CENTRO UNIVERSITÁRIO AUGUSTO MOTTA

Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas
Campo Grande - Noturno

Projeto para o módulo de BackEnd

Tema:

Adriel Furtado, Lucas Tadashi e Gabryel Fellipe.

https://github.com/d3kinho/projeto-formadora/tree/main/Projeto%20-%20estacao%2050

Rio de Janeiro, RJ
Setembro/ 2025

Sumário

1.Introdução

Este projeto tem por objetivo implementar uma aplicação WEB integrando o FRONTEND com o BACKEND, fazendo uso de banco de dados juntamente com o uso do PHP. Um dos obstáculos encontrados, até o momento, foi a necessidade do uso de um servidor, que foi remediada com o uso do XAMPP, que possibilita um servidor local, que é o necessário para o desenvolver e testar aplicações.

1.1. Objetivo

O objetivo deste projeto trata-se de uma loja virtual de uma lanchonete, onde um usuário pode se cadastrar, ver a descrição de um produto e por fim comprar, visando facilitar e modernizar a compra dos produtos da lanchonete.

2.Levantamento de Requisitos

Requisitos Funcionais (RF)

RF-01. Autenticação de Usuário: A aplicação deve permitir o cadastro de novos usuários e o login de usuários existentes. Usuários comuns podem se cadastrar pelo sistema, enquanto usuários "master" são criados diretamente no banco de dados, via script SQL.

RF-02. Cadastro de Usuário: O formulário de cadastro deve conter campos para nome completo, data de nascimento, sexo, nome materno, CPF, e-mail, telefones (celular e fixo), endereço completo, login, senha e confirmação de senha. Todos os campos, exceto o login e a confirmação de senha, são de preenchimento obrigatório.

- O campo de login deve ter exatamente 6 caracteres alfabéticos.
- A senha deve ser criptografada e ter 8 caracteres alfabéticos.
- O campo CPF deve ter conferência do Dígito Verificador.
- O CEP deve ser validado e, se possível, preencher os demais campos de endereço automaticamente via API.
- O sistema deve redirecionar o usuário para a tela de login após um cadastro bem-sucedido.

RF-02. Login: A tela deve ter campos para login e senha. Após a validação, o sistema deve acionar a tela de segundo fator de autenticação (2FA).

RF-04. Segundo Fator de Autenticação (2FA): Todos os usuários, tanto "master" quanto "comum", terão o segundo fator de autenticação. O sistema deve fazer uma pergunta de segurança aleatória (nome da mãe, data de nascimento

ou CEP) e verificar a resposta com os dados no banco de dados. Após 3 tentativas sem sucesso, o usuário deve ser redirecionado para a tela de login.

RF-05. Gestão de Usuários (Acesso Master):

- Consulta de Usuários: O usuário "master" deve poder visualizar uma lista de todos os usuários.
- Busca: Deve haver um campo de pesquisa onde o usuário "master" pode buscar usuários comuns por uma "substring" (parte do nome).
- Exclusão de Usuários: A tela de consulta deve apresentar um botão de exclusão para cada usuário comum, permitindo que o usuário "master" remova usuários após a confirmação.

RF-06. Alteração de Senha: Somente o usuário comum pode alterar a sua própria senha.

RF-07. Visualização de Logs: O usuário "master" deve poder consultar um log de autenticação, com um filtro de busca por nome ou CPF. O log deve mostrar o dia e a hora, nome do usuário e o 2FA exibido, em ordem cronológica inversa (mais recentes primeiro).

RF-08. Telas de Erro: O sistema deve exibir uma tela de erro quando algo inesperado acontecer, como uma falha na autenticação.

RF-09. Exibição de Informações:

- **Tela principal:** A tela principal deve ter um menu e exibir informações dos produtos, com suas respectivas descrições.
- **Modelo do BD:** A tela do modelo do banco de dados deve ter uma imagem da modelagem utilizada no sistema.

RF-010. Funcionalidades de Acessibilidade

- PDF: Inserir um botão para baixar a lista de usuários em formato PDF.
- Contraste: A aplicação deve prover uma barra com a opção de troca de contraste (fundo escuro/fonte clara e vice-versa).
- Tamanho da Fonte: A aplicação deve permitir o aumento e a diminuição do tamanho das fontes.

Requisitos não funcionais (RNF)

RNF-01. Identidade Visual e Design:

- O site deve usar uma identidade visual clara.
- Após o login, o login do usuário deve ser exibido no canto superior direito de todas as telas.

- O sistema deve exibir um menu e uma opção de logout em todas as telas após a autenticação.
- Mensagens de feedback ao usuário, como em formulários, devem usar modais ou toasts para uma estética elegante, evitando alerts. A identidade visual deve ser consistente em todos os feedbacks.

RNF-02. Tecnologia:

- A linguagem de programação para o Back-End deve ser PHP.
- O banco de dados a ser utilizado é o MySQL.
- O sistema deve ser responsivo e pode usar frameworks de front-end como Materialize, Bootstrap ou outros.

RNF-02. Segurança: A senha do usuário deve ser criptografada.

RNF-04. Arquitetura: O site deve ter um conjunto de páginas interconectadas.

3.Desenvolvimento

Sprint 1: Estrutura e Back-End Essencial

Objetivo: Configurar o ambiente e desenvolver a base da aplicação, incluindo o sistema de autenticação.

Backlog:

- **Configuração do Ambiente:** Instalar e configurar o ambiente de desenvolvimento (XAMPP com PHP e MySQL).
- **Modelagem e Banco de Dados:** Criar o script SQL para as tabelas de usuário, logs, e as tabelas necessárias, seguindo o modelo ER.
- **Lógica de Cadastro e Login:** Implementar a lógica de back-end para o cadastro de usuários. Isso inclui a validação dos campos, como CPF e CEP com API , e a criptografia da senha.
- **Lógica de Login:** Implementar a funcionalidade que valida o login e a senha no banco de dados.
- **Usuário Master:** Inserir o usuário master via script SQL no banco de dados.

Sprint 2: Autenticação e Funções Básicas

Objetivo: Configurar o ambiente e desenvolver a base da aplicação, incluindo o sistema de autenticação.

Backlog:

- **Implementação do 2FA:** Criar a lógica para a tela de Segundo Fator de Autenticação (2FA). A geração da pergunta deve ser aleatória e a resposta deve ser validada com o banco de dados. O sistema deve redirecionar para a tela de login após 3 tentativas sem sucesso.
- **Implementação do 2FA:** Desenvolver a funcionalidade de logout.
- **Lógica de Alteração de Senha:** Criar o sistema que permite ao usuário comum alterar a própria senha.

Sprint 3: Funcionalidades do Perfil Master

Objetivo: Desenvolver todas as funcionalidades exclusivas do perfil master.

Backlog:

- **Controle de Acesso:** Implementar o gerenciamento de sessão de usuário para controlar o acesso de acordo com o perfil.
- **Consulta de Usuários:** Criar a lógica de back-end para a tela de Consulta de Usuário, que exibe a lista de todos os usuários.
- **Busca:** Desenvolver a funcionalidade de pesquisa que permite ao usuário master buscar por uma substring no nome dos usuários comuns.
- **Exclusão de Usuários:** Implementar a lógica de exclusão, incluindo a confirmação para remover o usuário comum.
- **Tela de Logs:** Criar a lógica para a página de logs de autenticação, permitindo a consulta por nome, CPF ou todos, exibindo os dados em ordem mais recente.

Sprint 4: Acessibilidade, Testes e Entrega

Objetivo: Desenvolver todas as funcionalidades exclusivas do perfil master.

Backlog:

- Implementar a funcionalidade para baixar a lista de usuários em PDF
- Desenvolver o mecanismo de troca de contraste e de aumento/diminuição do tamanho das fontes.
- Testes e Validação: Realizar testes de ponta a ponta em todas as funcionalidades, garantindo que o sistema é responsivo e que não há erros.

Documentação e Entrega:

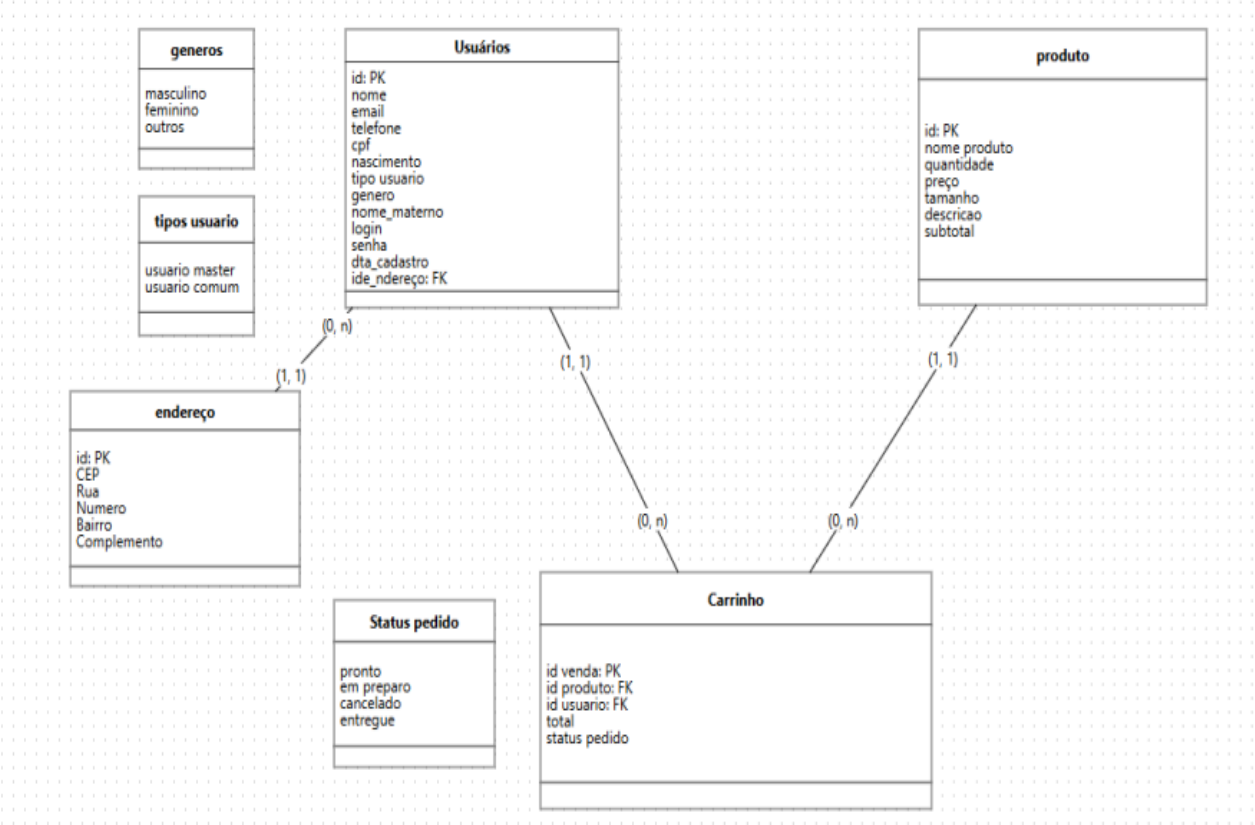
- Finalizar a documentação do projeto, incluindo o modelo ER e o link do GitHub
- Preparar a apresentação do projeto.

4.Teste Realizados

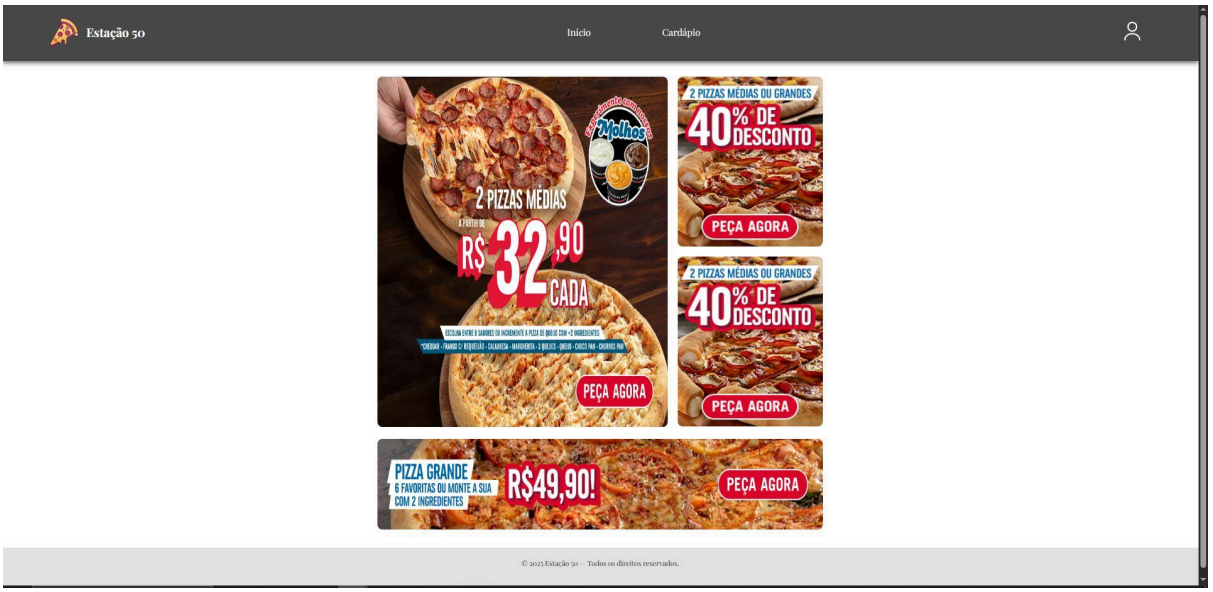
5.Considerações Finais

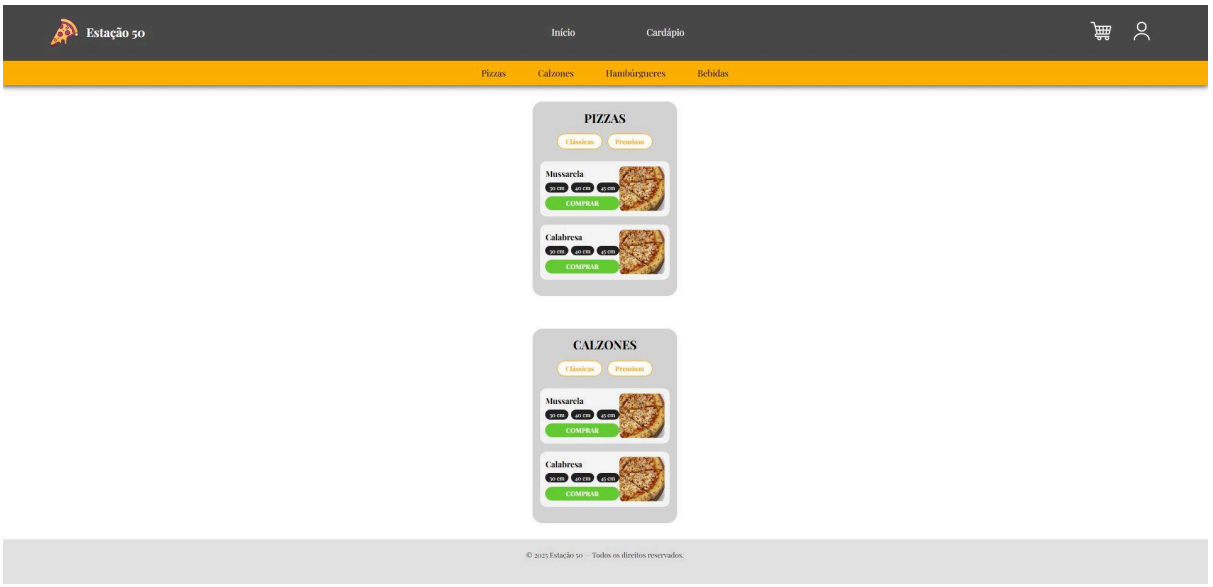
6.Anexos

6.1. Anexo A: Modelo ER



6.2. Anexo B: Prints de telas





6.3. Anexo C: Tabelas

```
CREATE DATABASE delivery;

use delivery;
create TABLE usuarios(
    id_usuario int AUTO_INCREMENT PRIMARY KEY,
    nome varchar(100) unique not null,
    email varchar(100) unique not null,
    telefone varchar(11) not null,
    cpf varchar(12) unique not null,
    nascimento date,
```

```
    tipo_usuario varchar(10),
    sexo varchar (10),
    nome_materno varchar(100),
    id_endereco int,
    login varchar(100) not null,
    senha varchar(100) not null,
    dt_cadastro date
);

use delivery;
create table endereco(
    id_endereco int AUTO_INCREMENT PRIMARY KEY,
    cep varchar(100) not null,
    rua varchar(100) not null,
    numero int not null,
    bairro varchar(100),
    complemento varchar(100)
);

use delivery;
ALTER TABLE usuarios
ADD CONSTRAINT id_endereco
FOREIGN KEY (id_endereco)
REFERENCES endereco(id_endereco);

use delivery;
create table produtos(
    id_produto int AUTO_INCREMENT PRIMARY key,
    nome_produto varchar(100) not null,
    quantidade int,
    preco float,
    tamanho float,
    descricao text,
    subtotal float
);

use delivery;
create table carrinho(
    id_venda int PRIMARY key,
    id_produto int,
    id_usuario int,
    total float,
    status_pedido varchar(100),
```

```
          FOREIGN KEY(id_produto) REFERENCES
produtos(id_produto),
          FOREIGN KEY(id_usuario) REFERENCES
usuarios(id_usuario)
     );
```