

CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS

ALUNOS:

Davi Augusto Marçal Rodrigues

Matheus Augusto de Carvalho Silva

Samuel Matias Almeida Irineu

TRABALHO PRÁTICO: PROJETO E-COMMERCE

Contagem

2025

Sumário

1 INTRODUÇÃO	3
1.1 Objetivos	3
2 DESENVOLVIMENTO	4
3 RESULTADOS	6
4 CONCLUSÃO	14
REFERÊNCIAS BIBLIOGRÁFICAS	16
APÊNDICE A - Repositório no GitHub	17
APÊNDICE B – Vídeo da aplicação em funcionamento	17

1 INTRODUÇÃO

O desenvolvimento de aplicações web tornou-se cada vez mais frequente com a democratização do acesso à internet. Esse crescimento impulsionou a adoção de diferentes paradigmas e padrões de projeto que, embora originalmente aplicados em outros contextos, demonstraram grande eficácia quando adaptados ao ambiente web. O paradigma de Programação Orientada a Objetos (POO) destaca-se, sendo amplamente utilizado na indústria de software por oferecer uma estrutura padronizada para a criação de classes, que facilita a reutilização e organização de códigos, bem como a manutenção e evolução do sistema (HENRIQUE, 2014).

Além disso, o padrão *Model-View-Controller* (MVC) também é amplamente usado, pois oferece vantagens como maior organização do código, separando responsabilidades entre *model*, *view* e *controller*. Isso facilita a manutenção, melhora a reutilização de componentes, permite evoluir partes da aplicação sem afetar outras e torna o desenvolvimento mais ágil e estruturado (HIGOR, 2014).

1.1 Objetivos

O objetivo geral deste trabalho consiste em desenvolver um sistema de cadastro e venda de produtos, no caso uma loja de celulares.

Os objetivos específicos do trabalho que refere-se aos requisitos que o sistema deve cumprir, estão listados abaixo:

- Implementar as funcionalidades principais:
 - Cadastro de usuário;
 - Cadastro de produtos;
 - Cadastro de vendas;
 - Relatório de produtos;
 - Relatório de vendas.
- Mitigar vulnerabilidades, utilizando validação, sanitização e *prepared statements*;
- Realizar tratamento de erros;
- Implementar POO;
- Implementar *Data Access Object* (DAO);
- Utilizar o padrão MVC.

As seções seguintes descrevem detalhadamente o processo de desenvolvimento do projeto, incluindo as estratégias adotadas, as estruturas de dados desenvolvidas e as decisões técnicas tomadas ao longo do trabalho. Além disso, serão apresentados os resultados obtidos e a conclusão.

2 DESENVOLVIMENTO

Nesta seção será apresentado o processo de implementação do sistema e as principais estratégias de desenvolvimento utilizadas na criação do site proposto. Além disso, será discutida a implementação do MVC e de ferramentas que garantiram a integridade e segurança dos dados salvos no banco de dados.

A implementação do padrão MVC serviu de norte para o desenvolvimento. O projeto foi dividido em três pastas principais: model, controller e view. As classes implementadas no model que correspondem às entidades do sistema, foram implementadas com base no diagrama Modelo de Classes implementado em (COSTA, 2025), com pequenas alterações para melhor adaptação ao projeto.

Há relações de associação e composição entre as entidades do sistema. A classe `Produto` possui dois atributos que são objetos, um da classe `Fabricante`, o outro da classe `Categoria`. São instanciados por meio de métodos `set` que recebem os respectivos objetos. Esses objetos recebidos são declarados e instanciados fora da classe, ou seja, não há uma relação de dependência de existência entre eles. Além disso, `Produto` possui um array de objetos da classe `Caracteristica`, e cada característica é instanciada diretamente dentro de `Produto`. Isso significa que cada característica depende de um produto para existir, dessa forma, se um produto for destruído, suas características também serão.

Para cada entidade foi criada uma classe de DAO, que é responsável pela manipulação dos dados referentes à respectiva entidade no banco de dados. As informações antes de serem armazenadas são validadas e sanitizadas. A validação serviu para testar os dados de entrada, podendo apresentar uma resposta rápida ao usuário, em casos de erro. A sanitização serviu para tratar os dados já validados, à fim de garantir a integridade deles, antes de armazená-los. Além disso, foram utilizadas *prepared statements* na comunicação com o banco de dados. O

uso desse recurso serve para utilizar o mesmo comando repetidamente e para proteger o sistema contra SQL Injection (PHP, entre 2005 e 2015).

Para realizar a conexão com o banco de dados foi criado uma classe `Conexao` que possui um atributo da classe `PDO`¹ (representa a conexão entre uma aplicação PHP e um banco de dados). Nessa classe `Conexao` é utilizado o padrão de projeto *Singleton*, pois esse padrão permite uma conexão estável em todo o sistema. O *Singleton* utilizado foi implementado conforme o especificado em (SANTOS, 2025) com adaptações para o contexto proposto.

As páginas da view exibe as informações para o usuário e recebe os dados de entrada do mesmo. Qualquer interação ou ação do usuário ocorre na parte da view. Optou-se por não utilizar o paradigma de POO na implementação das views.

Os códigos na parte de controller fazem a comunicação entre as páginas view e o model. As requisições do usuário, como clicar em um botão por exemplo, são enviadas usando o método GET. Já os dados de cadastro do usuário e cadastro de produtos, são enviados utilizando POST. Quando há uma requisição que envolve comunicação com o banco de dados, o controller se comunica com o model, à fim de obter os dados requeridos pela view. O tratamento básico de erros na comunicação com o banco de dados, utilizando blocos `try/catch`, foram implementadas no controller.

A ferramenta Visual Studio Code (VS Code) foi utilizada para o desenvolvimento do sistema, além disso, foi utilizada também a extensão Live Share do VS Code que permite colaboração em tempo real no código. Por fim, o versionamento de código entre os colaboradores do projetos foi realizado no GitHub, com *commits* direto na *branch* principal, a *main*.

No desenvolvimento do programa foi utilizado Inteligências Artificiais (IA) Generativas como ChatGPT, Claude e Grok para implementar parte dos códigos. As classes DAO, os códigos de marcação que serviram de base para as views, bem como as classes mais complexas de tratamento de erros no sistema, foram parcialmente implementadas pelas IAs, passando por adaptações para o projeto desenvolvido.

¹ https://www.php.net/manual/pt_BR/class.pdo.php

3 RESULTADOS

Como resultado foi possível a criação de um website de vendas de celulares, com as principais funcionalidades, descritas na seção 1.1, utilizando POO, MVC e ferramentas que garantam a segurança dos dados.

Na camada model foi realizada a implementação das classes responsáveis pela representação das entidades presentes no banco de dados. Essas classes refletem a lógica de negócio e servem como base para a criação dos DAOs. Dentro dos DAOs foram desenvolvidas as funções que acessam, consultam e manipulam as informações do banco.

Figura 1 – Tela de cadastro

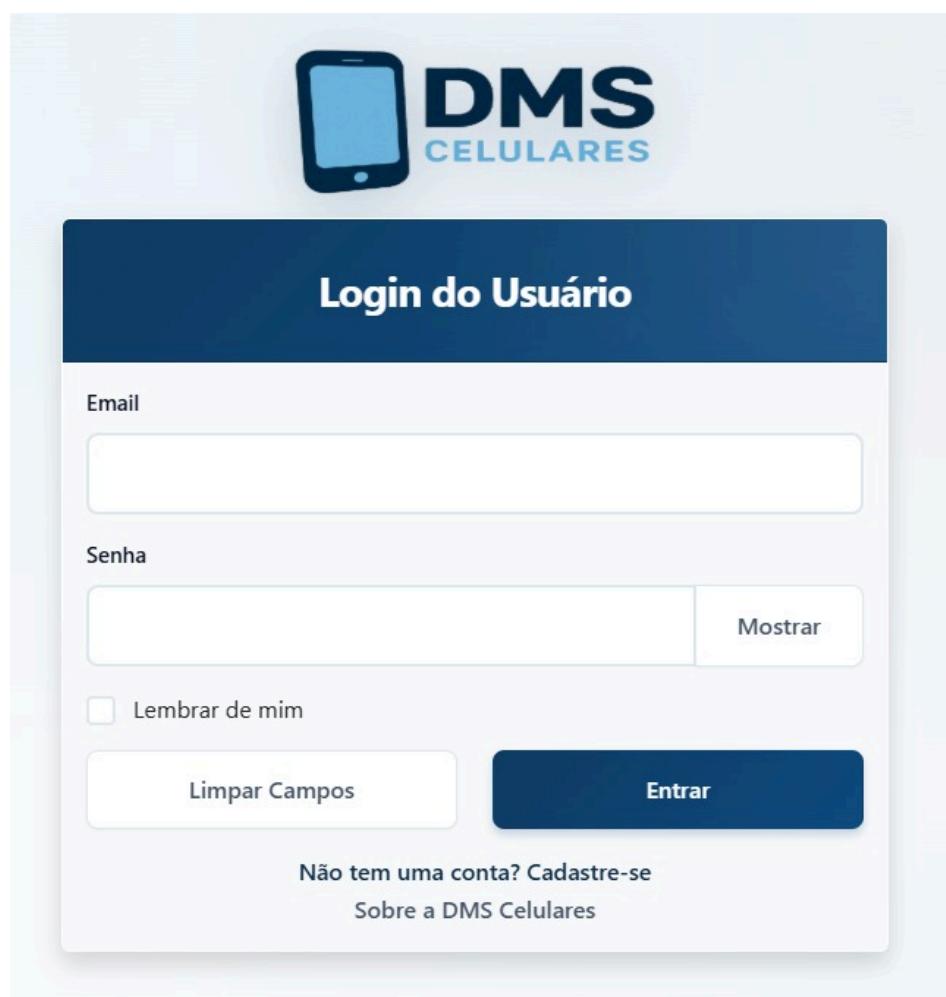


Fonte: Autoria própria.

Legenda: representação da tela de cadastro da loja desenvolvida.

Na camada view foram elaboradas três áreas principais. A primeira corresponde às interfaces de login e cadastro do usuário. Por meio dessas telas iniciais o usuário pode acessar as demais áreas do sistema, que são a área do cliente e a área do administrador, sendo esse acesso condicionado às permissões configuradas para cada perfil. A Figura 1 e a Figura 2 apresentam a tela de cadastro e a tela de login respectivamente.

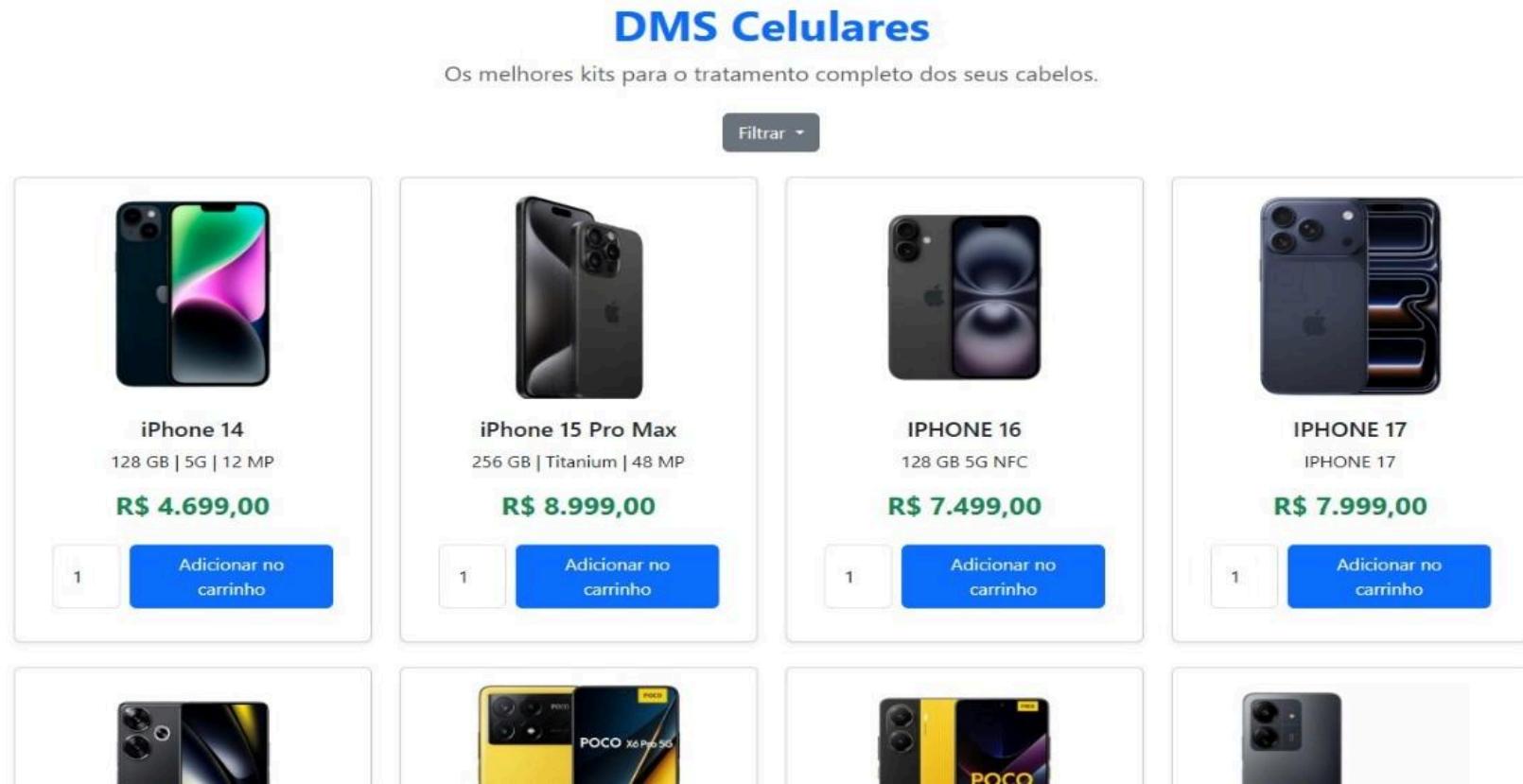
Figura 2 – Tela de login



Fonte: Autoria própria.

Legenda: representação da tela de login da loja desenvolvida.

Figura 3 – Tela de Produtos



Fonte: Autoria própria.

Legenda: representação da tela inicial de produtos da área do cliente na loja desenvolvida.

Figura 4 – Tela de Carrinho de Compras

 Carrinho de Compras					
Produto	Preço	Quantidade		Subtotal	Ações
 IPHONE 16	R\$ 7.499,00	<button>-</button> 2 <button>+</button>		R\$ 14.998,00	<button>Excluir</button>
 POCO X6 Pro	R\$ 2.199,00	<button>-</button> 1 <button>+</button>		R\$ 2.199,00	<button>Excluir</button>
 S25 ULTRA	R\$ 5.890,00	<button>-</button> 1 <button>+</button>		R\$ 5.890,00	<button>Excluir</button>

[Atualizar Carrinho](#)

Total da Compra: R\$ 15.588,00

[Limpar Carrinho](#) [Confirmar Compra](#)

Fonte: Autoria própria.

Legenda: representação da tela de carrinho de compras da área do cliente na loja desenvolvida.

Figura 5 – Tela de Minhas Compras

ID da Compra	Data	Valor Total
4	2025-12-10	R\$ 15.588,00
3	2025-12-07	R\$ 23.997,00
2	2025-12-07	R\$ 2.399,00
1	2025-12-07	R\$ 20.796,00

Fonte: Autoria própria.

Legenda: representação da tela de minhas compras da área do cliente na loja desenvolvida.

Figura 6 – Aba de cadastro de produtos

Novo Produto

Nome

Descrição

URL da Imagem

Estoque <input type="text" value="0"/>	Preço de Custo <input type="text" value="0"/>
Preço de Venda <input type="text" value="0"/>	Categoria <input type="text" value="Selecione"/>
Fabricante <input type="text" value="Selecione"/>	
Características (nome:valor por linha) <input type="text" value="Ex.: \nCor: Azul\nMemória: 128GB"/>	

Cadastrar produto

Fonte: Autoria própria.

Legenda: representação da aba de cadastro de produtos na área do administrador na loja desenvolvida.

Figura 7 – Aba de produtos cadastrados

Produtos cadastrados						Total: 16
Nome	Categoria	Fabricante	Preço	Estoque	Ações	
iPhone 14	4K	APPLE	R\$ 4.699,00	30	Editar	<button>Excluir</button>
iPhone 15 Pro Max	4K	APPLE	R\$ 8.999,00	20	Editar	<button>Excluir</button>
IPHONE 16	4K	APPLE	R\$ 7.499,00	10	Editar	<button>Excluir</button>
IPHONE 17	4K	APPLE	R\$ 7.999,00	28	Editar	<button>Excluir</button>
POCO F6	PRO GAMING	XIAOMI	R\$ 2.599,00	18	Editar	<button>Excluir</button>
POCO X6 Pro	PRO GAMING	XIAOMI	R\$ 2.199,00	22	Editar	<button>Excluir</button>
POCO X7 PRO	PRO GAMING	XIAOMI	R\$ 2.399,00	10	Editar	<button>Excluir</button>
Redmi 13C	ULTRA	XIAOMI	R\$ 899,00	40	Editar	<button>Excluir</button>
Redmi Note 13 Pro+	ULTRA	XIAOMI	R\$ 2.899,00	14	Editar	<button>Excluir</button>
REDMI NOTE 14 ULTRA	ULTRA	XIAOMI	R\$ 2.399,99	8	Editar	<button>Excluir</button>
S25 ULTRA	4K	SAMSUNG	R\$ 5.890,00	15	Editar	<button>Excluir</button>
SAMSUNG A26	AMOLED	SAMSUNG	R\$ 1.490,50	13	Editar	<button>Excluir</button>
Samsung Galaxy A55	AMOLED	SAMSUNG	R\$ 2.299,00	25	Editar	<button>Excluir</button>
Samsung Galaxy S23 FE	AMOLED	SAMSUNG	R\$ 3.799,00	17	Editar	<button>Excluir</button>

Fonte: Autoria própria.

Legenda: representação da aba de produtos cadastrados na área do administrador na loja desenvolvida.

Figura 8 – Tela de Vendas Realizadas

The screenshot shows a user interface for managing sales. At the top, there is a logo consisting of three vertical bars in green, blue, and red, followed by the text "Vendas Realizadas" in a bold, blue, sans-serif font. Below the title are two input fields: "Data Inicial" and "Data Final", each with a date picker icon (a small calendar icon) and a placeholder "mm/dd/yyyy". To the right of these fields are two buttons: "Filtrar" (Filter) in blue and "Limpar Filtro" (Clear Filter) in grey. A horizontal line separates this header from the main content area. The main content is a table with four columns: "ID da Compra" (Purchase ID), "Data" (Date), "Valor Total" (Total Value), and "ID do Cliente" (Client ID). The table contains four rows of data:

ID da Compra	Data	Valor Total	ID do Cliente
4	10/12/2025 00:00	R\$ 15.588,00	4
3	07/12/2025 00:00	R\$ 23.997,00	4
2	07/12/2025 00:00	R\$ 2.399,00	4
1	07/12/2025 00:00	R\$ 20.796,00	4

Fonte: Autoria própria.

Legenda: representação da tela de vendas realizadas da área do administrador na loja desenvolvida

Na área destinada ao cliente foram implementadas três telas principais. A primeira é a tela de listagem dos dispositivos disponíveis para compra, como é possível visualizar na Figura 3, onde são exibidos os celulares cadastrados e onde o usuário pode aplicar filtros, visualizar detalhes e adicionar produtos ao carrinho. A segunda tela corresponde ao carrinho de compras, representado na Figura 4, no qual o usuário pode aumentar a quantidade de itens, remover produtos e finalizar a aquisição. A terceira tela apresenta o histórico de compras realizadas anteriormente, demonstrado pela Figura 5, permitindo ao usuário consultar informações de pedidos passados e acompanhar o que já foi adquirido.

Também foi desenvolvida a área de administração dentro da camada view para o administrador. Essa área possui duas telas principais. A primeira é responsável pela gestão dos produtos disponíveis para venda, permitindo ao administrador adicionar novos dispositivos, alterar informações de produtos já cadastrados ou remover itens do catálogo assim como é possível visualizar com as Figuras 6 e 7. A segunda tela apresenta o registro de vendas realizadas, exibindo as transações mais recentes e oferecendo uma visão geral do desempenho comercial da plataforma, tal qual a Figura 8.

No controller foram desenvolvidos diversos componentes, entre eles uma classe destinada ao tratamento de erros, códigos responsáveis pelo controle do sistema e classes que gerenciam tokens CSRF. A classe de tratamento de erros armazena e trata as falhas que podem ocorrer durante a execução do programa, avaliando cada ação realizada e determinando se foi concluída com sucesso ou se houve algum problema a ser tratado. Os códigos responsáveis pela coordenação geral do sistema, garantem a comunicação adequada entre a view e o model e assegurando que as funcionalidades sejam executadas. Por fim, as classes responsáveis pelo CSRF atuam no gerenciamento dos tokens de acesso utilizados por cada usuário, protegendo o sistema contra ataques e acessos indevidos.

4 CONCLUSÃO

O projeto atingiu de forma satisfatória os objetivos propostos. O objetivo geral foi plenamente alcançado por meio do desenvolvimento eficaz de um site de venda de celulares, implementado com as funcionalidades essenciais esperadas em uma plataforma de comércio eletrônico voltada para esse tipo de produto.

No que se refere aos objetivos específicos, foram devidamente implementados. As funcionalidades principais foram executadas conforme o planejado. Do ponto de vista do usuário final, o sistema permite a criação de contas, autenticação por meio de login, compra de produtos, adição de itens ao carrinho e visualização de compras anteriores. Já na perspectiva do administrador, o sistema possibilita o cadastro de novos produtos, a atualização de itens já existentes e o gerenciamento de produtos vendidos, garantindo controle adequado sobre o catálogo e o fluxo comercial.

Além disso, outros objetivos específicos também foram contemplados. As vulnerabilidades identificadas durante o desenvolvimento foram mitigadas, e o sistema utiliza mecanismos que reforçam a integridade dos dados e a segurança do usuário. O tratamento de erros foi padronizado, proporcionando maior clareza e consistência na identificação e resolução de falhas, além de facilitar a manutenção do código.

O paradigma de POO foi utilizado, especialmente na camada model. O padrão de projeto DAO foi corretamente adotado e as classes que representam as entidades do sistema foram implementadas e se relacionam entre si. Da mesma forma, a arquitetura MVC foi aplicada de maneira adequada, quanto às responsabilidades de cada camada, que desempenham suas funções específicas dentro do sistema.

O projeto não apenas cumpriu os requisitos estabelecidos, mas também demonstrou boa aderência às práticas recomendadas de engenharia de software, oferecendo uma solução funcional, organizada e tecnicamente consistente.

REFERÊNCIAS BIBLIOGRÁFICAS

- COSTA, Elizabeth Duane Santos da. **2025_WEB2_e-commerce.** [pdf]. Contagem, MG, 2025. Disponível em: https://docs.google.com/presentation/d/1qFgmhxlelrr75PAbkHxjN3u8SQojP59XmF_P2tL4GUM/edit?slide=id.p#slide=id.p. Acesso em: 9 dez. 2025.
- HENRIQUE. **Os 4 pilares da Programação Orientada a Objetos.** [S. l.], 2014. Disponível em: <https://www.devmmedia.com.br/os-4-pilares-da-programacao-orientada-a-objetos/9264>. Acesso em: 9 dez. 2025.
- HIGOR. **Introdução ao Padrão MVC.** [S. l.], 2013. Disponível em: <https://www.devmmedia.com.br/introducao-ao-padrao-mvc/29308>. Acesso em: 9 dez. 2025.
- PHP. **A classe PDO.** [S. l.], [entre 2005 e 2015]. Disponível em: https://www.php.net/manual/pt_BR/class pdo.php. Acesso em: 10 dez. 2025.
- PHP. **Instruções Preparadas.** [S. l.], [entre 2005 e 2015]. Disponível em: https://www.php.net/manual/pt_BR/mysql.quickstart.prepared-statements.php. Acesso em: 9 dez. 2025.
- SANTOS, Alisson Rodrigo dos. **LP2-26-Padrões de projeto-Singleton.** [pdf]. Contagem, MG, 2025. Disponível em: https://ava.cefetmg.br/pluginfile.php/84903/mod_resource/content/1/LP2-26-Padr%C3%B5es%20de%20projeto-Singleton.pdf UM/edit?slide=id.p#slide=id.p. Acesso em: 10 dez. 2025.

APÊNDICE A - Repositório no GitHub

GRUPO-8-20205. **TrabalhoPraticoWEB**. Contagem, MG, 2025. 1 repositório Git. Disponível em: <https://github.com/Grupo-8-2025/TrabalhoPraticoWEB/tree/main>. Acesso em: 10 dez. 2025.

APÊNDICE B – Vídeo da aplicação em funcionamento

DAVI. **Video TrabalhoWEB**. Contagem, MG, 2025. 1 vídeo (1:30 min). Disponível em: <https://www.youtube.com/watch?v=-iGGfeWK93c>. Acesso em: 10 dez. 2025.