Marketplace de itens usados

Natã Cezer bordignon, João Pedro Strada, Gabriel Martins

Departamento de Sistemas e Computação
Universidade Regional Integrada (URI) – Erechim, RS, Brasil

103574@aluno.uricer.edu.br,
104095@aluno.uricer.edu.br,103799@aluno.uricer.edu.br

Abstract. *This paper presents the architecture and implementation of a fullstack system for a second-hand item marketplace. The backend, built with Node.js, Express, TypeORM, and PostgreSQL, offers endpoints for user and product management, JWT-based authentication, and integration with Docker services. The frontend, developed separately, consumes these APIs and provides an intuitive interface for user interaction. The project demonstrates a modular and scalable design, covering both server-side and client-side development.*

Resumo. *Este artigo apresenta a arquitetura e a implementação de um sistema fullstack para um marketplace de itens usados. O backend, desenvolvido com Node.js, Express, TypeORM e PostgreSQL, oferece endpoints para gerenciamento de usuários e produtos, autenticação baseada em JWT e integração com serviços via Docker. O frontend, desenvolvido separadamente, consome essas APIs e fornece uma interface intuitiva para interação dos usuários. O projeto demonstra um design modular e escalável, cobrindo tanto o desenvolvimento do lado do servidor quanto do lado do cliente.*

1. Introdução

A venda de produtos usados pela internet se tornou algo comum no dia a dia das pessoas, mas ainda assim muitas plataformas existentes acabam sendo complicadas de usar ou deixam a desejar em termos de organização e segurança. Pensando nisso, este projeto teve como objetivo criar um sistema completo (fullstack) para um marketplace de itens usados, que fosse simples, funcional e fácil de manter.

O sistema foi dividido entre backend e frontend. No backend, foram utilizadas tecnologias como Node.js, Express, TypeORM e PostgreSQL, além de autenticação com JWT e uso de contêineres Docker para facilitar o ambiente de desenvolvimento. Já no frontend, foi construída uma interface independente que se comunica com a API, permitindo o cadastro de usuários, gerenciamento de produtos e outras funcionalidades essenciais para o funcionamento do marketplace.

2. Trabalhos relacionados

Plataformas como OLX e Enjoei são exemplos populares de marketplaces de itens usados no Brasil. Elas permitem que usuários anunciem produtos, conversem com interessados e realizem negociações de forma independente. Apesar de serem amplamente utilizadas, essas plataformas possuem estruturas mais complexas, funcionalidades voltadas para grande escala e muitas vezes exigem integrações com sistemas de pagamento, avaliação e verificação.

Diferente dessas soluções consolidadas, este projeto foi desenvolvido com foco acadêmico, com o objetivo de aplicar na prática conceitos de desenvolvimento web fullstack. O sistema foi pensado para ser simples e funcional, com arquitetura modular, autenticação segura, integração com banco de dados relacional e um ambiente containerizado. Além disso, o controle total do código-fonte permite fácil expansão e personalização, o que torna a aplicação ideal para estudos, testes e evolução contínua em sala de aula ou projetos pessoais.

3. Tecnologias Utilizadas

O sistema foi desenvolvido com uma arquitetura fullstack, utilizando diversas tecnologias no backend, frontend e para suporte e testes. No backend, foram utilizadas as seguintes tecnologias: Node.js, ambiente de execução JavaScript para o servidor; Express, framework minimalista para criação de APIs e gerenciamento de rotas; TypeORM, biblioteca ORM para mapeamento objeto-relacional com PostgreSQL; PostgreSQL, sistema gerenciador de banco de dados relacional utilizado para armazenar as informações do sistema; JWT (JSON Web Token), mecanismo para autenticação e autorização de usuários; e Docker, ferramenta de contêinerização utilizada para facilitar o desenvolvimento, testes e deploy do ambiente.

O Node.js permite a execução de código JavaScript fora do navegador, tornando possível o desenvolvimento de aplicações web no lado do servidor. O Express complementa o Node.js ao oferecer recursos para criar rotas, lidar com requisições HTTP e estruturar o backend de forma simples e eficiente. O TypeORM facilita a comunicação com o banco de dados PostgreSQL, permitindo a criação e manipulação de tabelas e registros por meio de objetos e métodos JavaScript/TypeScript, sem a necessidade de escrever consultas SQL manuais. O PostgreSQL é o sistema de banco de dados relacional adotado, escolhido por sua robustez, segurança e suporte a operações complexas de consulta. Para garantir a segurança e o controle de acesso, foi implementada a autenticação baseada em JWT, que permite validar a identidade dos usuários por meio de tokens seguros.

No frontend, a aplicação foi construída com HTML e CSS para a estruturação e estilização das páginas web, enquanto o React foi utilizado para criar componentes interativos e reativos. O React permite que o frontend consuma a API exposta pelo backend e atualize a interface de forma dinâmica, sem a necessidade de recarregar a página, proporcionando uma melhor experiência para o usuário.

Além disso, foram utilizadas ferramentas de apoio para o desenvolvimento e testes: Docker, para isolar e executar os serviços do projeto em contêineres, garantindo a padronização do ambiente de execução; Adminer, ferramenta web para gerenciamento visual do banco de dados PostgreSQL; e Insomnia, uma ferramenta de testes para requisições HTTP, utilizada para validar os endpoints e a comunicação entre frontend e backend durante o desenvolvimento do sistema.

4. Arquitetura do Sistema

O sistema foi desenvolvido com uma arquitetura fullstack baseada na separação entre frontend e backend. O backend expõe uma API RESTful, enquanto o frontend, desenvolvido em React, HTML e CSS, consome essa API para fornecer uma interface interativa aos usuários. A comunicação entre as camadas é feita através de requisições HTTP no formato JSON.

4.1 Usuário

O módulo de usuários permite o cadastro, login e autenticação. As principais rotas são:

- POST /api/users/register: realiza o cadastro de novos usuários no sistema.
- POST /api/users/login: autentica usuários existentes, retornando um token JWT para autorização de requisições futuras.
- GET /api/users/auth: valida o token JWT e autentica o usuário, garantindo o acesso às funcionalidades protegidas.

4.2 Produto

O módulo de produtos permite o cadastro, consulta, edição e remoção de itens no marketplace. As rotas implementadas são:

- POST /api/products/register: cadastra novos produtos no sistema, incluindo informações como nome, preço, categoria e descrição.
- PUT /api/products/:id: edita as informações de um produto existente, identificado pelo seu ID.
- DELETE /api/products/:id: remove um produto do sistema.
- GET /api/products/all: retorna a lista de todos os produtos cadastrados no marketplace.
- GET /api/products/:id: consulta as informações detalhadas de um produto específico.

4.3 Carrinho

O módulo de carrinho permite a adição, consulta, edição e remoção de itens no carrinho de compras do usuário. As rotas disponíveis são:

- POST /api/cart/add: adiciona produtos ao carrinho.
- GET /api/cart/all: retorna todos os itens adicionados ao carrinho pelo usuário.
- PUT /api/cart/:id: edita a quantidade ou os detalhes de um item específico no carrinho.
- GET /api/cart/:id: consulta as informações de um item específico no carrinho.
- DELETE /api/cart/:id: remove um item específico do carrinho.

O sistema segue o padrão MVC (Model-View-Controller), no qual os controllers são responsáveis pela lógica de negócio, os models representam as tabelas no banco de dados e as rotas direcionam as requisições para os controllers correspondentes. A camada de persistência foi implementada com o TypeORM, mapeando as entidades para tabelas no PostgreSQL, enquanto a autenticação é gerenciada via JWT, garantindo a segurança das operações.

Referencias

VIEIRA, Rosângela da Silva; AMARAL, Hellen Fernanda. *Plataformas de marketplace e o impacto no comportamento do consumidor: estudo de caso da OLX*. Revista de Gestão e Negócios, v. 13, n. 1, p. 78-95, 2021. Disponível em: https://revistadegestaoenegocios.com.br. Acesso em: 02 jun. 2025.

KOTLER, Philip; KELLER, Kevin Lane. *Administração de marketing*. 15. ed. São Paulo: Pearson, 2016.

CHRISTENSEN, Clayton. *O Dilema da Inovação: quando as novas tecnologias levam as grandes empresas à falência*. São Paulo: MBooks, 2011.

POSTGRESQL. *Documentação oficial do PostgreSQL*. Disponível em: https://www.postgresql.org/docs/. Acesso em: 02 jun. 2025.

DOCKER. *Documentação oficial do Docker*. Disponível em: https://docs.docker.com/. Acesso em: 02 jun. 2025.

REACT. *Documentação oficial do React*. Disponível em: https://react.dev/. Acesso em: 02 jun. 2025.

TYPEORM. *Documentação oficial do TypeORM*. Disponível em: https://typeorm.io/. Acesso em: 02 jun. 2025.