Notas de Design e Desafios — Meli Challenge

Escolhas de Design

A escolha de tecnologias e arquitetura foi feita para refletir uma aplicação real, com foco em modularidade, reutilização de componentes e uma experiência de usuário fluida.

Arquitetura geral

- Optei por uma arquitetura clean e separada entre frontend e backend, para refletir uma aplicação real.
- Em um repositorio no modelo monorepo tenho as pastas `backend/` e `frontend/`, cada uma com sua estrutura organizada.
- O backend foi desenvolvido em **FastAPI**, que é uma framework leve e muito rápido para criação de APIs RESTful, com documentação automática (Swagger/OpenAPI).
- O frontend foi feito com **React + Next.js**, seguindo o pedido de "página de detalhes prototipada" e explorando as capacidades modernas de UX.

Backend

- Organização em:
  - `app/routes/` — rotas separadas
  - `app/services/` — lógica de serviço (ex: carregamento de produtos)
  - `app/schemas/` — definição dos modelos de resposta com **pydantic**
- Persistência: conforme o enunciado, usei **arquivo JSON local** para simular banco de dados.

Frontend

- Componentes 100% reutilizáveis e componentizados:
  - `ProductSummary`
  - `PaymentMethods`
  - `BuyBox`
  - `ProductGallery`
  - `ProductDescription`
  - `Breadcrumb`
  - `ProductNotFound`
- Estilo inspirado visualmente no **Mercado Livre** (cores, espaçamento, layout em colunas).
- Usei **TailwindCSS** para agilidade no protótipo.
- O protótipo está **responsivo** (mobile-friendly).

Makefile

- Criei um **Makefile** de raiz para que qualquer usuário consiga subir a aplicação com:
  - `make dev` → sobe backend + frontend
  - `make test-backend` → executa testes com coverage
  - `make lint-backend` e `make lint-frontend`

Desafios enfrentados e como resolvi

1) Imagens dos produtos

- O maior desafio no frontend foi lidar com as imagens do JSON:
  - Como o enunciado pede persistência local e não uso de banco/CDN, as imagens eram relativas (ex: `/images/products/...`).
  - No **Next.js** isso gera questões de path e otimização.
- Resolvi ajustando a configuração para servir **imagens estáticas** da pasta `public/images/products`.

---

2) UX da galeria de imagens

- Implementei um `ProductGallery` com miniaturas similares ao Mercado Livre.

---

3) Design do BuyBox

- Implementação do **modal de loja**, com quantidade de vendas.
- Challenge: como expressar o "indicador de qualidade" da loja.
- Resolvi com um **modal estilizado** e barras de progresso "mockadas".

---

4) Testes e cobertura

- Para o backend atingi **> 80% de coverage** com **pytest + coverage**.
- O frontend como é um protótipo, mantive **testes manuais** e um **linter rigoroso**. Tratando o erro caso o produto não seja encontrado.

Diagrama de uso / fluxo