

Arquitetura de Sistemas Distribuídos



E-commerce

- Italo Fideles Vieira do Nascimento
- Jully Anne Roman Palhano Dutra
- Lucas Morais Barcelos
- Pedro Henrique Nunes Alves
- Victor Hugo Vasquez da Silva
- Vinícius Pereira Coelho

Orientador:
Kleber Jacques Ferreira de Souza



Plano

01 INTRODUÇÃO

02 ESPECIFICAÇÕES DO PROJETO

03 APIS E WEB SERVICES

04 APLICAÇÃO WEB E MOBILE



Introdução

O e-commerce transformou a forma de comprar e vender. Com internet e mobile, o digital virou um dos principais canais, dominado por plataformas como Mercado Livre, Amazon e Shopee. Consumidores procuram praticidade, variedade e segurança.

Desafios dos pequenos vendedores

📢 Baixa divulgação • 📦 Gestão de estoque/vendas • 📊 Falta de análises

→ Necessitam de soluções simples, escaláveis e integradas.

Problema para consumidores

- Falta de uma plataforma unificada
- Uso de vários aplicativos, aumentando tempo e esforço



Objetivos

Objetivo Geral

Desenvolver a **Zabbix Store**, uma plataforma de e-commerce que conecte fornecedores e compradores em um ambiente seguro, escalável e com recursos de análise de vendas.

Objetivos Específicos

- 🛒 Cadastro simples e intuitivo de fornecedores e produtos
- 📊 Painel (dashboard) para acompanhamento e análise de vendas
- 🔍 Busca e filtragem eficiente de produtos
- 🔒 Segurança nas transações e proteção de dados
- 🌐 Acesso via web e dispositivos móveis



Catálogo de Serviços

Serviços de Autenticação e Autorização

- Serviço de Registro de Usuário
- Serviço de Gerenciamento de Conta
- Serviço de Login
- Serviço de Gerenciamento de Sessão

Serviços de Catálogo de Produtos

- Serviço de Cadastro de Produto
- Serviço de Consulta e Busca de Produtos
- Serviço de Gerenciamento de Estoque

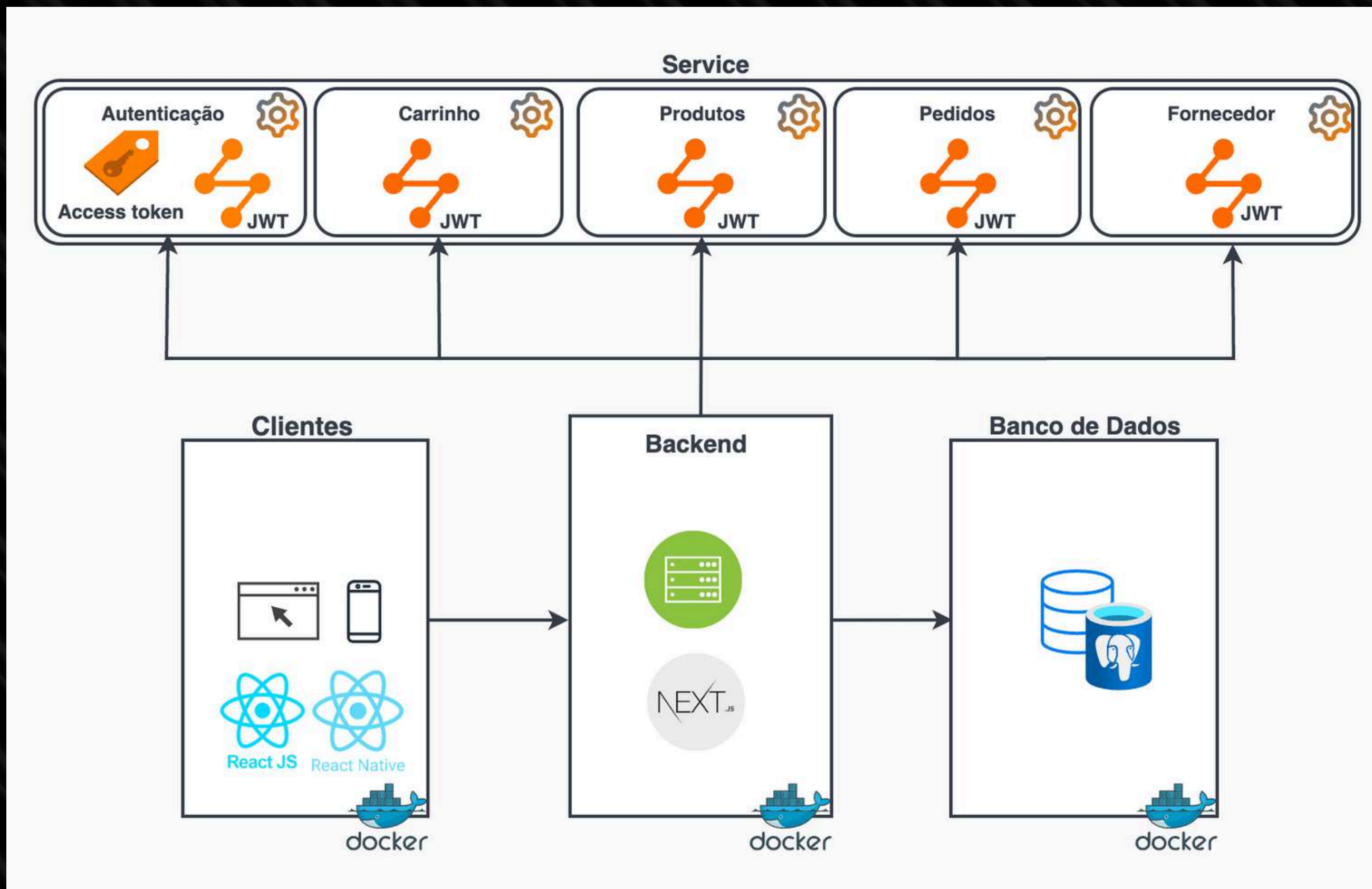
Serviços de Carrinho e Pedidos

- Serviço de Carrinho de Compras
- Serviço de Criação de Pedido





Arquitetura da Solução





Tecnologias Utilizadas

Backend



NestJS



Prisma ORM

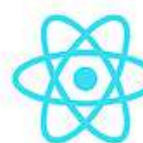


PostgreSQL



API REST

Frontend Web

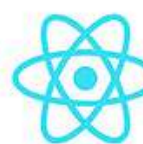


React



Swagger

Aplicativo Mobile



React Native

Ferramentas e IDEs



Visual
Studio
Code



Postman
Insomnia



Git &
GitHub



Fluxo de Dados da Aplicação

O fluxo de dados da ZabbixStore é baseado em uma arquitetura cliente-servidor, onde o frontend React consome APIs REST do backend NestJS, que por sua vez interage com o banco de dados PostgreSQL através do Prisma ORM.

Arquitetura Geral

WEB

```
Frontend (React) ↔ Backend (NestJS) ↔ Banco de Dados (PostgreSQL)
  ↓                ↓                ↓
- Contextos React  - Controllers      - Prisma ORM
- Hooks customizados - Services        - Schema definido
- Componentes      - DTOs/Validação    - Migrations
- LocalStorage     - JWT Auth          - Relacionamentos
```

Mobile

```
Frontend Mobile (React Native/Expo) ↔ Backend (NestJS) ↔ Banco de Dados (PostgreSQL)
  ↓                ↓                ↓
- React Navigation  - Controllers      - Prisma ORM
- Context API        - Services        - Schema definido
- Componentes       - DTOs/Validação    - Migrations
- AsyncStorage      - JWT Auth          - Relacionamentos
- Hooks customizados
```

↑
TOP Header
🏠 Home
🔑 Área de Login
👤 Área Logada
🛒 Carrinho de Compras

📦 Produto Selecionado
❤️ Lista de Favoritos
📊 Painel Administrativo - Visão Geral
📦 Painel Administrativo - Produtos
ℹ️ Sobre

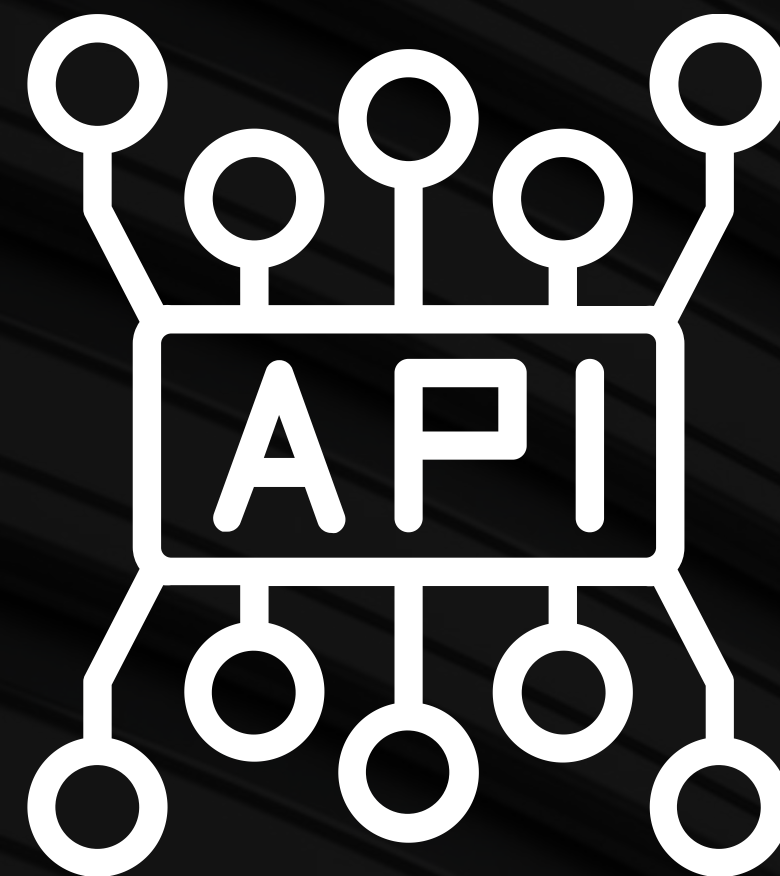


APIs e Web Services

🔗 APIs para integração segura

Funcionalidades:

- Autenticação de usuários
- Gerenciamento de produtos
- Validação de endereços
- Processamento de pedidos





Considerações de Segurança

Autenticação e Autorização

Uso de JWT com expiração definida e controle de acesso por perfis (cliente e fornecedor).



Proteção de Dados Sensíveis

Senhas armazenadas com hash seguro (bcrypt ou argon2).

Comunicação Segura

Toda a transmissão de dados ocorre via HTTPS e CORS configurado apenas para domínios autorizados.

Mitigação de Ataques Comuns

Prevenção contra SQL Injection (Prisma e validações), XSS (sanitização de entradas e modo estrito do React), CSRF (tokens e cabeçalhos seguros) e ataques de força bruta (rate limiting).

Monitoramento

Logs de atividades suspeitas para rastreamento e auditoria



Casos de Teste por Funcionalidade

Header

- Teste de Renderização
- Teste de Interações
- Teste de Funcionalidades
- Teste de Estados
- Teste de Responsividade
- Teste de Acessibilidade
- Teste por Tipo de Usuário

Home

- Teste de Renderização
- Teste de Interações
- Teste de Estados
- Teste de Responsividade
- Teste de Acessibilidade
- Teste por Tipo de Usuário

Autenticação (Login/Cadastro)

- Teste de Login
- Teste de Cadastro
- Teste de Validações
- Teste de Estados
- Teste de Responsividade
- Teste de Acessibilidade
- Teste de Segurança
- Teste por Tipo de Usuário

Área Logada do Usuário

- Teste de Renderização
- Teste de Interações
- Teste de Estados
- Teste de Validações
- Teste de Responsividade
- Teste de Acessibilidade
- Teste de Segurança
- Teste por Tipo de Usuário

Carrinho de Compras

- Teste de Renderização
- Teste de Interações
- Teste de Estados
- Teste de Validações
- Teste de Responsividade
- Teste de Acessibilidade
- Teste por Tipo de Usuário

Página de Produtos

- Teste de Renderização
- Teste de Interações
- Teste de Estados
- Teste de Validações
- Teste de Responsividade
- Teste de Acessibilidade
- Teste por Tipo de Usuário

Lista de Favoritos

- Teste de Renderização
- Teste de Interações
- Teste de Estados
- Teste de Validações
- Teste de Responsividade
- Teste de Acessibilidade
- Teste por Tipo de Usuário

Painel Administrativo

- Teste de Visão Geral
- Teste de Gerenciamento de Produtos
- Teste de Validações
- Teste de Responsividade
- Teste de Acessibilidade
- Teste de Segurança
- Teste por Tipo de Usuário

Foram feitos mais de 200
Casos de Teste



Hospedagem da Aplicação

Hospedagem da Aplicação

Infraestrutura (AWS EC2 + Docker)

- Instância EC2 com Docker Compose
- Containers: Frontend, Backend, Banco de Dados (Postgres)
- Rede e Segurança: VPC com sub-redes públicas/privadas, Security Groups (80/443 públicas; 5432/API internas)
- Load Balancer: opcional para escalabilidade futura
- Justificativa: controle, flexibilidade e baixo custo inicial; futura migração para ECS/Fargate ou Kubernetes

CI/CD (GitHub Actions)

- Pipeline: Build → Testes unitários → Deploy
- Deploy: Staging automático via SSH; Produção com aprovação manual
- Gerenciamento de segredos: GitHub Secrets + AWS Secrets Manager
- Justificativa: integração direta com repositório e deploy simplificado



Hospedagem da Aplicação

■ Mobile (APK)

- Ferramentas: Expo Go + EAS Build
- APK unsigned para QA interno; .aab planejado para publicação futura
- Distribuição interna inicialmente
- Justificativa: build simplificado, sem custos de publicação

📊 Monitoramento e Logs

- Infraestrutura: AWS CloudWatch (métricas EC2 + logs de containers)
- Aplicação: logs centralizados com Request ID + integração com Sentry
- Justificativa: alertas e análise simplificados com monitoramento nativo



Implantação

Pipelines

Hetzner

CX22
#108697257

● hc-zabbix-store
🌐 91.99.239.230 🌐 2a01:4f8:c2c:b832::/64 ➕ Add Flo

Overview Graphs Backups Snapshots Load Balancers Networking Firewa

🏠 2
VCPU

💾 4 GB
RAM

💾 40 GB
DISK LOCAL

💰 2.41
USAGE

📶 0/20 TB
TRAFFIC OUT

← Build and Deploy

✅ Build and Deploy #5

🏠 Summary

Jobs

✅ Build Zabbix - Frontend

✅ Build Zabbix - Backend

✅ Deploy to Hetzner

Run details

🕒 Usage

📄 Workflow file

Manually triggered last week

👤 dutrajully 🔑 e1ad5b3 📌 main

Status

Success

Total duration

2m 6s

Artifacts

—

build_push.yml

on: workflow_dispatch

✅ Build Zabbix - Frontend 43s

✅ Build Zabbix - Backend 1m 13s

✅ D.../ Deploy to Hetzner Ser... 46s

DockerHub

Name
pnunesdev/zabbixstore-backend
pnunesdev/zabbixstore-frontend

ZABBIX

Eletrônicos Fashion Casa Sobre

Buscar produtos...

Cadastrar Entrar

Descubra o melhor em cada categoria

Na Zabbix, você encontra uma seleção cuidadosa de produtos para todas as suas necessidades. Qualidade, estilo e preços justos.

Ver Produtos → Sobre Nós

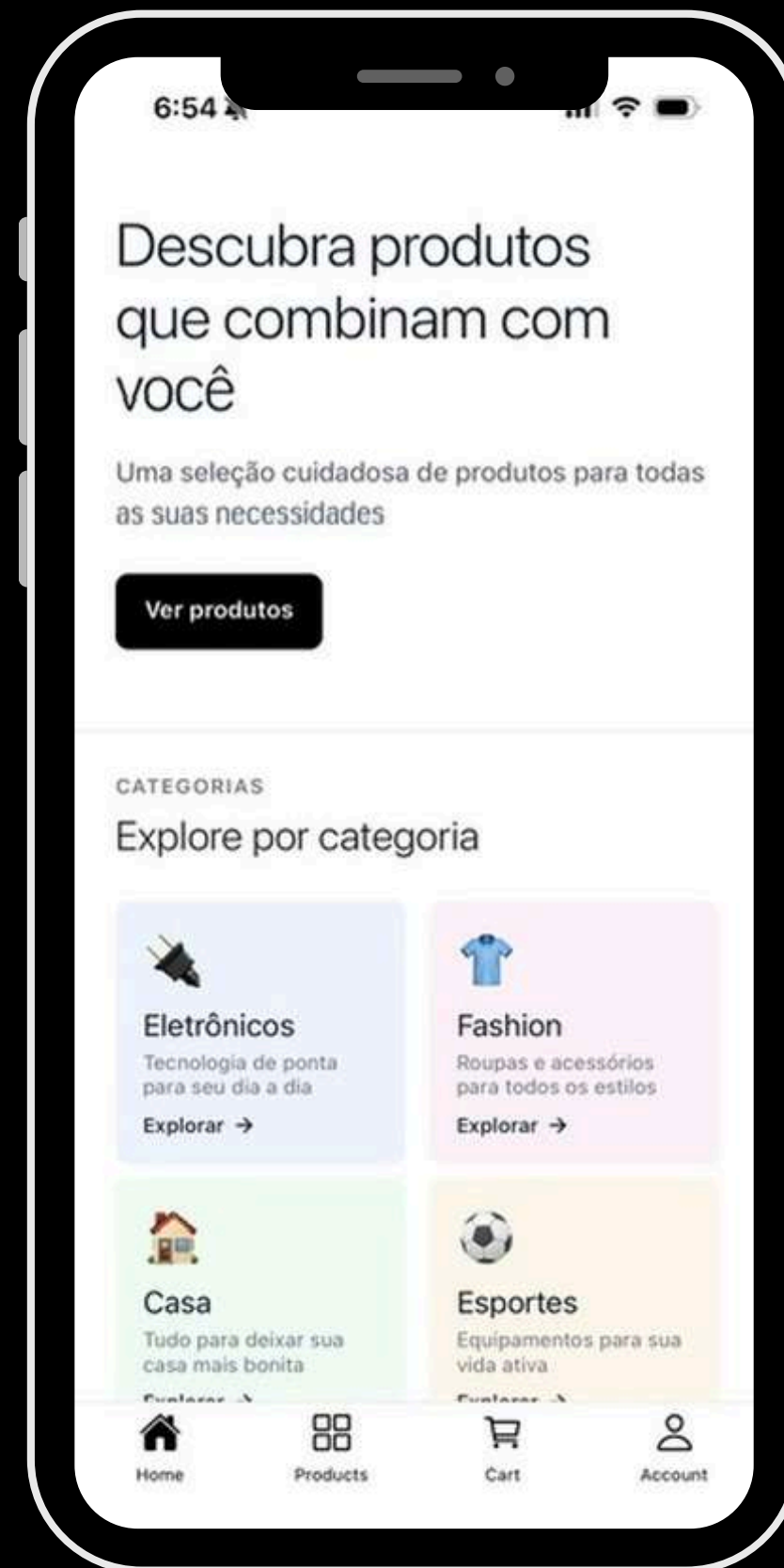
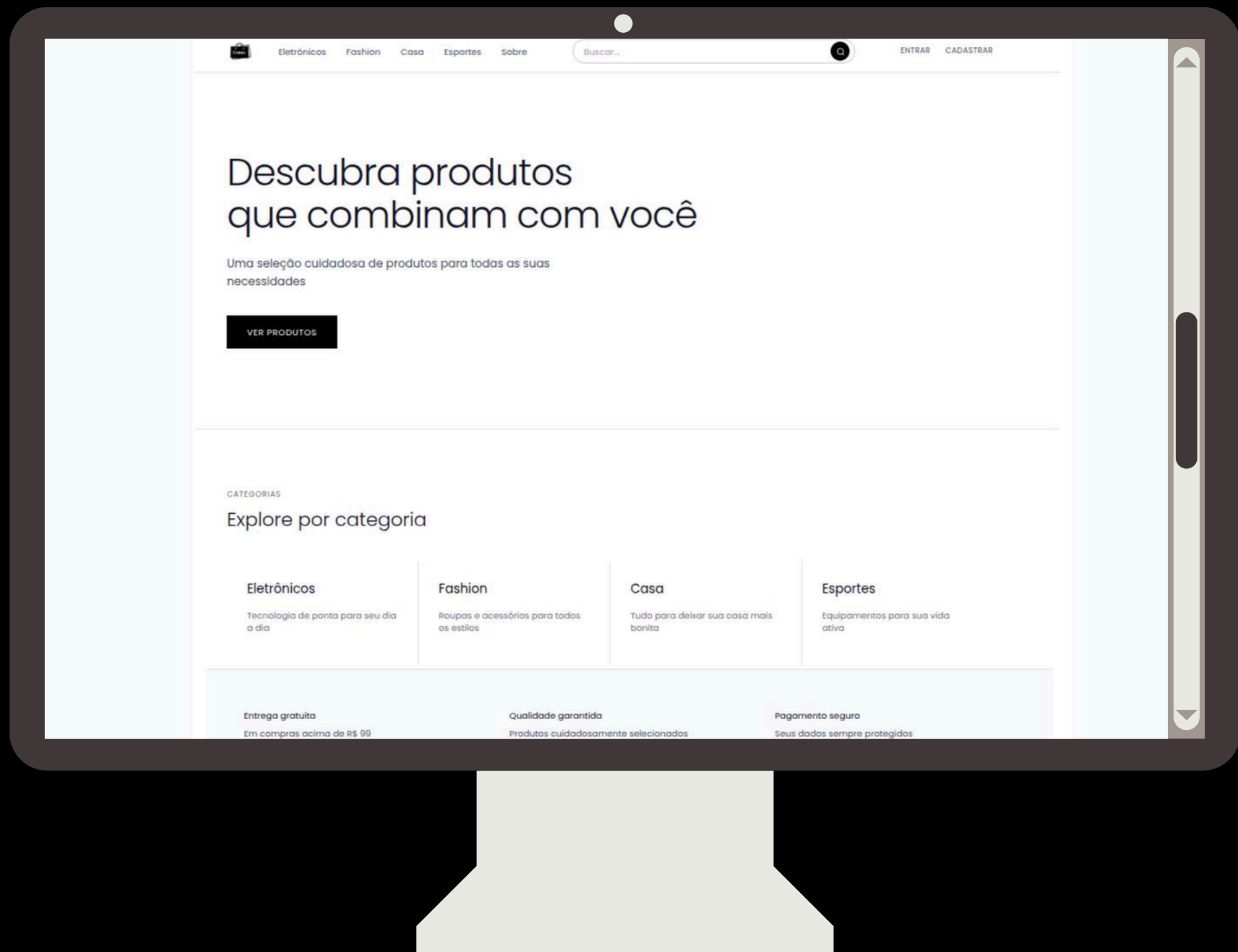
ZABBIX



Aplicação

Website

Mobile





Obrigado