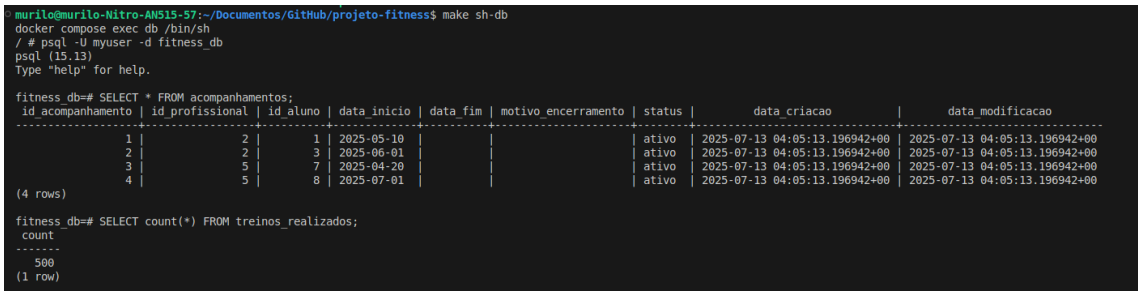
[Testes] Configuração dos Testes:



```python
# /backend/tests/conftest.py

import pytest
from fastapi.testclient import TestClient
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker, Session

from src.database import Base, get_db
from src.main import app

SQLALCHEMY_DATABASE_URL = "sqlite:///./test.db"
engine = create_engine(SQLALCHEMY_DATABASE_URL, connect_args={"check_same_thread": False})
TestingSessionLocal = sessionmaker(autocommit=False, autoflush=False, bind=engine)

# Fixture para garantir que as tabelas sejam criadas e limpas para cada teste
@pytest.fixture(scope="function")
def db_session() -> Session:
    # Limpa dados de execuções anteriores
    Base.metadata.drop_all(bind=engine)
    # Cria as tabelas
    Base.metadata.create_all(bind=engine)

    db = TestingSessionLocal()
    try:
        yield db
    finally:
        db.close()

# Fixture que cria o cliente de API e sobrescreve a dependência do banco
@pytest.fixture(scope="function")
def client(db_session: Session):
    def override_get_db():
        yield db_session

    app.dependency_overrides[get_db] = override_get_db
    yield TestClient(app)
    del app.dependency_overrides[get_db]
```

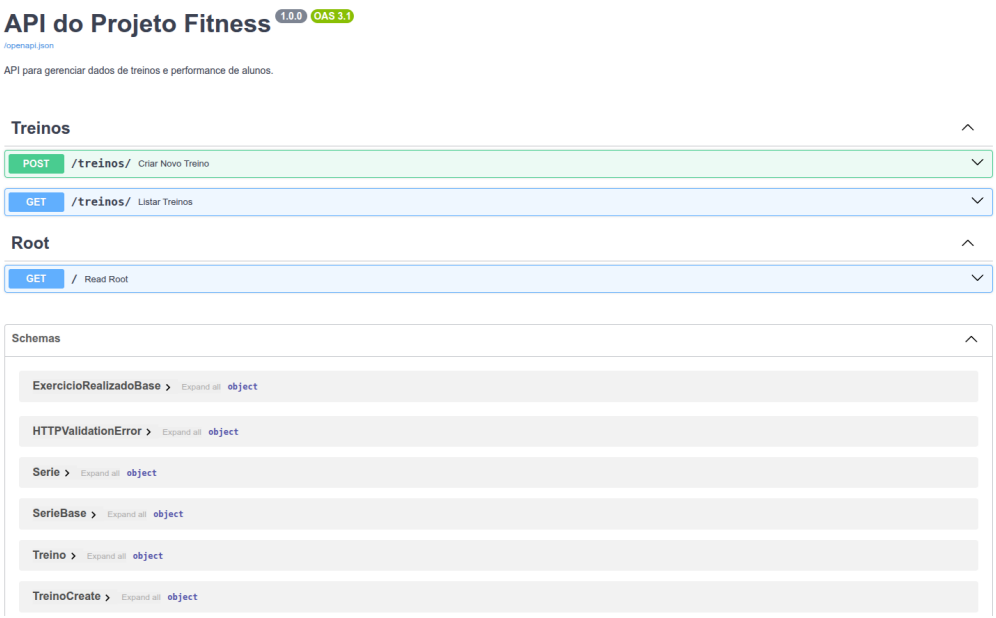
Evidência dos resultados:

Prova 1 (Banco Populado):

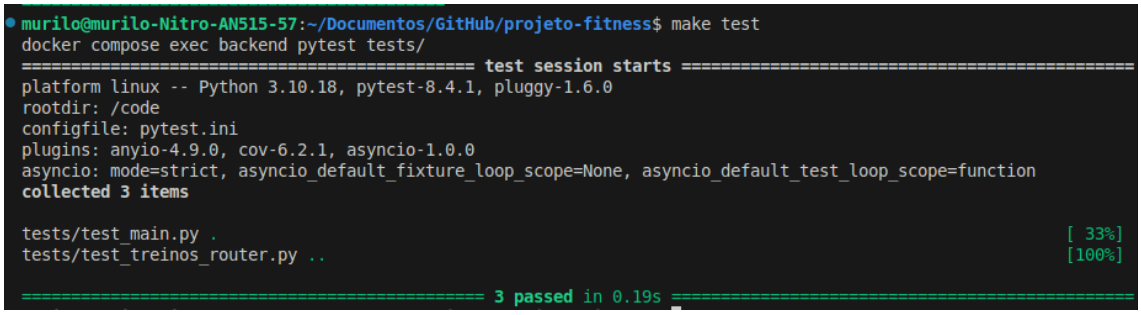


```
murilo@murilo-Nitro-AN515-57:~/Documentos/GitHub/projeto-fitness$ make sh-db
docker compose exec db /bin/sh
/ # psql -U myuser -d fitness_db
psql (15.13)
Type "help" for help.

fitness_db=# SELECT * FROM acompanhamentos;
 id_acompanhamento | id_profissional | id_aluno | data_inicio | data_fim | motivo_encerramento | status |         data_criacao          |       data_modificacao
-------------------+-----------------+----------+-------------+----------+---------------------+--------+-------------------------------+-------------------------------
                 1 |               2 |        1 | 2025-05-10  |          |                     | ativo  | 2025-07-13 04:05:13.196942+00 | 2025-07-13 04:05:13.196942+00
                 2 |               2 |        3 | 2025-06-01  |          |                     | ativo  | 2025-07-13 04:05:13.196942+00 | 2025-07-13 04:05:13.196942+00
                 3 |               5 |        7 | 2025-04-20  |          |                     | ativo  | 2025-07-13 04:05:13.196942+00 | 2025-07-13 04:05:13.196942+00
                 4 |               5 |        8 | 2025-07-01  |          |                     | ativo  | 2025-07-13 04:05:13.196942+00 | 2025-07-13 04:05:13.196942+00
(4 rows)

fitness_db=# SELECT count(*) FROM treinos_realizados;
 count
-------
   500
(1 row)
```

Prova 2 (API Funcionando):



Prova 3 (Testes Passando):



2.1.2 Retrospectiva da Sprint

Pontos Positivos e Ganhos de Produtividade

A decisão de investir tempo na estruturação inicial do projeto provou ser um grande acelerador. A adoção de uma arquitetura limpa, inspirada nos princípios do MVC e separada em pacotes (`src`, `scripts`, `tests`), tornou o código mais legível e fácil de manter.

A utilização do **Docker e Docker Compose** desde o início eliminou a clássica síndrome do "funciona na minha máquina", garantindo um ambiente de desenvolvimento consistente e replicável. A criação de um **`Makefile`** como um "painel de controle" centralizou os comandos complexos em atalhos simples (como `make up`, `make db-init`, `make test`), o que aumentou significativamente a produtividade e reduziu a chance de erros manuais.

Principais Desafios e Soluções Aplicadas

A Sprint 1 apresentou desafios técnicos consideráveis, que foram superados através de um processo iterativo de depuração e refatoração:

- **Configuração do Ambiente Docker:** O maior desafio foi garantir a sincronia perfeita entre o ambiente local (VS Code) e o contêiner em execução. Enfrentamos e resolvemos problemas de permissão de usuário não-root, conflitos entre versões do `docker-compose` e falhas de "hot-reload", culminando em uma configuração robusta que usa um usuário não-root com permissões corretas e um `Dockerfile` otimizado.
- **Gerenciamento de Caminhos (Python Path):** A refatoração para uma estrutura de pacotes profissional (`src`) expôs erros de `ModuleNotFoundError`. A solução inicial de `sys.path.append` se mostrou frágil ao entrar em conflito com as ferramentas de formatação. O problema foi resolvido de forma definitiva ao transformar o projeto em um **pacote Python instalável** com um arquivo `pyproject.toml` e executando os scripts como módulos (`python -m`), uma prática padrão da indústria.
- **Setup da Suíte de Testes:** A implementação dos testes com `pytest` foi um micro-projeto em si. Depuramos uma série de erros, desde `NameError` por falta de imports até um persistente erro de `readonly database` com o SQLite. A solução foi refatorar a configuração para usar **`pytest fixtures`** de forma isolada, com um ciclo de vida de setup e teardown bem definido no arquivo `conftest.py`, garantindo que cada teste rode em um ambiente 100% limpo.

Principais Aprendizados

Além da construção da API funcional, a Sprint 1 foi uma imersão em boas práticas de engenharia de software:

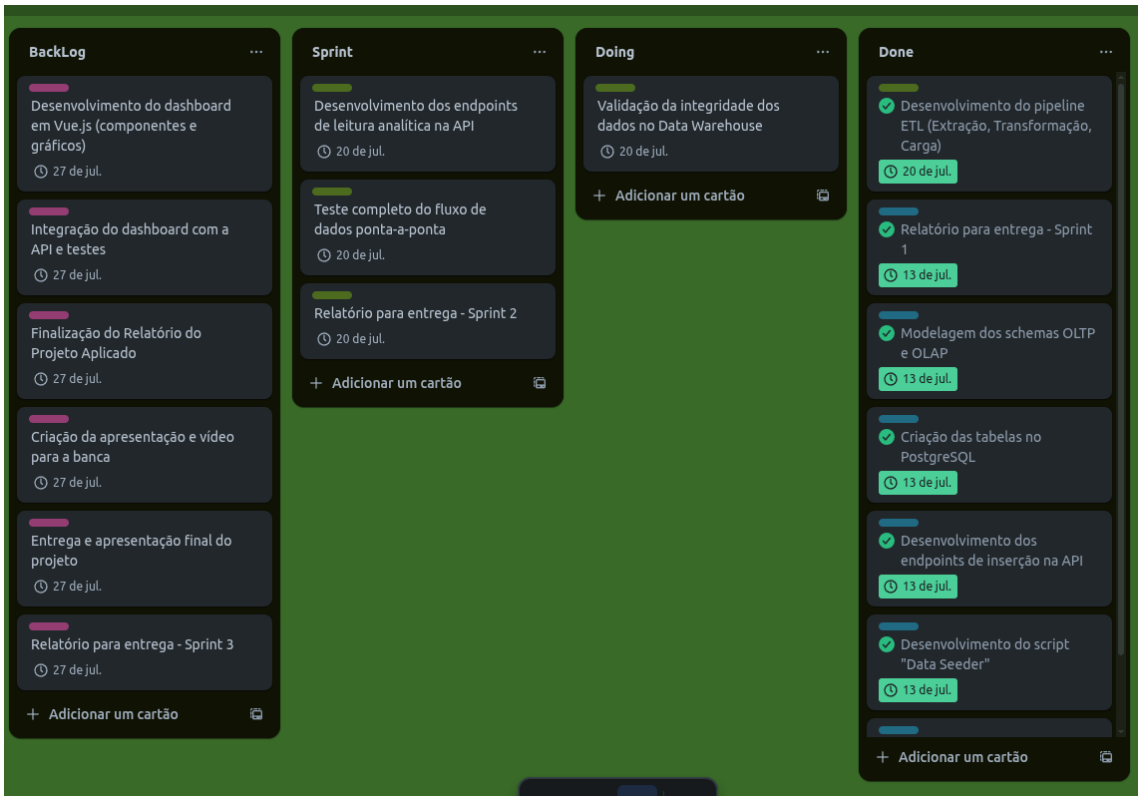
- **O Valor da Análise Estática:** A introdução do `flake8` (linter) e `mypy` (type checker) simula uma "etapa de build" para o Python. Aprender a configurar e satisfazer essas ferramentas previne uma classe inteira de erros estruturais antes mesmo de o código ser executado.
- **A Importância do Isolamento nos Testes:** O principal aprendizado técnico foi a necessidade absoluta de isolar o estado de cada teste. A experiência demonstrou na prática por que a gestão de um banco de dados de teste limpo para cada execução é um pilar fundamental para a confiabilidade de uma suíte de testes automatizados.
- **Modelagem de Dados Evolutiva:** O modelo de dados evoluiu de uma simples tabela de usuários para um schema complexo com herança de tabelas e colunas de auditoria. Isso demonstrou como a compreensão das regras de negócio (diferentes personas, histórico de relacionamentos) impacta diretamente o design da arquitetura de dados.

2.2 Sprint 2

2.2.1 Solução

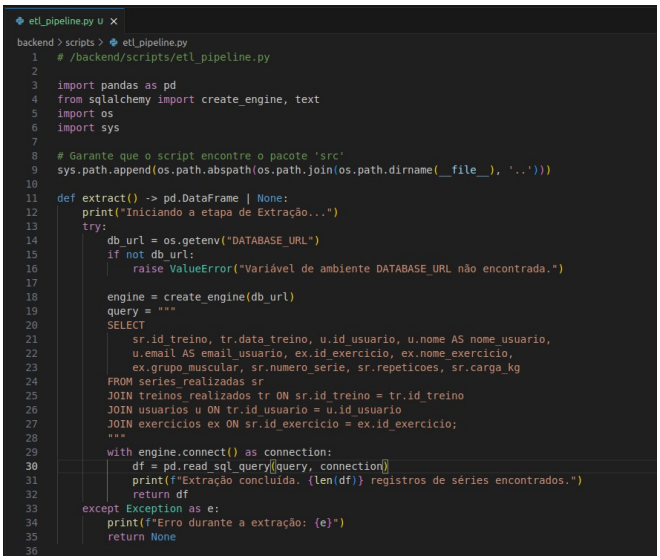
Aqui você apresenta as provas concretas do que foi desenvolvido nesta sprint.

Evidência do planejamento:



Evidência da execução de cada requisito:

Para "Desenvolvimento do pipeline ETL":



```python
# /backend/scripts/etl_pipeline.py

import pandas as pd
from sqlalchemy import create_engine, text
import os
import sys

# Garante que o script encontre o pacote 'src'
sys.path.append(os.path.abspath(os.path.join(os.path.dirname(__file__), '..')))

def extract() -> pd.DataFrame | None:
    print("Iniciando a etapa de Extração...")
    try:
        db_url = os.getenv("DATABASE_URL")
        if not db_url:
            raise ValueError("Variável de ambiente DATABASE_URL não encontrada.")

        engine = create_engine(db_url)
        query = """
        SELECT
            sr.id_treino, tr.data_treino, u.id_usuario, u.nome AS nome_usuario,
            u.email AS email_usuario, ex.id_exercicio, ex.nome_exercicio,
            ex.grupo_muscular, sr.numero_serie, sr.repeticoes, sr.carga_kg
        FROM series_realizadas sr
        JOIN treinos_realizados tr ON sr.id_treino = tr.id_treino
        JOIN usuarios u ON tr.id_usuario = u.id_usuario
        JOIN exercicios ex ON sr.id_exercicio = ex.id_exercicio;
        """
        with engine.connect() as connection:
            df = pd.read_sql_query(query, connection)
            print(f"Extração concluída. {len(df)} registros de séries encontrados.")
            return df
    except Exception as e:
        print(f"Erro durante a extração: {e}")
        return None
```

```python
    return None

def transform(df: pd.DataFrame):
    if df is None or df.empty:
        print("DataFrame de entrada vazio. A transformação não pode continuar.")
        return None, None, None, None
    print("Iniciando a etapa de Transformação...")
    df['data_treino'] = pd.to_datetime(df['data_treino'])
    dim_tempo = df[['data_treino']].drop_duplicates()
    dim_tempo.rename(columns={'data_treino': 'id_data'}, inplace=True)
    dim_tempo['ano'] = dim_tempo['id_data'].dt.year
    dim_tempo['mes'] = dim_tempo['id_data'].dt.month
    dim_tempo['dia'] = dim_tempo['id_data'].dt.day
    dim_tempo['dia_da_semana'] = dim_tempo['id_data'].dt.day_name()
    dim_aluno = df[['id_usuario', 'nome_usuario', 'email_usuario']].drop_duplicates()
    dim_aluno.rename(columns={'id_usuario': 'id_aluno', 'nome_usuario': 'nome_aluno', 'email_usuario': 'email'}, inplace=True)
    dim_exercicio = df[['id_exercicio', 'nome_exercicio', 'grupo_muscular']].drop_duplicates()
    df['volume_total_carga'] = df['repeticoes'] * df['carga_kg']
    fct_treinos = df.groupby(['id_treino', 'data_treino', 'id_usuario', 'id_exercicio']).agg(
        total_series=('numero_serie', 'count'),
        total_repeticoes=('repeticoes', 'sum'),
        maior_carga_kg=('carga_kg', 'max'),
        volume_total_carga=('volume_total_carga', 'sum')
    ).reset_index()
    fct_treinos.rename(columns={'data_treino': 'id_data', 'id_usuario': 'id_aluno'}, inplace=True)
    print("Transformação concluída.")
    return dim_tempo, dim_aluno, dim_exercicio, fct_treinos

def load(df: pd.DataFrame, table_name: str, engine):
    """Carrega um DataFrame em uma tabela do DWH. Usa 'append' para adicionar os dados."""
    if df is None: return
    print(f"Carregando dados na tabela DWH: '{table_name}'...")
    try:
        df.to_sql(table_name, engine, if_exists='append', index=False)
        print(f"Carga de dados na tabela '{table_name}' concluída com sucesso.")
    except Exception as e:
        print(f"Erro ao carregar dados na tabela '{table_name}': {e}")
        raise

# --- FUNÇÃO PARA ORQUESTRAR A LIMPEZA ---
def truncate_dwh_tables(engine):
    """Limpa as tabelas do DWH na ordem correta para evitar erros de FK."""
    print("Iniciando limpeza das tabelas do Data Warehouse...")
    try:
        with engine.connect() as connection:
            # Começamos a transação
            transaction = connection.begin()
            # Limpamos a tabela FATO primeiro, pois ela depende das outras
            connection.execute(text("TRUNCATE TABLE fct_treinos RESTART IDENTITY CASCADE;"))
            # Depois, limpamos as dimensões
            connection.execute(text("TRUNCATE TABLE dim_tempo RESTART IDENTITY CASCADE;"))
            connection.execute(text("TRUNCATE TABLE dim_aluno RESTART IDENTITY CASCADE;"))
            connection.execute(text("TRUNCATE TABLE dim_exercicio RESTART IDENTITY CASCADE;"))
            # Finalizamos a transação
            transaction.commit()
            print("Tabelas do DWH limpas com sucesso.")
    except Exception as e:
        print(f"Erro ao limpar as tabelas do DWH: {e}")
        raise

if __name__ == "__main__":
    db_url = os.getenv("DATABASE_URL")
    if not db_url:
        print("Erro: A variável de ambiente DATABASE_URL não está configurada.")
    else:
        engine = create_engine(db_url)

        # 1. Extrai os dados
        df_extraido = extract()

        if df_extraido is not None and not df_extraido.empty:
            # 2. Transforma os dados
            dim_tempo, dim_aluno, dim_exercicio, fct_treinos = transform(df_extraido)

            # 3. Limpa o DWH antes de carregar
            truncate_dwh_tables(engine)

            # 4. Carrega as tabelas na ordem correta (dimensões primeiro)
            load(dim_tempo, "dim_tempo", engine)
            load(dim_aluno, "dim_aluno", engine)
            load(dim_exercicio, "dim_exercicio", engine)
            load(fct_treinos, "fct_treinos", engine)

            print("\nProcesso de ETL concluído com sucesso!")
```

Para "Desenvolvimento dos endpoints de leitura analítica na API":

backend > src > routers > analytics_router.py

```python
# /backend/src/routers/analytics_router.py

from fastapi import APIRouter, Depends, HTTPException
from sqlalchemy.orm import Session

from .. import schemas, services
from ..database import get_db

router = APIRouter(prefix="/analytics", tags=["Analytics"])


@router.get("/performance/{aluno_id}", response_model=schemas.AnalyticsResponse)
def get_performance_data(aluno_id: int, db: Session = Depends(get_db)):
    """
    Endpoint para buscar os dados de performance de um aluno específico
    a partir do Data Warehouse.
    """
    dados_do_banco = services.get_aluno_performance(db=db, aluno_id=aluno_id)

    if not dados_do_banco:
        raise HTTPException(
            status_code=404,
            detail=f"Nenhum dado de performance encontrado para o aluno com ID {aluno_id}.",   # noqa: E501
        )

    # Agora, cada 'linha' no resultado tem os nomes das colunas que definimos na query
    nome_aluno_encontrado = dados_do_banco[0].nome_aluno

    # Usamos uma list comprehension para montar a lista de forma mais eficiente
    performance_list = [
        schemas.PerformanceData.from_orm(linha) for linha in dados_do_banco
    ]

    return schemas.AnalyticsResponse(
        id_aluno=aluno_id,
        nome_aluno=nome_aluno_encontrado,
        performance=performance_list,
    )

```

Evidência dos resultados:

Prova 1 (Log de Execução do ETL):

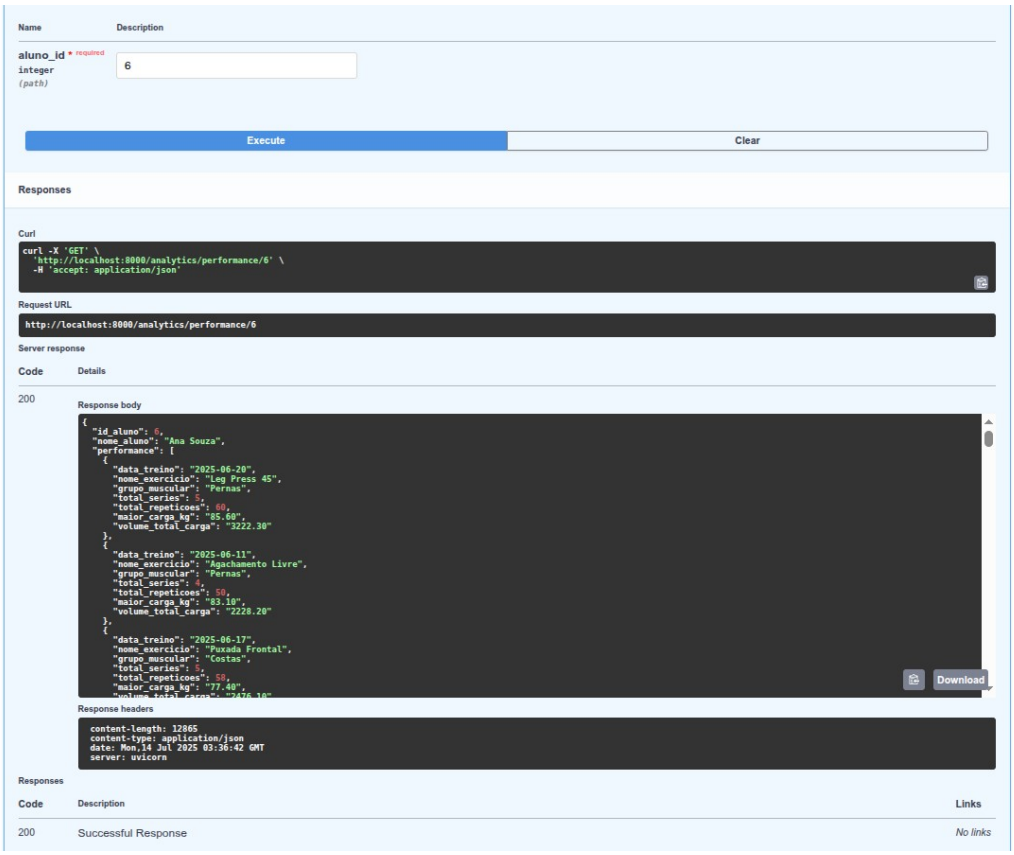
```
murilo@murilo-Nitro-AN515-57:~/Documentos/GitHub/projeto-fitness$ make etl-run
docker compose exec backend python -m scripts.etl_pipeline
Iniciando a etapa de Extração...
Extração concluída. 2009 registros de séries encontrados.
Iniciando a etapa de Transformação...
Transformação concluída.
Iniciando limpeza das tabelas do Data Warehouse...
Tabelas do DWH limpas com sucesso.
Carregando dados na tabela DWH: 'dim_tempo'...
Carga de dados na tabela 'dim_tempo' concluída com sucesso.
Carregando dados na tabela DWH: 'dim_aluno'...
Carga de dados na tabela 'dim_aluno' concluída com sucesso.
Carregando dados na tabela DWH: 'dim_exercicio'...
Carga de dados na tabela 'dim_exercicio' concluída com sucesso.
Carregando dados na tabela DWH: 'fct_treinos'...
Carga de dados na tabela 'fct_treinos' concluída com sucesso.

Processo de ETL concluído com sucesso!
```

Prova 2 (Data Warehouse Populado):

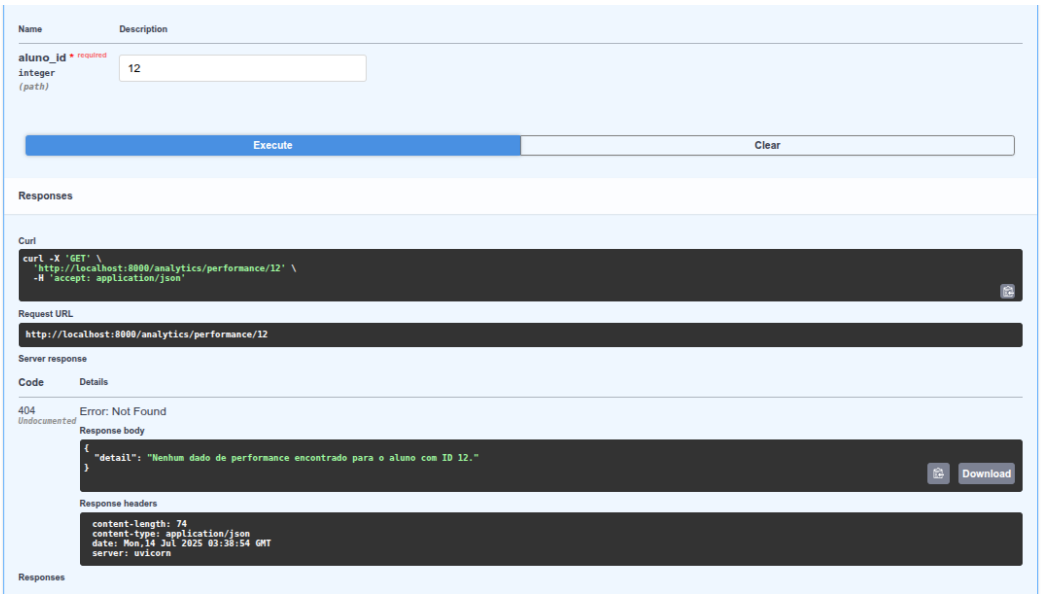
```
murilo@murilo-Nitro-AN515-57:~/Documentos/GitHub/projeto-fitness$ make sh-db
docker compose exec db /bin/sh
/ # psql -U myuser -d fitness_db
psql (15.13)
Type "help" for help.

fitness_db=# SELECT count(*) FROM fct_treinos;
 count
-------
   500
(1 row)

fitness_db=# SELECT * FROM dim_aluno LIMIT 5;
 id_aluno |   nome_aluno    |        email        |         data_criacao          |       data_modificacao
----------+-----------------+---------------------+-------------------------------+-------------------------------
        9 | Gabriel Santos  | gabriel.s@email.com | 2025-07-14 01:17:10.015857+00 | 2025-07-14 01:17:10.015857+00
        1 | Bruno Alves     | bruno@email.com     | 2025-07-14 01:17:10.015857+00 | 2025-07-14 01:17:10.015857+00
       10 | Julia Martins   | julia.m@email.com   | 2025-07-14 01:17:10.015857+00 | 2025-07-14 01:17:10.015857+00
        3 | Mariana Costa   | mariana@email.com   | 2025-07-14 01:17:10.015857+00 | 2025-07-14 01:17:10.015857+00
        6 | Ana Souza       | ana.s@email.com     | 2025-07-14 01:17:10.015857+00 | 2025-07-14 01:17:10.015857+00
(5 rows)

fitness_db=#
```

Prova 3 (Endpoint Analítico Robusto)

Artefato 1 (Caminho de sucesso – return 200):



Artefato 2 (Tratamento de Erro – return 404):



2.2.2 Retrospectiva da Sprint

A Sprint 2 foi focada no coração do projeto de Engenharia de Dados: a construção do pipeline ETL e a criação de uma API para expor os dados já processados do Data Warehouse (DWH).

- **Pontos Positivos:** A estrutura de dados bem definida na Sprint 1 provou seu valor, facilitando a extração e a transformação dos dados com a biblioteca Pandas. A lógica de agregação para criar a tabela de fatos (`fct_treinos`) e as dimensões foi implementada de forma eficiente. A criação de um endpoint de análise (`/analytics`) para consumir dados do DWH foi concluída com sucesso.
- **Principais Desafios e Soluções Aplicadas:** O principal desafio técnico surgiu durante a etapa de carga (Load) do ETL. O erro `DependentObjectsStillExist` ocorreu porque a estratégia inicial de `if_exists='replace'` tentava apagar as tabelas de dimensão antes da tabela de fatos, violando as restrições de chave estrangeira. A solução foi refatorar o pipeline para uma abordagem mais robusta: criar uma função `truncate_dwh_tables` que limpa os dados na ordem correta (fatos primeiro, depois dimensões) e alterar a estratégia de carga para `if_exists='append'`. Isso tornou o pipeline idempotente e seguro para re-execuções.
- **Principal Aprendizado:** O aprendizado chave desta sprint foi a importância da **orquestração e da ordem das operações em um pipeline de ETL**. Ficou claro na prática que a integridade referencial de um Data Warehouse em Estrela precisa ser respeitada tanto na carga quanto na limpeza dos dados. Além disso, a implementação do tratamento de erro 404 para o endpoint analítico reforçou o conceito de construir APIs robustas que lidam com casos de borda (como a não existência de dados para um determinado ID) de forma explícita e informativa.

2.3 Sprint 3

2.3.1 Solução

- Evidência da execução de cada requisito:
- Evidência dos resultados:

2.3.2 Retrospectiva da Sprint

3. Considerações Finais

3.1 Resultados

Por meio de um texto detalhado, apresente os principais resultados alcançados pelo seu Projeto Aplicado.

Cite os pontos positivos e negativos, as dificuldades enfrentadas e as experiências vivenciadas durante todo o processo.

3.2 Próximos passos

Descreva quais são os próximos passos que poderão contribuir com o aprimoramento da solução apresentada pelo seu Projeto Aplicado.