PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS

CineAA

Goiânia

2025

Allan Vitor Correia

CineAA

Dissertação apresenta o trabalho de conclusão de curso de ciência da computação - Pontifícia Universidade Católica de Goiás – PUC Goiás, como parte das exigências para a graduação do mesmo. Orientador: EUGENIO JULIO M C CARVALHO

Goiânia

2025

Sumário

Lista de figuras

1 Introdução

O avanço das tecnologias de informação e comunicação transformou a forma como as pessoas consomem conteúdo audiovisual. Com o crescimento das plataformas digitais, como *Netflix*, *Prime Video* e *Disney+*, o consumo de filmes e séries passou a ocupar um papel central na rotina das pessoas. Nesse contexto, observa-se também o aumento do interesse por sistemas que permitem organizar, avaliar e compartilhar experiências cinematográficas de maneira personalizada (GOMES; LIMA, 2022).

De acordo com a pesquisa TIC Domicílios (CETIC, 2023), cerca de **84% da população brasileira com acesso à internet utiliza plataformas digitais para entretenimento**, especialmente serviços de streaming e redes sociais voltadas à recomendação de conteúdo. Essa tendência revela um cenário em que o público busca não apenas assistir, mas também **interagir e compartilhar opiniões sobre produções audiovisuais**, evidenciando uma demanda por soluções web que unam organização, avaliação e interação social.

Nesse contexto, o projeto **Cine Rank** propõe o desenvolvimento de um **sistema web voltado à catalogação, avaliação e recomendação de filmes**, permitindo que usuários criem listas personalizadas, avaliem títulos já assistidos, mantenham listas de filmes que desejam ver e compartilhem recomendações com amigos dentro da própria plataforma. O sistema busca aliar **usabilidade, interatividade e design responsivo**, utilizando tecnologias modernas como HTML5, CSS3, JavaScript, PHP e PostgreSQL, integradas por meio da arquitetura **MVC (Model-View-Controller)**.

Segundo Pressman e Maxim (2020), o desenvolvimento de sistemas de software modernos deve priorizar a experiência do usuário e a escalabilidade da aplicação, adotando práticas que facilitem manutenção e evolução contínua. Assim, o **Cine Rank** não se limita a um catálogo estático de filmes, mas propõe um ambiente dinâmico de interação social e recomendação, aproximando-se conceitualmente de redes como *Letterboxd* e *MyAnimeList*, porém com foco na simplicidade e acessibilidade da web.

A escolha desse tema fundamenta-se na relevância do cinema como manifestação cultural e na crescente necessidade de plataformas que organizem a experiência cinematográfica dos usuários, oferecendo um espaço colaborativo de troca de opiniões e descobertas. Além disso, o projeto busca aplicar na prática os conhecimentos adquiridos ao longo do curso de Engenharia de Computação, com ênfase em **engenharia de software, banco de dados, arquitetura web e programação full-stack**.

1.1 OBJETIVO GERAL

Desenvolver e documentar um **sistema web denominado Cine Rank**, que permita aos usuários **catalogar, avaliar, comentar e recomendar filmes**, com funcionalidades de interação social, listas personalizadas e interface responsiva,

utilizando tecnologias web modernas e banco de dados relacional PostgreSQL.

1.2 OBJETIVOS ESPECÍFICOS

- Realizar o levantamento e documentação dos **requisitos funcionais e não funcionais** do sistema;

- Projetar o **modelo conceitual e lógico** do banco de dados relacional;

- Implementar o **backend** utilizando **PHP** e **PostgreSQL**, aplicando o padrão **MVC**;

- Desenvolver o **frontend** com **HTML5, CSS3 e JavaScript**, priorizando a responsividade e a acessibilidade;

- Implementar funcionalidades para **criação e gerenciamento de listas de filmes**, **avaliação com notas**, e **recomendações entre usuários**;

- Integrar o sistema a um **módulo de autenticação** seguro para cadastro e login de usuários;

- Realizar **testes de funcionalidade e usabilidade**, avaliando o desempenho e a consistência da aplicação;

- Elaborar a documentação final conforme o modelo da **PUC Goiás**, descrevendo as etapas de desenvolvimento e os resultados obtidos.

1.3 JUSTIFICATIVA

O crescimento do consumo de conteúdo audiovisual em plataformas digitais transformou o modo como os usuários interagem com o cinema. Com a grande quantidade de títulos disponíveis em serviços de streaming e mídias sociais, tornou-se cada vez mais difícil para os espectadores **organizar, avaliar e compartilhar suas experiências cinematográficas** de forma estruturada. Nesse contexto, surge a necessidade de ferramentas que permitam **gerenciar listas de filmes assistidos ou desejados**, além de **avaliar e trocar recomendações** com outros usuários (GOMES; LIMA, 2022).

O projeto **Cine Rank** nasce dessa necessidade, aliada ao interesse pessoal do autor pelo cinema e pela área de desenvolvimento web. A proposta é criar um ambiente digital que una **organização, interação e avaliação**, possibilitando ao usuário registrar suas experiências e descobrir novas produções a partir das recomendações de outros membros.

Além disso, o projeto tem caráter **acadêmico e tecnológico**, servindo como aplicação prática dos conhecimentos adquiridos no curso de **Engenharia de Computação**, com ênfase em **engenharia de software, banco de dados e desenvolvimento full-stack**. Assim, o *Cine Rank* se apresenta não apenas como uma plataforma útil ao público interessado em cinema, mas também como um **exercício de integração entre teoria e prática**, demonstrando a capacidade de planejar, projetar e implementar um sistema web completo e funcional.

1.4 STAKEHOLDERS

Os stakeholders do projeto **Cine Rank** são os indivíduos e grupos que têm interesse direto ou indireto no desenvolvimento, uso e resultados do sistema. São eles:

- **Allan Vitor Correia** — desenvolvedor e responsável técnico pelo projeto, encarregado da análise, modelagem, implementação e documentação do sistema;

- **Usuários finais** — pessoas interessadas em catalogar filmes, avaliar produções, criar listas personalizadas e interagir com outros usuários na plataforma;

- **Eugênio (Orientador)**— responsáveis por acompanhar, avaliar e validar o projeto no contexto acadêmico, garantindo a aplicação correta dos conceitos de engenharia de software;

- **Pai do autor** — idealizador inicial da proposta, cuja sugestão originou a ideia central do sistema e serviu de inspiração para o desenvolvimento do *Cine Rank*.

2 DESCRIÇÃO GLOBAL

Este capítulo apresenta uma visão geral do sistema *Cine Rank*, abrangendo suas principais funcionalidades, arquitetura, interfaces e o ambiente de operação. O projeto foi concebido para oferecer uma solução prática e acessível para usuários que desejam **organizar, avaliar e recomendar filmes**, utilizando recursos de interação social e interface web responsiva.

O *Cine Rank* foi desenvolvido com base nas boas práticas de engenharia de software, buscando atender aos princípios de **usabilidade, desempenho, segurança e manutenibilidade**, conforme recomendam Pressman e Maxim (2020). O sistema foi estruturado em **arquitetura cliente-servidor**, adotando o padrão **MVC (Model-View-Controller)** para promover a separação entre as camadas de apresentação, lógica de negócio e acesso a dados.

As tecnologias utilizadas incluem **HTML5**, responsável pela estrutura semântica das páginas; **CSS3**, empregado na estilização e responsividade; **JavaScript**, para controle dinâmico da interface e comunicação assíncrona; **PHP**, como linguagem principal do back-end; e **PostgreSQL**, como sistema de gerenciamento de banco de dados relacional. Essa combinação forma uma base sólida e moderna para o desenvolvimento de aplicações web completas, conforme orientações do *World Wide Web Consortium (W3C)* (2018).

O *Cine Rank* também foi projetado para ser compatível com múltiplos dispositivos — computadores, tablets e smartphones —, permitindo que os usuários acessem suas listas e avaliações em qualquer lugar. Essa característica reforça a importância da **responsividade** e da **acessibilidade digital**, aspectos considerados essenciais no desenvolvimento web contemporâneo (W3C, 2018).

O sistema apresenta uma proposta diferenciada por unir o caráter informativo de um catálogo cinematográfico ao aspecto interativo de uma rede social, oferecendo uma experiência mais rica e colaborativa. Cada usuário pode criar seu perfil, cadastrar filmes, inserir notas e comentários, além de visualizar as avaliações de outros usuários, favorecendo o **engajamento e a troca de recomendações** dentro da comunidade.

2.1 ASPECTO GERAL DO PRODUTO

O *Cine Rank* é um **sistema web de catalogação e recomendação de filmes**, cujo propósito é oferecer aos usuários um espaço para **gerenciar suas experiências cinematográficas** e **compartilhar opiniões com outras pessoas**. O produto foi desenvolvido para operar integralmente via navegador, dispensando instalação de software adicional.

O sistema é composto por uma série de módulos interconectados que contemplam as seguintes funcionalidades principais:

- **Cadastro e autenticação de usuários:** permite a criação de contas pessoais, login seguro e gerenciamento de perfis.

- **Catálogo de filmes:** exibe títulos cadastrados, com informações como título, gênero, sinopse, ano e nota média.

- **Listas personalizadas:** possibilita ao usuário criar listas de filmes que já assistiu, pretende assistir ou deseja recomendar.

- **Sistema de avaliações:** cada usuário pode atribuir notas e comentários a filmes, contribuindo para a nota média geral.

- **Interação social:** permite visualizar recomendações e listas públicas de amigos cadastrados no sistema.

- **Pesquisa e filtragem:** o usuário pode buscar filmes por título, gênero ou nota, facilitando a navegação no catálogo.

- **Interface responsiva:** o layout se adapta automaticamente a diferentes tamanhos de tela e dispositivos.

O sistema foi implementado utilizando **HTML5, CSS3 e JavaScript** no front-end e **PHP com PostgreSQL** no back-end. Essa combinação garante uma

aplicação **leve, escalável e segura**, conforme apontam W3C (2018) e Pressman e Maxim (2020).

Além disso, a escolha por tecnologias de código aberto contribui para a **acessibilidade e manutenibilidade** do projeto, permitindo futuras expansões, como a integração com APIs externas (por exemplo, *The Movie Database – TMDb*) para o carregamento automático de informações sobre filmes.

2.1.1 INTERFACES DO SISTEMA

Esta seção descreve as interfaces do sistema **CineAA**, abrangendo os aspectos visuais, estruturais, de software, comunicação e banco de dados. O objetivo é apresentar como o sistema interage com o usuário, com outros módulos internos e com o ambiente de execução, garantindo uma visão completa das conexões funcionais que sustentam a aplicação.

2.1.1.1 Interface Gráfica

A interface gráfica do **CineAA** foi projetada para proporcionar uma navegação intuitiva e agradável, com foco na experiência do usuário e na responsividade. O layout é construído em **HTML5**, estilizado com **CSS3** e dinamizado por **JavaScript**, possibilitando interações assíncronas e fluídas. As principais telas do sistema são descritas a seguir:

- **Tela de Login e Cadastro:**
  Exibe os campos de entrada para *e-mail* e *senha*, além de botões para acessar a conta ou realizar o cadastro de um novo usuário. Inclui mensagens de erro e validações básicas no front-end. O formulário de registro contém campos para nome, e-mail, CPF, telefone e senha.

- **Tela Principal (Catálogo de Filmes):**
  Apresenta o catálogo completo de filmes cadastrados, exibindo pôster, título, gênero, diretor, ano de lançamento e média de avaliação. O usuário pode navegar, filtrar por gênero e acessar a página de detalhes de cada filme.

- **Tela de Detalhes do Filme:**
  Exibe informações detalhadas do filme selecionado, incluindo sinopse,

elenco principal, avaliação média e botões para adicionar o filme a listas personalizadas ou ao catálogo pessoal (assistidos, para assistir, recomendados).

- **Tela de Perfil do Usuário:**
  Apresenta as informações do usuário autenticado (nome, foto de perfil e biografia), bem como suas listas de filmes criadas, avaliações recentes e recomendações recebidas. Permite edição de perfil e alteração de senha.

- **Tela de Listas Personalizadas:**
  Permite ao usuário criar, editar ou excluir listas próprias, definindo nome, descrição e visibilidade (pública ou privada). As listas exibem os filmes associados, podendo ser atualizadas dinamicamente.

- **Tela de Recomendações:**
  Exibe recomendações recebidas de outros usuários. Cada recomendação contém o nome do remetente, o título do filme e uma mensagem opcional. O destinatário pode aceitar ou recusar a sugestão.

- **Tela de Amigos:**
  Apresenta a lista de amigos adicionados, convites pendentes e sugestões de novas amizades. Cada item exibe o nome do usuário e opções de aceitar, recusar ou remover a amizade.

- **Tela de Notificações:**
  Centraliza alertas de novas recomendações, solicitações de amizade e atualizações de listas. O sistema indica o status de leitura e o tipo de notificação, permitindo o gerenciamento individual.

- **Tela de Recuperação de Senha:**
  Oferece ao usuário a possibilidade de redefinir sua senha por meio do e-mail cadastrado, validando as informações e atualizando o registro correspondente no banco de dados.

Todas as interfaces seguem um design **responsivo**, adaptando-se automaticamente a diferentes tamanhos de tela (desktop, tablet e dispositivos móveis). O conjunto visual utiliza cores neutras e ícones descritivos, priorizando clareza e consistência entre as páginas.

2.1.1.2 Interface de Software

O **CineAA** adota uma **arquitetura MVC (Model–View–Controller)**, separando de forma clara as responsabilidades de cada camada do sistema:

- **Model:** implementado em **PHP**, responsável pela manipulação de dados e interação com o banco de dados PostgreSQL. Contém as classes e funções que gerenciam as entidades *User*, *Movie*, *List*, *Friendship* e *Recommendation*.

- **View:** composta por páginas **HTML** e arquivos **CSS**, responsáveis pela interface visual e exibição das informações processadas. Os dados são integrados de forma dinâmica através de **JavaScript** e **requisições AJAX**.

- **Controller:** conjunto de scripts **PHP** intermediários entre o front-end e o banco de dados. Controla as requisições HTTP, valida dados, executa consultas SQL e retorna respostas em formato JSON ou HTML, dependendo da operação.

A comunicação entre o front-end e o back-end é feita por **requisições assíncronas (AJAX)**, utilizando o protocolo **HTTP/HTTPS**, o que permite atualização parcial de conteúdo sem recarregar toda a página.

O sistema também possui **scripts auxiliares** para autenticação, controle de sessões, upload de imagens e manipulação de dados de filmes, garantindo segurança e consistência nas transações.

2.1.1.3 Interface de Hardware e Comunicação

O sistema **CineAA** foi projetado para ser executado em ambiente **local (localhost)** utilizando o pacote **XAMPP**, que integra o **Apache Server**, **PHP** e o **PostgreSQL** como servidor de banco de dados.

O sistema requer hardware mínimo para operação:

- Processador: Dual Core 2.0 GHz ou superior;
- Memória RAM: 4 GB (recomendado 8 GB);
- Armazenamento: 500 MB livres para arquivos e banco de dados;
- Navegador compatível: Google Chrome, Mozilla Firefox ou Microsoft Edge (versões atuais).

A comunicação entre o servidor e o cliente ocorre por meio de protocolo **HTTP** via porta **80** (ou **443** quando em HTTPS). O sistema é compatível com redes locais (LAN) e pode ser hospedado em servidores web convencionais sem alterações estruturais.

Em termos de segurança, o sistema implementa autenticação via sessão PHP e criptografia de senhas utilizando o algoritmo **bcrypt**, evitando acesso não autorizado aos dados dos usuários.

2.1.1.4 Interface de Banco de Dados

O banco de dados **CineAA** foi implementado em **PostgreSQL**, e sua modelagem segue o paradigma **relacional**, utilizando chaves primárias e estrangeiras para manter a integridade referencial entre as tabelas. A estrutura é composta pelas seguintes entidades principais:

- **users:** armazena as informações pessoais e credenciais dos usuários (nome, e-mail, CPF, telefone, senha criptografada, imagem de perfil e biografia).

- **movies:** contém os dados dos filmes cadastrados, incluindo título, gênero, diretor, elenco principal, sinopse, ano de lançamento, pôster e nota média.

- **personal_catalog:** associa usuários a filmes, registrando status (“assistido”, “para assistir”, “assistindo”, “recomendado”), notas e observações pessoais.

- **custom_lists:** permite que o usuário crie listas personalizadas de filmes, públicas ou privadas, com nome e descrição definidos.

- **list_movies:** tabela de ligação entre listas e filmes, garantindo a relação muitos-para-muitos.

- **friendships:** gerencia as relações de amizade entre usuários, armazenando status da solicitação (pendente, aceita, rejeitada).

- **recommendations:** registra recomendações enviadas entre usuários, associando remetente, destinatário e filme, com status e mensagens personalizadas.

- **notifications:** centraliza alertas de eventos no sistema (novas amizades, recomendações, atualizações), controlando o status de leitura.

O relacionamento entre as entidades é regido por **chaves estrangeiras** e **restrições de integridade**, conforme o modelo apresentado. Essa estrutura garante a coerência dos dados e o correto funcionamento das funcionalidades interativas do sistema.

3 REQUISITOS ESPECÍFICOS

3.1 História de Usuários – Necessidades

As histórias de usuário descrevem as principais necessidades e interações esperadas pelos usuários do sistema **CineAA**, sob a perspectiva funcional. Elas foram elaboradas com base no levantamento de requisitos e nas funcionalidades do sistema.

Id	Descrição da História de Usuário	Fonte (Stakeholder)
HU001	Como usuário, quero criar uma conta informando meus dados pessoais para poder acessar o sistema.	Usuário final
HU002	Como usuário, quero realizar login com e-mail e senha para acessar meu perfil e minhas listas.	Usuário final
HU003	Como usuário autenticado, quero editar meus dados pessoais e atualizar minha foto de perfil.	Usuário final
HU004	Como usuário, quero visualizar um catálogo de filmes com informações detalhadas.	Usuário final
HU005	Como usuário, quero avaliar filmes com notas	Usuário final

HU006	Como usuário, quero adicionar filmes às minhas listas personalizadas (assistidos, para assistir, recomendados).	Usuário final
HU007	Como usuário, quero criar listas personalizadas com nome e descrição próprios.	Usuário final
HU008	Como usuário, quero enviar recomendações de filmes a amigos.	Usuário final
HU009	Como usuário, quero aceitar ou recusar solicitações de amizade.	Usuário final
HU010	Como usuário, quero visualizar e gerenciar minhas notificações.	Usuário final
HU011	Como usuário, quero redefinir minha senha caso esqueça.	Usuário final
HU012	Como administrador (dev), quero ter acesso às informações do banco de dados para fins de manutenção e teste.	Desenvolvedor

3.2 Requisitos Funcionais (RF)

A seguir, estão listados os **requisitos funcionais** do sistema, organizados conforme o padrão da PUC Goiás. Cada requisito possui um identificador único, prioridade e descrição.

Identificador	Nome	Prioridade	Autor	Descrição
RF001	Cadastro de Usuário	Essencial	Allan Vitor Correia	O sistema deve permitir o cadastro de novos usuários informando nome, e-mail, CPF, telefone e senha.
RF002	Autenticação de Usuário	Essencial	Allan Vitor Correia	O sistema deve permitir que o usuário realize login por meio de e-mail e senha.
RF003	Edição de Perfil	Desejável	Allan Vitor Correia	O sistema deve permitir que o usuário edite suas informações pessoais e foto de perfil.
RF004	Catálogo de Filmes	Essencial	Allan Vitor Correia	O sistema deve exibir uma lista de filmes cadastrados, contendo título, gênero, ano, diretor e nota média.

RF005	Avaliação de Filmes	Essencial	Allan Vitor Correia	O sistema deve permitir que o usuário atribua notas e comentários aos filmes.
RF006	Criação de Listas Personalizadas	Essencial	Allan Vitor Correia	O sistema deve permitir que o usuário crie listas personalizadas de filmes, definindo nome, descrição e visibilidade.
RF007	Adição de Filmes a Listas	Essencial	Allan Vitor Correia	O sistema deve permitir que o usuário adicione filmes às suas listas personalizadas.
RF008	Gerenciamento de Catálogo Pessoal	Essencial	Allan Vitor Correia	O sistema deve permitir que o usuário classifique filmes como assistidos, para assistir, assistindo ou recomendados.
RF009	Sistema de Recomendações	Essencial	Allan Vitor Correia	O sistema deve permitir que o usuário recomende filmes para outros usuários registrados.
RF010	Sistema de Amizades	Desejável	Allan Vitor Correia	O sistema deve permitir o envio, aceitação e rejeição de solicitações de amizade.

RF011	Notificações	Desejável	Allan Vitor Correia	O sistema deve notificar o usuário sobre eventos relevantes, como novas amizades e recomendações.
RF012	Recuperação de Senha	Desejável	Allan Vitor Correia	O sistema deve oferecer funcionalidade de redefinição de senha via e-mail.
RF013	Busca de Filmes	Essencial	Allan Vitor Correia	O sistema deve permitir a busca de filmes por título, gênero ou avaliação.
RF014	Remoção de Itens	Desejável	Allan Vitor Correia	O sistema deve permitir que o usuário remova filmes de suas listas ou catálogo pessoal.
RF015	Logout Seguro	Essencial	Allan Vitor Correia	O sistema deve encerrar a sessão do usuário de forma segura, removendo dados temporários.

3.3 REQUISITOS DE QUALIDADE OU NÃO FUNCIONAIS (RQNF)

Os requisitos não funcionais especificam as características de qualidade que o sistema deve apresentar, assegurando desempenho, segurança, confiabilidade e facilidade de uso.

Identificador	Nome	Prioridade	Autor	Descrição
RQNF001	Usabilidade	Essencial	Allan Vitor Correia	O sistema deve possuir uma interface simples, intuitiva e responsiva, facilitando o uso por diferentes perfis de usuários.
RQNF002	Desempenho	Essencial	Allan Vitor Correia	As páginas devem carregar em tempo médio inferior a 3 segundos em conexões banda larga.
RQNF003	Segurança	Essencial	Allan Vitor Correia	As senhas dos usuários devem ser criptografadas utilizando o algoritmo *bcrypt*.
RQNF004	Confiabilidade	Essencial	Allan Vitor Correia	O sistema deve garantir integridade dos dados e evitar inconsistências durante operações simultâneas.

RQNF005	Compatibilidade	Essencial	Allan Vitor Correia	O sistema deve ser compatível com os navegadores Google Chrome, Mozilla Firefox e Microsoft Edge (versões atuais).
RQNF006	Portabilidade	Desejável	Allan Vitor Correia	O sistema deve funcionar em diferentes dispositivos (desktop, tablet e smartphone).
RQNF007	Manutenibilidade	Desejável	Allan Vitor Correia	O código deve ser modularizado e seguir boas práticas de desenvolvimento, facilitando futuras atualizações.
RQNF008	Escalabilidade	Opcional	Allan Vitor Correia	O sistema deve permitir expansão de suas funcionalidades sem necessidade de reformulação completa da base de dados.
RQNF009	Acessibilidade	Desejável	Allan Vitor Correia	A interface deve empregar contraste adequado, ícones descritivos e navegação clara, atendendo a princípios básicos de acessibilidade digital.
RQNF010	Armazenamento Seguro	Essencial	Allan Vitor Correia	O sistema deve armazenar dados de forma segura, utilizando PostgreSQL com controle de acesso e restrição de permissões.

3.4 REGRAS DE DOMÍNIO OU NEGÓCIO (RDN)

As regras de domínio descrevem políticas e restrições lógicas que regem o funcionamento do sistema **CineAA**, garantindo coerência entre os processos e as informações manipuladas.

Identificador	Nome	Prioridade	Autor	Descrição
RDN001	Autenticidade de Usuário	Essencial	Allan Vitor Correia	Cada usuário deve possuir um e-mail único no sistema, utilizado como identificador para login.
RDN002	Classificação de Filmes	Essencial	Allan Vitor Correia	Um filme só pode pertencer a uma lista específica do usuário (assistidos, para assistir, assistindo ou recomendado).
RDN003	Recomendações	Essencial	Allan Vitor Correia	O usuário pode recomendar um mesmo filme para outro usuário apenas uma vez, evitando duplicidade.

RDN004	Gerenciamento de Amizades	Desejável	Allan Vitor Correia	A relação de amizade só é efetivada após a aceitação da solicitação por ambas as partes.
RDN005	Notificações de Sistema	Essencial	Allan Vitor Correia	O sistema deve gerar notificações automáticas para eventos de amizade e recomendações recebidas.
RDN006	Privacidade de Listas	Desejável	Allan Vitor Correia	As listas personalizadas podem ser definidas como públicas ou privadas pelo usuário.
RDN007	Avaliação de Filmes	Essencial	Allan Vitor Correia	Cada usuário pode avaliar um mesmo filme apenas uma vez, podendo atualizar sua nota posteriormente.
RDN008	Integridade de Dados	Essencial	Allan Vitor Correia	A exclusão de um usuário deve remover automaticamente suas listas, recomendações e amizades vinculadas.
RDN009	Controle de Sessão	Essencial	Allan Vitor Correia	O sistema deve encerrar automaticamente a sessão do usuário após período de inatividade prolongada.
RDN010	Validação de Campos	Essencial	Allan Vitor Correia	Todos os campos de formulários devem ser validados antes do envio

				para o servidor, evitando entradas inválidas.

3.5 RESTRIÇÕES (R)

As restrições descrevem os limites técnicos e operacionais do sistema, considerando o ambiente de execução, ferramentas e recursos disponíveis para o desenvolvimento.

Identificador	Nome	Prioridade	Autor	Descrição
R001	Ambiente de Execução	Essencial	Allan Vitor Correia	O sistema deve ser executado localmente em ambiente XAMPP (Apache, PHP e PostgreSQL).
R002	Banco de Dados Relacional	Essencial	Allan Vitor Correia	O sistema deve utilizar exclusivamente o SGBD PostgreSQL.
R003	Linguagem de Programação	Essencial	Allan Vitor Correia	O back-end deve ser implementado em PHP, versão 8 ou superior.

R004	Conectividade	Essencial	Allan Vitor Correia	É necessária conexão ativa entre o servidor Apache e o PostgreSQL para funcionamento do sistema.
R005	Hospedagem Local	Desejável	Allan Vitor Correia	O sistema deve funcionar integralmente no servidor local antes de migração para ambiente web.
R006	Acesso Autenticado	Essencial	Allan Vitor Correia	Somente usuários cadastrados e autenticados podem acessar funcionalidades restritas.
R007	Dependência de Navegador	Essencial	Allan Vitor Correia	O sistema deve ser acessado por navegadores compatíveis com HTML5 e CSS3.
R008	Armazenamento de Imagens	Desejável	Allan Vitor Correia	As imagens de perfil e pôsteres de filmes devem ser armazenadas no diretório do servidor local.
R009	Controle de Sessão	Essencial	Allan Vitor Correia	O sistema deve utilizar sessões PHP para controlar o login e o logout dos usuários.
R010	Conformidade de Código	Desejável	Allan Vitor Correia	O código deve seguir boas práticas de nomenclatura, indentação e organização de diretórios.

4 DESCRIÇÃO DOS DADOS DO SISTEMA (DDS)

Esta seção apresenta a estrutura dos dados utilizados pelo sistema CineAA, descrevendo as tabelas que compõem o banco de dados relacional PostgreSQL. Cada conjunto de dados foi projetado de forma a garantir integridade, coerência e desempenho, respeitando as boas práticas de modelagem relacional (ELMASRI; NAVATHE, 2016).

DDS001 – Tabela `users`

Identificador	Descrição	Tipo / Tamanho	Formato	Restrições	Comentário
id	Identificador único do usuário	int(11)	Numérico	PK, AI	Chave primária
name	Nome completo do usuário	varchar(100)	Texto	Not null	—

email	Endereço de e-mail do usuário	varchar(100)	Texto	Not null, Unique	Usado no login
cpf	Cadastro de Pessoa Física	varchar(14)	000.000.000-00	Unique	Dado opcional
phone	Telefone de contato	varchar(15)	(00)00000-0000	—	—
password	Senha criptografada	varchar(255)	Hash	Not null	Criptografada com bcrypt
profile_picture	Caminho da imagem de perfil	varchar(255)	URL/local path	—	Imagem armazenada localmente
bio	Biografia do usuário	text	Texto livre	—	Campo opcional
created_at	Data de criação do registro	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—
updated_at	Data da última atualização	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—

DDS002 – Tabela `movies`

Identificador	Descrição	Tipo / Tamanho	Formato	Restrições	Comentário
id	Identificador do filme	int(11)	Numérico	PK, AI	Chave primária
genre	Gênero do filme	varchar(100)	Texto	—	—

title	Título do filme	varchar(255)	Texto	Not null	—
release_year	Ano de lançamento	int(11)	AAAA	—	—
director	Nome do diretor	varchar(255)	Texto	—	—
main_cast	Atores principais	text	Texto livre	—	Lista de nomes
description	Sinopse ou resumo do filme	text	Texto livre	—	—
rating	Nota média do filme	decimal(3,1)	0.0–10.0	Default 0	Atualizada dinamicamente
poster_url	Caminho ou URL do pôster	varchar(255)	URL/local path	—	—
created_at	Data de criação do registro	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—
updated_at	Data da última atualização	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—

DDS003 – Tabela `personal_catalog`

Identificador	Descrição	Tipo / Tamanho	Formato	Restrições	Comentário
id	Identificador do catálogo pessoal	int(11)	Numérico	PK, AI	—

user_id	ID do usuário associado	int(11)	Numérico	FK → users(id)	—
movie_id	ID do filme associado	int(11)	Numérico	FK → movies(id)	—
status	Situação do filme	enum	watched / watching / to_watch / recomendado	—	Define categoria
rating	Nota atribuída pelo usuário	int(11)	0–10	—	—
notes	Observações pessoais	text	Texto livre	—	—
created_at	Data de criação	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—
updated_at	Data de atualização	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—

DDS004 – Tabela `custom_lists`

Identificador	Descrição	Tipo / Tamanho	Formato	Restrições	Comentário
id	Identificador da lista	int(11)	Numérico	PK, AI	—
user_id	Usuário criador da lista	int(11)	Numérico	FK → users(id)	—
name	Nome da lista	varchar(100)	Texto	Not null	—

description	Descrição da lista	text	Texto livre	—	—
is_public	Indicador de visibilidade	tinyint(1)	0 / 1	Default 1	1 = pública
created_at	Data de criação	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—
updated_at	Data de atualização	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—

DDS005 – Tabela `list_movies`

Identificador	Descrição	Tipo / Tamanho	Formato	Restrições	Comentário
list_id	Identificador da lista	int(11)	Numérico	FK → custom_lists(id)	—
movie_id	Identificador do filme	int(11)	Numérico	FK → movies(id)	—

Essa tabela implementa o relacionamento *muitos-para-muitos* entre filmes e listas personalizadas.

DDS006 – Tabela `friendships`

Identificador	Descrição	Tipo / Tamanho	Formato	Restrições	Comentário
id	Identificador da amizade	int(11)	Numérico	PK, AI	—

user_id	ID do usuário solicitante	int(11)	Numérico	FK → users(id)	—
friend_id	ID do amigo	int(11)	Numérico	FK → users(id)	—
status	Estado da amizade	enum	pending / accepted / rejected	Default pending	—
created_at	Data da solicitação	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—
updated_at	Última atualização	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—

DDS007 – Tabela `recommendations`

Identificador	Descrição	Tipo / Tamanho	Formato	Restrições	Comentário
id	Identificador da recomendação	int(11)	Numérico	PK, AI	—
from_user_id	Usuário que enviou a recomendação	int(11)	Numérico	FK → users(id)	—
to_user_id	Usuário que recebeu a recomendação	int(11)	Numérico	FK → users(id)	—
movie_id	Filme recomendado	int(11)	Numérico	FK → movies(id)	—
message	Mensagem personalizada	text	Texto livre	—	Opcional

status	Status da recomendação	enum	pending / accepted / rejected	Default pending	—
created_at	Data de envio	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—
updated_at	Data de atualização	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—

DDS008 – Tabela `notifications`

Identificador	Descrição	Tipo / Tamanho	Formato	Restrições	Comentário
id	Identificador da notificação	int(11)	Numérico	PK, AI	—
user_id	Usuário destinatário da notificação	int(11)	Numérico	FK → users(id)	—
type	Tipo de notificação	varchar(50)	Texto	—	Ex: amizade, recomendação
reference_id	ID de referência ao evento	int(11)	Numérico	—	Pode referenciar amizade ou recomendação
message	Conteúdo textual da notificação	text	Texto livre	—	—

is_read	Indicador de leitura	tinyint(1)	0 / 1	Default 0	1 = lida
created_at	Data de criação	timestamp	AAAA-MM-DD HH:MM:SS	Default current	—

6 Tecnologias do projeto

6.1 HTML5

O HTML5 é mantido pelo **World Wide Web Consortium (W3C)**, que define os padrões e recomendações oficiais para a linguagem (W3C, 2014). O HTML5 (HyperText Markup Language – versão 5) é a mais recente evolução da linguagem de marcação padrão para a construção de páginas web. O HTML5 sendo hoje amplamente aceito e suportado por todos os navegadores.

Diferentemente das versões anteriores, o HTML5 introduziu **novas tags semânticas** (como <header>, <footer>, <section>, <article> e <aside>), que tornam o código mais legível tanto para desenvolvedores quanto para mecanismos de busca e ferramentas de acessibilidade. Essa semântica favorece a organização do

conteúdo e contribui para boas práticas de **SEO (Search Engine Optimization)**, aspecto essencial em aplicações que demandam visibilidade.

Outro avanço significativo foi o **suporte nativo a multimídia**, por meio das tags <audio> e <video>, eliminando a necessidade de plugins de terceiros. Além disso, elementos como <canvas> (para renderização de gráficos e animações em tempo real), <svg> (para imagens vetoriais escaláveis) e a API de **geolocalização** ampliaram as possibilidades de interatividade e usabilidade diretamente no navegador.

Do ponto de vista da integração tecnológica, o HTML5 foi projetado para trabalhar em conjunto com o **CSS3 e o JavaScript**, compondo o que é chamado de **tríade da web**. Essa integração permite a construção de sistemas completos e dinâmicos, como é o caso do CineAA. Outro ponto de destaque é a **responsividade**, facilitada pelo uso de atributos e elementos que, aliados a folhas de estilo, permitem que a aplicação seja acessada em diferentes dispositivos (computadores, tablets e smartphones) sem perda de qualidade.

O HTML5 também se destacou por introduzir **APIs nativas** que ampliaram significativamente o desenvolvimento web:

- **Local Storage e Session Storage**, que permitem armazenar dados localmente no navegador, mesmo em cenários offline;
- **IndexedDB**, banco de dados orientado a objetos dentro do navegador para manipulação de grandes volumes de dados;
- **WebSockets**, que viabilizam comunicação em tempo real entre cliente e servidor;
- **API de Drag and Drop**, que facilita a criação de interfaces mais interativas e intuitivas.

Outro aspecto importante é a evolução dos **formulários**, com novos tipos de entrada como email, date, number e url. Esses campos melhoram a experiência do usuário, especialmente em dispositivos móveis, onde o teclado virtual se adapta ao tipo de dado solicitado, além de simplificarem a validação de informações no lado do cliente.

Em termos de **segurança**, o HTML5 reduziu a dependência de plugins externos vulneráveis e passou a oferecer suporte mais robusto a conexões seguras

via HTTPS, além de mecanismos que favorecem boas práticas de desenvolvimento seguro.

O **HTML5** também é essencial para o conceito de *Progressive Web Apps* (PWAs), que unem características de sites e aplicativos móveis. Por meio de recursos como *Service Workers*, é possível implementar cache inteligente, funcionamento offline e envio de notificações *push*, tornando a experiência mais próxima à de um aplicativo nativo. Segundo o W3C (2014), o HTML5 consolidou-se como o padrão de referência no desenvolvimento web, trazendo avanços significativos em semântica, acessibilidade, integração multimídia e suporte a aplicações modernas.

O **HTML5** apresenta uma série de qualidades que o consolidaram como padrão de referência no desenvolvimento web. Entre as principais, destacam-se:

- **Compatibilidade universal:** Por ser um padrão consolidado pelo W3C e amplamente aceito pelos navegadores modernos (como Chrome, Firefox, Safari e Edge), o HTML5 garante que páginas desenvolvidas sejam exibidas corretamente em diferentes plataformas, sem a necessidade de versões específicas ou complementos adicionais. Isso reduz custos de manutenção e aumenta a eficiência do desenvolvimento.

- **Semântica aprimorada:** As novas tags semânticas introduzidas pelo HTML5 permitem uma organização lógica e clara do conteúdo, facilitando a leitura do código tanto por desenvolvedores quanto por mecanismos de busca. Essa característica também melhora a acessibilidade e contribui diretamente para práticas de SEO (*Search Engine Optimization*), aumentando a visibilidade das aplicações web.

- **Integração com multimídia:** O suporte nativo a áudio, vídeo, gráficos e imagens vetoriais escaláveis elimina a dependência de *plugins* externos como Flash ou Silverlight. Isso não apenas simplifica o desenvolvimento de conteúdos interativos, como também aumenta a segurança e a performance da aplicação.

- **Performance otimizada:** Ao reduzir a necessidade de tecnologias de terceiros e oferecer suporte a recursos nativos, o HTML5 possibilita páginas mais leves e de carregamento mais rápido. Essa característica é essencial em um

cenário em que os usuários demandam experiências fluidas e imediatas.

- **Acessibilidade ampliada:** O HTML5 trabalha em conjunto com padrões como o WAI-ARIA (*Accessible Rich Internet Applications*), que garantem compatibilidade com leitores de tela e outras tecnologias assistivas. Dessa forma, ele contribui para a inclusão digital e para o cumprimento de boas práticas de acessibilidade em sistemas web.

- **Interoperabilidade e integração:** Projetado para atuar em conjunto com o CSS3 e o JavaScript, o HTML5 permite a construção de sistemas completos e altamente interativos. Além disso, sua arquitetura facilita a integração com *frameworks* e bibliotecas modernas, como React, Angular e Vue.js, potencializando o desenvolvimento ágil e escalável.

- **Durabilidade tecnológica:** Por ser um padrão global constantemente atualizado, o HTML5 garante que os conteúdos criados hoje continuarão acessíveis no futuro, evitando a obsolescência tecnológica e assegurando longevidade às aplicações.

- **Portabilidade:** Aplicações em HTML5 podem ser acessadas em múltiplos dispositivos conectados à internet, incluindo computadores, tablets, smartphones, televisores inteligentes e até consoles de jogos, o que amplia o alcance e a usabilidade dos sistemas.

- **Melhorias nos formulários:** Novos tipos de campos (*email, date, number, url, color*) e atributos (*required, pattern, placeholder*) facilitam a validação de dados no lado do cliente e aprimoram a experiência do usuário, sobretudo em dispositivos móveis, nos quais o teclado virtual se adapta ao tipo de dado solicitado.

- **Offline e armazenamento local:** Recursos como *Local Storage*, *Session Storage* e *IndexedDB* permitem que informações sejam armazenadas diretamente no navegador, viabilizando o funcionamento parcial das aplicações mesmo sem conexão com a internet.

- **Segurança aprimorada:** A eliminação da dependência de *plugins* externos reduziu significativamente vulnerabilidades comuns. Além disso, o HTML5

possui suporte robusto a conexões seguras via HTTPS e a mecanismos de controle de permissões para APIs (como geolocalização, acesso à câmera e microfone).

- **Suporte a aplicações modernas (PWAs):** O HTML5 é peça-chave no desenvolvimento de *Progressive Web Apps*, que combinam características de sites e aplicativos móveis. Isso possibilita funcionalidades como notificações *push*, cache inteligente e funcionamento offline, tornando a experiência próxima à de um aplicativo nativo (W3C, 2014).

No projeto CineAA, o HTML5 foi amplamente utilizado para estruturar as páginas principais do sistema, como index.html (tela inicial e de login), catalogo.html (exibição dos filmes disponíveis), perfil.html (informações do usuário) e amigos.html (gerenciamento da rede social), apresentadas respectivamente nas Figuras 1 e 2. Em todas essas páginas, o uso de elementos semânticos contribuiu para uma melhor organização do código, facilitando tanto a manutenção quanto a futura expansão do sistema.

Figura 1 - index.html

Fonte: autoria própria.

Figura 2 – amigos.html

Fonte: autoria própria.

6.2 CSS3

O CSS é um dos principais pilares do desenvolvimento web moderno, permitindo separar a estrutura do conteúdo da sua apresentação visual. As especificações evoluem constantemente para atender novas demandas de design, responsividade e acessibilidade. De acordo com o *World Wide Web Consortium (W3C, 2025)*, o **CSS Snapshot 2025** reúne os módulos estáveis da linguagem, funcionando como uma visão geral consolidada das recomendações atuais.

Enquanto o HTML5 é responsável pela **estruturação do conteúdo**, o CSS3 atua como complemento, permitindo a personalização visual, a responsividade e a criação de interfaces mais atrativas e interativas. Essa separação entre estrutura e estilo garante maior organização ao projeto, facilita a manutenção do sistema e promove a reutilização de código, reduzindo esforços em grandes aplicações web.

Uma das principais diferenças em relação às versões anteriores é que o CSS3 foi desenvolvido em **módulos independentes**, como *Backgrounds and Borders*, *Selectors*, *Media Queries*, *Animations* e *Flexbox*. Essa modularização possibilitou a inclusão de novos recursos sem comprometer a compatibilidade com

versões antigas, além de permitir que os navegadores implementassem gradualmente as novidades de acordo com sua evolução.

Entre os avanços mais significativos do **CSS3** destacam-se: transições, animações, gradientes, sombras, bordas arredondadas, *flexbox*, *grid layout* e *media queries* para design responsivo. Esses recursos reduziram a necessidade de imagens externas ou *scripts* adicionais, tornando o carregamento das páginas mais leve, eficiente e seguro. Segundo o W3C (2018), esses recursos ampliaram a flexibilidade no desenvolvimento de interfaces e consolidaram o CSS3 como padrão fundamental para a web moderna.

A responsividade é considerada uma das maiores contribuições do CSS3, alcançada principalmente por meio das *media queries*, que permitem ajustar automaticamente o layout da página de acordo com o tamanho da tela e o dispositivo utilizado. Esse recurso tornou-se indispensável no cenário atual, no qual usuários acessam aplicações em múltiplas plataformas, como computadores, tablets e smartphones (MOZILLA DEVELOPER NETWORK, 2025).

Os recursos visuais avançados também merecem destaque:

- Sombras em elementos e textos (*box-shadow, text-shadow*), que aumentam a profundidade da interface;

- Gradientes lineares e radiais (*linear-gradient, radial-gradient*), que permitem efeitos de cor modernos sem imagens externas;

- Transformações 2D/3D (*transform, rotate, scale, skew*), que possibilitam efeitos interativos e dinâmicos;

- Transições e animações, que permitem criar efeitos de suavidade em interações de usuário, elevando a experiência visual a um nível semelhante ao de aplicativos nativos.

Além disso, o CSS3 ampliou o uso de seletores avançados, permitindo maior precisão na estilização e manipulação de elementos específicos. Também introduziu pseudo-classes e pseudo-elementos mais robustos (*:nth-child, :not, ::before, ::after*),

que oferecem maior controle sobre a personalização sem necessidade de *scripts* adicionais.

Outro recurso de destaque são as **CSS Custom Properties** (variáveis), que permitem definir valores reutilizáveis para cores, espaçamentos e tipografias, aumentando a consistência do design e facilitando a manutenção de grandes projetos.

O CSS3 também desempenha um papel relevante em acessibilidade, já que propriedades como *contrast, visibility, pointer-events* e o uso adequado de *focus* contribuem para criar interfaces mais inclusivas, compatíveis com tecnologias assistivas e em conformidade com padrões internacionais de usabilidade.

Do ponto de vista de integração tecnológica, o CSS3 atua de forma complementar ao HTML5 e ao JavaScript, compondo a chamada tríade da web. Essa combinação permite desenvolver *Progressive Web Apps* (PWAs) com interfaces modernas, responsivas e fluidas, compatíveis com múltiplos dispositivos e próximas da experiência de aplicativos móveis nativos (W3C, 2018; MOZILLA DEVELOPER NETWORK, 2025).

Entre as principais qualidades do CSS3, destacam-se:

- Separação entre conteúdo e estilo: garante organização, manutenção simplificada e reaproveitamento de código;
- Responsividade nativa: possibilita adaptar layouts a diferentes dispositivos sem duplicação de conteúdo;
- Design moderno: suporte a gradientes, sombras, cantos arredondados, tipografia personalizada (via *@font-face*) e animações complexas;
- Redução da dependência de imagens: muitos efeitos podem ser criados diretamente no código, deixando a página mais leve;
- Alto desempenho: renderização otimizada pelos navegadores, aumentando a velocidade de carregamento;
- Flexibilidade de layout: recursos como *flexbox* e *grid layout* permitem organizar elementos de forma dinâmica, escalável e padronizada;
- Compatibilidade universal: suporte garantido por todos os navegadores modernos;

Customização escalável: uso de variáveis CSS para padronizar estilos e facilitar manutenção;

- Interatividade avançada: transições, animações e pseudo-elementos tornam as interfaces mais dinâmicas;
- Contribuição para acessibilidade: propriedades visuais e estruturais que facilitam a adaptação de conteúdos para diferentes públicos.

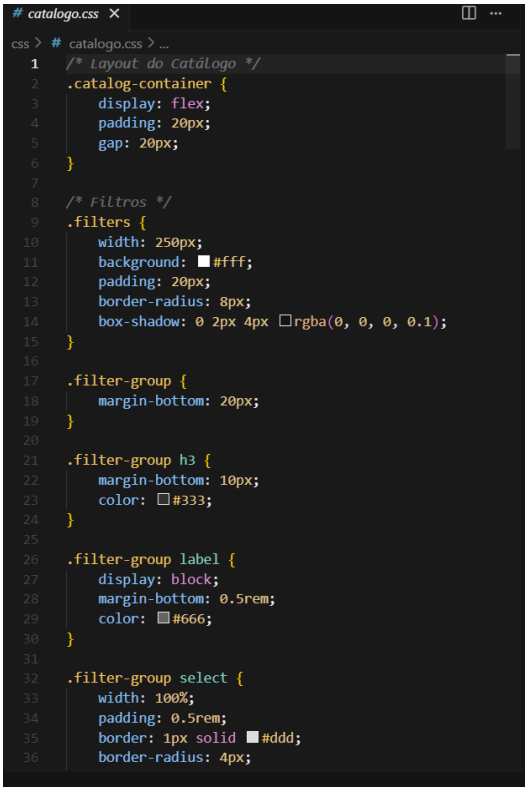
No projeto **CineAA**, o CSS3 desempenhou papel fundamental na construção da identidade visual da aplicação. Arquivos como css/style.css, css/catalogo.css e css/perfil.css, representados nas Figuras 3 e 4, foram responsáveis por definir paletas de cores, tipografia, organização de elementos na tela e responsividade das páginas. Por exemplo:

- O **style.css** foi utilizado para padronizar o design geral da aplicação, incluindo botões, formulários e cabeçalhos;
- O **catalogo.css** estruturou a apresentação do catálogo de filmes, organizando os cards em grades flexíveis e responsivas;
- O **perfil.css** permitiu a personalização da página de perfil, garantindo que as informações do usuário fossem exibidas de forma clara e organizada.

Figura 3 - css/style

Fonte: autoria própria.

Figura 4 - css/catálogo



```css
/* Layout do Catálogo */
.catalog-container {
    display: flex;
    padding: 20px;
    gap: 20px;
}

/* Filtros */
.filters {
    width: 250px;
    background: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

.filter-group {
    margin-bottom: 20px;
}

.filter-group h3 {
    margin-bottom: 10px;
    color: #333;
}

.filter-group label {
    display: block;
    margin-bottom: 0.5rem;
    color: #666;
}

.filter-group select {
    width: 100%;
    padding: 0.5rem;
    border: 1px solid #ddd;
    border-radius: 4px;
```

Fonte: autoria própria.

6.3 JavaScript

Criado em 1995 pela Netscape, o JavaScript surgiu com o propósito de tornar a web mais interativa e dinâmica. Inicialmente utilizado apenas para pequenas validações em formulários, rapidamente evoluiu para se consolidar como a principal linguagem de programação para o desenvolvimento web no lado do cliente (front-end). Atualmente, além de ser suportado por todos os navegadores modernos, sua versatilidade permite também o uso no lado do servidor (com **Node.js**), em aplicações móveis (com **React Native**), em softwares desktop multiplataforma (como os desenvolvidos em **Electron**) e até mesmo em dispositivos IoT.

O JavaScript é uma linguagem de programação dinâmica e baseada em protótipos, amplamente utilizada no desenvolvimento web para criar páginas interativas e dinâmicas. Segundo o *Mozilla Developer Network (MDN, 2025)*, o JavaScript é uma linguagem leve, interpretada e baseada em objetos com funções de primeira classe, mais conhecida como a linguagem de script para páginas web, mas usada também em vários outros ambientes sem browser, tais como Node.js, Apache CouchDB e Adobe Acrobat.

O grande diferencial do JavaScript está em sua execução diretamente no navegador, o que possibilita manipular a estrutura da página por meio do **DOM (Document Object Model)**. Essa capacidade permite alterar conteúdos em tempo real, validar dados de formulários, aplicar efeitos visuais e responder a interações do usuário de maneira imediata, sem a necessidade de recarregar a página. Essa característica garante uma experiência mais fluida, interativa e próxima de aplicações nativas.

Outro avanço significativo foi a introdução de técnicas para **requisições assíncronas**, como o **AJAX** e, mais recentemente, o uso da **API fetch()**, **Promises** e **async/await**. Esses recursos permitem que as páginas se comuniquem com o servidor sem recarregamento completo, otimizando a performance, melhorando a experiência de navegação e viabilizando aplicações modernas conhecidas como **Single Page Applications (SPA)**.

Além disso, o JavaScript consolidou um **ecossistema robusto**, com uma ampla gama de **bibliotecas** (como jQuery, D3.js, Chart.js) e **frameworks** (como **React**, **Angular** e **Vue.js**) que potencializam a produtividade dos desenvolvedores, trazendo soluções estruturadas para aplicações complexas. Entretanto, mesmo sem esses recursos externos, a linguagem em sua forma nativa já oferece meios suficientes para implementar interatividade de alto nível destacando as principais características do JavaScript .

- **Interatividade avançada**: permite criar interfaces altamente dinâmicas, onde botões, menus, formulários e componentes visuais respondem imediatamente à ação do usuário. Esse dinamismo é essencial em aplicações que buscam maior engajamento e usabilidade.
- **Execução no cliente**: reduz significativamente a carga sobre os servidores, uma vez que boa parte do processamento ocorre diretamente no navegador. Isso se traduz em menor consumo de recursos de infraestrutura e maior escalabilidade em sistemas web.
- **Comunicação assíncrona**: com o uso de AJAX, fetch() e WebSockets, é possível atualizar apenas partes específicas da página, manter conexões em tempo real e construir aplicações que se comportam como softwares desktop, sem recarregamentos constantes.
- **Compatibilidade universal**: o JavaScript é suportado nativamente em todos os navegadores modernos, o que garante que aplicações desenvolvidas com ele funcionem de maneira consistente em diferentes dispositivos e plataformas.

- **Integração tecnológica**: atua em conjunto com **HTML5**, **CSS3** e diversas **APIs do navegador** (como geolocalização, armazenamento local, notificações e acesso a hardware como câmera e microfone), possibilitando o desenvolvimento de soluções completas.
- **Ecossistema expansivo**: conta com milhares de bibliotecas e frameworks que aceleram o processo de desenvolvimento e fornecem soluções já testadas e otimizadas, permitindo que os desenvolvedores foquem em funcionalidades de alto nível.
- **Facilidade de aprendizado e ampla comunidade**: sua sintaxe simples facilita a curva de aprendizado para iniciantes, enquanto sua vasta documentação e comunidade ativa fornecem suporte constante para desenvolvedores de todos os níveis.
- **Alta performance em tempo real**: com o avanço dos motores JavaScript (como V8 do Google Chrome e SpiderMonkey da Mozilla), a linguagem passou a oferecer performance suficiente para aplicações complexas, incluindo **jogos 2D/3D**, animações gráficas e simulações em tempo real.
- **Portabilidade e ubiquidade**: o JavaScript não está restrito apenas ao navegador. Com o Node.js, ele conquistou espaço no backend, além de viabilizar o desenvolvimento de aplicativos móveis e desktop. Essa versatilidade faz com que seja uma das linguagens mais utilizadas no mundo.
- **Escalabilidade e modularização**: com a introdução de módulos (ES6 Modules) e padrões modernos de programação, o JavaScript tornou-se adequado não apenas para pequenos scripts, mas também para projetos de larga escala, sustentando sistemas corporativos e plataformas globais.
- **Atualizações constantes**: o JavaScript é uma linguagem em constante evolução, recebendo melhorias anuais por meio do ECMAScript. Isso garante que a linguagem acompanhe as novas demandas do desenvolvimento web e mantenha sua relevância no futuro.

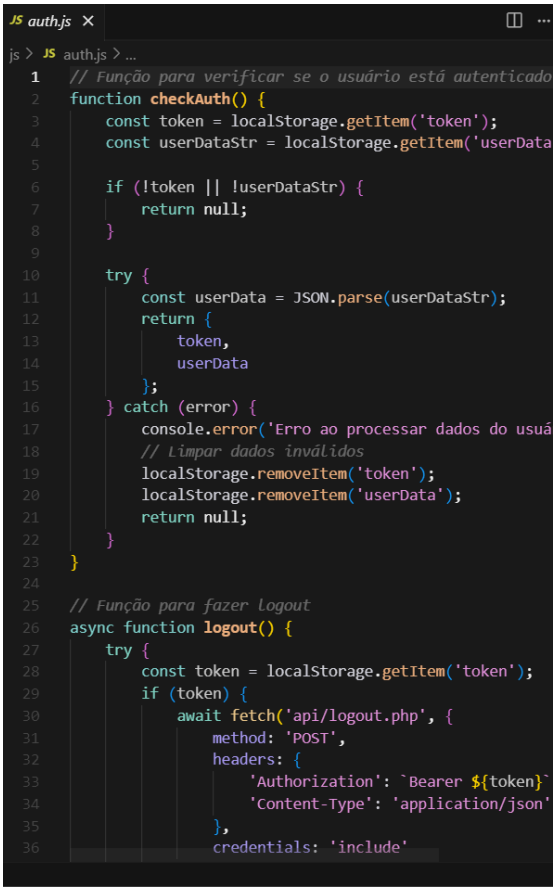
No contexto do projeto **CineAA**, o JavaScript desempenhou papel fundamental na interatividade e na usabilidade do sistema. Alguns exemplos práticos de aplicação incluem:

- O arquivo **js/auth.js** foi responsável pela autenticação de usuários, validando dados inseridos e interagindo com a API em PHP.
- O arquivo **js/catalogo.js** gerenciou o carregamento dinâmico dos filmes disponíveis, permitindo uma navegação fluida e organizada.

- O arquivo **js/perfil.js** possibilitou a atualização imediata das informações do usuário, refletindo mudanças diretamente na interface.
- O arquivo **js/recommendations.js** foi utilizado para exibir recomendações personalizadas de filmes, ampliando a experiência do usuário.

Essas funcionalidades foram implementadas por meio da manipulação do **DOM** e do uso de **requisições assíncronas**, garantindo ao usuário uma navegação fluida, sem interrupções ou recarregamentos desnecessários, como evidenciado na Figura 5.

Figura 5 - js/auth.js



Fonte: autoria própria.

6.4 PHP

O PHP, inicialmente criado por Rasmus Lerdorf em 1994 como um conjunto de scripts para rastrear visitas ao seu currículo online, evoluiu significativamente ao longo dos anos. Segundo o manual oficial do PHP (PHP Manual, 2025), o PHP/FI foi uma versão aprimorada que introduziu funcionalidades como interpretação automática de variáveis de formulário e sintaxe embutida em HTML, características

que marcaram o início da linguagem como a conhecemos hoje, Atualmente, o PHP é uma das linguagens mais utilizadas em sistemas web dinâmicos, estando presente em grandes plataformas como **WordPress, Drupal, Joomla**, e em aplicações de larga escala, como Facebook e Wikipedia (em versões iniciais de suas arquiteturas).

O grande diferencial do PHP está na sua capacidade de **gerar páginas dinâmicas** a partir de scripts executados no servidor, que podem interagir diretamente com bancos de dados e com o front-end. Quando um usuário acessa uma página desenvolvida em PHP, o servidor processa o código e retorna ao navegador apenas o resultado em HTML, CSS ou JavaScript, garantindo **segurança**, uma vez que a lógica do sistema não fica exposta ao cliente.

Outra característica relevante é a integração nativa com bancos de dados relacionais, especialmente MySQL e MariaDB, mas também com PostgreSQL, SQLite e outros sistemas de gerenciamento. Essa facilidade permite desenvolver aplicações completas, capazes de cadastrar, consultar, atualizar e excluir dados de forma eficiente, possibilitando a construção de sistemas dinâmicos e interativos. Segundo o **PHP Group (2025)**, a linguagem mantém suporte contínuo a diversos SGBDs, além de atualizações que garantem compatibilidade com arquiteturas modernas.

Além disso, o PHP possui sintaxe acessível e flexível, facilitando a aprendizagem por desenvolvedores iniciantes, enquanto sua ampla comunidade e extensa documentação garantem suporte contínuo e evolução constante da linguagem (LERDORF; GUTMANS; SURASKI, 2002).

Principais qualidades do PHP:

- **Execução no servidor:** garante que toda a lógica da aplicação seja processada antes de ser enviada ao cliente, aumentando a segurança e prevenindo acesso não autorizado a dados sensíveis.

- **Integração nativa com bancos de dados:** suporte a múltiplos SGBDs, permitindo consultas complexas, manipulação eficiente de dados e compatibilidade com diferentes arquiteturas de banco.

- **Flexibilidade e simplicidade:** curva de aprendizado acessível, sintaxe intuitiva e compatibilidade com outras linguagens de programação.

- **Ampla adoção e comunidade ativa:** grande disponibilidade de bibliotecas, *frameworks* (como Laravel, CodeIgniter, Symfony) e ferramentas, além de tutoriais, fóruns e suporte técnico.

- **Portabilidade:** compatível com diferentes sistemas operacionais, como Windows, Linux e macOS, possibilitando o desenvolvimento e implantação em diversos ambientes.

- **Desempenho adequado:** especialmente quando combinado com técnicas de cache, servidores otimizados (Apache, Nginx) e práticas de programação eficientes.

- **Integração com front-end:** permite mesclar código PHP dentro de páginas HTML e interagir com JavaScript, agilizando o desenvolvimento de interfaces dinâmicas.

- **Segurança:** oferece recursos nativos e práticas recomendadas para prevenção de ataques comuns, como injeção de SQL, XSS e CSRF, garantindo proteção em aplicações web.

- **Escalabilidade:** suporta desde pequenos sites até aplicações corporativas de grande porte, sendo capaz de atender demandas crescentes de usuários e dados.

- **Suporte a padrões modernos:** *frameworks* e arquiteturas como MVC (*Model-View-Controller*), APIs REST e integração com serviços externos tornam o PHP adequado para projetos complexos e estruturados.

No projeto **CineAA**, o PHP foi essencial para implementar as funcionalidades do **back-end**, funcionando como elo entre a interface do usuário e o banco de dados, conforme ilustrado na Figura 6. Alguns exemplos de aplicação incluem:

- O arquivo **login.php** gerenciou o processo de autenticação, verificando credenciais e controlando sessões de acesso.
- O arquivo **recuperar_senha.php**, em conjunto com o **PHP Mailer**, permitiu o envio seguro de e-mails de recuperação de senha.
- A pasta **api/** concentrou scripts que atuaram como controladores da aplicação:

- **api/register.php** implementou o registro de novos usuários;
- **api/login.php** controlou sessões e autenticação;
- **api/movies.php** gerenciou o catálogo de filmes, fornecendo dados em tempo real para o front-end.

- O arquivo **config/database.php** centralizou informações de conexão com o banco, garantindo organização, manutenção facilitada e consistência no código.

Figura 6 - login.php

Fonte: autoria própria.

Com o PHP, o CineAA pôde oferecer funcionalidades fundamentais como cadastro, login, gerenciamento de perfis, catálogo de filmes e recomendações personalizadas. A linguagem também possibilitou a implementação da **arquitetura MVC**, separando claramente as camadas de **Model**, **View** e **Controller**, o que facilita manutenção futura, escalabilidade e extensão de funcionalidades.

Dessa forma, o **PHP** não apenas suportou a lógica do servidor, mas também garantiu **dinamismo, segurança, integração com bancos de dados e escalabilidade**, consolidando-se como uma base essencial para o funcionamento

completo do CineAA e reafirmando seu papel como uma das linguagens mais importantes no desenvolvimento web moderno.

6.5 SQL e Banco de Dados Relacionais

O SQL (Structured Query Language) é a linguagem padrão para gerenciamento de bancos de dados relacionais, permitindo a criação, manipulação e consulta de dados de forma eficiente. Segundo *Elmasri e Navathe (2016)*, o SQL oferece recursos para definição de esquema, manipulação de dados e controle de acesso, sendo essencial para o desenvolvimento de sistemas que dependem de armazenamento estruturado e integridade dos dados.

Um **banco de dados relacional** organiza as informações em **tabelas** (também chamadas de relações), compostas por **linhas (registros)** e **colunas (atributos)**. Essa estrutura permite a **normalização dos dados**, evitando redundâncias desnecessárias e garantindo **integridade e consistência**. Cada tabela é interligada por meio de **chaves primárias** e **chaves estrangeiras**, permitindo consultas complexas e manutenção eficiente das informações.

A linguagem **SQL (Structured Query Language)** é utilizada para manipular e consultar dados em bancos relacionais, sendo um padrão adotado pela maioria dos SGBDs (Sistemas de Gerenciamento de Bancos de Dados), como MySQL, PostgreSQL, Oracle e SQL Server. O SQL é organizado em **subconjuntos principais**, cada um voltado a uma função específica:

- **DDL (Data Definition Language)**: responsável pela definição da estrutura do banco de dados, incluindo criação de tabelas (**CREATE TABLE**), alteração (**ALTER**) e exclusão (**DROP**).
- **DML (Data Manipulation Language)**: utilizada para inserir, atualizar e excluir registros (**INSERT**, **UPDATE**, **DELETE**).
- **DQL (Data Query Language)**: focada em consultas, principalmente através do comando **SELECT**, permitindo filtrar, agrupar e ordenar dados de acordo com diferentes critérios.

- **DCL (Data Control Language)**: controla permissões e segurança do banco, definindo quais usuários podem acessar, modificar ou visualizar dados (**GRANT**, **REVOKE**).
- **TCL (Transaction Control Language)**: garante consistência em operações críticas, permitindo agrupar comandos em **transações** que podem ser confirmadas (**COMMIT**) ou revertidas (**ROLLBACK**).

Principais qualidades do SQL e dos bancos de dados relacionais:

- **Organização estruturada**: os dados são armazenados em tabelas inter-relacionadas, facilitando manutenção, consultas e integridade.
- **Integridade e consistência**: mecanismos como **constraints**, **triggers** e **transações** asseguram que as informações permaneçam corretas e confiáveis.
- **Flexibilidade em consultas**: o SQL permite filtrar, agrupar e ordenar dados, realizar **junções** complexas, utilizar **subqueries** e funções agregadas, proporcionando grande poder de análise.
- **Segurança**: controle granular de acesso por usuário, definição de permissões e uso de **procedures** e **views** contribuem para proteger dados sensíveis.
- **Escalabilidade**: bancos relacionais podem suportar desde aplicações de pequeno porte até sistemas corporativos de grande escala, garantindo desempenho consistente.
- **Ampla adoção**: SQL é um padrão universalmente reconhecido, o que torna o conhecimento da linguagem aplicável em praticamente qualquer SGBD.
- **Suporte a transações complexas**: essencial em sistemas que exigem confiabilidade, como aplicativos de e-commerce ou redes sociais, garantindo que operações múltiplas sejam executadas de forma atômica.
- **Interoperabilidade**: permite integração com linguagens de programação (PHP, Python, JavaScript, Java) e frameworks, viabilizando a construção de sistemas completos e dinâmicos.
- **Capacidade analítica**: possibilita geração de relatórios, estatísticas e insights a partir de grandes volumes de dados, apoiando decisões estratégicas.

Uso no projeto CineAA:

No projeto **CineAA**, o SQL e o **modelo relacional** foram fundamentais para estruturar e manipular os dados do sistema. O banco de dados foi definido no arquivo **database/schema.sql**, conforme ilustrado na Figura 7, incluindo as principais tabelas:

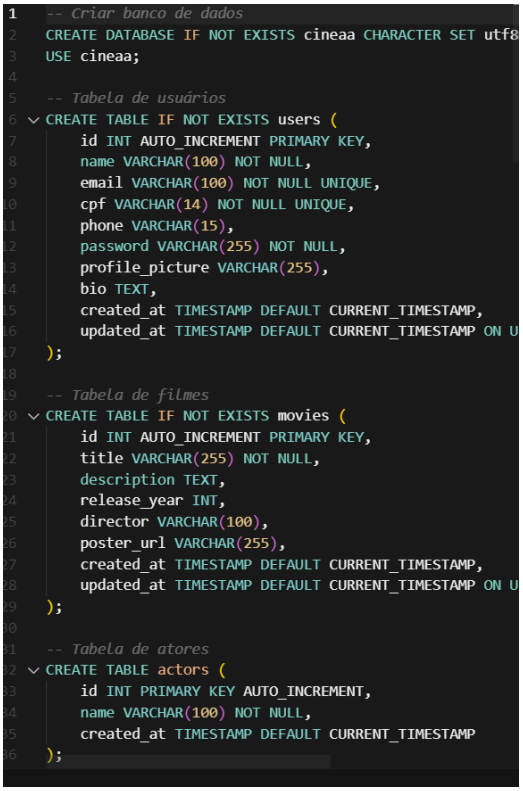
- **Usuários**: armazena informações cadastrais, credenciais e dados de autenticação.

- **Filmes**: catálogo de filmes disponíveis, com atributos como título, gênero, ano, descrição e avaliação média.
- **Amizades**: representa relações sociais entre os usuários, permitindo interações semelhantes a uma rede social.
- **Comentários**: registra opiniões e interações dos usuários sobre os filmes.
- **Atividades**: rastreia ações dos usuários, como avaliações, recomendações ou interações com amigos.

Exemplos de aplicação no CineAA incluem:

- **INSERT**: utilizado para cadastrar novos usuários durante o registro (**api/register.php**) ou adicionar filmes ao catálogo.
- **SELECT**: empregado para autenticar credenciais no login (**login.php**) e exibir dinamicamente o catálogo de filmes (**catalogo.html**) via integração com **api/movies.php**.
- **UPDATE**: utilizado no **update-user-info.php** para modificar informações do perfil do usuário.
- **DELETE**: aplicado em operações de exclusão de comentários ou amizades, garantindo flexibilidade de gerenciamento e manutenção da integridade dos dados.

Figura 7-database



```sql
-- Criar banco de dados
CREATE DATABASE IF NOT EXISTS cineaa CHARACTER SET utf8
USE cineaa;

-- Tabela de usuários
CREATE TABLE IF NOT EXISTS users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    cpf VARCHAR(14) NOT NULL UNIQUE,
    phone VARCHAR(15),
    password VARCHAR(255) NOT NULL,
    profile_picture VARCHAR(255),
    bio TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U
);

-- Tabela de filmes
CREATE TABLE IF NOT EXISTS movies (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    release_year INT,
    director VARCHAR(100),
    poster_url VARCHAR(255),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON U
);

-- Tabela de atores
CREATE TABLE actors (
    id INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(100) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Fonte: autoria própria.

Além disso, o banco de dados do CineAA utilizou **constraints de integridade**, **chaves primárias e estrangeiras**, **índices** para otimizar consultas e **transações** para operações críticas, assegurando consistência e confiabilidade do sistema. A estrutura relacional permitiu que as funcionalidades do CineAA funcionassem de forma **segura, eficiente e escalável**, atendendo às necessidades de uma aplicação web moderna.

Dessa forma, o SQL e o modelo relacional desempenharam papel essencial no projeto, oferecendo **organização, confiabilidade, segurança e flexibilidade**, tornando possível que todas as camadas do CineAA — front-end, back-end e banco de dados — trabalhassem de maneira integrada e eficient

6.6 XAMPP

Para o desenvolvimento local do projeto, utilizei o **XAMPP**, uma ferramenta que provê um ambiente integrado com Apache, MySQL/MariaDB, PHP e Perl. Esse ambiente simula as condições de um servidor real, permitindo construir, testar e depurar aplicações web de forma segura antes de sua publicação. Conforme descrito no blog *Mercado Online Digital*, o XAMPP facilita a instalação de um servidor web local, tornando possível realizar operações sobre o banco de dados localmente, como criação de tabelas, inserção de dados e consultas.

O **XAMPP** é um **pacote de software livre** que reúne diversas ferramentas essenciais para o desenvolvimento e execução de aplicações web em ambiente local. O nome é um acrônimo que representa seus principais componentes:

- **X** → **multiplataforma**: compatível com Windows, Linux e macOS, garantindo portabilidade e flexibilidade no desenvolvimento.
- **A** → **Apache**: um dos servidores web mais utilizados no mundo, responsável por processar requisições HTTP e entregar páginas web.
- **M** → **MySQL/MariaDB**: sistemas de gerenciamento de banco de dados relacionais, essenciais para armazenar, consultar e manipular informações.
- **P** → **PHP**: linguagem de programação voltada ao desenvolvimento web dinâmico.
- **P** → **Perl**: linguagem complementar presente no pacote, usada principalmente para scripts e tarefas administrativas.

Criado pela organização **Apache Friends**, o XAMPP foi projetado para **simplificar a configuração de ambientes de desenvolvimento**, eliminando a necessidade de instalar e configurar manualmente cada componente. Com apenas

alguns cliques, o desenvolvedor tem acesso a um **ambiente completo**, integrado e pronto para rodar aplicações dinâmicas, simulando um servidor real.

Principais qualidades do XAMPP :

1. **Facilidade de instalação e configuração**
   a. Instalação rápida e intuitiva, sem necessidade de ajustes complexos.
   b. Configuração mínima necessária para começar a desenvolver, tornando-o ideal para iniciantes e estudantes.
2. **Ambiente completo de desenvolvimento**
   a. Reúne **servidor web (Apache)**, **banco de dados (MySQL/MariaDB)** e linguagens de programação (**PHP e Perl**) em um único pacote.
   b. Permite testar aplicações web completas localmente, sem depender de servidores externos.
3. **Portabilidade e compatibilidade**
   a. Funciona em diferentes sistemas operacionais.
   b. Facilita a migração de projetos entre computadores, mantendo a consistência do ambiente de desenvolvimento.
4. **Integração com PHP e SQL**
   a. Já configurado para rodar aplicações PHP que interagem com bancos de dados MySQL/MariaDB.
   b. Permite testar rotinas de back-end, autenticação, operações CRUD e integrações com front-end.
5. **Ferramentas de administração incluídas**
   a. **phpMyAdmin**: interface gráfica para gerenciar bancos de dados, criar tabelas, executar consultas SQL e depurar dados de forma prática.
   b. Suporte a logs de Apache e MySQL, permitindo identificar erros rapidamente.
6. **Gratuito e de código aberto**
   a. Sem custos de licença, sendo acessível para estudantes, pesquisadores e desenvolvedores independentes.
   b. Possibilidade de personalização e estudo do funcionamento interno de cada componente.
7. **Ideal para testes locais e desenvolvimento seguro**
   a. Permite testar funcionalidades de uma aplicação antes de publicá-la em produção.
   b. Evita exposição de dados e vulnerabilidades durante o desenvolvimento.
8. **Suporte a múltiplos projetos simultâneos**

a. É possível configurar múltiplos virtual hosts, permitindo desenvolver e testar diferentes sistemas ao mesmo tempo, no mesmo ambiente.

9. **Flexibilidade e extensibilidade**

a. Permite instalar módulos adicionais, extensões PHP e bibliotecas, tornando o ambiente adaptável às necessidades de cada projeto.

b. Suporte a SSL, envio de e-mails locais e integração com APIs externas, simulando funcionalidades de servidores reais.

10. **Documentação e comunidade ativa**

a. Extensa documentação oficial e tutoriais disponíveis na internet.

b. Grande comunidade de usuários que compartilham soluções, dicas e melhores práticas.

Uso do XAMPP no projeto CineAA :

No projeto **CineAA**, o XAMPP desempenhou papel central como **ambiente de desenvolvimento e testes**, permitindo que todas as funcionalidades do sistema fossem executadas localmente de forma integrada.

Exemplos de utilização:

- **Apache**: processou os arquivos .php do sistema, como **login.php**, **register.php** e os scripts da pasta **api/**.
- **MySQL/MariaDB**: armazenou informações definidas em **database/schema.sql**, como usuários, filmes, amizades, comentários e atividades.
- **phpMyAdmin**: facilitou a administração e depuração do banco de dados, permitindo a execução de queries, inserção de dados de teste e verificação da integridade das tabelas.

A integração entre **Apache + PHP + MySQL** garantiu que funcionalidades essenciais, como **autenticação, cadastro de usuários, exibição do catálogo de filmes e recomendações personalizadas**, fossem implementadas e testadas localmente com rapidez e segurança.

Além disso, o XAMPP permitiu:

- Testar **novas funcionalidades** sem impacto em produção.
- Simular **cenários reais de servidor web**, garantindo que a aplicação se comportasse corretamente em ambientes de produção.
- Acelerar o ciclo de desenvolvimento e depuração, economizando tempo e recursos.

Dessa forma, o **XAMPP** foi uma ferramenta indispensável no projeto CineAA, fornecendo um ambiente seguro, completo e eficiente, essencial para o

desenvolvimento ágil, testes robustos e integração das diversas camadas do sistema.

6.7 PHPMailer

O PHPMailer é uma das bibliotecas mais populares em PHP para o envio de e-mails de forma segura, eficiente e simplificada. Criado originalmente em 2001 por Brent R. Matzelle, o projeto nasceu para superar as limitações da função nativa *mail()* do PHP, que apresenta dificuldades de configuração, baixa confiabilidade e falta de recursos avançados. Com o tempo, o PHPMailer evoluiu para uma solução robusta, amplamente utilizada em aplicações de pequeno, médio e grande porte. Atualmente, é mantido como projeto de código aberto, possuindo uma comunidade ativa que garante sua atualização constante e adequação às boas práticas de segurança (Mailtrap, 2024; Wikipedia, s.d.)..

Um dos principais diferenciais do **PHPMailer** é a sua capacidade de enviar e-mails utilizando servidores SMTP (*Simple Mail Transfer Protocol*). Isso garante maior confiabilidade na entrega e permite integração com provedores populares, como Gmail, Outlook, Yahoo Mail, além de servidores corporativos que utilizam autenticação avançada. Essa característica o torna uma solução profissional, adequada para sistemas que dependem da comunicação direta com o usuário (PHPMailer, 2025).

Funcionalidades e qualidades do PHPMailer

- **Envio via SMTP:** suporta autenticação segura com diferentes servidores de e-mail. Garante maior taxa de entrega e reduz a probabilidade de que mensagens sejam classificadas como spam.

- **Suporte a criptografia:** compatível com protocolos de segurança como SSL e TLS, fundamentais para proteger credenciais e o conteúdo transmitido. Essencial em ambientes que demandam sigilo, como sistemas acadêmicos, corporativos e de comércio eletrônico.

- **Envio de anexos:** permite incluir múltiplos arquivos nos e-mails, como PDFs, imagens, planilhas e documentos. Facilita processos como envio de relatórios, comprovantes ou instruções automatizadas.

- **Formatos de mensagem flexíveis:** suporta envio em texto simples ou em HTML, possibilitando e-mails mais visuais, responsivos e personalizados. Permite a criação de *templates* profissionais, alinhando a comunicação da aplicação com a identidade visual do projeto.

- **Compatibilidade e integração:** fácil integração com sistemas baseados em PHP, incluindo *frameworks* (Laravel, CodeIgniter, Symfony) e bancos de dados. Funciona em conjunto com APIs externas, viabilizando envios em massa ou comunicação personalizada com os usuários.

- **Ampla documentação e comunidade ativa:** disponibiliza exemplos práticos, guias e suporte da comunidade global. Facilita o aprendizado, implementação e manutenção, mesmo por desenvolvedores iniciantes.

- **Substituto robusto para mail():** resolve problemas da função nativa, como dificuldades de configuração, falhas em servidores compartilhados e baixa entrega. Se consolidou como padrão de mercado para envio de e-mails em PHP.

- **Confiabilidade e escalabilidade:** pode ser usado tanto em aplicações simples quanto em sistemas complexos de larga escala. É capaz de lidar com alto volume de mensagens, quando configurado em conjunto com servidores de e-mail otimizados.

- **Boas práticas de segurança:** suporte a autenticação por OAuth 2.0, garantindo maior proteção em integrações com serviços modernos, como Gmail e Outlook. Possibilidade de configurar políticas de envio que dificultam ataques de *spoofing* e *phishing*.

Uso do PHPMailer no projeto CineAA

No projeto **CineAA**, o PHPMailer foi empregado de forma estratégica para garantir a **recuperação de contas de usuários**, agregando confiabilidade e praticidade à aplicação, como demonstrado na **Figura 8**.

Exemplos de utilização incluem:

- No arquivo **recuperar_senha.php**, o PHPMailer foi configurado para enviar um **link de redefinição de senha** diretamente ao e-mail cadastrado do usuário.
- O script gerava automaticamente mensagens em **HTML**, tornando o e-mail visualmente mais amigável e profissional.
- Foram utilizados protocolos seguros via **SMTP com TLS/SSL**, garantindo que as credenciais de login e o conteúdo transmitido não fossem interceptados.

Figura 8 - `recuperar_senha.php`

Fonte: autoria própria.

Além disso, o uso do PHPMailer abriu espaço para futuras funcionalidades de comunicação dentro do CineAA, como:

- Envio de **notificações automáticas**, informando sobre novas recomendações ou interações entre amigos.
- Possibilidade de envio de **boletins informativos (newsletters)** com sugestões de filmes.
- Comunicação personalizada, reforçando a ideia de rede social de filmes.

Importância no contexto do CineAA

O PHPMailer foi crucial para demonstrar **boas práticas de segurança** e para oferecer uma experiência de usuário mais **completa e confiável**. Ao implementar

essa biblioteca, o projeto garantiu que os usuários pudessem **recuperar suas contas de maneira prática e segura**, sem depender de processos manuais.

Dessa forma, o PHPMailer não apenas cumpriu uma função técnica, mas também se destacou como uma ferramenta que **enriqueceu a usabilidade, aumentou a segurança e profissionalizou a comunicação do sistema**.

6.8 Arquitetura MVC

A Arquitetura MVC (*Model-View-Controller*) é um dos padrões de projeto mais influentes e utilizados no desenvolvimento de software moderno. Criada na década de 1970 por Trygve Reenskaug, sua principal contribuição está na separação de responsabilidades dentro de uma aplicação, dividindo-a em três camadas interdependentes, mas bem definidas. Esse modelo traz maior clareza, modularidade e escalabilidade ao sistema, tornando-se especialmente relevante no desenvolvimento de aplicações web. Segundo o artigo *Introdução ao Padrão MVC: Primeiros passos na Arquitetura MVC* da DevMedia (s.d.), esse modelo facilita a manutenção, o reaproveitamento de código e a evolução dos sistemas ao longo do tempo

O funcionamento do MVC é baseado em três componentes principais:

- **Model (Modelo):** é a camada responsável pela lógica de negócios, regras da aplicação e manipulação dos dados. Aqui estão concentradas as operações de armazenamento, consulta, validação e processamento das informações, geralmente vinculadas diretamente ao banco de dados por meio de SQL ou de técnicas modernas como **ORM (Object-Relational Mapping)**.
- **View (Visão):** encarregada da interface com o usuário, ou seja, da apresentação das informações de forma clara, interativa e acessível. Essa camada utiliza HTML, CSS e JavaScript para estruturar e estilizar os dados, mas não contém regras de negócio — sua função é exibir ao usuário os resultados tratados pelo Model e enviados pelo Controller.
- **Controller (Controlador):** atua como o elo de ligação entre o usuário e o sistema. É responsável por receber as requisições (cliques, formulários, URLs), processar os dados recebidos, acionar o Model para realizar operações necessárias e, por fim, devolver a resposta para a View, que a apresenta ao usuário.

Entre as **principais qualidades** da arquitetura MVC, destacam-se:

- **Separação de responsabilidades:** cada camada possui uma função bem definida, o que facilita a compreensão do sistema, reduz dependências e diminui a chance de erros.
- **Organização e clareza:** o código torna-se mais limpo e estruturado, possibilitando que diferentes desenvolvedores trabalhem em paralelo em cada camada.
- **Reutilização e modularidade:** partes da aplicação podem ser reaproveitadas em outros contextos, aumentando a eficiência no desenvolvimento.
- **Escalabilidade:** novas funcionalidades podem ser adicionadas sem comprometer a estrutura já existente.
- **Testabilidade:** a divisão em camadas facilita a criação de testes automatizados, garantindo maior confiabilidade no sistema.
- **Manutenibilidade:** problemas podem ser identificados e corrigidos com maior rapidez, já que o impacto de alterações é restrito a cada camada.
- **Alinhamento com boas práticas de engenharia de software:** o MVC é um padrão consolidado, adotado em frameworks robustos como Laravel, Spring, Django e Rails.

No projeto **CineAA**, a arquitetura MVC foi aplicada para organizar a comunicação entre a interface do usuário e o banco de dados, garantindo um fluxo de informações estruturado e confiável (Figuras 9 e 10). Na prática, sua implementação se deu da seguinte forma:

- **Model:** localizado na pasta *models/*. Entre os principais arquivos, destacam-se:
  - *User.php* – operações relacionadas ao gerenciamento de usuários (cadastro, autenticação e atualização de dados).
  - *Movie.php* – manipulação e manutenção do catálogo de filmes.
  - *Comment.php* – controle das interações entre usuários, como comentários e feedbacks.
- **View:** composta principalmente por arquivos HTML e CSS presentes na raiz do projeto e na pasta *css/*. Exemplos incluem *catalogo.html*, *perfil.html* e *catalogo.css*. Essa camada foi responsável por exibir de forma amigável e organizada as informações recuperadas do sistema.
- **Controller:** implementado em arquivos PHP dentro da pasta *api/*. Exemplos são *login.php*, *register.php*, *movies.php* e *recommendations.php*, que receberam requisições dos usuários, acionaram os modelos correspondentes e enviaram as respostas devidamente processadas ao front-end.

Um exemplo prático de funcionamento foi o processo de login no CineAA:

1. O usuário preenchia o formulário da **View** (HTML/CSS).
2. O **Controller** (*api/login.php*) recebia a requisição e acionava o **Model** (*User.php*) para verificar as credenciais no banco.
3. O resultado era devolvido à **View**, que apresentava uma mensagem de sucesso ou erro e redirecionava o usuário de acordo com o resultado.

Figura 9-models

Fonte: autoria própria.

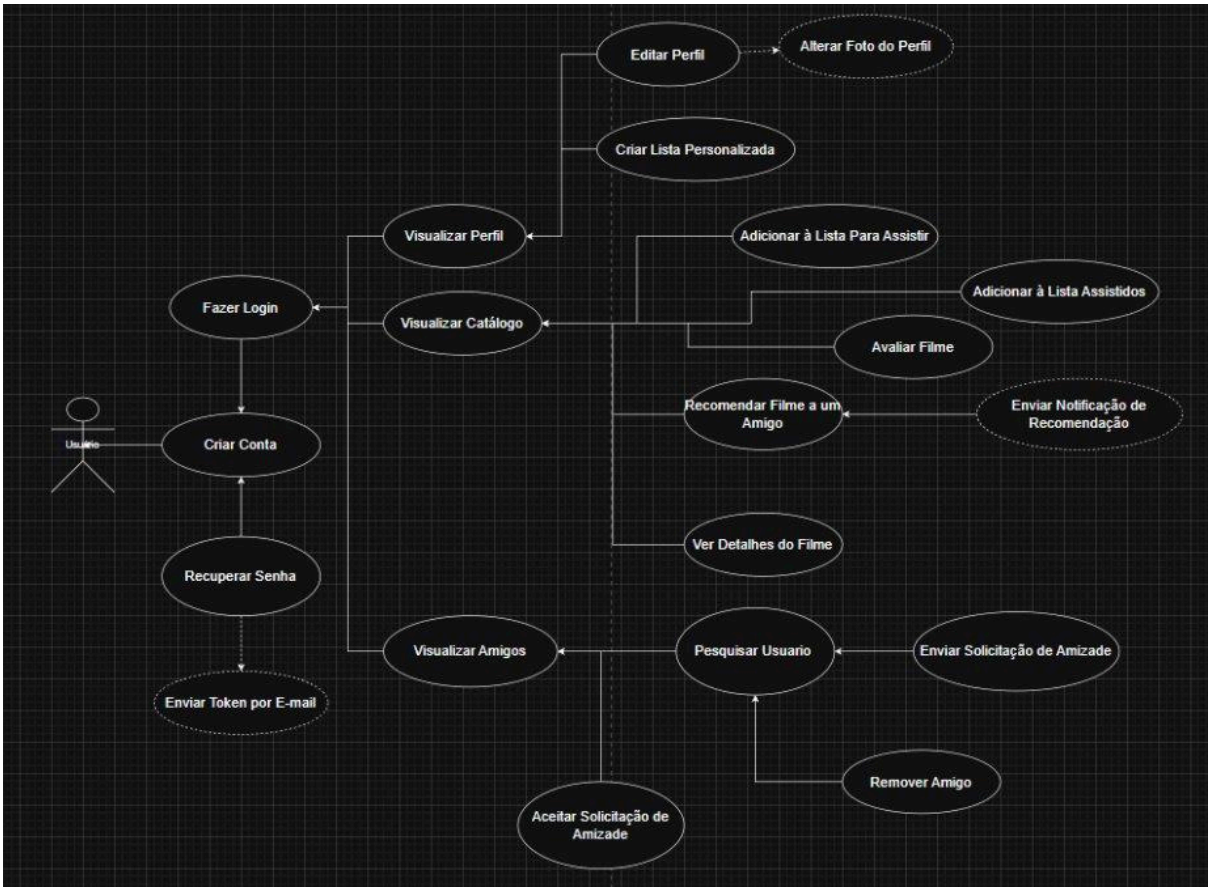
Figura 10-api

Fonte: autoria própria.

Assim, a utilização da arquitetura MVC no CineAA não apenas trouxe **organização e clareza estrutural**, mas também **aproximou o projeto de padrões profissionais**, garantindo escalabilidade, manutenibilidade e robustez. Esse alinhamento com boas práticas da engenharia de software torna o sistema mais confiável, seguro e preparado para evoluções futuras.

7 Fluxograma

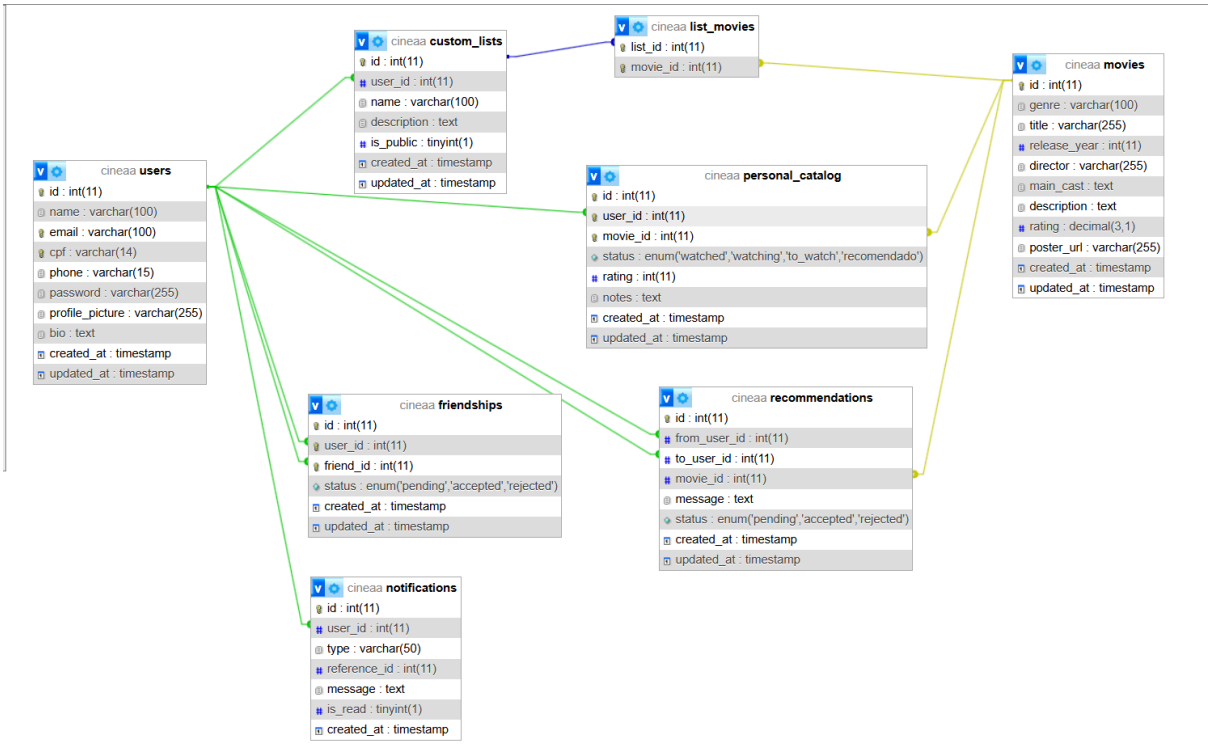
Figura 11-fluxograma



Fonte: autoria própria.

8 Diagrama de domínio

Figura 12-Diagrama de domínio



Fonte: autoria própria.

Referências

BARRA, L. et al. *Aplicações móveis e a sociedade da informação*. São Paulo: Atlas, 2017.

CETIC.BR. *Pesquisa sobre o uso das tecnologias de informação e comunicação no Brasil – TIC Domicílios 2023*. Disponível em: https://cetic.br. Acesso em: 22 out. 2025.

GOMES, R.; LIMA, P. *Comportamento do consumidor de streaming no Brasil: tendências e preferências*. *Revista de Mídia e Sociedade*, v. 9, n. 2, p. 45–59, 2022.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de Software: uma abordagem profissional*. 9. ed. Porto Alegre: AMGH, 2020.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de Software: uma abordagem profissional*. 9. ed. Porto Alegre: AMGH, 2020.

W3C – World Wide Web Consortium. *HTML5 Recommendation*. 2018. Disponível em: https://www.w3.org/TR/html5/. Acesso em: 22 out. 2025.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de Software: uma abordagem profissional*. 9. ed. Porto Alegre: AMGH, 2020.

W3C – World Wide Web Consortium. *HTML5 Recommendation*. 2018. Disponível em: https://www.w3.org/TR/html5/. Acesso em: 22 out. 2025.

SOMMERVILLE, I. *Engenharia de Software*. 10. ed. São Paulo: Pearson, 2019.

ELMASRI, R.; NAVATHE, S. B. *Sistemas de Banco de Dados*. 7. ed. São Paulo: Pearson, 2016.

DATE, C. J. *Introdução a Sistemas de Bancos de Dados*. 8. ed. Rio de Janeiro: Elsevier, 2004.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S. *Database System Concepts*. 7th ed. New York: McGraw-Hill, 2019.

PRESSMAN, R. S.; MAXIM, B. R. *Engenharia de Software: uma abordagem profissional*. 9. ed. Porto Alegre: AMGH, 2020.

WHATWG. *HTML Living Standard*. Disponível em: https://html.spec.whatwg.org/multipage/. Acesso em: 12 set. 2025.

W3C. *HTML5: A vocabulary and associated APIs for HTML and XHTML*. W3C Recommendation, 2014. Disponível em: https://www.w3.org/TR/html5/. Acesso em: 18 set. 2025

WORLD WIDE WEB CONSORTIUM (W3C). *CSS Snapshot 2025*. Nota do Grupo. 09 set. 2025. Disponível em: https://www.w3.org/TR/css-2025/. Acesso em: 15 set. 2025.

W3C. *Cascading Style Sheets (CSS) Snapshot 2018*. W3C Working Group Note, 2018. Disponível em: https://www.w3.org/TR/css-2018/. Acesso em: 18 set. 2025.

MOZILLA DEVELOPER NETWORK. *CSS: Cascading Style Sheets*. MDN Web Docs, 2025. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/CSS. Acesso em: 18 set. 2025.

MOZILLA FOUNDATION. *JavaScript*. Disponível em:https://developer.mozilla.org/pt-BR/docs/Web/JavaScript. Acesso em: 15 set. 2025.

ECMA INTERNATIONAL. *ECMAScript 2023 Language Specification*. ECMA-262, 14th edition, jun. 2023. Disponível em: https://262.ecma-international.org/. Acesso em: 18 set. 2025.

MOZILLA DEVELOPER NETWORK. *JavaScript*. MDN Web Docs, 2025. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript. Acesso em: 18 set. 2025

HP MANUAL. *Histórico do PHP*. Disponível em: https://www.php.net/manual/en/history.php.php. Acesso em: 15 set. 2025.

LERDORF, R.; GUTMANS, Z.; SURASKI, A. *Programming PHP*. 2. ed. Sebastopol: O'Reilly Media, 2002.

PHP GROUP. *PHP Manual*. 2025. Disponível em: https://www.php.net/manual/pt_BR/. Acesso em: 18 set. 2025.

ELMASRI, R.; NAVATHE, S. *Introdução a sistemas de bancos de dados*. 8. ed. tradução da 8ª edição americana. São Paulo: Pearson, 2016.

GARCIA, Guilherme. *XAMPP: O que é, Como Funciona, Vantagens e Instalação da Ferramenta*. Mercado Online Digital, 22 maio 2024. Disponível em: https://mercadoonlinedigital.com/blog/xampp/. Acesso em: 15 set. 2025.

MAILTRAP. *PHPMailer Guide: Configuration, SMTP Setup, and Email Sending*. Mailtrap Blog, 22 mar. 2024. Disponível em: https://mailtrap.io/blog/phpmailer/. Acesso em: 15 set. 2025.

WIKIPEDIA. *PHPMailer*. Disponível em: https://en.wikipedia.org/wiki/PHPMailer. Acesso em: 15 set. 2025.

PHPMailer. *PHPMailer Documentation*. 2025. Disponível em: https://github.com/PHPMailer/PHPMailer. Acesso em: 18 set. 2025.

DEVMEDIA. *Introdução ao Padrão MVC: Primeiros passos na Arquitetura MVC*. Disponível em: https://www.devmedia.com.br/introducao-ao-padrao-mvc/29308. Acesso em: 15 set. 2025.

Pontifícia Universidade Católica de Goiás
Pontifical Catholic University of Goias
Av. Universitária, 1069, Setor Universitário
Caixa Postal 86 - CEP 74.605-010
Goiânia - Goiás - Brasil

RESOLUÇÃO n° 038/2020 – CEPE

ANEXO I

APÊNDICE ao TCC

Termo de autorização de publicação de produção acadêmica

O estudante ALLAN VITOR CORREIA OLIVEIRA do Curso de Ciênia da Computação matrícula 2020.1.0028.0044-6, telefone: (62)98420-9333 e-mail allanvitorcorreia20@gmail.com, na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do Autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado Sistema Gestor de filmes e Indicações, gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto(PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 30 de setembro de 2025.

gov.br Documento assinado digitalmente
ALLAN VITOR CORREIA OLIVEIRA
Data: 01/10/2025 10:03:17-0300
Verifique em https://validar.iti.gov.br

Assinatura do autor: _____

Nome completo do autor: ALLAN VITOR CORREIA OLIVEIRA

gov.br Documento assinado digitalmente
EUGENIO JULIO MESSALA CANDIDO CARVALHO
Data: 01/10/2025 10:42:47-0300
Verifique em https://validar.iti.gov.br

Assinatura do professor-orientador: _____
Nome completo do professor-orientador: