

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
DO RIO GRANDE DO NORTE
CAMPUS CAICÓ

ALAN PEREIRA DA SILVA
FABIAN MESSY CASSIANO FERNANDES DA SILVA
FLAUBER SAUAN CANDIDO ALVES
GUILHERME DAVID ARAÚJO DE OLIVEIRA
JOÃO PAULO FERNANDES ALVES

GERENCIADOR DE ESTUDOS PESSOAIS

RELATÓRIO DE PROJETO INTEGRADOR

CAICÓ-RN
2025

SUMÁRIO

RESUMO.....	2
1. INTRODUÇÃO.....	3
2. OBJETIVOS.....	3
2.1 OBJETIVO GERAL.....	3
2.2 OBJETIVOS ESPECÍFICOS.....	3
3. METODOLOGIA.....	3
4. RESULTADOS.....	4
5. CONCLUSÃO.....	4
6. REFERÊNCIAS BIBLIOGRÁFICAS.....	4

RESUMO

O projeto foi criado com o intuito de criar uma plataforma voltada à venda de cursos online, visando aplicar conhecimentos adquiridos ao longo da formação acadêmica, além dos conhecimentos adquiridos, foi necessário buscar novas ferramentas e tecnologias para o desenvolvimento. O sistema foi desenvolvido ao longo de 3 semanas, tendo etapas de planejamento, análise de requisitos e implementação do código.

1. INTRODUÇÃO

O crescimento da educação digital evidenciou a necessidade de plataformas que possibilitem a oferta e a organização de cursos online de forma estruturada, acessível e segura.

Com o avanço das tecnologias da informação e a ampliação do ensino a distância, tornou-se fundamental o desenvolvimento de sistemas capazes de gerenciar conteúdos educacionais, usuários e processos de aquisição de cursos em um ambiente digital integrado.

Nesse contexto, o projeto foi desenvolvido como uma solução acadêmica para simular uma plataforma de venda de cursos, permitindo a aplicação prática de conceitos teóricos relacionados ao desenvolvimento de sistemas web. Com o desafio a mais de conhecer novas tecnologias e desenvolver o sistema

2. OBJETIVOS

2.1 OBJETIVO GERAL

Propor uma ferramenta digital destinada ao ensino através da disponibilização de cursos, com o objetivo de incentivar o ensino e eficiência nos estudos.

2.2 OBJETIVOS ESPECÍFICOS

- Implementar uma interface intuitiva, acessível e responsiva, que facilite a navegação e o uso da plataforma por diferentes estudantes de diferentes níveis de familiaridade com a tecnologia.

3. METODOLOGIA

O frontend foi desenvolvido com HTML, CSS (framework - Tailwind), JavaScript (framework - React). Tendo como objetivo principal a construção de uma interface intuitiva.

O banco de dados será estruturado em SQLite, com tabelas normalizadas e relacionamentos entre usuários e cursos. As senhas serão armazenadas de forma segura, utilizando criptografia.

O backend foi desenvolvido utilizando o microframework Flask, em Python, sendo responsável por toda a lógica do sistema. Além disso, foram incluídos módulos de autenticação e autorização, funcionalidades completas de CRUD (criação, leitura, atualização e exclusão de dados).

A parte de documentação foi desenvolvida por Flauber Sauan e Alan Pereira. O backend e frontend foram desenvolvidos por Guilherme David, Fabian Messy e João Paulo.

Os principais endpoints foram:

- **GET/items:** Irá ler os itens, pegar o tipo e o status.

Condições:

- Se tiver tipo, variável items será uma lista dos items com tipo.
- Se tiver status, variável items será uma lista dos items com status.
- O retorno será a lista dos items em formato json.

- **POST/items:** Recebe **JSON**, valida e cria (ou **400** se falhar).

Condições:

- Se tiver erro, retorna o erro em json com o padrão 400;

Caso não tenha erro:

- vai ler os items com a função 'read_item'.
- definir um novo id que será o id máximo em items + 1.
- um item terá o padrão: "id" : novo_id, **dados
- items lista receberá o item, e o item será escrito no items.json;
- Retorna o item em json, no padrão 201.

- **PUT/items:** Busca pelo ID: se houver erro no `validate item`, retorna 400; se não encontrar, 404; se ok, atualiza e retorna o JSON.

Se não houver erro:

- Items = items em json;
- Busca o ID, sobrescreve os dados e retorna a lista.
- Busca o ID: se achar, atualiza e retorna o JSON; se não, retorna 404.

- **PATCH/items**: Atualiza via **ID**: se o novo status for inválido, retorna **400**; se ok, aplica o **JSON**.

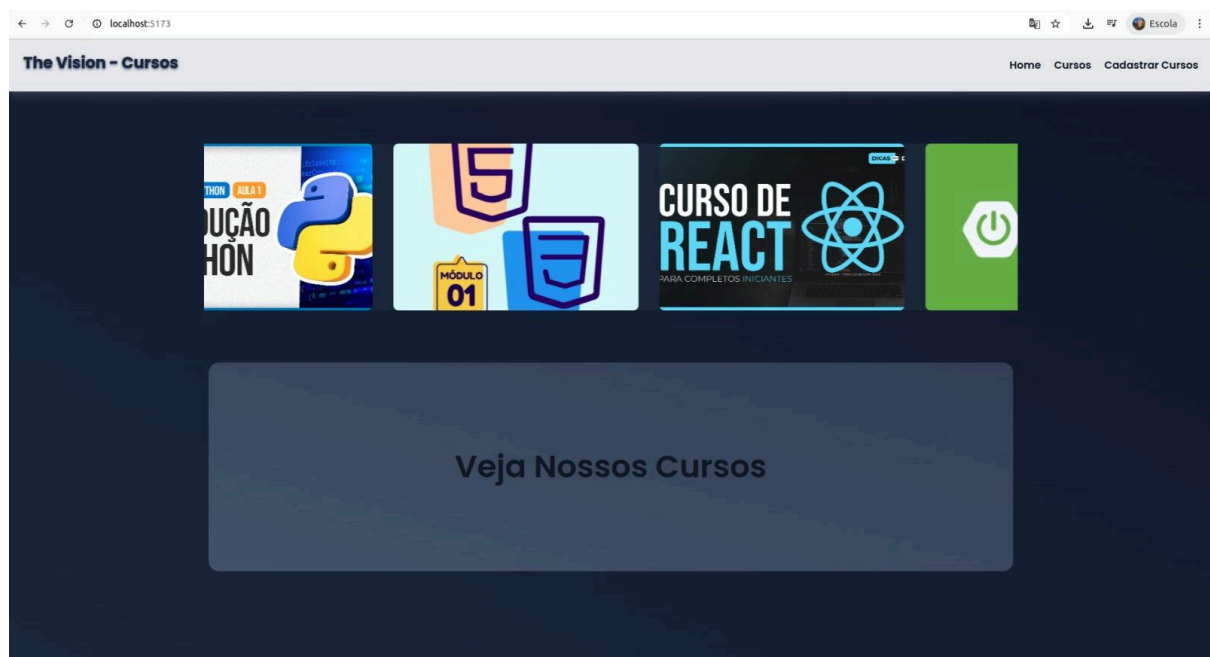
Se não houver erro:

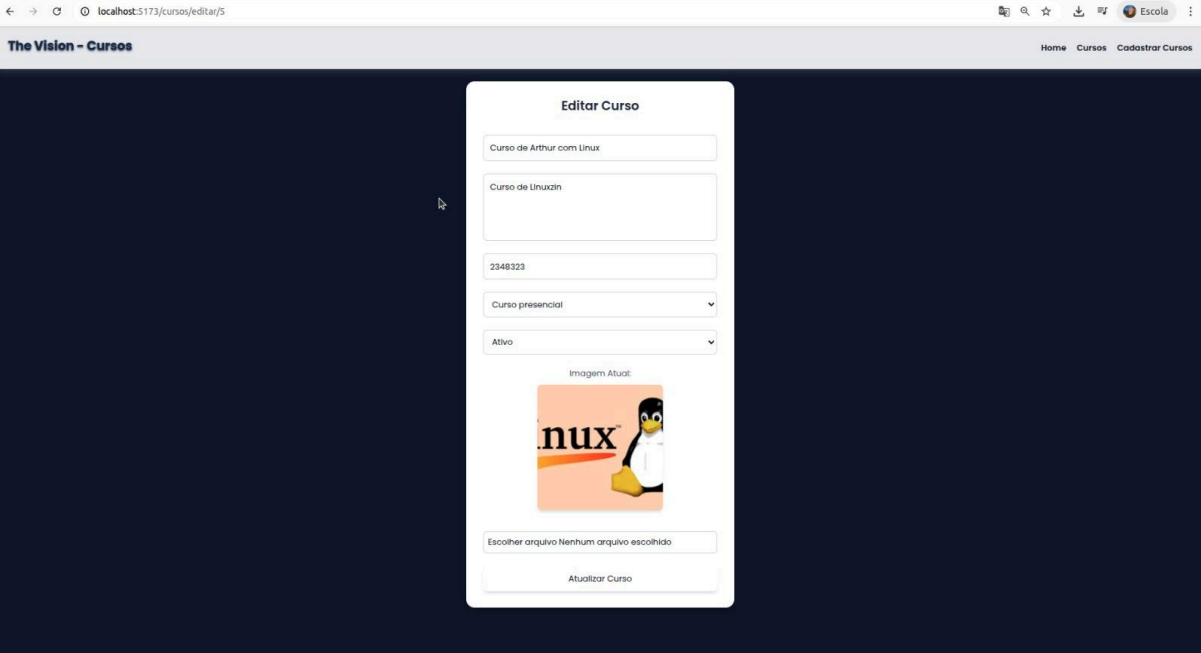
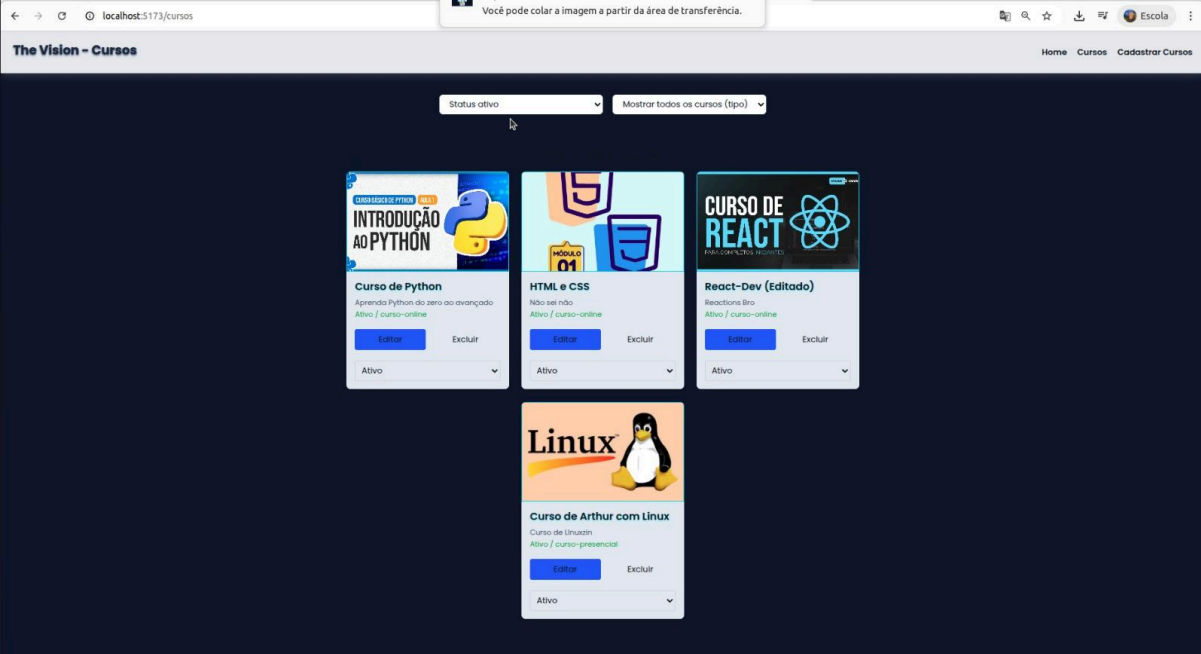
- Items = itens em json;
- Busca o **ID** e **atualiza** o status.
- Retorna o item (**JSON**) ou **404** se não existir.

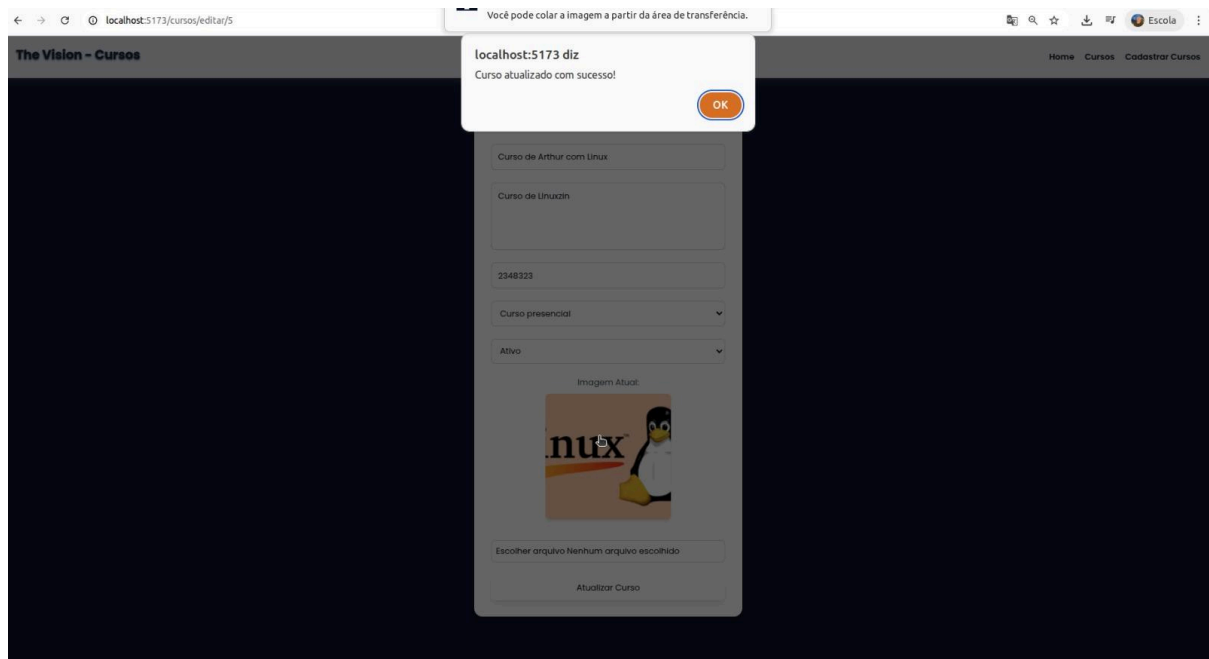
- **DELETE/items**: Irá deletar um item via id, recebendo os dados json.

- Filtra o array removendo o **ID** escolhido.
- O retorno então, será uma string vazia no padrão 204.

4. RESULTADOS







5. CONCLUSÃO

Apesar de atender aos objetivos propostos, o sistema conta com algumas limitações, em relação ao caráter acadêmico.

Uma das principais limitações foi a ausência de integração com sistemas reais de pagamento, sendo o processo de compra apenas simulado. Essa decisão foi necessária para evitar riscos financeiros e questões legais, mas restringiu a validação do sistema em um ambiente comercial real.

Outra limitação refere-se à escala da aplicação, uma vez que o sistema não foi projetado para suportar um grande volume de acessos simultâneos. A infraestrutura utilizada é adequada para fins de estudo e demonstração, mas não contempla requisitos avançados de desempenho, como balanceamento de carga e alta disponibilidade.

Também se destaca como limitação a restrição de funcionalidades, como a ausência de sistema de avaliações, certificação automática e acompanhamento detalhado do progresso dos usuários, funcionalidades comuns em plataformas educacionais comerciais.

6. REFERÊNCIAS BIBLIOGRÁFICAS

TAILWIND CSS. *Tailwind*. Disponível em: <https://tailwindcss.com/>. Acesso em: 03 fev. 2026.

REACT. *React*. Disponível em: <https://pt-br.react.dev/learn> Acesso em: 02 fev. 2026.