**Prompt para o desenvolvimento - “Sistema de Comparação de Produtos MercadoLivre”**

Objetivo Principal

Desenvolva um sistema completo de comparação de produtos para e-commerce inspirado no MercadoLivre, implementando uma solução full-stack moderna com frontend React e backend Flask. O sistema deve permitir aos usuários comparar produtos lado a lado com interface intuitiva e responsiva, incluindo API RESTful para gerenciamento de dados de produtos.

Especificações Técnicas Gerais

Stack Tecnológica

• Frontend: React 18 com TypeScript, Vite como build tool

• Backend: Flask (Python 3.11+) com estrutura modular

• Estilização: Tailwind CSS para design responsivo

• Componentes UI: Biblioteca de componentes moderna (shadcn/ui ou similar)

• Banco de Dados: JSON file storage para MVP (facilmente migrável para PostgreSQL)

• API: RESTful com suporte a CORS

• Deployment: Configuração para deploy em ambiente de produção

Arquitetura do Projeto

Plain Text

product-comparison-system/ ├── frontend/ │ ├── src/ │ │ ├── components/ │ │ │ ├── ui/ │ │ │ ├── ProductCard.tsx │ │ │ ├── ComparisonTable.tsx │ │ │ └── ProductFilter.tsx │ │ ├── pages/ │ │ │ └── ComparisonPage.tsx │ │ ├── hooks/ │ │ │ └── useProducts.ts │ │ ├── types/ │ │ │ └── product.ts │ │ ├── utils/ │ │ │ └── api.ts │ │ ├── App.tsx │ │ └── main.tsx │ ├── public/ │ ├── package.json │ ├── vite.config.ts │ ├── tailwind.config.js │ └── tsconfig.json ├── backend/ │ ├── src/ │ │ ├── routes/ │ │ │ ├── products.py │ │ │ └── comparison.py │ │ ├── models/ │ │ │ └── product.py │ │ ├── data/ │ │ │ └── products.json │ │ ├── utils/ │ │ │ └── helpers.py │ │ └── main.py │ ├── requirements.txt │ └── README.md └── README.md

Especificações Detalhadas do Frontend

Componentes Principais

1. ProductCard Component

Crie um componente ProductCard.tsx que exiba:

• Imagem do produto com lazy loading e fallback

• Nome do produto com truncamento inteligente

• Preço atual formatado em moeda brasileira (R$)

• Preço original (se houver desconto) com texto riscado

• Porcentagem de desconto em badge destacado

• Avaliação em estrelas (1-5 estrelas) com número de avaliações

• Status de estoque (disponível, últimas unidades, indisponível)

• Informações de frete (grátis, prazo de entrega)

• Selo MercadoLivre (se aplicável)

• Botão "Adicionar à Comparação" com estado ativo/inativo

• Hover effects suaves e transições CSS

1. ComparisonTable Component

Desenvolva uma tabela de comparação que inclua:

• Layout responsivo que funcione em desktop, tablet e mobile

• Cabeçalho fixo com imagens e nomes dos produtos

• Linhas de especificações organizadas por categorias:

• Informações básicas (preço, marca, modelo)

• Especificações técnicas (dimensões, peso, características)

• Avaliações e reviews

• Informações de entrega e garantia

• Destaque de diferenças entre produtos com cores visuais

• Funcionalidade de scroll horizontal em dispositivos móveis

• Botões de ação (comprar, favoritar, remover da comparação)

• Exportação da comparação em PDF ou imagem

1. ProductFilter Component

Implemente um sistema de filtros avançado:

• Filtro por categoria (eletrônicos, casa, moda, etc.)

• Filtro por faixa de preço com slider duplo

• Filtro por marca com checkbox múltipla

• Filtro por avaliação (4+ estrelas, 3+ estrelas, etc.)

• Filtro por condição (novo, usado, recondicionado)

• Filtro por frete grátis

• Ordenação (menor preço, maior preço, mais vendidos, melhor avaliado)

• Busca por texto com autocomplete

• Filtros salvos para usuários frequentes

Interface do Usuário

Design System

• Paleta de cores: Inspirada no MercadoLivre

• Primária: #3483FA (azul MercadoLivre)

• Secundária: #00A650 (verde)

• Accent: #FFE600 (amarelo)

• Neutros: #666666, #999999, #EEEEEE

• Backgrounds: #FFFFFF, #F5F5F5

• Tipografia:

• Fonte principal: 'Proxima Nova' ou 'Inter' como fallback

• Tamanhos: 12px, 14px, 16px, 18px, 24px, 32px

• Pesos: 400 (regular), 600 (semibold), 700 (bold)

• Espaçamentos: Sistema baseado em 8px (8, 16, 24, 32, 48, 64px)

• Border radius: 4px para elementos pequenos, 8px para cards

• Shadows: Sombras sutis para elevação de elementos

Layout Responsivo

• Desktop (1200px+): Layout de 3-4 colunas para produtos

• Tablet (768px-1199px): Layout de 2-3 colunas

• Mobile (320px-767px): Layout de 1-2 colunas com stack vertical

• Breakpoints customizados para otimização da experiência

Estados e Interações

• Loading states com skeletons animados

• Empty states com ilustrações e call-to-actions

• Error states com mensagens claras e opções de retry

• Success states com feedback visual positivo

• Hover effects em todos os elementos interativos

• Focus states para acessibilidade com teclado

• Transições suaves (300ms ease-in-out)

Funcionalidades Avançadas do Frontend

1. Comparação Inteligente

• Máximo de 4 produtos na comparação simultânea

• Persistência local da comparação (localStorage)

• Compartilhamento da comparação via URL

• Histórico de comparações realizadas

• Sugestões de produtos similares para comparar

1. Busca e Descoberta

• Busca em tempo real com debounce

• Sugestões de busca baseadas em popularidade

• Filtros inteligentes que se adaptam aos resultados

• Busca por imagem (funcionalidade futura)

• Recomendações personalizadas

1. Performance e Otimização

• Lazy loading de imagens e componentes

• Virtualização de listas longas

• Caching de requisições API

• Compressão de imagens automática

• Bundle splitting para carregamento otimizado

Especificações Detalhadas do Backend

Estrutura da API

1. Endpoints Principais

GET /api/products

Retorna lista paginada de produtos com filtros opcionais:

JSON

{ "products": [ { "id": "prod\_001", "name": "iPhone 15 Pro Max 256GB", "brand": "Apple", "category": "smartphones", "price": { "current": 7999.99, "original": 8999.99, "currency": "BRL", "discount\_percentage": 11 }, "images": [ "https://example.com/iphone15-1.jpg", "https://example.com/iphone15-2.jpg" ], "specifications": { "display": "6.7 polegadas Super Retina XDR", "storage": "256GB", "camera": "48MP + 12MP + 12MP", "battery": "4422 mAh", "os": "iOS 17", "weight": "221g", "dimensions": "159.9 x 76.7 x 8.25 mm" }, "rating": { "average": 4.8, "count": 1247, "distribution": { "5": 856, "4": 312, "3": 58, "2": 15, "1": 6 } }, "availability": { "in\_stock": true, "quantity": 45, "shipping": { "free": true, "estimated\_days": 2, "methods": ["standard", "express"] } }, "seller": { "name": "Apple Store Oficial", "rating": 4.9, "verified": true }, "features": [ "Resistente à água IP68", "Carregamento sem fio", "Face ID", "5G" ], "created\_at": "2024-01-15T10:30:00Z", "updated\_at": "2024-01-20T14:22:00Z" } ], "pagination": { "page": 1, "per\_page": 20, "total": 156, "total\_pages": 8 }, "filters\_applied": { "category": "smartphones", "price\_range": [1000, 10000], "brand": ["Apple", "Samsung"] } }

GET /api/products/{id}

Retorna detalhes completos de um produto específico

POST /api/comparison

Cria uma nova comparação entre produtos:

JSON

{ "product\_ids": ["prod\_001", "prod\_002", "prod\_003"], "comparison\_id": "comp\_abc123", "created\_at": "2024-01-20T15:30:00Z" }

GET /api/comparison/{comparison\_id}

Retorna dados formatados para comparação

GET /api/categories

Retorna lista de categorias disponíveis

GET /api/brands

Retorna lista de marcas disponíveis

1. Sistema de Filtros

Implemente filtros avançados que suportem:

• Múltiplos valores por filtro (OR logic)

• Combinação de filtros (AND logic entre diferentes tipos)

• Ranges numéricos (preço, avaliação, etc.)

• Filtros aninhados (subcategorias)

• Ordenação múltipla (preço ASC, rating DESC)

1. Validação e Tratamento de Erros

• Validação de entrada com mensagens claras

• Rate limiting para prevenir abuso

• Logging estruturado para debugging

• Códigos de erro padronizados:

• 400: Bad Request (dados inválidos)

• 404: Not Found (produto não encontrado)

• 429: Too Many Requests (rate limit)

• 500: Internal Server Error

Dados de Exemplo

Produtos de Demonstração

Crie pelo menos 50 produtos de exemplo distribuídos nas categorias:

1. Smartphones (15 produtos)

• iPhone 15 Pro Max, iPhone 15, iPhone 14

• Samsung Galaxy S24 Ultra, S24, S23

• Xiaomi 14 Pro, 13 Pro, Redmi Note 13

• Google Pixel 8 Pro, Pixel 8

• OnePlus 12, 11T Pro

• Motorola Edge 40 Pro, Edge 30

1. Laptops (12 produtos)

• MacBook Pro M3, MacBook Air M2

• Dell XPS 13, XPS 15, Inspiron 15

• Lenovo ThinkPad X1, IdeaPad Gaming

• HP Pavilion, Envy x360

• Asus ZenBook, ROG Strix

• Acer Aspire 5

1. Televisores (10 produtos)

• Samsung QLED 65", 55", 43"

• LG OLED 65", 55"

• Sony Bravia 65", 55"

• TCL 55", 43"

• Philips 50"

1. Eletrodomésticos (8 produtos)

• Geladeira Brastemp, Electrolux

• Máquina de lavar Consul, LG

• Micro-ondas Panasonic, Philco

• Air fryer Mondial, Britânia

1. Casa e Decoração (5 produtos)

• Sofá 3 lugares, Poltrona reclinável

• Mesa de jantar, Rack para TV

• Luminária LED

Especificações Detalhadas por Categoria

Para cada categoria, defina especificações relevantes:

Smartphones:

• Tela (tamanho, resolução, tecnologia)

• Processador (marca, modelo, velocidade)

• Memória RAM e armazenamento

• Câmeras (megapixels, recursos)

• Bateria (capacidade, carregamento)

• Sistema operacional

• Conectividade (5G, Wi-Fi, Bluetooth)

• Resistência (IP rating)

Laptops:

• Processador (Intel/AMD, geração, cores)

• Memória RAM (capacidade, tipo)

• Armazenamento (SSD/HDD, capacidade)

• Tela (tamanho, resolução, tipo)

• Placa de vídeo (integrada/dedicada)

• Bateria (duração estimada)

• Peso e dimensões

• Portas e conectividade

Funcionalidades Específicas a Implementar

1. Sistema de Comparação Avançado

Algoritmo de Comparação Inteligente

Desenvolva um sistema que:

• Identifica automaticamente especificações comparáveis entre produtos

• Destaca diferenças significativas com cores e ícones

• Calcula scores de compatibilidade entre produtos

• Sugere produtos alternativos baseado na comparação atual

• Gera resumos das principais diferenças

Visualização de Dados

• Gráficos comparativos para especificações numéricas

• Barras de progresso para ratings e scores

• Ícones intuitivos para features booleanas

• Códigos de cor para facilitar identificação rápida

• Tooltips informativos para especificações técnicas

1. Sistema de Busca Inteligente

Funcionalidades de Busca

• Busca por texto livre com processamento de linguagem natural

• Busca por especificações ("smartphone com 128GB")

• Busca por faixa de preço ("celular até 2000 reais")

• Busca por marca e modelo com autocomplete

• Correção automática de termos digitados incorretamente

• Sinônimos e variações (celular = smartphone = telefone)

Algoritmo de Relevância

Implemente scoring baseado em:

• Correspondência exata de termos (peso 40%)

• Correspondência parcial em nome e descrição (peso 30%)

• Popularidade do produto (peso 15%)

• Avaliações dos usuários (peso 10%)

• Disponibilidade em estoque (peso 5%)

1. Sistema de Recomendações

Engine de Recomendação

• Produtos similares baseado em especificações

• Produtos complementares (acessórios, cases, etc.)

• Produtos em promoção na mesma categoria

• Produtos mais vendidos para usuários similares

• Produtos recém-lançados relevantes

Personalização

• Histórico de navegação para melhorar sugestões

• Preferências de marca baseadas em interações

• Faixa de preço preferida calculada automaticamente

• Categorias de interesse identificadas pelo comportamento

1. Performance e Otimização

Caching Strategy

• Cache de produtos populares em memória

• Cache de resultados de busca frequentes

• Cache de imagens com CDN simulation

• Invalidação inteligente quando dados são atualizados

Otimizações de Banco de Dados

• Índices otimizados para consultas frequentes

• Paginação eficiente com cursor-based pagination

• Agregações pré-calculadas para filtros

• Compressão de dados JSON quando apropriado

Especificações de Design e UX

Design Visual

Identidade Visual

• Logo: Inspirado no MercadoLivre mas único

• Cores primárias: Azul (#3483FA), Verde (#00A650), Amarelo (#FFE600)

• Cores secundárias: Cinzas neutros para texto e backgrounds

• Gradientes: Sutis para CTAs e elementos de destaque

Componentes de Interface

Cards de Produto

• Aspect ratio 3:4 para imagens de produto

• Padding consistente (16px interno)

• Border radius de 8px

• Shadow sutil para elevação

• Hover effect com elevação aumentada

• Estados visuais claros (loading, error, success)

Botões e CTAs

• Botão primário: Azul MercadoLivre com hover mais escuro

• Botão secundário: Outline azul com hover preenchido

• Botão de perigo: Vermelho para ações destrutivas

• Tamanhos: Small (32px), Medium (40px), Large (48px)

• Estados: Default, Hover, Active, Disabled, Loading

Formulários e Inputs

• Input fields com border sutil e focus state azul

• Labels sempre visíveis (não floating)

• Validation states com cores e ícones apropriados

• Placeholders informativos mas não substitutos de labels

Experiência do Usuário

Fluxos Principais

Fluxo de Comparação

1. Descoberta: Usuário navega ou busca produtos
2. Seleção: Adiciona produtos à comparação (máximo 4)
3. Comparação: Visualiza tabela comparativa detalhada
4. Decisão: Analisa diferenças e toma decisão
5. Ação: Acessa loja para comprar ou salva para depois

Fluxo de Busca

1. Entrada: Usuário digita termo de busca
2. Sugestões: Sistema oferece autocomplete
3. Resultados: Lista produtos relevantes
4. Refinamento: Usuário aplica filtros
5. Seleção: Escolhe produtos para comparar ou visualizar

Micro-interações

• Animação de loading durante carregamento de dados

• Feedback visual ao adicionar produto à comparação

• Transições suaves entre estados

• Confirmações para ações importantes

• Tooltips para elementos que precisam explicação

Acessibilidade

• Contraste adequado (WCAG AA compliance)

• Navegação por teclado em todos os elementos

• Screen reader support com ARIA labels

• Focus indicators visíveis e consistentes

• Texto alternativo para todas as imagens

• Tamanhos de toque adequados (44px mínimo)

Configuração e Deploy

Configuração do Ambiente

Frontend (React + Vite)

JSON

{ "name": "product-comparison-frontend", "version": "1.0.0", "type": "module", "scripts": { "dev": "vite", "build": "tsc && vite build", "preview": "vite preview", "lint": "eslint . --ext ts,tsx --report-unused-disable-directives --max-warnings 0" }, "dependencies": { "react": "^18.2.0", "react-dom": "^18.2.0", "react-router-dom": "^6.8.0", "axios": "^1.3.0", "lucide-react": "^0.263.1", "clsx": "^1.2.1", "tailwind-merge": "^1.14.0" }, "devDependencies": { "@types/react": "^18.2.15", "@types/react-dom": "^18.2.7", "@typescript-eslint/eslint-plugin": "^6.0.0", "@typescript-eslint/parser": "^6.0.0", "@vitejs/plugin-react": "^4.0.3", "autoprefixer": "^10.4.14", "eslint": "^8.45.0", "eslint-plugin-react-hooks": "^4.6.0", "eslint-plugin-react-refresh": "^0.4.3", "postcss": "^8.4.27", "tailwindcss": "^3.3.0", "typescript": "^5.0.2", "vite": "^4.4.5" } }

Backend (Flask)

Plain Text

Flask==2.3.3 Flask-CORS==4.0.0 Flask-RESTful==0.3.10 python-dotenv==1.0.0 marshmallow==3.20.1 Werkzeug==2.3.7 gunicorn==21.2.0

Variáveis de Ambiente

Plain Text

**Backend**

FLASK\_ENV=development FLASK\_DEBUG=True API\_PORT=5000 CORS\_ORIGINS=http://localhost:3000,http://localhost:5173 DATABASE\_URL=sqlite:///products.db SECRET\_KEY=your-secret-key-here

**Frontend**

VITE\_API\_BASE\_URL=http://localhost:5000/api VITE\_APP\_NAME=Product Comparison System VITE\_APP\_VERSION=1.0.0

Scripts de Inicialização

start.sh (Script principal)

Bash

#!/bin/bash echo "Iniciando Product Comparison System..."

**Backend**

echo "Iniciando backend Flask..." cd backend python -m venv venv source venv/bin/activate pip install -r requirements.txt python src/main.py & BACKEND\_PID=$!

**Frontend**

echo "Iniciando frontend React..." cd ../frontend npm install npm run dev & FRONTEND\_PID=$!

echo "Sistema iniciado!" echo "Frontend: http://localhost:5173" echo "Backend API: http://localhost:5000" echo "Para parar o sistema, pressione Ctrl+C"

**Cleanup function**

cleanup() { echo "Parando serviços..." kill $BACKEND\_PID $FRONTEND\_PID exit 0 }

trap cleanup SIGINT wait

Instruções de Implementação Passo a Passo

Fase 1: Setup Inicial (30 minutos)

1. Crie a estrutura de diretórios conforme especificado
2. Configure o backend Flask com estrutura modular
3. Configure o frontend React com Vite e TypeScript
4. Instale dependências e configure scripts de build
5. Teste a comunicação básica entre frontend e backend

Fase 2: Backend Core (45 minutos)

1. Implemente o modelo de dados para produtos
2. Crie os endpoints básicos da API (/products, /categories, /brands)
3. Adicione dados de exemplo (50+ produtos)

Implemente filtros e paginação

1. Configure CORS e middleware de logging
2. Teste todos os endpoints com dados de exemplo

Fase 3: Frontend Base (60 minutos)

1. Configure Tailwind CSS e sistema de design
2. Crie componentes UI básicos (Button, Card, Input)
3. Implemente ProductCard component
4. Crie página de listagem de produtos
5. Adicione sistema de filtros básico
6. Implemente busca em tempo real
7. Teste responsividade em diferentes tamanhos de tela

Fase 4: Sistema de Comparação (45 minutos)

1. Implemente ComparisonTable component
2. Adicione funcionalidade de adicionar/remover produtos da comparação
3. Crie endpoint de comparação no backend
4. Implemente persistência local da comparação
5. Adicione compartilhamento via URL
6. Teste fluxo completo de comparação

Fase 5: Refinamentos e Otimização (30 minutos)

1. Adicione loading states e error handling
2. Implemente lazy loading de imagens
3. Otimize performance com memoização
4. Adicione animações e micro-interações
5. Teste acessibilidade básica
6. Ajuste design e responsividade final

Fase 6: Deploy e Documentação (15 minutos)

1. Configure scripts de build para produção
2. Teste build de produção
3. Crie documentação básica (README.md)
4. Configure variáveis de ambiente
5. Teste deploy local

Critérios de Qualidade e Validação

Funcionalidades Obrigatórias

• ✅ Listagem de produtos com filtros funcionais

• ✅ Busca em tempo real com resultados relevantes

• ✅ Comparação lado a lado de até 4 produtos

• ✅ Interface responsiva (desktop, tablet, mobile)

• ✅ API RESTful com endpoints documentados

• ✅ Dados de exemplo realistas e variados

• ✅ Performance adequada (carregamento < 3s)

Critérios de Qualidade

• Código limpo e bem estruturado

• Componentes reutilizáveis e modulares

• Tratamento de erros adequado

• Loading states em todas as operações assíncronas

• Validação de dados no frontend e backend

• Acessibilidade básica (contraste, navegação por teclado)

• Design consistente com sistema estabelecido

Testes de Validação

1. Teste de navegação: Usuário consegue navegar facilmente entre produtos
2. Teste de busca: Busca retorna resultados relevantes rapidamente
3. Teste de comparação: Comparação funciona com diferentes tipos de produtos
4. Teste de responsividade: Interface funciona bem em diferentes dispositivos
5. Teste de performance: Aplicação carrega rapidamente mesmo com muitos produtos
6. Teste de usabilidade: Interface é intuitiva para usuários não técnicos

Considerações Finais

Este prompt fornece especificações completas para desenvolver um sistema de comparação de produtos profissional e funcional. O sistema deve ser implementado seguindo as melhores práticas de desenvolvimento web moderno, com foco em experiência do usuário, performance e manutenibilidade do código.

A implementação deve resultar em uma aplicação que demonstre competência técnica em desenvolvimento full-stack, design de APIs, interface de usuário moderna e arquitetura de software escalável. O produto final deve ser uma demonstração convincente de habilidades em desenvolvimento web contemporâneo, adequado para apresentação em contextos profissionais ou acadêmicos.

Lembre-se de manter o código bem documentado, seguir convenções de nomenclatura consistentes e implementar tratamento adequado de erros em todas as camadas da aplicação. O objetivo é criar não apenas uma aplicação funcional, mas um exemplo de excelência em desenvolvimento de software.