



ESCOLA SENAI “A. JACOB LAFER”
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

ANA LAUREN DOURADO PEREIRA
ISABELA PAIOLA
LAURA ARAUJO DUTRA DOS SANTOS
MARIA EDUARDA DOS SANTOS ROSA
PEDRO HENRIQUE DA SILVA RODRIGUES
YASMIN LOPES BORBA

PROJETO 3º TERMO - Streaming de filmes, uma aplicação web que permita a locação de filmes, séries, novelas e desenhos; podendo se logar como administrador e usuário.

SANTO ANDRÉ

2025

ANA LAUREN DOURADO PEREIRA
ISABELA PAIOLA
LAURA ARAUJO DUTRA DOS SANTOS
MARIA EDUARDA DOS SANTOS ROSA
PEDRO HENRIQUE DA SILVA RODRIGUES
YASMIN LOPES BORBA

PROJETO 3º TERMO - Streaming de filmes uma
aplicação web que permita a locação de filmes, séries, novelas e
desenhos; podendo se logar como administrador e usuário.

Trabalho apresentado ao
curso Técnico em Desenvolvimento de
Sistemas da escola SENAI A. Jacob
Lafer

Orientador: Professor Raul e
Professora Nadja.

SANTO ANDRÉ
2025

AGRADECIMENTO

A escola SENAI, pelo fornecimento de materiais que foram fundamentais para a realização da pesquisa.

Ao Professor Raul o mago e Professora Nadja, pela excelente orientação e ensinamentos.

Aos colegas da turma, pelo companheirismo ao longo deste trabalho.

A todos aqueles que contribuíram para a realização do trabalho.

RESUMO

O projeto visa desenvolver um sistema de locação inovador para plataformas de streaming, alinhado às novas tendências do mercado de locação digital. A proposta é criar uma solução robusta e escalável de gestão de conteúdo, que se adapte às necessidades dinâmicas dos serviços de streaming e ofereça uma experiência personalizada, eficiente e altamente interativa. A ideia central do sistema é otimizar o processo de acesso e gerenciamento de conteúdo, permitindo que um único usuário utilize a plataforma de forma ágil e segura, por meio de métodos modernos de autenticação. Embora o registro de múltiplos usuários não esteja previsto, essa funcionalidade poderá ser implementada futuramente, caso o cliente deseje. Os administradores terão controle completo sobre o sistema, incluindo a gestão de conteúdo e a configuração da plataforma, garantindo uma experiência fluida e sem fricções para o usuário.

Palavras-chave: Streaming 1. Locação 2. Filmes 3. Usuários 4. Administrador 5.

Summary

The project aims to develop an innovative registration system for streaming platforms, aligned with new trends in the digital rental market. The proposal is to create a robust and scalable user and content management solution that adapts to the dynamic needs of streaming services, offering a personalized, efficient and highly interactive experience for customers.

The central idea of the system is to optimize the registration and management process, allowing users to register quickly and securely, using modern authentication methods. This flexibility will provide a frictionless onboarding experience, allowing users to quickly access the desired content.

Keywords: streaming 1. Location 2. Movies 3. Users 4. Administrator 5.

Sumário

1. INTRODUÇÃO.....	8
1.1 Justificativa	8
2.1 OBJETIVOS	9
3. LEVANTAMENTO DE REQUISITOS	9
3.2 Requisitos não Funcionais:	11
4. Desenvolvimento	12
4.1 Solução Inicial	12
4.2 Scrum e Sprint	12
5. Protótipo.....	14
6. Cronograma	27
7. Kanban.....	29
8. CONCLUSÃO.....	30
9. Referências	31
10. APÊNDICE	33
10.1 Página mobile	33
11 CÓDIGOS.....	48
11.2 DATA.....	49
12 MODELS	51
12.1 desenh.php.....	51
12.2 filme.php	52
12.3 item.php	53
12.4 novela.php.....	55
12.5 serie.php.....	56
13 NODE MODULES	57
13.1 public	57
13.2 services	70
13.3 views.....	79

14 json.sprint.....	108
14.1 filmes.php.....	108
14.2 salvar.php.....	109
14.3 usuarios.php.....	111

1. INTRODUÇÃO

Este trabalho tem como objetivo apresentar o CineHome, um serviço de streaming de filmes e séries, focado na locação de conteúdos audiovisuais via plataforma online. Inspirado em grandes players do mercado, como Amazon Prime Video e Netflix, que popularizaram o consumo sob demanda de conteúdo audiovisual SILVA, João. *O impacto dos serviços de streaming no consumo audiovisual*. São Paulo: Editora Digital, 2021., o CineHome oferece uma alternativa baseada na locação individual ou por pacotes. O CineHome oferece aos usuários a possibilidade de acessar uma ampla gama de títulos, mediante o pagamento de uma locação por título ou pacote. A plataforma visa proporcionar uma experiência personalizada, com interface amigável, recomendação inteligente de conteúdo e múltiplas opções de pagamento. Este projeto busca atender às demandas de consumidores que desejam mais flexibilidade no consumo de conteúdos audiovisuais, sem a necessidade de assinatura mensal.

1.1 Justificativa

O mercado de streaming tem apresentado um crescimento exponencial nos últimos anos, impulsionado pela crescente demanda por conteúdo digital acessível a qualquer hora e em qualquer lugar. De acordo com o estudo *Inside Video 2024*, produzido pela Kantar IBOPE Media, em 2023, o consumo de vídeo em diferentes formatos alcançou 99,63% da população brasileira, evidenciando a ampla penetração desse tipo de conteúdo no país. A transformação digital e a mudança nos hábitos de consumo de mídia, especialmente nos contextos de entretenimento, educação e cultura, tornam imprescindível a criação de soluções inovadoras que atendam às necessidades de um público cada vez mais exigente e conectado. Segundo o artigo "Transformação digital: como os hábitos de consumo mudaram no Brasil", a digitalização crescente tem remodelado profundamente os hábitos de consumo no país, com consumidores cada vez mais conectados e exigindo experiências de compra integradas e personalizadas.

Além disso, no campo educacional, a cultura digital tem revolucionado

não apenas o que se aprende, mas também como se aprende, trazendo novas possibilidades para o processo de ensino-aprendizagem Espaço Virtual.

Essas mudanças destacam a importância de desenvolver soluções

inovadoras que atendam às demandas de um público cada vez mais exigente e conectado.

Nesse cenário, o projeto de locação de streaming surge como uma

resposta estratégica para atender a esse novo padrão de consumo, oferecendo uma plataforma dinâmica e escalável que permita a locação de conteúdos digitais de maneira simples, acessível e personalizada.

2.1 OBJETIVOS

O objetivo deste projeto de locação de streaming é criar (desenvolver/proporcionar) uma plataforma digital intuitiva e eficiente para o aluguel de filmes, séries, desenhos e novelas, proporcionando uma experiência de entretenimento personalizada e acessível. O sistema visa atender tanto usuários quanto administradores, permitindo que os primeiros possam explorar e alugar conteúdo multimídia de forma prática, enquanto os administradores gerenciam o catálogo, a disponibilidade dos itens e os processos de locação. A plataforma também busca oferecer recursos como cálculo automático de preços de aluguel, gerenciamento de assinaturas e histórico de locações, criando um ambiente digital dinâmico e simplificado para o público em geral.

3. LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos é uma fase fundamental no processo de desenvolvimento de sistemas, que tem como objetivo identificar, documentar e entender as necessidades e expectativas dos stakeholders (partes interessadas) em relação ao sistema ou produto a ser desenvolvido. Ele serve para garantir que o projeto atenda aos objetivos do cliente e aos requisitos do

usuário, evitando falhas de trabalho e mal-entendidos durante a execução do projeto. O Termo de Abertura do Projeto (TAP) é essencial para garantir que o projeto atenda aos objetivos do cliente e aos requisitos do usuário, evitando falhas de trabalho e mal-entendidos durante a execução do projeto.

3.1 Requisitos Funcionais:

Um requisito funcional é uma especificação que descreve o que um sistema ou software deve fazer, ou seja, as funcionalidades que ele deve oferecer. Ele define os comportamentos, ações e operações que o sistema precisa realizar para atender às necessidades dos usuários e cumprir com os objetivos do projeto. Exemplos incluem: cadastro de usuários, processamento de pagamentos ou envio de e-mails automáticos.

- A aplicação deve permitir que usuários façam login com nome de usuário e senha.
- Cadastro de novos usuários deve ser permitido com validação de dados (nome, e-mail, senha).

Página de Administradores.

- Administradores devem ser autenticados para acessar a área administrativa.
- Eles devem poder gerenciar filmes, séries e desenhos: cadastrar, editar e remover.
- Visualização dos registros de aluguel e transações.

Página de Usuários

- Usuários devem poder navegar e pesquisar conteúdos no catálogo.
- Aluguel de filmes, séries e desenhos com escolha de duração e pagamento.
- Utilizar arquivos JSON (usuários.json, itens.json) para armazenar informações dos usuários e dos conteúdos Autenticação e Autorização.
- Sistema de login com autenticação básica (usuário/senha).

- Controle de acesso com base no perfil de administrador ou usuário comum.

3.2 Requisitos não Funcionais:

O levantamento de requisitos não funcionais refere-se à identificação e documentação das características e qualidades do sistema que não estão diretamente relacionadas às funcionalidades específicas que ele deve realizar, mas que são essenciais para seu desempenho, usabilidade, segurança e outros aspectos de qualidade. Enquanto os requisitos funcionais descrevem o que o sistema deve fazer, os requisitos não funcionais descrevem como o sistema deve se comportar e as condições que ele deve atender.

Desempenho

- O sistema deve responder em até 3 segundos para qualquer requisição.
- O streaming de vídeos deve ser eficiente e não ter interrupções.

Usabilidade

- A interface deve ser intuitiva, com navegação clara.
- O layout deve ser responsivo para funcionar bem em dispositivos móveis e desktops.

Segurança

- As senhas devem ser criptografadas antes de serem armazenadas.
- Validação de dados de entrada para evitar injeções de SQL ou outros tipos de ataques.
- Autenticação básica (nome de usuário + senha) e controle de acesso.

Qualidade do Código

- O código deve ser bem estruturado, modulado e comentado para facilitar a manutenção.

- A aplicação deve seguir boas práticas de desenvolvimento, como o uso de padrões de projeto quando necessário.

4. Desenvolvimento

A metodologia para o desenvolvimento com o levantamento de requisitos por meio de reuniões com as partes interessadas, identificando suas necessidades. Em seguida, realizamos uma análise e planejamento detalhados, priorizando funcionalidades e definindo cronogramas. Na etapa de desenho da arquitetura, estabelecemos a estrutura do banco de dados e escolhemos as tecnologias apropriadas, criando Wireframes para as interfaces. O desenvolvimento ocorreu em sprints ágeis, com entregas incrementais e feedback contínuo. Essa metodologia garantiu uma solução eficaz e alinhada.

4.1 Solução Inicial

A solução inicial para o projeto de locação de streaming consiste em desenvolver uma plataforma web moderna e intuitiva, com foco na experiência do usuário e na gestão eficiente do catálogo de filmes. A plataforma será dividida em duas interfaces principais: uma para os administradores e outra para os usuários.

Para os administradores, o sistema permitirá a adição, edição e gerenciamento de filmes, séries, novelas e desenhos no catálogo, além de permitir o controle da disponibilidade e cálculo dos valores de locação.

Para os usuários, a plataforma oferece um catálogo de fácil navegação, onde será possível buscar e filtrar conteúdos, calcular o preço do aluguel e realizar a locação diretamente. A solução também incluirá funcionalidades para o gerenciamento de assinaturas e histórico de alugueis.

4.2 Scrum e Sprint

O Scrum é uma estrutura ágil que ajuda equipes a organizar o trabalho em ciclos curtos de desenvolvimento, chamados de sprints. A equipe de Scrum se compromete a lançar o trabalho no final de cada sprint e adotam práticas e estruturas para atingir esse ritmo.

Durante a Sprint 1, foram discutidas e planejadas as funcionalidades iniciais do sistema, como o cadastro de usuários, a interface de busca e a seleção de filmes e séries. Também foi feito o protótipo de alta fidelidade, utilizando o Figma.

A Sprint 2 do projeto foi focada no desenvolvimento do sistema de locadora, com ênfase na criação do frontend responsivo e interativo, baseado no protótipo definido anteriormente. Utilizando tecnologias como HTML, CSS, JavaScript (ou framework escolhido), foram implementadas as principais funcionalidades, como autenticação de usuários (com dois perfis distintos), gestão de itens (adicionar, excluir, alugar e devolver), e cálculo de valor estimado do aluguel. Os dados foram armazenados em arquivos JSON, com validações de segurança aplicadas. Ao final, a aplicação foi testada para garantir o bom funcionamento.

Na Sprint 3, foram concluídas etapas essenciais para a consolidação do projeto. A equipe concentrou esforços em três frentes principais:

- **Elaboração do backend com o website completo:** Nesta fase, a estrutura do backend foi totalmente desenvolvida, integrando-se ao frontend do site de maneira funcional e estável. Isso incluiu a configuração de rotas, banco de dados, autenticação de usuários, e demais funcionalidades dinâmicas necessárias para que o site operasse de forma completa. O sistema passou a ter suporte para interações reais, armazenamento e manipulação de dados, proporcionando uma experiência mais robusta para o usuário.
- **Entrega do relatório técnico (padrão ABNT):** Foi elaborado e entregue o relatório técnico do projeto, seguindo as normas da ABNT. O documento detalha todas as etapas do desenvolvimento, desde a concepção da ideia até a implementação técnica, incluindo cronograma, tecnologias utilizadas, divisão de tarefas, metodologia aplicada, e os resultados obtidos até o momento.
- **Apresentação do projeto (Canva):** Também foi preparada uma apresentação visual para exposição do projeto. O material apresenta os principais pontos do trabalho, como objetivos, funcionalidades implementadas, interface desenvolvida e perspectivas futuras. A apresentação tem como objetivo comunicar de forma clara e visual o andamento e os resultados do projeto,

podendo ser utilizada em reuniões, bancas avaliadoras ou demonstrações públicas.

5. Protótipo

O protótipo da aplicação web para o gerenciamento de locação de filmes facilita o aluguel de filmes, séries, desenhos e novelas, permitindo que tanto usuários quanto administradores façam login na plataforma. O administrador pode adicionar filmes ao catálogo, gerenciar a disponibilidade e calcular o valor do aluguel, enquanto o usuário pode calcular o preço do aluguel, alugar filmes e acessar uma página inicial com suas assinaturas e histórico de aluguéis. Além disso, há uma aba de catálogo onde o usuário pode explorar filmes, séries, novelas e desenhos para locação.

Imagem 1 - Logo

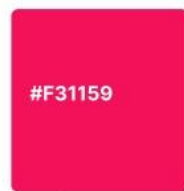


Fonte: Autoria própria (2025).

Na figura 1 a logo da CineHome foi projetada para refletir a experiência de assistir filmes em casa. O sofá foi escolhido como elemento visual, simbolizando o ambiente acolhedor e familiar. O vermelho foi destacado, associado ao cinema, para transmitir emoções intensas, como paixão e emoção, características comumente exploradas nas narrativas cinematográficas. Com essa logo queríamos passar a mensagem de conforto com elementos que representam o cinema em nossa casa.

Imagem 2 – Guia de cores

Cores



Primária -1



Primária



Primária +1



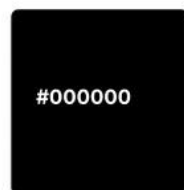
Secundária -1



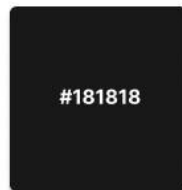
Secundária



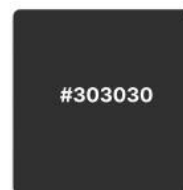
Secundária +1



Dark -1



Dark



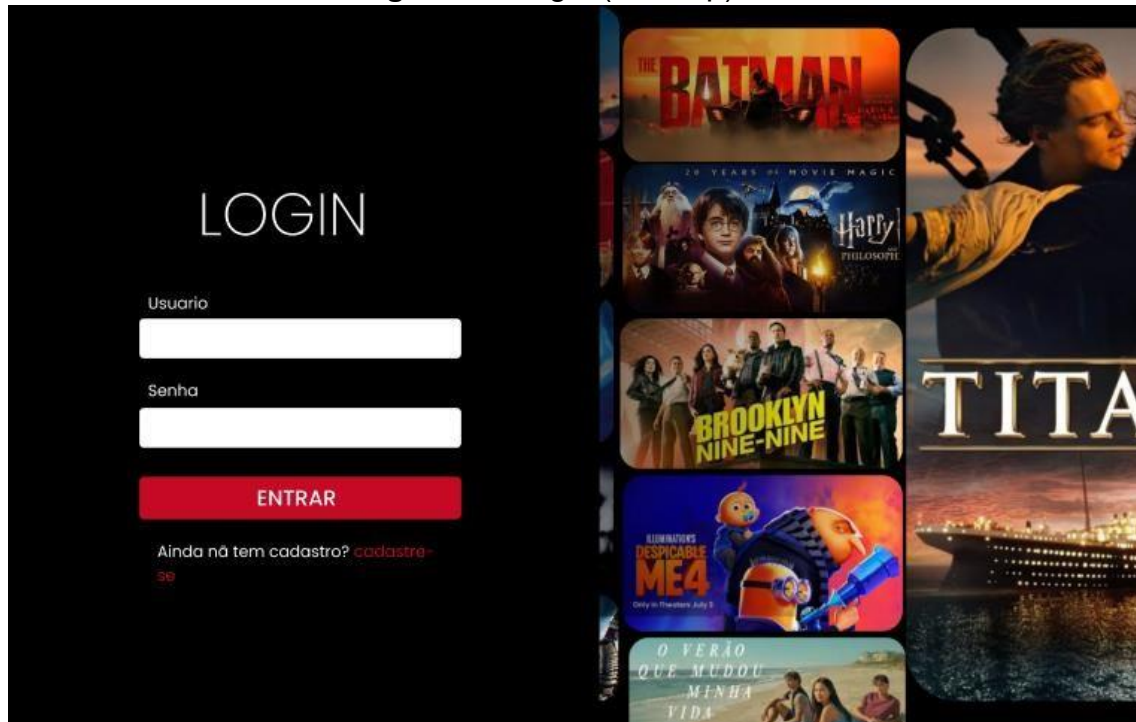
Dark +1



Fonte: Autoria própria (2025).

A figura 2 é a Tipografia é a arte e técnica de organizar textos de forma visualmente harmoniosa e funcional. Ela envolve a escolha e o uso adequado de fontes, tamanhos, espaçamentos, alinhamentos e hierarquias, com o objetivo de tornar a leitura mais agradável e a comunicação mais eficaz. Uma boa tipografia não apenas transmite informação, mas também reforça a identidade visual e a estética de um projeto.

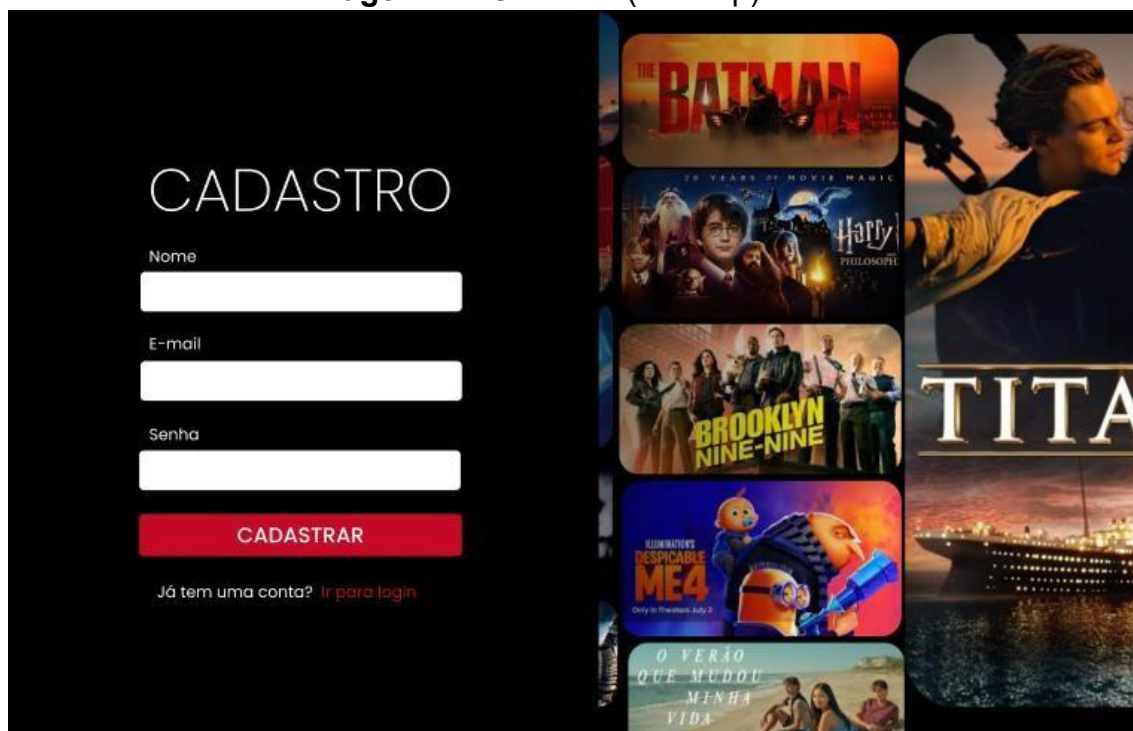
Imagem 3 – Login (desktop)



Fonte: Autoria própria (2025).

A figura 3 mostra a página de login do site CineHome, com o formulário de login à esquerda em fundo escuro e estilizado. Inclui campos para usuário e senha, um botão vermelho “ENTRAR” e um link dizendo “Ainda não tem cadastro? cadastre-se”. À direita, há uma colagem com capas de filmes e séries, destacando o tema de cinema e entretenimento do site.

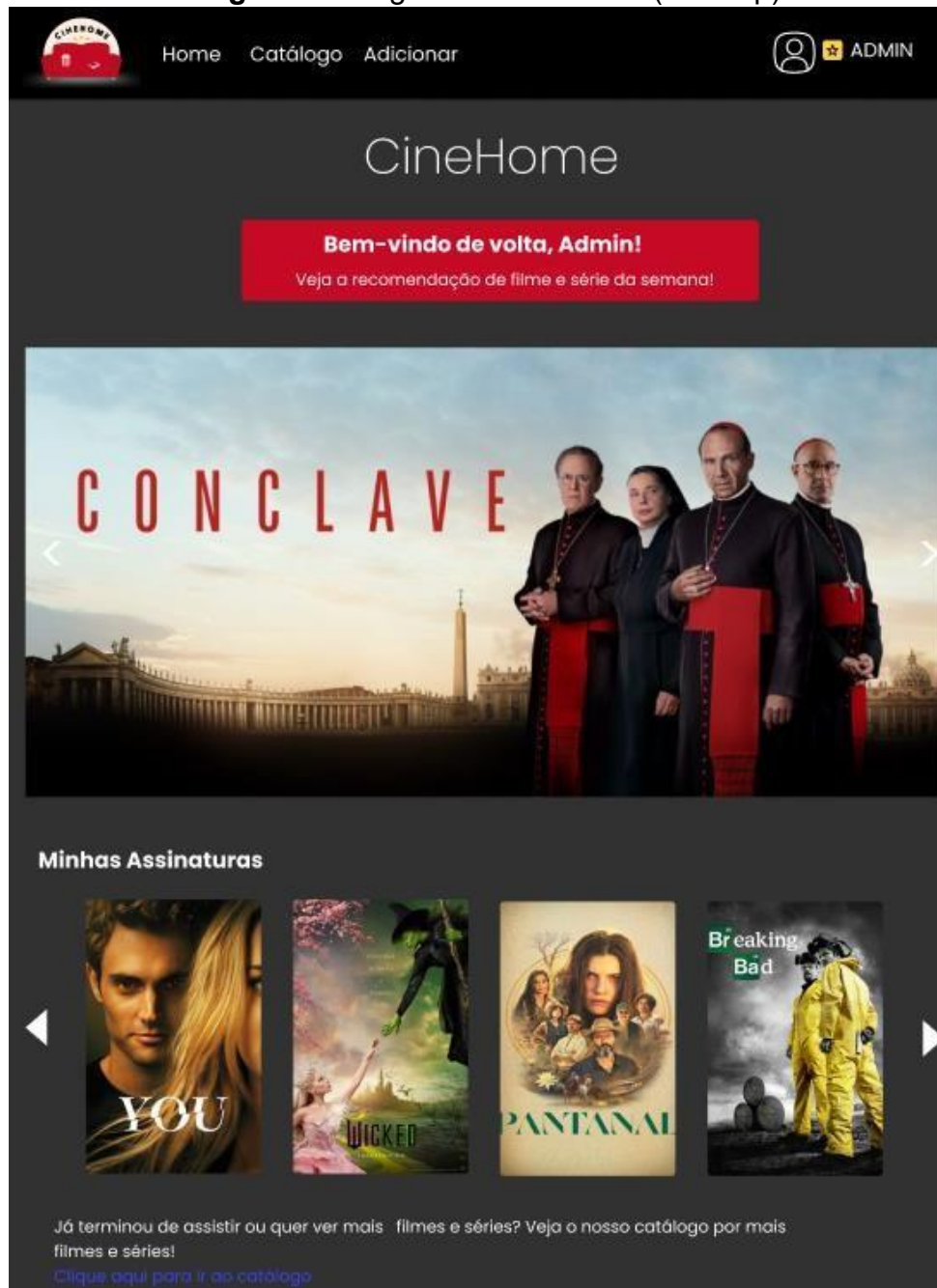
Imagem 4 – Cadastro (desktop)



Fonte: Autoria própria (2025).

A figura 4 mostra a página de cadastro do site CineHome, com um design escuro e moderno. À esquerda, há um formulário com o título “CADASTRO”, seguido pelos campos Nome, E-mail e Senha. Abaixo, há um botão vermelho escrito “CADASTRAR”. No fim do formulário, aparece a frase “Já tem uma conta? Ir para login”, com um link em vermelho que leva à página de login. À direita, a imagem de fundo exibe capas de filmes e séries conhecidas, destacando o tema de entretenimento do site.

Imagem 5 – Página administrador (desktop)



Adicionar

Adicionar ao catalogo

Título

Sinopse

Genêro

Tipo

Adicionar poster do filme

Adicionar

Calcular Preço

Tipo

Dias

Calcular



Como Treinar Seu Dragão

Filme | Animação | Aventura

1

Alugar

Disponível

Editar

Deletar



Dexter

Serie | Drama

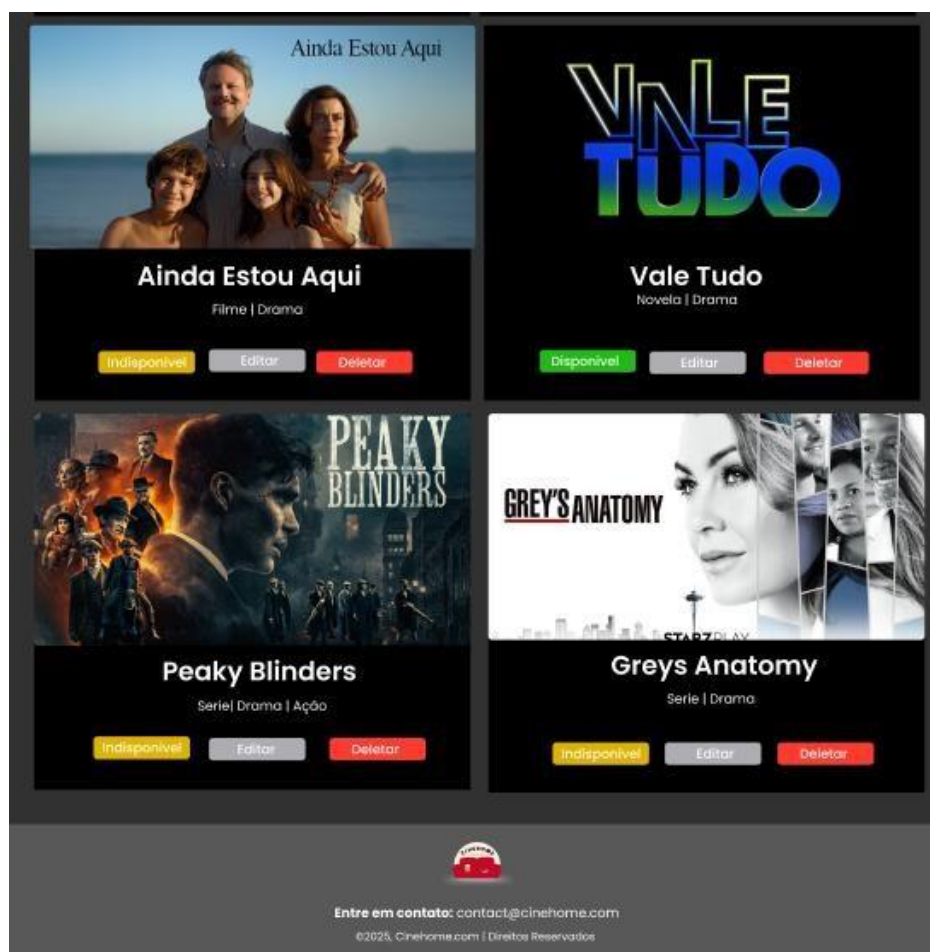
1

Alugar

Disponível

Editar

Deletar



Fonte: Autoria própria (2025).

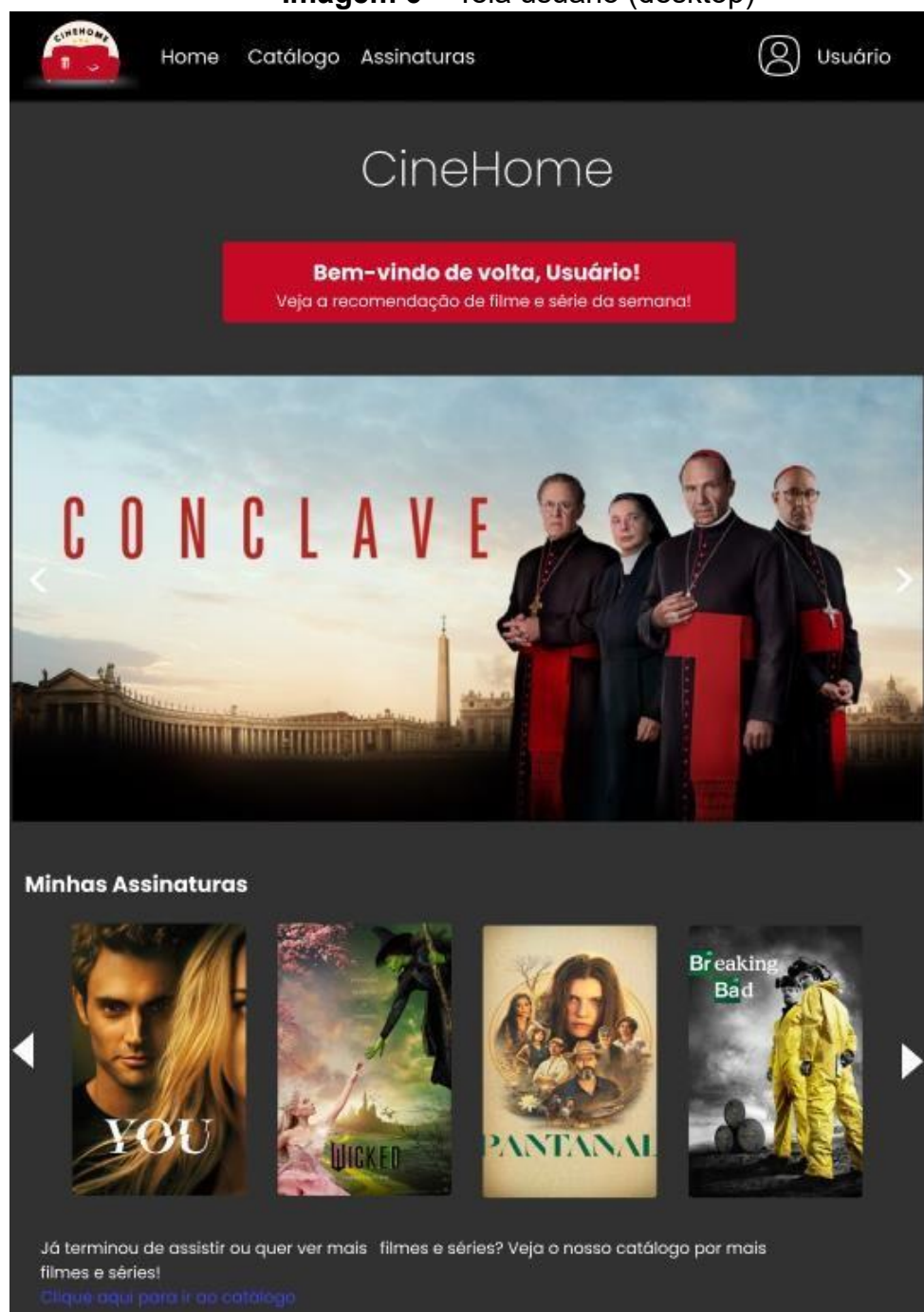
A figura 5 mostra a página do CineHome é uma interface administrativa para gerenciar um catálogo de filmes, séries, novelas e desenhos. No topo, há um menu de navegação com links para "Home", "Catálogo" e "Adicionar", além da identificação do usuário logado como "Admin".

Ela exibe uma mensagem de boas-vindas, seguida por um carrossel de destaque com banners de mídia. A seção "Minhas Assinaturas" mostra um carrossel com pôsteres das produções favoritas ou assistidas recentemente.

A área de administração inclui um formulário para adicionar novos títulos e uma calculadora para preço de aluguel. Abaixo, há uma grade com cards das mídias cadastradas, contendo imagem, título, tipo, gênero e botões de ações como "Alugar", "Editar", "Deletar" e status de disponibilidade.

Por fim, no rodapé da página, há o logo da CineHome, informações de contato por e-mail (contact@cinhome.com), e a nota de direitos autorais:

Imagem 6 – Tela usuário (desktop)





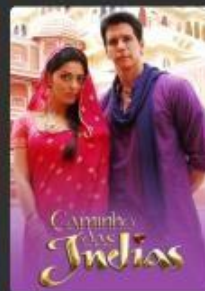
Filmes



Séries



Novelas



Desenhos



Previsão

Calcular Preço

Tipo

Dias

Calcular



Como Treinar Seu Dragão

Filme | Animação | Aventura

Disponível

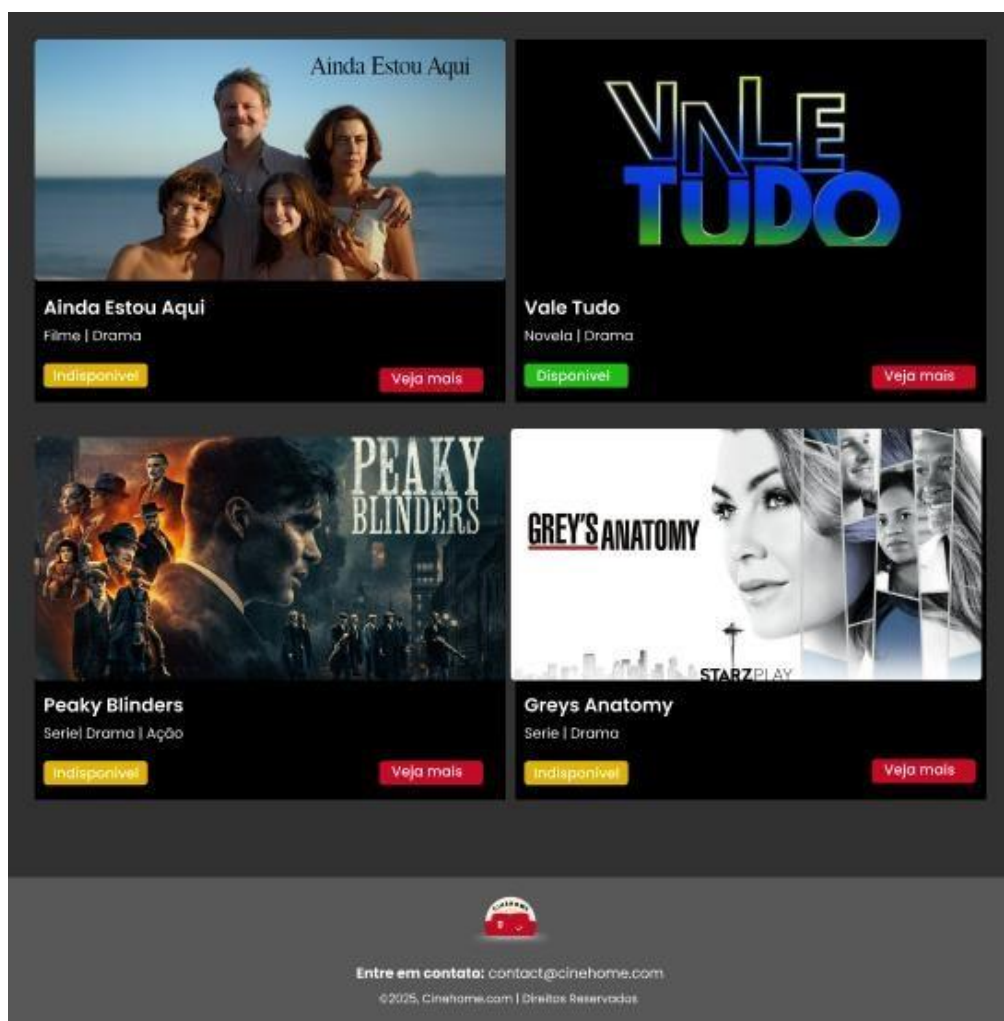


Dexter

Série | Drama

Disponível

[Veja mais](#)



Fonte: Autoria própria (2025).

A figura 6 mostra a página principal da plataforma CineHome apresenta uma interface moderna com fundo escuro e destaques em vermelho. No topo, há um cabeçalho com o logotipo da marca à esquerda, seguido pelo menu de navegação com as opções "Home", "Catálogo" e "Assinaturas", além do ícone de usuário no canto direito.

Logo abaixo, há uma mensagem de boas-vindas personalizada ao usuário, destacada em um banner vermelho, incentivando-o a ver a recomendação da semana. Em seguida, há um banner principal em formato de carrossel exibindo a recomendação em destaque.

A seção "Minhas Assinaturas" apresenta um carrossel com os títulos já acessados ou favoritos do usuário.

Mais abaixo, a página é dividida em seções por categoria de conteúdo: Filmes, Séries, Novelas e Desenhos. Cada categoria é exibida em formato de carrossel horizontal, com títulos populares.

A seção "Previsão" permite ao usuário calcular o preço de uma assinatura com base no tipo de conteúdo e no número de dias desejados. Há campos para preenchimento e um botão de "Calcular".

Na parte inferior, a plataforma destaca alguns títulos com informações sobre disponibilidade. Filmes e séries são apresentados com botões indicando se estão "Disponíveis", "Indisponíveis" ou para "Ver mais".

Por fim, no rodapé da página, há o logo da CineHome, informações de contato por e-mail (contact@cinehome.com), e a nota de direitos autorais: ©2025, Cinehome.com | Direitos Reservados.

6. Cronograma

Um cronograma é uma ferramenta usada para organizar tarefas em um determinado período. Ele ajuda a planejar o que deve ser feito, quando e em qual ordem. É muito útil para garantir que tudo seja realizado no prazo, seja em trabalhos escolares, projetos profissionais ou eventos.

Imagem 7 – Cronograma

Cronograma- Sprint 3

Scrum Master: Isabela Paiola

ATIVIDADE	RESPONSÁVEL	DATA DE INÍCIO	DATA FIM	STATUS
Planejamento do Cronograma de Desenvolvimento	Ana Lauren	12/05/2025	12/05/2025	 Concluído
Melhorias do Frontend (prototipagem e codagem)	(Equipe de designers) Yasmin Lopes, Maria Eduarda, Isabela Paiola, Ana Lauren.	28/04/2025	12/05/2025	 Concluído
Elaboração do Backend com o website completo	(Equipe de desenvolvedores) Yasmin Lopes, Maria Eduarda, Isabela Paiola, Pedro Henrique.	29/04/2025	20/05/2025	 Concluído
Desenvolvimento dos Slides de Apresentação	(Equipe de designers) Laura Dutra, Maria Eduarda, Ana Lauren, Yasmim Lopes.	12/04/2025	20/05/2025	 Concluído
Atualização do Kanban (organização e acompanhamento)	Laura Dutra.	29/04/2025	29/04/2025	 Concluído
Documentação do Sistema (organização e acompanhamento)	Isabela Paiola, Ana Lauren.	29/04/2025	20/05/2025	 Concluído

Fonte: Autoria própria (2025).

A figura 7 mostra um cronograma de atividades relacionadas ao desenvolvimento de um projeto.

7. Kanban

Kanban é um método de organização visual usado para controlar tarefas e melhorar a produtividade. Ele usa quadros divididos em colunas, como “A Fazer”, “Em Andamento” e “Concluído”. Assim, é possível acompanhar o andamento do trabalho de forma clara e prática. É muito usado em empresas, escolas e projetos pessoais.

Imagem 8 – Kanban



Fonte: Autoria própria (2025).

A figura 8 mostra um quadro Kanban da plataforma Trello, utilizado para o gerenciamento de tarefas do projeto CineHome. O quadro está dividido em três colunas, chamadas de "Sprints", que representam diferentes fases do projeto.

8. CONCLUSÃO

O projeto CineHome surgiu como uma resposta às mudanças no comportamento do consumidor digital, que valoriza cada vez mais a flexibilidade, a personalização e a acessibilidade no consumo de conteúdos audiovisuais. Com base em referências consolidadas do mercado de streaming, como Amazon Prime Video e Netflix, a proposta da plataforma se diferencia ao oferecer um modelo baseado em locação avulsa ou por pacotes, sem a obrigatoriedade de assinatura mensal.

Durante o desenvolvimento, foram levantados e organizados requisitos funcionais e não funcionais que garantem a eficiência, segurança e usabilidade do sistema. A metodologia ágil Scrum, aplicada em sprints, permitiu uma abordagem iterativa, com entregas incrementais, testes constantes e validações contínuas, promovendo uma evolução coerente do projeto. Além disso, o uso de ferramentas como protótipos em Figma, cronograma e quadro Kanban foram fundamentais para o planejamento, organização e controle das etapas do desenvolvimento.

O protótipo resultante oferece uma experiência completa, com áreas exclusivas para usuários e administradores, funcionalidades como cadastro e aluguel de conteúdos, histórico de locações, gerenciamento de catálogo e cálculo de preços. A identidade visual, composta pela logomarca e paleta de cores, reforça o conceito de conforto e entretenimento em casa, elementos centrais da proposta do CineHome.

Dessa forma, o projeto não apenas atende às necessidades do público-alvo, como também demonstra a aplicação de boas práticas em análise de requisitos, design centrado no usuário e desenvolvimento de sistemas. O CineHome representa uma solução escalável e alinhada com as tendências do mercado digital, preparada para evoluir conforme as exigências de um consumidor cada vez mais conectado e exigente.

9. Referências

- BRASIL. "**Lei Geral de Proteção de Dados Pessoais**". Disponível em: <https://www.gov.br/secretariageral/legislacao/lei-geral-de-protecao-dedadospessoais>. Acesso em: 24 mar. 2025.
- GITHUB. "**GitHub Docs: Getting Started with GitHub**". Disponível em: https://github.com/Isa-Paiola/Sprint_3-semester Acesso em: 24 mar. 2025.
- TRELLO. "**Trello Kanban CineHome**". Disponível em: <https://trello.com/b/2tX7cEz5/cinehome-%E2%AD%90> Acesso em: 24 mar. 2025.
- FIGMA. "**Protótipo Figma CineHome**". Disponível em: <https://www.figma.com/design/5WyvWOHJNtcQC6zrmqe1Y/Streaming-deFilmes?node-id=0-1&p=f&t=CyJkzV593V2kGOTY-0> Acesso em: 24 mar. 2025.
- CONTÁBEIS. **Transformação digital: como os hábitos de consumo mudaram no Brasil**. 2023. Disponível em: <https://www.contabeis.com.br>. Acesso em: 15 abr. 2025.
- KANTAR IBOPE MEDIA. *Inside Video 2024: conteúdo em vídeo no Brasil*. 2024. Disponível em: <https://kantaribobemedia.com/conteudo/conteudo-emvideo-2024>. Acesso em: 15 abr. 2025.
- KANTAR IBOPE MEDIA. *Inside Video 2024: conteúdo em vídeo no Brasil*. 2024. Disponível em: <https://kantaribobemedia.com/conteudo/conteudo-emvideo-2024>. Acesso em: 15 abr. 2025.
- **ESCRITÓRIO DE PROJETOS**. Termo de abertura do projeto: o que é, como fazer, exemplo. Disponível em:


<https://escritoriodeprojetos.com.br/termode-abertura-do-projeto/>. Acesso em: 15 abr. 2025.

- **VISURE SOLUTIONS.** O que são Requisitos Funcionais: Exemplos, Definição, Guia Completo. Disponível em: <https://visuresolutions.com/pt/blog/requisitos-funcionais/>. Acesso em: 15 abr. 2025.
- **VISURE SOLUTIONS.** O que são Requisitos Não Funcionais: Exemplos, Definição, Guia Completo. Disponível em: <https://visuresolutions.com/pt/blog/requisitos-n%C3%A3o-Funcionais/>. Acesso em: 15 abr. 2025.
- **ATLASSIAN.** *Agile vs Scrum: entenda as diferenças e quando usar cada um.* 2024. Disponível em: <https://www.atlassian.com/br/agile/scrum/agile-vsscrum>. Acesso em: 15 abr. 2025.
- **DOMESTIKA.** *O que cada cor significa no cinema.* 2024. Disponível em: <https://www.domestika.org/pt/blog/2788-o-que-cada-cor-significa-no-cinema>. Acesso em: 15 abr. 2025.
- **TRELLO.** Quadro no Trello. Disponível em: <https://trello.com/b/2tX7cEz5>. Acesso em: 16 abr. 2025.

10. APÊNDICE

10.1 Página mobile

Imagem 9 – Login

A mobile login screen with a dark background. The word "LOGIN" is centered at the top in a large, white, sans-serif font. Below it, the label "Usuário" is followed by a white rectangular input field. Further down, the label "Senha" is followed by another white rectangular input field. Below the password field is a red rectangular button with the word "ENTRAR" in white, uppercase letters. At the bottom, the text "Ainda não tem cadastro?" is displayed, followed by a red link that says "Cadastre-se".

LOGIN

Usuário

Senha

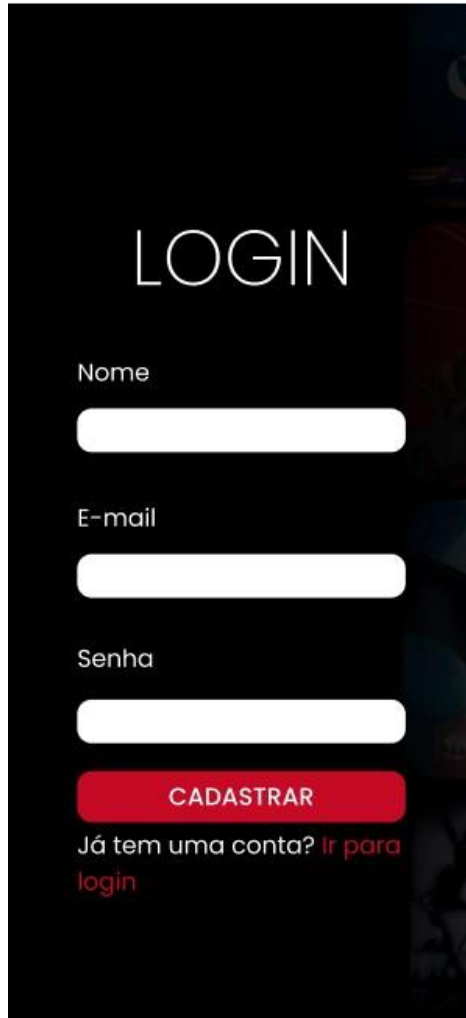
ENTRAR

Ainda não tem cadastro?

[Cadastre-se](#)

Fonte: Autoria própria (2025).

Imagem 10 – Cadastro

A mobile application interface for login and registration. The background is dark with a faint, abstract pattern. The word "LOGIN" is displayed in large, white, sans-serif capital letters. Below it, there are three white input fields for "Nome", "E-mail", and "Senha". A red button with the text "CADASTRAR" in white is positioned below the password field. At the bottom, there is a link that says "Já tem uma conta? Ir para login" in a light red color.

LOGIN

Nome

E-mail

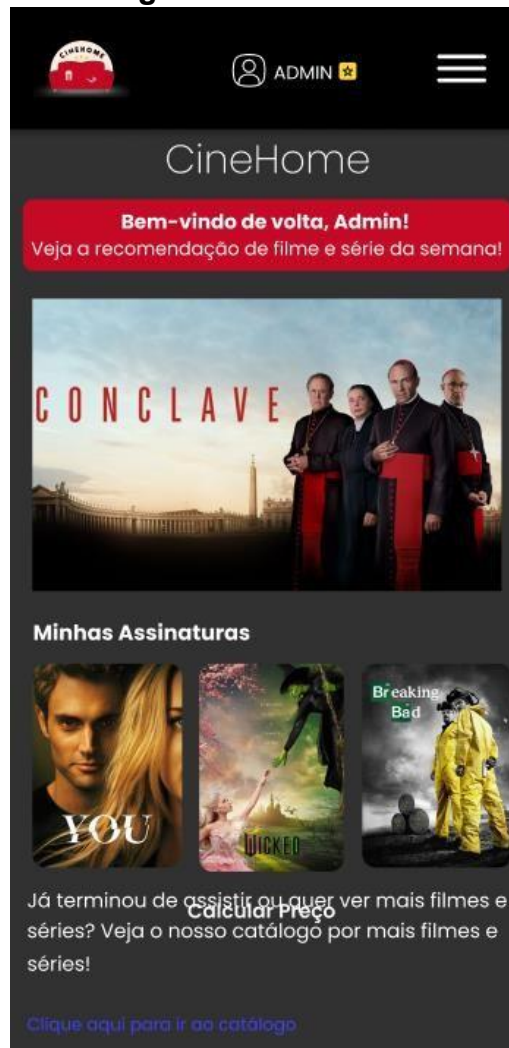
Senha

CADASTRAR

Já tem uma conta? [Ir para login](#)

Fonte: Autoria própria (2025).

Imagem 11 – Administrador



Adicionar

Adicionar ao catalogo

Título

Sinopse

Genêro

Tipo

Adicionar poster do filme

clique e escolha o poster

Adicionar

Calcular Preço

Tipo

Dias

Calcular



Como Treinar Seu Dragão

Filme | Animação | Aventura

Disponível

Veja mais



Dexter

Serie | Drama

Disponível

Veja mais



Ainda Estou Aqui

Filme | Drama

Indisponível

Veja mais

VALE TUDO

Vale Tudo

Novela | Drama

Disponível

Veja mais



Peaky Blinders

Serie | Drama | Ação

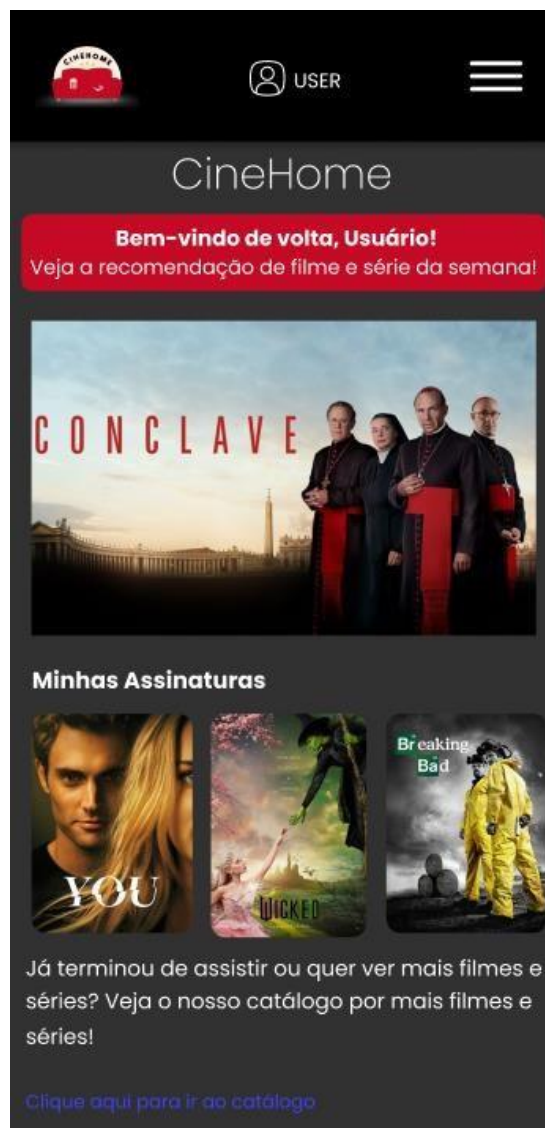
Indisponível

Veja mais



Fonte: Autoria própria (2025).

Imagem 12 – Usuário



Filmes



Séries



Novelas



Desenhos



Previsão

Calcular Preço

Tipo

Dias

Calcular



Como Treinar Seu Dragão

Filme | Animação | Aventura

Disponível

Veja mais




Dexter

Serie | Drama

Disponível

Veja mais




Ainda Estou Aqui

Ainda Estou Aqui

Filme | Drama

Indisponível

Veja mais




Vale Tudo

Novela | Drama

Disponível

Veja mais




Peaky Blinders

Serie | Drama | Ação

Indisponível

Veja mais




Greys Anatomy

Serie | Drama

Indisponível

Veja mais



Entre em contato: contact@cinehome.com

©2025, Cinehome.com | Direitos Reservados

Fonte: Autoria própria (2025).

10.2 Páginas web desktop

Imagem 13 – Cadastro

localhost/streaming-teste/codigo/public/cadastro.php

CADASTRO

Nome

E-mail

Senha

CADASTRAR

Já tem uma conta? [Ir para login](#)

Fonte: Autoria própria (2025).

Imagem 14 – Página Login

localhost/streaming-teste/codigo/public/login.php

LOGIN

Usuário

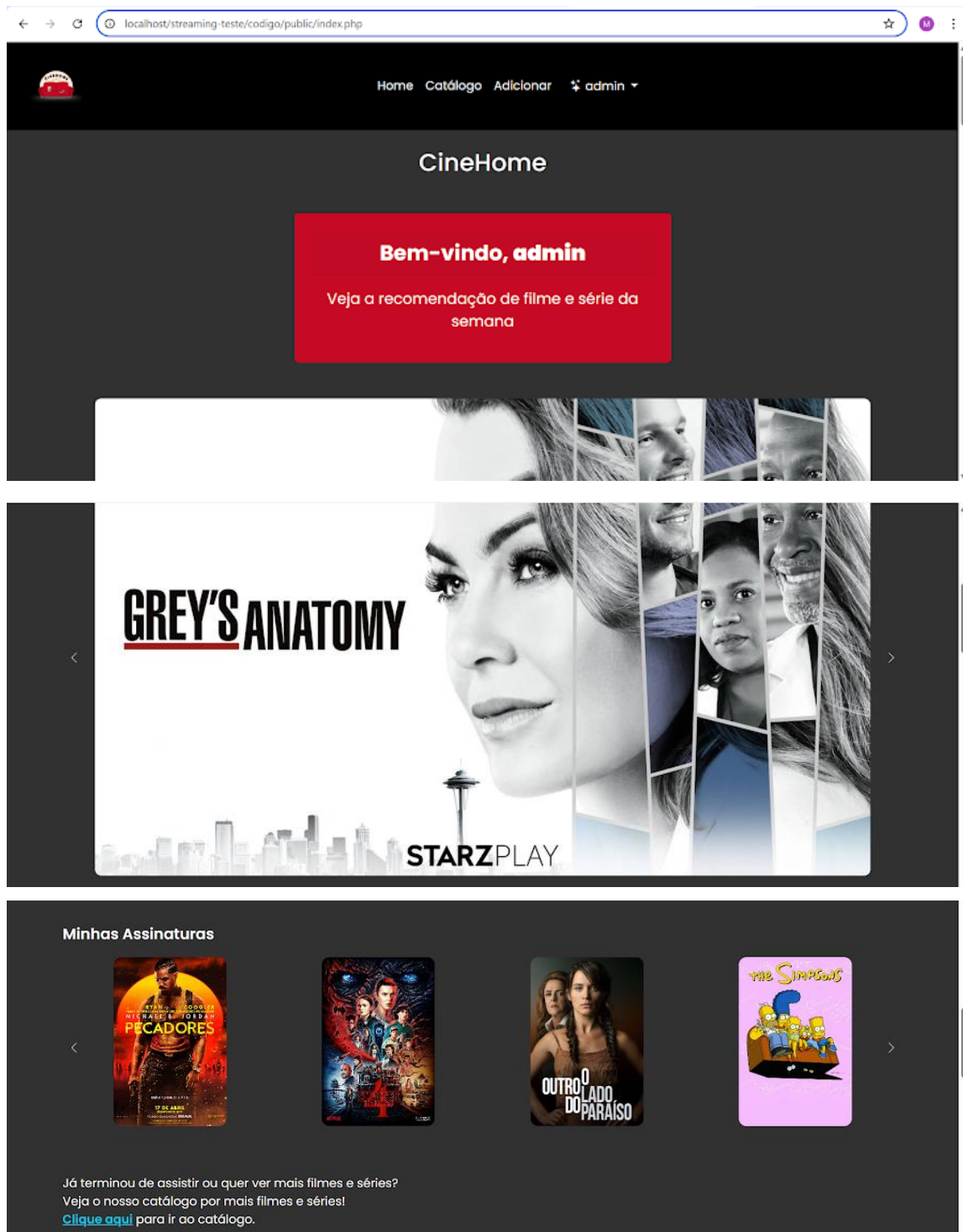
Senha

Entrar

Ainda não tem cadastro? [Cadastre-se](#)

Fonte: Autoria própria (2025).

Imagem 15 – Página Administrador



Adicionar

Adicionar ao catálogo

Título

Sinopse

Gênero

Selecione

Tipo

Selecione

Imagem

Selecione uma imagem existente (opcional)

Adicionar

Calcular Preço


Tipo

Selecione

Tempo em dias

Calcular

Item adicionado com sucesso!




Como Treinar Seu Dragão 2

Filme

Disponível

Alugar

EditarDeletar



Dexter

Série

Disponível

Alugar

EditarDeletar



Ainda estou aqui

Filme

Alugando

Deletar

EditarDeletar



Conclave

Filme

Disponível

Alugar

EditarDeletar

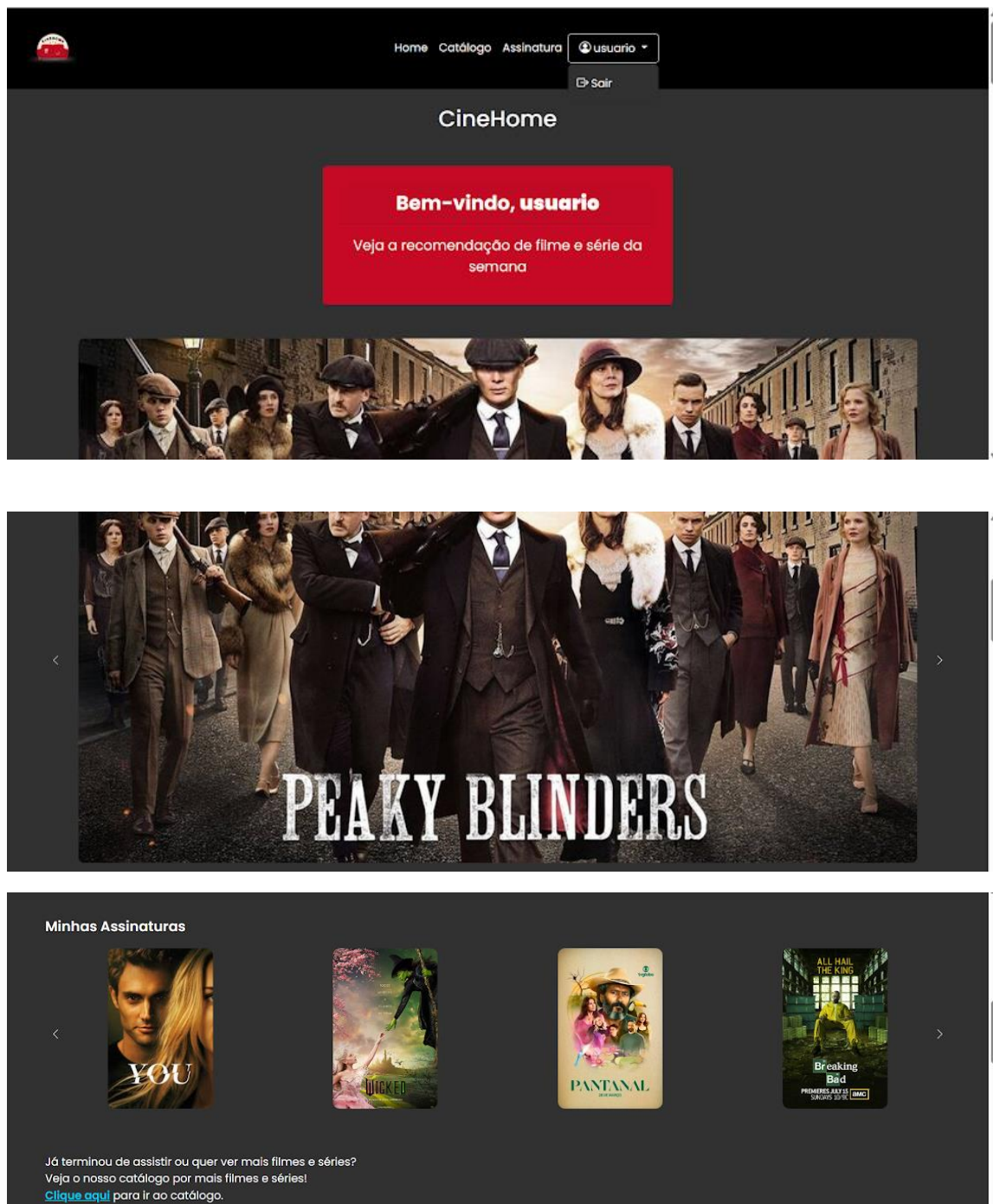


Entre em contato: contact@cinehome.com






© 2025, Cinehome.com | Direitos Reservados

Fonte: Autoria própria (2025).






Imagem 16 – Página Usuário








FILMES








SÉRIES



NOVELAS



DESENHOS



Previsão

Calcular Preço

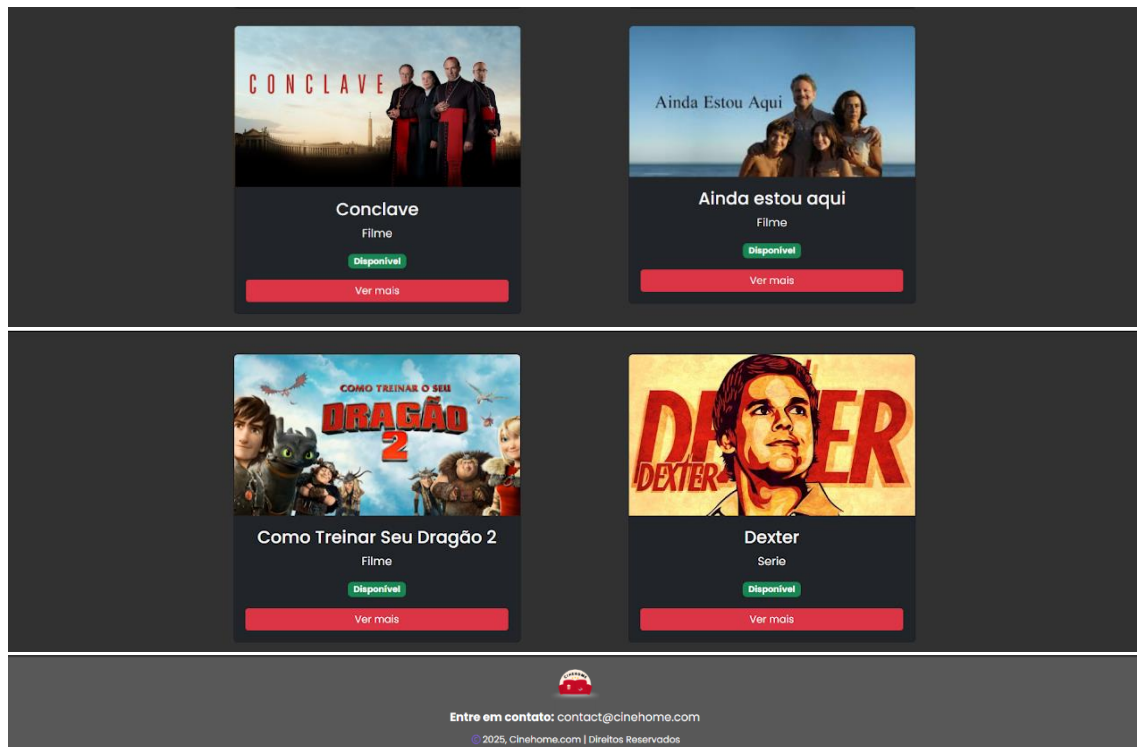
Tipo

Selecione

Tempo em dias

Calcular

Previsão de valor para 3 dias: R\$ 15,00



Fonte: Autoria própria (2025).

11 CÓDIGOS

11.1 CONFIG

Imagem 17 – config.php

```
<?php

// Arquivo de configuração com as constantes do sistema

define('ARQUIVO_JSON', __DIR__ . '/../data/itens.json');
define('ARQUIVO_USUARIOS', __DIR__ . '/../data/usuarios.json');
define('DIARIA_FILME', 20.00);
define('DIARIA_SERIE', 5.00);
define('DIARIA_NOVELA', 2.00);
define('DIARIA_DESENHO', 5.00);

// Definições de caminhos
define('UPLOAD_DIR', __DIR__ . '/../img/uploads/');

// Configurações de upload
define('MAX_FILE_SIZE', 5 * 1024 * 1024); // 5MB
define('ALLOWED_TYPES', ['image/jpeg', 'image/png', 'image/gif']);
```

Fonte: Autoria própria (2025).

Na Figura 17 o código PHP define constantes que configuram o funcionamento do sistema. Ele especifica os caminhos dos arquivos onde os dados dos itens e dos usuários serão armazenados, além do diretório onde as

imagens enviadas serão salvas. Também define os valores das diárias para diferentes tipos de mídias: R\$ 20,00 para filmes, R\$ 5,00 para séries e desenhos, e R\$ 2,00 para novelas. Por fim, estabelece regras para o envio de imagens, permitindo apenas arquivos nos formatos JPEG, PNG e GIF, com tamanho máximo de 5 megabytes.

11.2 DATA

Imagem 18 – intens.json

```
[
  {
    "tipo": "filme",
    "titulo": "Como Treinar Seu Drag\u00e3o 2",
    "sinopse": "Cinco anos se passaram desde que Solu\u00e7\u00e3o estabeleceu a paz com os drag\u00f5es e vive em harmonia, na Ilha de Berk, com Banguela. Eles voam, apostam corridas e se divertem muito. Em uma destas aventuras, descobrem uma caverna secreta cheia de drag\u00f5es. Agora, a dupla luta para proteger Berk de um guerreiro perigoso, chamado Drago Bludvist, que deseja controlar todos os drag\u00f5es existentes.",
    "genero": "infantil",
    "disponivel": true,
    "imagem": "img/comotreinarseudragao2.jpg"
  },
  {
    "tipo": "serie",
    "titulo": "Dexter",
    "sinopse": "A s\u00e9rie Dexter acompanha Dexter Morgan, um especialista forense que, por um lado, trabalha no departamento de pol\u00edcia de Miami-Dade como analista de padr\u00f5es de sangue, mas que, por outro lado, tem uma vida dupla como assassino em s\u00e9rie. Ele mata apenas assassinos que n\u00e3o foram condenados e que escaparam da lei, seguindo um \"C\u00f3digo\" criado pelo seu pai, Harry. ",
    "genero": "drama",
    "disponivel": true,
    "imagem": "img/dexter.jpg"
  },
  {
    "tipo": "filme",
    "titulo": "Ainda estou aqui",
    "sinopse": "No in\u00edcio da d\u00e9cada de 1970, o Brasil enfrenta o endurecimento da ditadura militar. No Rio de Janeiro, a fam\u00edlia Paiva - Rubens, Eunice e seus cinco filhos - vive \u00e0 beira da praia em uma casa de portas abertas para os amigos. Um dia, Rubens Paiva \u00e9 levado por militares \u00e0 paisana e desaparece.
```

```
Eunice - cuja busca pela verdade sobre o destino de seu marido se
estenderia por d\u00e9cadas - \u00e9 obrigada a se reinventar e
tra\u00e7ar um novo futuro para si e seus filhos.",
  "genero": "drama",
  "disponivel": true,
  "imagem": "img/aindaestouaqui.jpg"
}
]
```

Fonte: Autoria própria (2025).

Na Figura 18 o conteúdo apresentado é uma lista de obras (filmes e séries) registrada em formato JSON, utilizada para um sistema de catálogo digital. Cada obra possui informações como tipo (filme ou série), título, sinopse, gênero, disponibilidade e imagem ilustrativa. No total, há três obras cadastradas.

Imagem 19 – usuarios.json

```
[
  {
    "username": "admin",
    "password":
"$2y$10$GdeGCU5pOrEgksL3MmLXvu1JE8/V/KZOP1KYLkm6ukLwytV4uU8ii",
    "perfil": "admin"
  },
  {
    "username": "usuario",
    "password":
"$2y$10$XrG4efucfJVVWrFW82gDK0BTKlvQ91ptheoXlwbdevubx9eMlQGmJ2",
    "perfil": "usuario"
  },
  {
    "username": "Yasmin",
    "password":
"$2y$10$XrG4efucfJVVWrFW82gDK0BTKlvQ91ptheoXlwbdevubx9eMlQGmJ2",
    "perfil": "usuario"
  }
]
```

Fonte: Autoria própria (2025).

Na Figura 19 o conteúdo apresenta uma lista de usuários cadastrados em um sistema, armazenada em formato JSON. Cada usuário possui um nome

de login (username), uma senha criptografada (password) e um perfil de acesso (perfil). Há três usuários registrados.

11.3 INTERFACES

Imagem 20 – locavel.php

```
<?php
namespace Interfaces;

// Interface que define os métodos necessários para um item ser locável

interface Locavel {
    public function alugar() : string;
    public function devolver() : string;
    public function isDisponivel(): bool;
}
```

Fonte: Autoria própria (2025).

A Figura 20 mostra o código que define uma interface chamada Locavel, que serve como um modelo para qualquer item que possa ser alugado no sistema. Essa interface obriga as classes que a implementarem a possuir três funcionalidades.

12 MODELS

12.1 desenh.php

Imagem 21 – desenho.php

```
<?php
namespace Models;
use Interfaces\Locavel;

//classe que representa as motos
class Desenho extends Item implements Locavel {

    public function calcularAluguel(int $dias):float{
        return $dias * DIARIA_DESENHO;
    }

    public function alugar():string{
        if($this->disponivel){
            $this->disponivel = false;
            return "Desenho '{$this->titulo}' alugada com sucesso!";
        }
    }
}
```

```

        return "Desenho '{$this->titulo}' não está disponível.";
    }

    public function devolver() : string{
        if(!$this->disponivel){
            $this->disponivel = true;
            return "Desenho '{$this->titulo}' devolvida com sucesso!";
        }
        return "Desenho '{$this->titulo}' já está disponível.";
    }
}

```

Fonte: Autoria própria (2025).

A Figura 21 mostra a classe Desenho representa desenhos animados que podem ser alugados. Ela possui métodos para calcular o valor do aluguel, alugar e devolver o item, respeitando a lógica de disponibilidade. Tudo isso segue um padrão definido pela interface Locavel, garantindo que o sistema trate todos os itens alugáveis de forma padronizada.

12.2 filme.php

Imagem 22 – filme.php

```

<?php
namespace Models;
use Interfaces\Locavel;

// Classe que representa um carro

class Filme extends Item implements Locavel {

    public function calcularAluguel(int $dias):float {
        return $dias * DIARIA_FILME;
    }

    public function alugar(): string {
        if ($this->disponivel){
            $this->disponivel = false;
            return "Filme '{$this->titulo}' alugado com sucesso!";
        }
        return "Filme '{$this->titulo}' não está disponível.";
    }

    public function devolver(): string {
        if (!$this->disponivel){
            $this->disponivel = true;

```

```

        return "Filme '{$this->titulo}' devolvido com sucesso!";
    }
    return "Filme '{$this->titulo}' já está disponível.";
}
}

```

Fonte: Autoria própria (2025).

A Figura 22 mostra o código que define a classe Filme, que representa um filme que pode ser alugado. Essa classe herda de uma classe base chamada Item e implementa uma interface chamada Locavel, o que significa que ela deve ter métodos para alugar, devolver e calcular o valor do aluguel.

O método calcularAluguel calcula o preço do aluguel multiplicando a quantidade de dias pelo valor da diária do filme. O método alugar verifica se o filme está disponível; se estiver, marca como alugado e retorna uma mensagem de sucesso, caso contrário, informa que o filme não está disponível. O método devolver faz o contrário: se o filme estiver alugado, marca como disponível e confirma a devolução; se já estiver disponível, informa que não há necessidade de devolução.

12.3 item.php

Imagem 23 – item.php

```

<?php
namespace Models;

// Classe abstrata para todos os tipos de itens

abstract class Item {
    protected string $titulo;
    protected string $genero;
    protected string $sinopse;
    protected bool $disponivel;
    protected ?string $imagem;

    public function __construct(string $titulo, string $sinopse, string
$genero, ?string $imagem = null) {
        $this->titulo = $titulo;
        $this->genero = $genero;
        $this->sinopse = $sinopse;
        $this->disponivel = true;
        $this->imagem = $imagem;
    }
}

```

```

}

// Função para cálculo de aluguel
abstract public function calcularAluguel(int $dias): float;

public function isDisponivel(): bool {
    return $this->disponivel;
}

public function getTitulo(): string {
    return $this->titulo;
}

public function getGenero(): string {
    return $this->genero;
}

public function getSinopse(): string {
    return $this->sinopse;
}

public function setDisponivel(bool $disponivel): void {
    $this->disponivel = $disponivel;
}

public function getImagem(): string {
    if ($this->imagem === null || empty($this->imagem)) {
        return 'img/no-image.jpg';
    }
    // Remove barras invertidas e normaliza o caminho
    $path = str_replace('\\', '/', $this->imagem);
    // Remove ./ do início se existir
    $path = preg_replace('/^\.\//', '', $path);
    return $path;
}

public function setImagem(?string $imagem): void {
    if ($imagem !== null) {
        // Normaliza o caminho antes de salvar
        $imagem = str_replace('\\', '/', $imagem);
        // Remove ./ do início se existir
        $imagem = preg_replace('/^\.\//', '', $imagem);
        $this->imagem = $imagem;
    } else {
        $this->imagem = null;
    }
}
}

```

Fonte: Autoria própria (2025).

A Figura 23 mostra a classe Item é um modelo base abstrato para diferentes tipos de itens no sistema, armazenando informações comuns como título, gênero, sinopse, disponibilidade e imagem. Ela obriga a implementação do método para calcular o valor do aluguel e oferece métodos para acessar e modificar esses dados, além de tratar a imagem padrão e o caminho das imagens.

12.4 novela.php

Imagem 24 – novela.php

```
<?php
namespace Models;
use Interfaces\Locavel;

//classe que representa as motos
class Novela extends Item implements Locavel {

    public function calcularAluguel(int $dias):float{
        return $dias * DIARIA_NOVELA;
    }

    public function alugar():string{
        if($this->disponivel){
            $this ->disponivel = false;
            return "Novela '{$this->titulo}' alugada com sucesso!";
        }
        return "Novela '{$this->titulo}' não está disponível.";
    }

    public function devolver() : string{
        if(!$this->disponivel){
            $this ->disponivel = true;
            return "Novela '{$this->titulo}' devolvida com sucesso!";
        }
        return "Novela '{$this->titulo}' já está disponível.";
    }
}
```

Fonte: Autoria própria (2025).

A Figura 24 mostra a classe *Novela* representa uma novela que pode ser alugada no sistema. Ela herda da classe base *Item* e implementa a interface *Locavel*, garantindo os métodos para calcular aluguel, alugar e devolver.

O método *calcularAluguel* multiplica os dias pelo valor da diária da novela. O método *alugar* verifica se a novela está disponível; se sim, marca como alugada e retorna uma mensagem de sucesso; se não, informa que não está disponível. O método *devolver* faz o contrário: se estiver alugada, marca como disponível e confirma a devolução; caso contrário, informa que já está disponível.

12.5 serie.php

Imagem 25 – serie.php

```
<?php
namespace Models;
use Interfaces\Locavel;

//classe que representa as motos
class Serie extends Item implements Locavel {

    public function calcularAluguel(int $dias):float{
        return $dias * DIARIA_SERIE;
    }

    public function alugar():string{
        if($this->disponivel){
            $this ->disponivel = false;
            return "Serie '{$this->titulo}' alugada com sucesso!";
        }
        return "Serie '{$this->titulo}' não está disponível.";
    }

    public function devolver() : string{
        if(!$this->disponivel){
            $this ->disponivel = true;
            return "Serie '{$this->titulo}' devolvida com sucesso!";
        }
        return "Serie '{$this->titulo}' já está disponível.";
    }
}
```

Fonte: Autoria própria (2025).

A Figura 25 mostra a classe *Serie* representa uma série que pode ser alugada no sistema. Ela estende a classe base *Item* e implementa a interface *Locavel*, garantindo a implementação dos métodos para calcular o valor do aluguel, alugar e devolver.

O método `calcularAluguel` calcula o preço multiplicando o número de dias pelo valor da diária da série. O método `alugar` verifica se a série está disponível; se estiver, marca como alugada e retorna uma mensagem de sucesso; caso contrário, informa que não está disponível. O método `devolver` faz o contrário: se estiver alugada, marca como disponível e confirma a devolução; se já estiver disponível, informa que não precisa ser devolvida.

13 NODE MODULES

13.1 public

Imagem 26 – cadastro.php

```
<?php
session_start();

// Limpa o cadastro se vier com reset
if (isset($_GET['reset']) && $_GET['reset'] == 1) {
    unset($_SESSION['usuario']);
    header("Location: cadastro.php");
    exit;
}

if (isset($_SESSION['usuario'])) {
    $nome = $_SESSION['usuario']['nome'];
}
?>

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
    <title>CineHome - Cadastro</title>

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.
css"/>
```

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.13.1/font/bootstrap-icons.min.css">

<style>
  @import
url('https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,20
0;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;
1,600;1,700;1,800;1,900&family=Roboto+Mono:ital,wght@0,100..700;1,100..700
&display=swap&quot; rel="stylesheet');

  * {
    font-family: 'Poppins', sans-serif;
    margin: 0;
    padding: 0;
    box-sizing: border-box;
  }

  body, html {
    height: 100%;
    overflow: hidden;
  }

  .bg-container {
    position: relative;
    width: 100%;
    height: 100vh;
    background: url('../img/foto_filmes.png') no-repeat left center;
    background-size: cover;
  }

  .form-overlay {
    position: absolute;
    left: 0;
    top: 0;
    height: 100%;
    width: 40%;
    display: flex;
    justify-content: center;
    align-items: center;
    background: rgba(0, 0, 0, 0.6);
  }

  .form-box {
    width: 100%;
    max-width: 360px;
    padding: 2rem;
    background-color: rgba(0, 0, 0, 0.7);
    border-radius: 12px;
    color: white;
```

```
    z-index: 2;
  }

  .common-input {
    background-color: #fff;
    color: #000 !important;
  }

  .common-input:focus {
    background-color: #ffffffdd;
    color: #000 !important;
    box-shadow: none;
  }

  .btn-danger {
    background-color: #D22424;
    border: none;
    padding: 0.75rem;
    font-weight: bold;
    letter-spacing: 1px;
    transition: background-color 0.2s ease;
  }

  .btn-danger:hover {
    background-color: #a71c1c;
  }

  .form-label {
    font-size: 1.2rem;
    font-weight: 500;
  }

  .text-link {
    text-align: center;
    margin-top: 1rem;
  }

  .text-link a {
    color: #ff4c4c;
    text-decoration: none;
  }

  .text-link a:hover {
    text-decoration: underline;
  }

  @media (max-width: 768px) {
    .form-overlay {
      width: 100%;
    }
  }
}
```

```

        background-color: rgba(0, 0, 0, 0.85);
    }

    .bg-container {
        background-position: top center;
        background-size: contain;
    }
}
</style>
</head>
<body>

<div class="bg-container">
    <div class="form-overlay">
        <div class="form-box">
            <h1 class="mb-4 text-center" style="letter-spacing:
2px;">CADASTRO</h1>

            <?php
            if (isset($nome)) {
                echo "<div class='alert alert-info text-center'>
                    Olá, <strong>$nome</strong>! Você já está cadastrado no
CineHome.
                </div>
                <div class='text-center mt-3'>
                    <a href='login.php' class='btn btn-danger w-100'>Ir para a
página de login</a>
                </div>
                <div class='text-link'>
                    <a href='cadastro.php?reset=1'>Voltar para o cadastro</a>
                </div>";
            } elseif ($_SERVER["REQUEST_METHOD"] == "POST") {
                $nome = htmlspecialchars($_POST["nome"]);
                $email = htmlspecialchars($_POST["email"]);
                $senha = htmlspecialchars($_POST["senha"]);

                $_SESSION['usuario'] = [
                    'nome' => $nome,
                    'email' => $email,
                    'senha' => $senha,
                ];

                echo "<div class='alert alert-success text-center'>
                    Cadastro realizado com sucesso! Bem-vindo,
<strong>$nome</strong>.
                </div>
                <div class='text-center mt-3'>
                    <a href='login.php' class='btn btn-danger w-100'>Ir para a
página de login</a>
            }
        </div>
    </div>
</div>

```

```

        </div>
        <div class='text-link'>
            <a href='cadastro.php?reset=1'>Voltar para o cadastro</a>
        </div>";
    } else {
        ?>

        <form method="POST" action="">
            <div class="mb-3">
                <label for="nome" class="form-label">Nome</label>
                <input type="text" name="nome" id="nome" class="form-control
common-input" required>
            </div>
            <div class="mb-3">
                <label for="email" class="form-label">E-mail</label>
                <input type="email" name="email" id="email" class="form-control
common-input" required>
            </div>
            <div class="mb-4">
                <label for="senha" class="form-label">Senha</label>
                <input type="password" name="senha" id="senha" class="form-
control common-input" required>
            </div>
            <button type="submit" class="btn btn-danger w-
100">CADASTRAR</button>
        </form>

        <div class="text-link">
            Já tem uma conta? <a href="login.php">Ir para login</a>
        </div>

        <?php } ?>
    </div>
</div>
</div>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle
.min.js"></script>
</body>
</html>

```

Fonte: Autoria própria (2025).

A Figura 26 mostra o código PHP cria uma página de cadastro para o site CineHome. Ele usa sessões para armazenar os dados do usuário (nome, email, senha). Se o usuário já estiver cadastrado, mostra uma mensagem de boas-vindas e um link para login. Caso contrário, exibe um formulário para

cadastro. Ao enviar o formulário, os dados são salvos na sessão e uma mensagem de confirmação aparece. Também há uma opção para limpar o cadastro e reiniciar o processo. O visual usa Bootstrap e uma imagem de fundo com estilo responsivo.

Imagem 27 – index.php

```
<?php

// Incluir o autoload
require_once __DIR__ . '/../vendor/autoload.php';

// Incluir o arquivo com as variáveis
require_once __DIR__ . '/../config/config.php';

session_start();

// Importar as classes Locadora e Auth
use Services\{Locadora, Auth};

// Importar as classes de modelos
use Models\{Serie, Filme, Desenho, Novela};

// Verificar se o usuário está logado
if(!Auth::verificarLogin()){
    header('Location: login.php');
    exit;
}

// Condição para logout
if (isset($_GET['logout'])){
    (new Auth()->logout());
    header('Location: login.php');
    exit;
}

if (!is_dir('img/uploads')) {
    mkdir('img/uploads', 0777, true);
}
chmod('img/uploads', 0777);

// Criar uma instância da classe locadora
$locadora = new Locadora();

$mensagem = '';

$usuario = Auth::getUsuario();
```

```

// Verificar os dados do formulário via POST
if ($_SERVER['REQUEST_METHOD'] == 'POST') {

    // Verificar se requer permissão de administrador
    if (isset($_POST['adicionar']) || isset($_POST['deletar']) ||
isset($_POST['alugar']) || isset($_POST['devolver'])) {
        if (!Auth::isAdmin()) {
            $mensagem = "Você não tem permissão para realizar essa ação.";
            goto renderizar;
        }
    }

    if (isset($_POST['adicionar'])) {
        if (isset($_POST['titulo'], $_POST['sinopse'], $_POST['genero'],
$_POST['tipo'])) {
            try {
                // Debug - Mostrar todos os dados do POST
                error_log("Dados do POST: " . print_r($_POST, true));

                // Verifica se foi selecionada uma imagem existente
                $imagemPath = null;
                if (!empty($_POST['imagem_existente'])) {
                    $imagemPath = $_POST['imagem_existente'];
                    error_log("Imagem existente selecionada: " . $imagemPath);

                    // Verifica se o arquivo existe
                    if (!file_exists($imagemPath)) {
                        error_log("AVISO: Arquivo de imagem não encontrado: "
. $imagemPath);
                    }
                }
                // Se não foi selecionada uma imagem existente e foi feito
upload
                elseif (isset($_FILES['imagem']) && $_FILES['imagem']['error']
=== UPLOAD_ERR_OK) {
                    error_log("Dados do arquivo enviado: " .
print_r($_FILES['imagem'], true));
                    $imagemPath = $uploadHandler-
>handleUpload($_FILES['imagem']);
                    error_log("Nova imagem enviada: " . $imagemPath);
                }

                // Se nenhuma imagem foi fornecida, usa a imagem padrão
                if ($imagemPath === null) {
                    $imagemPath = 'img/no-image.jpg';
                    error_log("Usando imagem padrão: " . $imagemPath);
                }
            } catch (\RuntimeException $e) {

```

```

        $mensagem = "Erro ao fazer upload da imagem: " . $e-
>getMessage();
        goto renderizar;
    }
    $titulo = $_POST['titulo'];
    $sinopse = $_POST['sinopse'];
    $genero = $_POST['genero'];
    $tipo = $_POST['tipo'];

    if ($tipo == 'serie'){
        $item = new Serie($titulo, $sinopse, $genero, $imagemPath);
    } elseif($tipo == 'filme'){
        $item = new Filme($titulo, $sinopse, $genero, $imagemPath);
    } elseif ($tipo == 'desenho'){
        $item = new Desenho($titulo, $sinopse, $genero, $imagemPath);
    } elseif ($tipo == 'novela'){
        $item = new Novela($titulo, $sinopse, $genero, $imagemPath);
    } else {
        $mensagem = "Escolha um tipo válido.";
        goto renderizar;
    }

    $locadora->adicionarItem($item);

    $mensagem = "Item adicionado com sucesso!";
}
}
elseif(isset($_POST['alugar']))){

    $dias = isset($_POST['dias']) ? (int)$_POST['dias'] : 1;
    $mensagem = $locadora->alugarItem($_POST['titulo'], $dias);
}
elseif(isset($_POST['devolver']))){
    $mensagem = $locadora->devolverItem($_POST['titulo']);
}
elseif(isset($_POST['deletar']))){
    $mensagem = $locadora->deletarItem($_POST['titulo'],
$_POST['genero']);
}
elseif(isset($_POST['calcular']))){
    $dias = (int)$_POST['diasCalculo'];
    $tipo = $_POST['tipoCalculo'];
    $valor = $locadora->calcularPrevisaoAluguel($tipo, $dias);

    $mensagem = "Previsão de valor para {$dias} dias: R$ " .
number_format($valor, 2, ',', '.');
    $_POST = []; // Limpar os dados do formulário após alugar
}
}

```



```
renderizar:  
require_once __DIR__ . '/../views/template.php';
```

Fonte: Autoria própria (2025).

A Figura 27 mostra esse script PHP é o controlador principal da aplicação de uma locadora de vídeos. Ele começa incluindo arquivos de configuração e autoloader para carregar as dependências necessárias. Em seguida, inicia a sessão do usuário para controlar o login.

O sistema verifica se o usuário está autenticado. Se não estiver, redireciona para a página de login. Também permite que o usuário faça logout pela URL.

Depois, verifica se a pasta para armazenar imagens existe e, caso contrário, cria essa pasta com permissões adequadas.

É criada uma instância da classe Locadora, que gerencia os itens disponíveis, como séries, filmes, desenhos e novelas.

Quando o formulário é enviado via método POST, o script verifica se o usuário tem permissão para executar ações restritas, como adicionar, deletar, alugar ou devolver um item. Caso não tenha, exibe uma mensagem de erro.

Se o usuário quer adicionar um item, o sistema captura os dados enviados, trata o upload da imagem ou usa uma imagem padrão, cria o objeto do tipo correto (de acordo com o tipo selecionado: série, filme, desenho ou novela) e adiciona esse item na locadora.

Para as ações de aluguel e devolução, o sistema chama os métodos correspondentes para atualizar o status do item. Na ação de deletar, o item é removido da lista.

Também há uma funcionalidade para calcular o valor previsto do aluguel com base no tipo do item e na quantidade de dias informados.

Por fim, a aplicação exibe mensagens para o usuário indicando o resultado das ações e carrega o template da interface para mostrar a página atualizada.

Imagem 28 – login.php

```
<?php
```

```

// Incluir o autoload do composer para carregar automaticamente as classes
utilizadas
require_once __DIR__ . '/../vendor/autoload.php';
require_once __DIR__ . '/../config/config.php';

session_start();

use Services\Auth;

$mensagem = '';
$auth = new Auth();

// Se já estiver logado, redireciona para index
if (Auth::verificarLogin()) {
    header('Location: index.php');
    exit;
}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $username = $_POST['username'] ?? '';
    $password = $_POST['password'] ?? '';

    if ($auth->login($username, $password)) {
        header('Location: index.php');
        exit;
    } else {
        $mensagem = 'Usuário ou senha inválidos';
    }
}

?>
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
    <title>CineHome - Login</title>

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.
css"/>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.13.1/font/bootstrap-icons.min.css">

    <style>
        @import
url('https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,20
0;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;

```

```
1,600;1,700;1,800;1,900&family=Roboto+Mono:ital,wght@0,100..700;1,100..700
&display=swap&quot; rel="stylesheet");
```

```
* {
  font-family: 'Poppins', sans-serif;
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}
```

```
body, html {
  height: 100%;
  overflow: hidden;
}
```

```
.bg-container {
  position: relative;
  width: 100%;
  height: 100vh;
  background: url('../img/foto_filmes.png') no-repeat left center;
  background-size: cover;
}
```

```
.form-overlay {
  position: absolute;
  left: 0;
  top: 0;
  height: 100%;
  width: 40%;
  display: flex;
  justify-content: center;
  align-items: center;
  background: rgba(0, 0, 0, 0.6);
}
```

```
.form-box {
  width: 100%;
  max-width: 360px;
  padding: 2rem;
  background-color: rgba(0, 0, 0, 0.7);
  border-radius: 12px;
  color: white;
  z-index: 2;
}
```

```
.common-input {
  background-color: #fff;
  color: #000 !important;
}
```

```
.common-input:focus {
  background-color: #ffffffdd;
  color: #000 !important;
  box-shadow: none;
}

.btn-danger {
  background-color: #D22424;
  border: none;
  padding: 0.75rem;
  font-weight: bold;
  letter-spacing: 1px;
  transition: background-color 0.2s ease;
}

.btn-danger:hover {
  background-color: #a71c1c;
}

.form-label {
  font-size: 1.2rem;
  font-weight: 500;
}

.text-link {
  text-align: center;
  margin-top: 1rem;
}

.text-link a {
  color: #ff4c4c;
  text-decoration: none;
}

.text-link a:hover {
  text-decoration: underline;
}

@media (max-width: 768px) {
  .form-overlay {
    width: 100%;
    background-color: rgba(0, 0, 0, 0.85);
  }

  .bg-container {
    background-position: top center;
    background-size: contain;
  }
}
```

```

    }
    </style>
</head>
<body>
    <div class="bg-container">
        <div class="form-overlay">
            <div class="form-box">
                <h1 class="mb-4 text-center" style="letter-spacing:
2px;">LOGIN</h1>

                <?php if ($mensagem) :?>
                    <div class="alert alert-danger"><?=
htmlspecialchars($mensagem)?></div>
                <?php endif; ?>

                <form method="post" class="needs-validation" novalidate>
                    <div class="mb-3">
                        <label class="form-label">Usuário</label>
                        <input type="text" name="username" class="form-
control" required>
                    </div>
                    <div class="mb-3">
                        <label class="form-label">Senha</label>
                        <input type="password" name="password"
class="form-control" required>
                    </div>
                    <button type="submit" class="btn btn-danger w-
100">Entrar</button>
                </form>
                <div class="text-link">
                    Ainda não tem cadastro? <a href="cadastro.php">Cadastre-se</a>
                </div>
            </div>
        </div>
    </div>

    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle
.min.js"></script>
</body>
</html>

```

Fonte: Autoria própria (2025).

A Figura 28 mostra o código PHP cria uma página de login para o sistema CineHome. Ele começa carregando as configurações e classes necessárias, inicia a sessão e verifica se o usuário já está logado. Se estiver, redireciona para a página principal.

Quando o formulário de login é enviado, o sistema tenta autenticar o usuário usando o nome de usuário e senha fornecidos. Se o login for correto, o usuário é redirecionado para a página principal. Caso contrário, uma mensagem de erro aparece informando que o usuário ou a senha são inválidos.

A página HTML usa o Bootstrap para estilização e exibe um formulário com campos para usuário e senha, além de um botão para entrar. Também há um link para a página de cadastro para novos usuários.

13.2 services

Imagem 29 – auth.php

```
<?php
// define espaço para organização do código
namespace Services;

class Auth{
    private array $usuarios = [];

    // Método construtor
    public function __construct(){
        $this->carregarUsuarios();
    }

    // Método para carregar usuários do arquivo JSON
    private function carregarUsuarios(): void {

        // Verificar se existe o arquivo
        if(file_exists(ARQUIVO_USUARIOS)){
            // Lê o conteúdo e decodifica o JSON para o array
            $conteudo = json_decode(file_get_contents(ARQUIVO_USUARIOS),
true);

            // verificar se é um array
            $this->usuarios = is_array($conteudo) ? $conteudo : [];
        } else {
            $this-> usuarios = [
                [
                    'username' => 'admin',
                    'password' => password_hash('admin123',
PASSWORD_DEFAULT),
                    'perfil' => 'admin'
                ],
            ]
        }
    }
}
```

```

        'username' => 'usuario',
        'password' => password_hash('user123',
PASSWORD_DEFAULT),
        'perfil' => 'usuario'
    ]
};
$this->salvarUsuarios();
}

// função para salvar usuários no arquivo JSON
private function salvarUsuarios():void {
    $dir = dirname(ARQUIVO_USUARIOS);

    if(!is_dir($dir)){
        mkdir($dir, 0777, true);
    }
    file_put_contents(ARQUIVO_USUARIOS, json_encode($this->usuarios,
JSON_PRETTY_PRINT));
}

// Método para realizar login
public function login(string $username, string $password): bool{

    foreach ($this->usuarios as $usuario){
        if ($usuario['username'] === $username &&
password_verify($password, $usuario['password'])){
            $_SESSION['auth'] = [
                'logado' => true,
                'username' => $username,
                'perfil' => $usuario['perfil']
            ];
            return true; //login realizado
        }
    }
    return false; // não realizou login
}

public function logout():void{
    session_destroy();
}

// verificar se o usuário está logado
public static function verificarLogin():bool{
    return isset($_SESSION['auth']) && $_SESSION['auth']['logado'] ===
true;
}

public static function isPerfil(string $perfil): bool{

```

```

        return isset($_SESSION['auth']) && $_SESSION['auth']['perfil'] ===
$perfil;
    }

    public static function isAdmin():bool {
        return self::isPerfil('admin');
    }

    public static function isUser():bool {
        return self::isPerfil('usuario');
    }

    public static function getUsuario(): ?array {
        // retorna os dados da sessão ou nulo se não existir
        return $_SESSION['auth'] ?? null;
    }
}

```

Fonte: Autoria própria (2025).

A Figura 29 mostra a classe Auth gerencia o sistema de autenticação dos usuários. Ela carrega os usuários de um arquivo JSON, ou cria usuários padrão caso o arquivo não exista. Permite que um usuário faça login, validando o nome e a senha, criando uma sessão para manter o estado de login. Também possui métodos para logout e para verificar se o usuário está logado ou se possui um perfil específico, como administrador ou usuário comum. Além disso, pode retornar os dados do usuário atualmente logado.

Imagem 30 – locadora.php

```

<?php
namespace Services;

use Models\{Item, Filme, Serie, Desenho, Novela};

// classe para gerenciar a locação
class Locadora {
    private array $itens = [];

    public function __construct(){
        if (!defined('ARQUIVO_JSON')) {
            throw new \RuntimeException('Constante ARQUIVO_JSON não
definida');
        }
        $this->carregarItens();
    }
}

```



```

private function carregarItens(): void {
    if (file_exists(ARQUIVO_JSON)) {
        // decodifica o arquivo JSON
        $jsonContent = file_get_contents(ARQUIVO_JSON);
        if ($jsonContent === false) {
            error_log("Erro ao ler o arquivo JSON: " . ARQUIVO_JSON);
            return;
        }

        $dados = json_decode($jsonContent, true);
        if ($dados === null) {
            error_log("Erro ao decodificar JSON: " .
json_last_error_msg());
            return;
        }

        error_log("Dados carregados do JSON: " . print_r($dados,
true));

        foreach ($dados as $dado) {
            try {
                if ($dado['tipo'] === 'filme') {
                    $item = new Filme($dado['titulo'],
$dado['sinopse'], $dado['genero'], $dado['imagem'] ?? null);
                } else if ($dado['tipo'] === 'serie') {
                    $item = new Serie($dado['titulo'],
$dado['sinopse'], $dado['genero'], $dado['imagem'] ?? null);
                } else if ($dado['tipo'] === 'novela') {
                    $item = new Novela($dado['titulo'],
$dado['sinopse'], $dado['genero'], $dado['imagem'] ?? null);
                } else if ($dado['tipo'] === 'desenho') {
                    $item = new Desenho($dado['titulo'],
$dado['sinopse'], $dado['genero'], $dado['imagem'] ?? null);
                } else {
                    error_log("Tipo de item desconhecido: " .
$dado['tipo']);

                    continue;
                }

                $item->setDisponivel($dado['disponivel']);
                $this->itens[] = $item;
                error_log("Item carregado: " . $item->getTitulo() . "
com imagem: " . $item->getImagem());
            } catch (\Exception $e) {
                error_log("Erro ao criar item: " . $e->getMessage());
            }
        }
    } else {

```

```

        error_log("Arquivo JSON não encontrado: " . ARQUIVO_JSON);
    }
}

// Salvar item
private function salvarItens(): void {
    $dados = [];

    foreach ($this->itens as $item) {
        $dados[] = [
            'tipo' => ($item instanceof Filme) ? 'filme' :
                (($item instanceof Serie) ? 'serie' :
                    (($item instanceof Novela) ? 'novela' :
                        (($item instanceof Desenho) ? 'desenho' :
null))),
            'titulo' => $item->getTitulo(),
            'sinopse' => $item->getSinopse(),
            'genero' => $item->getGenero(),
            'disponivel' => $item->isDisponivel(),
            'imagem' => $item->getImagem()
        ];
    }

    $dir = dirname(ARQUIVO_JSON);
    if (!is_dir($dir)) {
        if (!mkdir($dir, 0777, true)) {
            error_log("Erro ao criar diretório: " . $dir);
            return;
        }
    }

    $jsonString = json_encode($dados, JSON_PRETTY_PRINT |
JSON_UNESCAPED_SLASHES);
    if ($jsonString === false) {
        error_log("Erro ao codificar JSON: " . json_last_error_msg());
        return;
    }

    $resultado = file_put_contents(ARQUIVO_JSON, $jsonString);
    if ($resultado === false) {
        error_log("Erro ao salvar arquivo JSON: " . ARQUIVO_JSON);
    } else {
        error_log("Dados salvos com sucesso em: " . ARQUIVO_JSON);
        error_log("Conteúdo salvo: " . $jsonString);
    }
}

// Adicionar novo item
public function adicionarItem(Item $item): void{

```

```

        $this->itens[] = $item;
        $this->salvarItens();
        $_POST = []; // Limpar os dados do formulário após adicionar
    }

    //Remover item
    public function deletarItem(string $titulo, string $genero): string{

        foreach ($this->itens as $key => $item){

            // verifica se o titulo e tipo correspondem
            if($item->getTitulo() === $titulo && $item->getGenero() ===
$genero){

                // remove o item do array
                unset($this->itens[$key]);

                // reorganizar os indices
                $this->itens = array_values($this->itens);

                // Salvar o novo estado
                $this->salvarItens();
                return "Item '{$titulo}' removido com sucesso!";
            }
        }
        return "Item não encontrado!";
    }

    // Alugar item por n dias
    public function alugarItem(string $titulo, int $dias = 1): string{

        // percorre a lista de itens
        foreach($this->itens as $item){

            if($item->getTitulo() === $titulo && $item->isDisponivel()){

                // calcular valor do aluguel
                $valorAluguel = $item->calcularAluguel($dias);

                // Marcar como alugado
                $mensagem = $item->alugar();

                $this->salvarItens();

                return $mensagem . "Valor do aluguel: R$" .
number_format($valorAluguel, 2, ',', '.');
            }
        }
        return "Item não disponível";
    }
}

```

```

// Devolver item

public function devolverItem(string $titulo): string{

    // Percorrer a lista
    foreach($this->itens as $item){

        if($item->getTitulo() === $titulo && !$item->isDisponivel()){

            // disponibilizar o item
            $mensagem = $item->devolver();

            $this->salvarItens();
            return $mensagem;
        }
    }
    return "Item já disponível ou não encontrado.";
}

// Retorna a lista de itens

public function listarItens():array{
    return $this->itens;
}

// Calcular previsão do valor
public function calcularPrevisaoAluguel(string $tipo, int $dias):
float {

    if($tipo === 'Filme'){
        return (new Filme('', '', '')) ->calcularAluguel($dias);
    } elseif($tipo === 'Desenho'){
        return (new Desenho('', '', '')) ->calcularAluguel($dias);
    } elseif($tipo === 'Novela'){
        return (new Novela('', '', '')) ->calcularAluguel($dias);
    } else {
        return (new Serie('', '', '')) ->calcularAluguel($dias);
    }
}
}

```

Fonte: Autoria própria (2025).

A Figura 30 mostra a classe Locadora é responsável por gerenciar uma coleção de itens para aluguel, como filmes, séries, novelas e desenhos. Quando a classe é criada, ela carrega esses itens a partir de um arquivo JSON que armazena as informações.

Se o arquivo JSON existir, ela lê e interpreta o conteúdo para preencher a lista de itens; caso contrário, exibe mensagens de erro no log. Cada item é instanciado como um objeto específico (Filme, Série, Novela ou Desenho), e suas informações são configuradas, inclusive o status de disponibilidade.

Imagem 31 – UploadHandler.php

```
<?php
namespace Services;

class UploadHandler {
    private string $uploadDir;
    private array $allowedTypes = ['image/jpeg', 'image/png',
'image/gif'];
    private int $maxFileSize = 5242880; // 5MB

    public function __construct(string $uploadDir = 'img/uploads/') {
        $this->uploadDir = $uploadDir;
        if (!is_dir($this->uploadDir)) {
            mkdir($this->uploadDir, 0777, true);
        }
    }

    public function handleUpload(array $file): ?string {
        if (!isset($file['error']) || is_array($file['error'])) {
            throw new \RuntimeException('Invalid parameters.');
```

```

        if (!in_array($mimeType, $this->allowedTypes)) {
            throw new \RuntimeException('Invalid file format.');
```

```
        }

        $extension = array_search($mimeType, [
            'jpg' => 'image/jpeg',
            'png' => 'image/png',
            'gif' => 'image/gif',
        ], true);

        $filename = sprintf('%s.%s',
            sha1_file($file['tmp_name']),
            $extension
        );

        if (!move_uploaded_file(
            $file['tmp_name'],
            $this->uploadDir . $filename
        )) {
            throw new \RuntimeException('Failed to move uploaded file.');
```

```
        }

        return 'img/uploads/' . $filename;
    }
}

```

Fonte: Autoria própria (2025).

A Figura 31 mostra a classe UploadHandler é responsável por fazer o upload de arquivos, especialmente imagens. Ela define um diretório padrão para salvar os arquivos enviados, que é "img/uploads/". Se esse diretório não existir, a classe o cria automaticamente.

Ao fazer o upload, o método handleUpload recebe um arquivo e verifica se ele foi enviado corretamente, sem erros. Caso haja algum erro, ele lança uma exceção informando qual foi o problema, como arquivo não enviado ou tamanho maior que o permitido.

O método também verifica se o arquivo não ultrapassa o tamanho máximo definido, que é 5 megabytes. Depois, utiliza uma função para identificar o tipo MIME do arquivo e verifica se ele está dentro dos tipos permitidos, que são imagens nos formatos JPEG, PNG e GIF.

Se o arquivo for válido, a classe define a extensão correta com base no tipo MIME e cria um nome único para o arquivo, utilizando o hash SHA-1 do

conteúdo do arquivo. Em seguida, move o arquivo para a pasta de uploads com o nome gerado.

Se o upload for bem-sucedido, o método retorna o caminho relativo da imagem salva. Caso contrário, ele lança uma exceção indicando o problema ocorrido durante o processo.

Essa classe garante que apenas imagens válidas e seguras sejam armazenadas no servidor, organizando os arquivos e prevenindo erros comuns no upload.

13.3 views

Imagem 32 – landingpage.php

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <!-- Importando Bootstrap e estilos personalizados -->
  <link rel="stylesheet"
href="node_modules/bootstrap/dist/css/bootstrap.min.css">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.13.1/font/bootstrap-icons.min.css">
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.6/dist/css/bootstrap.min.
css" rel="stylesheet" integrity="sha384-
4Q6Gf2aSP4eDXB8Miphr37CMZZQ5oXLH2yaXMJ2w8e2ZtHT17GptT4jmdRuHDT"
crossorigin="anonymous">
  <!-- Fav Icon -->
  <link rel="shortcut icon" href="img/Logo-Streaming-Filmes-_1_.ico"
type="image/x-icon">
  <!-- css interno -->
  <style>
    @import
url('https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,20
0;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;
1,600;1,700;1,800;1,900&family=Roboto+Mono:ital,wght@0,100..700;1,100..700
&display=swap' rel="stylesheet');
    *{
      font-family: 'Poppins', sans-serif;
      color: #fff !important;
      padding: 0;
      margin: 0;
      box-sizing: border-box;
```

```
}

/* fonte para títulos */
.title-font{
    font-family: 'Poppins', sans-serif;
}

.bem{
    font-family: 'Poppins', sans-serif;
}

/* links */
a{
    color: #eee ;
    text-decoration: none;
}
a:hover{
    text-decoration: underline;
}

select,
::picker(select) {
appearance: base-select;
}

/* containers */
.common-container{
    background-color: #00000020 !important;
    border: 3px solid #fff ;
    border-radius: 5px;
}
.borderless{
    border: none;
}

.carrossel-posters {
    max-width: 300px;
    margin: 0 auto;
}

.poster-img {
    height: 400px;
    width: auto;
    object-fit: cover;
    border-radius: 8px;
}
```



```
/* botões */
.common-btn{
    background-color: #00000020;
    color: #fff;
    border: 3px solid #fff;
    border-radius: 5px;
    padding: 0.5rem 1rem;
    max-width: fit-content;
    align-self: center;
}
.common-btn:hover{
    background-color: #fff;
    color: #000 !important;
}
.common-btn:active{
    background-color: #ddd !important;
    color: #000000 !important;
    border: 3px solid #ddd !important;
}
.common-btn:hover .bi, .common-btn:active .bi{
    color: #000 !important;
}

/* inputs */
.common-input{
    background-color: #ffffff88;
    border: none;
    border-left: 4px solid #fff;
    border-bottom: 4px solid #fff;
    color: #fff;
}
.common-input:focus{
    background-color: #ffffffbb;
    color: #FDFCF7;
    box-shadow: none;
    border-left: 4px solid #F31111;
    border-bottom: 4px solid #F31111;
}
option.common-input{
    background-color: #313131;
}

/* faz os posters de filmes e series ficar do tamanho correto */
.fmsr-poster{
    height: auto;
    width: 100%;
    max-width: 150px;
    max-height: 230px;
    border-radius: 5px;
}
```

```
        object-fit: cover;
    }
    .mini-poster{
        max-height: 75px;
        max-width: 150px;
        border: 1px solid #fff;
        border-radius: 5px;
    }
    body{
        background-color: #313131;
    }
    .dropdown-menu{
        background-color: #313131;
    }
    .dropdown-menu .dropdown-item{
        background: none;
    }
    .nav-color{
        background-color: #000000;
    }
    .nav-logo img{
        width: 60px;
        height: 60px;
    }
    .cover{
        background: linear-gradient(90deg, #000000 0%, #00000000 50%),
url(../img/foto_filmes.png) no-repeat;
        background-size: cover;
    }
    .log-container{
        width: 30vw;
        height: 70vh;
    }
    .log-container h1{
        width: 100%;
        text-align: center;
        margin-top: .5em;
    }
    .log-form{
        height: 80%;
        display: flex;
        flex-direction: column;
        justify-content: space-evenly;
    }
    footer{
        background-color: #585858;
    }
    .listagem{
        background-color: #00000000 !important;
```

```

}
.listagem thead{
    border: 1px solid #fff;
    border-radius: 5px 5px 0 0;
}
.listagem tbody{
    border: 1px solid #fff;
    border-radius: 0 0 5px 5px;
}
.listagem th, .listagem td{
    background-color: #ffffff10;
}
.days-input{
    color: #000 !important;
}

.limite{
    color: #D22424 !important;
    text-decoration: underline dashed;
}
@media (max-width: 767px){
    .fmsr-poster {
        width: 100%;
        max-width: 200px;
        height: auto;
        border-radius: 5px;
    }
    .borderless-sm{
        border: none;
    }
    .log-container {
        width: 100%;
        max-width: 400px;
        height: auto;
        padding: 2em;
        margin: auto;
    }
}
</style>
<title>CineHome</title>
</head>
<body>
    <!-- main -->
    <main class="container-fluid cover p-md-5 pe-1 px-1">
        <div class="container mt-md-3 ms-md-5 p-4 pe-lg-1 col-md-5 col-sm-12 common-container borderless-sm">
            <h1 class="title-font">CineHome</h1>

```

```

        <p style="font-size: 1.6em;" class="mb-5 lh-lg">O lugar onde
        você<br>transforma sua casa em cinema!</p>
        <a href="login.php" class="btn btn-danger btn-sm">Entrar</a>
    </div>
</main>
<div class="container-fluid">
    <h3 class="title-font m-2">Um catálogo que abrange todos os
    gostos!</h3>

    <!-- negócios de filme série -->
    <div class="container-fluid">
        <p class="mb-4 mt-5 fs-3 text-uppercase text-
        center">Filmes</p>
        <div class="row row-cols-2 row-cols-md-3 row-cols-lg-4 g-3 px-
        3">
            <div class="col">
                
            </div>
            <div class="col">
                
            </div>
            <div class="col">
                
            </div>
            <div class="col">
                
            </div>
        </div>
        <p class="mb-4 mt-5 fs-3 text-uppercase text-
        center">Séries</p>
        <div class="row row-cols-2 row-cols-md-3 row-cols-lg-4 g-3 px-
        3">
            <div class="col">
                
            </div>
            <div class="col">
                
            </div>
            <div class="col">
                
            </div>
            <div class="col">

```

```

        
    </div>
</div>
<p class="mb-4 mt-5 fs-3 text-uppercase text-
center">Novelas</p>
    <div class="row row-cols-2 row-cols-md-3 row-cols-lg-4 g-3 px-
3">
        <div class="col">
            
        </div>
        <div class="col">
            
        </div>
        <div class="col">
            
        </div>
        <div class="col">
            
        </div>
    </div>
    <p class="mb-4 mt-5 fs-3 text-uppercase text-
center">Desenhos</p>
    <div class="row row-cols-2 row-cols-md-3 row-cols-lg-4 g-3 px-
3">
        <div class="col">
            
        </div>
        <div class="col">
            
        </div>
        <div class="col">
            
        </div>
        <div class="col">
            
        </div>
    </div>
</div>
<!-- Rodapé -->

```

```

        <footer class="d-flex flex-column align-items-center justify-content-
center mt-5">
            <div class="nav-logo"></div>
            <p class="lh-1"><strong>Entre em contato:</strong>
contact@cinehome.com </p>
            <p class="lh-1" style="font-size: small;">©2025, Cinehome.com |
Direitos Reservados</p>
        </footer>

        <!-- js do bootstrap -->
        <script
src="node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>

```

Fonte: Autoria própria (2025).

A Figura 32 mostra a página chama-se CineHome. No topo, aparece o título “CineHome” em destaque. Logo abaixo, há um texto que diz: “O lugar onde você transforma sua casa em cinema!” e um botão vermelho escrito “Entrar” que direciona para a página de login. Em seguida, há um subtítulo que afirma: “Um catálogo que abrange todos os gostos!”. Depois, são apresentadas quatro categorias: filmes, séries, novelas e desenhos. Em cada categoria, várias imagens de pôsteres são exibidas em grades que se adaptam ao tamanho da tela. No rodapé, encontra-se o logotipo do CineHome, um e-mail para contato (contact@cinehome.com) e a frase “©2025, Cinehome.com | Direitos Reservados”. O fundo da página é escuro, com elementos e textos em tons claros, e a tipografia é moderna e limpa para facilitar a leitura.

Imagem 33 – template.php

```

<?php

require_once __DIR__ . '/../services/Auth.php';

use Services\Auth;

$usuario = Auth::getUsuario();

?>
<!DOCTYPE html>
<html lang="pt-br">
<head>

```

```

<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<!-- Importando Bootstrap e estilos personalizados -->
<link rel="stylesheet"
href="node_modules/bootstrap/dist/css/bootstrap.min.css">
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.13.1/font/bootstrap-icons.min.css">
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.6/dist/css/bootstrap.min.
css" rel="stylesheet">
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle
.min.js"></script>

<!-- Fav icon -->
<link rel="shortcut icon" href="img/Logo-Streaming-Filmes-_1_.ico"
type="image/x-icon">
<!-- CSS INTERNO -->
<style>
    @import
url('https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,20
0;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;
1,600;1,700;1,800;1,900&family=Roboto+Mono:ital,wght@0,100..700;1,100..700
&display=swap& rel="stylesheet');
    *{
        font-family: 'Poppins', sans-serif;
        color: #fff !important;
        padding: 0;
        margin: 0;
        box-sizing: border-box;
    }
    select option {
color: black !important;
}

    /* fonte para títulos */
    .title-font{
        font-family: 'Poppins', sans-serif;
    }

    /* links */
    a{
        color: #eee ;
        text-decoration: none;
    }
    a:hover{
        text-decoration: underline;
    }

```

```

select,
::picker(select) {
appearance: base-select;
}

/* containers */
.common-container{
background-color: #C60A26 !important;
border-radius: 5px;
max-width: 500px;
}
.borderless{
border: none;
}
option{
color: #000;
}

/* botões */
.common-btn{
background-color: #00000020;
color: #fff;

border-radius: 5px;
padding: 0.5rem 1rem;
max-width: fit-content;
align-self: center;
}
.common-btn:hover{
background-color: #fff;
color: #000 !important;
}
.common-btn:active{
background-color: #ccc !important;
color: #000000 !important;
border: none !important;
}
.common-btn:hover .bi, .common-btn:active .bi{
color: #000 !important;
}

/* Botões minimalistas do carrossel */
.carousel-control-prev,
.carousel-control-next {
width: 30px;
height: 30px;
top: 50%;
transform: translateY(-50%);
background: none;
border: none;

```



```

padding: 0;
opacity: 0.8;
}

.carousel-control-prev-icon,
.carousel-control-next-icon {
  background-color: transparent;
  /* border: solid white; */
  /* border-width: 0 4px 4px 0; */
  width: 16px;
  height: 16px;
}

.carousel-control-prev-icon {
  transform: rotate(0deg);
}

.carousel-control-next-icon {
  transform: rotate(0deg);
}

.carousel-item img {
  height: auto;
  max-height: 230px;
  width: 100%;
  max-width: 150px;
  object-fit: cover;
  border-radius: 10px;
}

.carousel-item {
  padding: 0 1rem;
}

.carousel-control-prev-icon,
.carousel-control-next-icon {
  border-radius: 50%;
}

.category-section {
  margin: 2rem 0;
}

/* Redução do tamanho das imagens do carrossel de banners */
#banners-container .carousel-inner img {
  max-height: 580px;
  max-width: 95%;
  margin: 0 auto;
  object-fit: cover;
  border-radius: 8px;
}

```

```

option.common-input{
    background-color: #313131;
}

body{
    background-color: #313131;
}
.dropdown-menu{
    background-color: #313131;
}
.dropdown-menu .dropdown-item{
    background: none;
}
.dropdown-menu.custom-width {
    width: 10px;
}
.nav-color{
    background-color: #000000;
}
.nav-logo img{
    width: 75px;
    height: 75px;
}
.navbar-toggler {
    border:none;
}
.navbar-toggler:focus,
.navbar-toggler:active {
    outline: none;
    box-shadow: none;
}

.navbar-toggler-icon {
    background-image: url("data:image/svg+xml,%3csvg
xmlns='http://www.w3.org/2000/svg&#39; viewBox='0 0 30 30'%3e%3cpath
stroke='white' stroke-width='2' d='M4 7h22M4 15h22M4
23h22'/%3e%3c/svg%3e");
}
.cover{
    background: linear-gradient(90deg, #000000 0%, #00000000 50%),
url(img/foto_filmes.png) no-repeat;
    background-size: cover;
}

.log-container{
    width: 30vw;
    height: 70vh;
}

```

```

.log-container h1{
    width: 100%;
    text-align: center;
    margin-top: .5em;
}
.log-form{
    height: 80%;
    display: flex;
    flex-direction: column;
    justify-content: space-evenly;

}
.custom-footer {
background-color: #585858;
color: #fff;
text-align: center;
}

footer{
    background-color: #585858;
}

.footer-logo img {
width: 60px;
height: 60px;
}

.days-input{
    color: #000 !important;
}

.limite{
    color: #D22424 !important;
    text-decoration: underline dashed;
}
@media (max-width: 767px){
    .fmsr-poster{
        max-width: 400px;
        height: 200px;
        margin-top: 0.5em;
    }
    .borderless-sm{
        border: none;
    }
    .log-container{
        width: 80vw;
        height: 80vh;
    }
}
/* Responsividade */

```

```

        .footer-logo img {
            width: 50px;
            height: 50px;
        }

        .custom-footer p {
            font-size: 0.85rem;
        }

        #banners-container .carousel-inner img {
            max-height: 200px;
        }
    }

</style>
<title>CineHome</title>
</head>
<body>
    <!-- Barra de navegação -->
    <nav class="navbar navbar-expand-lg navbar-color p-3">
        <div class="container-fluid justify-content-center">
            <a href="home_adm.html" class="navbar-brand nav-logo mx-auto">
                
            </a>
            <button class="navbar-toggler text-white border-white"
type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-
controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse justify-content-center text-
center" id="navbarNav">
                <ul class="navbar-nav mb-2 mb-lg-0">
                    <li class="nav-item">
                        <a class="nav-link" href="#HOME">Home</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="#CATALOGO">Catálogo</a>
                    </li>
                    <?php if (Auth::isUser()): ?>
                    <li class="nav-item">
                        <a class="nav-link" href="#ALUGUE">Assinatura</a>
                    </li>
                    <?php endif; ?>
                    <?php if (Auth::isAdmin()): ?>
                    <li class="nav-item">
                        <a class="nav-link" href="#ALUGUE">Adicionar</a>
                    </li>
                    <?php endif; ?>
                </ul>
            </div>
        </div>
    </nav>

```

```

        <!-- Menu -->
        <div class="dropdown nav-item ms-lg-auto w-100 w-lg-auto">
            <div class="d-flex justify-content-center justify-content-
lg-end">
                <div class="dropdown d-flex justify-content-left">
                    <button class="btn common-btn dropdown-toggle w-
100 text-center text-lg-end px-3 "type="button" id="adminDropdown" data-bs-
toggle="dropdown" aria-expanded="false">
                        <?php if (Auth::isUser()): ?><i class="bi bi-
person-circle"></i> <?php endif; ?>
                        <?php if (Auth::isAdmin()): ?>
                            <i class="bi bi-stars"></i> <?php endif; ?> <?=
htmlspecialchars($usuario['username'])?>
                        </button>
                        <ul class="dropdown-menu dropdown-menu-end w-100
text-center text-lg-end" aria-labelledby="adminDropdown" style="min-width:
unset;">
                            <li>
                                <div class="d-flex align-items-center gap-3
user-info">
                                    <!-- Botão Sair com ícone usando Bootstrap
Icons -->
                                    <a href="?logout=1" class="btn d-flex
align-items-center gap-1 d-flex justify-content-center" >
                                        <i class="bi bi-box-arrow-right"></i>
                                        Sair
                                    </a>
                                </div>
                            </li>
                        </ul>
                    </div>
                </div>
            </div>
        </div>
    </nav>

    <!-- Título -->
    <div class="text-center my-4">
        <h2 class="title-font">CineHome</h2>
    </div>

    <!-- ##### HOME ##### -->

```

```

<div id="HOME">
    <!-- Título e subtítulo -->
    <div class="container my-5 text-center align-center d-flex
justify-content-center">
        <div class="card common-container borderless p-4">
            <div class="card-header fs-3 fw-bold"><span class="welcome-
text">Bem-vindo, <strong><?=
htmlspecialchars($usuario['username'])?></strong></span></div>
            <div class="card-body">
                <p class="card-text fs-5">Veja a recomendação de filme e
série da semana</p>
            </div>
        </div>
    </div>
    <!-- Carrossel de Banners -->
    <div class="container my-5" id="banners-container">
        <div id="slider" class="carousel slide" data-bs-
ride="carousel">
            <div class="carousel-inner">
                <div class="carousel-item active">
                    
                </div>
                <div class="carousel-item">
                    
                </div>
                <div class="carousel-item">
                    
                </div>
            </div>
            <button class="carousel-control-prev" type="button" data-bs-
target="#slider" data-bs-slide="prev">
                <span class="carousel-control-prev-icon" aria-
hidden="true"></span>
                <span class="visually-hidden">Anterior</span>
            </button>
            <button class="carousel-control-next" type="button" data-bs-
target="#slider" data-bs-slide="next">
                <span class="carousel-control-next-icon" aria-
hidden="true"></span>
                <span class="visually-hidden">Próximo</span>
            </button>
        </div>
    </div>
    <!-- Carrossel de Assinaturas -->
    <div class="container mb-4">

```

```

        <div class="card-header fs-5 fw-semibold">Minhas
Assinaturas</div>
        <div id="carouselAssinaturas" class="carousel slide mt-3"
data-bs-ride="carousel">
        <div class="carousel-inner text-center">
            <!-- Página 1 -->
            <div class="carousel-item active">
                <div class="row justify-content-center">
                    <div class="col-6 col-md-3 mb-3 d-flex justify-
content-center">
                        
                    </div>
                    <div class="col-6 col-md-3 mb-3 d-flex justify-
content-center">
                        
                    </div>
                    <div class="col-6 col-md-3 mb-3 d-flex justify-
content-center">
                        
                    </div>
                    <div class="col-6 col-md-3 mb-3 d-flex justify-
content-center">
                        
                    </div>
                </div>
            </div>
            <!-- Página 2 -->
            <div class="carousel-item">
                <div class="row justify-content-center">
                    <div class="col-6 col-md-3 mb-3 d-flex justify-
content-center">
                        
                    </div>
                    <div class="col-6 col-md-3 mb-3 d-flex justify-
content-center">
                        
                    </div>
                    <div class="col-6 col-md-3 mb-3 d-flex justify-
content-center">
                        
                    </div>
                </div>
            </div>
        </div>
    </div>

```

```

        <div class="col-6 col-md-3 mb-3 d-flex justify-
content-center">
            
        </div>
    </div>
</div>
<div class="carousel-control-prev" type="button" data-bs-
target="#carrosselAssinaturas" data-bs-slide="prev">
    <span class="carousel-control-prev-icon"></span>
</button>
<div class="carousel-control-next" type="button" data-bs-
target="#carrosselAssinaturas" data-bs-slide="next">
    <span class="carousel-control-next-icon"></span>
</button>
</div>
</div>
<!-- Texto Final -->
<div class="container my-5">
    <p class="fs-6 catalogo-texto">
        Já terminou de assistir ou quer ver mais filmes e séries?<br
/>
        Veja o nosso catálogo por mais filmes e séries!<br />
        <a href="catalogo.php" class="text-decoration-underline text-
info fw-semibold">Clique aqui</a> para ir ao catálogo.
    </p>
</div>
</div>

<!-- ##### FIM HOME ##### -->

<!-- ##### CATALOGO ##### -->
<?php if (Auth::isUser()): ?>
    <!-- Barra de pesquisa -->
    <div class="search-bar text-center mb-4">
        <form class="d-flex justify-content-center" role="search">
            <input
                class="form-control me-2 bg-dark border-0 text-white"
                type="search"
                placeholder="Buscar títulos..."
                aria-label="Buscar"
                style="max-width: 400px;"
            />
            <button class="btn btn-outline-dark" type="submit">
                <i class="bi bi-search"></i>
            </button>
        </form>
    </div>

```



```

<?php endif; ?>

<?php if (Auth::isUser()): ?>
<!-- Seções de conteúdo -->
<div class="container" id="CATALOGO">
    <!-- Filmes -->
    <div class="category-section">
        <h5 class="text-uppercase text-center">Filmes</h5>
        <div id="carouselFilmes" class="carousel slide" data-bs-
ride="carousel">
            <div class="carousel-inner">
                <div class="carousel-item active">
                    <div class="d-flex justify-content-center flex-wrap
gap-3">
                        
                        
                        
                        
                        
                    </div>
                </div>
                <div class="carousel-item">
                    <div class="d-flex justify-content-center flex-wrap
gap-3">
                        
                        
                        
                        
                        
                    </div>
                </div>
            </div>
            <button class="carousel-control-prev" type="button" data-
bs-target="#carouselFilmes" data-bs-slide="prev">
                <span class="carousel-control-prev-icon"></span>
            </button>

```

```

        <button class="carousel-control-next" type="button" data-
bs-target="#carouselFilmes" data-bs-slide="next">
            <span class="carousel-control-next-icon"></span>
        </button>
    </div>
</div>
<br>

<!-- Séries -->
<div class="category-section">
    <h5 class="text-uppercase text-center">Séries</h5>
    <div id="carouselSeries" class="carousel slide" data-bs-
ride="carousel">
        <div class="carousel-inner">
            <div class="carousel-item active">
                <div class="d-flex justify-content-center flex-wrap
gap-3">
                    
                    
                    
                    
                    
                </div>
            </div>
            <div class="carousel-item">
                <div class="d-flex justify-content-center flex-wrap
gap-3">
                    
                    
                    
                    
                    
                </div>
            </div>
        </div>
        <button class="carousel-control-prev" type="button" data-
bs-target="#carouselSeries" data-bs-slide="prev">
            <span class="carousel-control-prev-icon"></span>
        </button>
    </div>

```

```

        <button class="carousel-control-next" type="button" data-
bs-target="#carouselSeries" data-bs-slide="next">
            <span class="carousel-control-next-icon"></span>
        </button>
    </div>
</div>
<br>

<!-- Novelas -->
<div class="category-section">
    <h5 class="text-uppercase text-center">Novelas</h5>
    <div id="carouselNovelas" class="carousel slide" data-bs-
ride="carousel">
        <div class="carousel-inner">
            <div class="carousel-item active">
                <div class="d-flex justify-content-center flex-wrap
gap-3">
                    
                    
                    
                    
                    
                </div>
            </div>
            <div class="carousel-item">
                <div class="d-flex justify-content-center flex-wrap
gap-3">
                    
                    
                    
                    
                    
                </div>
            </div>
        </div>
        <button class="carousel-control-prev" type="button" data-
bs-target="#carouselNovelas" data-bs-slide="prev">
            <span class="carousel-control-prev-icon"></span>
        </button>
    </div>

```

```

        <button class="carousel-control-next" type="button" data-
bs-target="#carouselNovelas" data-bs-slide="next">
            <span class="carousel-control-next-icon"></span>
        </button>
    </div>
</div>
<br>

<!-- Desenhos -->
<div class="category-section">
    <h5 class="text-uppercase text-center">Desenhos</h5>
    <div id="carouselDesenhos" class="carousel slide" data-bs-
ride="carousel">
        <div class="carousel-inner">
            <div class="carousel-item active">
                <div class="d-flex justify-content-center flex-wrap
gap-3">
                    
                    
                    
                    
                    
                </div>
            </div>
            <div class="carousel-item">
                <div class="d-flex justify-content-center flex-wrap
gap-3">
                    
                    
                    
                    
                    
                </div>
            </div>
        </div>
        <button class="carousel-control-prev" type="button" data-
bs-target="#carouselDesenhos" data-bs-slide="prev">
            <span class="carousel-control-prev-icon"></span>

```

```

        </button>
        <button class="carousel-control-next" type="button" data-
bs-target="#carouselDesenhos" data-bs-slide="next">
            <span class="carousel-control-next-icon"></span>
        </button>
    </div>
</div>
</div>
<?php endif; ?>

<!-- ##### FIM CATALOGO ##### -->

<!-- ##### ALUGUE ##### -->
<div id="ALUGUE">
    <!-- Container principal -->
    <div class="container mb-4 mt-4">
        <div class="row justify-content-center align-items-start gy-4
gx-4">

            <?php if (Auth::isAdmin()): ?>
                <!-- Título -->
                <div class="text-center my-4">
                    <h2 class="title-font">
                        Adicionar
                    </h2>
                </div>

                <!-- Formulário Adicionar ao catálogo -->
                <div class="col-md-6 col-lg-5 d-flex justify-content-center">
                    <div class="p-3 common-container bg-dark text-white w-100"
style="max-width: 400px;">
                        <h4 class="text-center">Adicionar ao catálogo</h4>
                        <form method="POST" class="needs-validation"
enctype="multipart/form-data" novalidate>
                            <!-- Título -->
                            <div class="mb-3">
                                <label class="form-label">Título</label>
                                <input type="text" name="titulo" class="form-control
common-input text-black" required>
                                <div class="invalid-feedback">Campo obrigatório</div>
                            </div>
                            <!-- Sinopse -->
                            <div class="mb-3">
                                <label class="form-label">Sinopse</label>
                                <textarea class="form-control common-input text-black"
name="sinopse" rows="2" required></textarea>
                                <div class="invalid-feedback">Campo obrigatório</div>
                            </div>

```

```

        <!-- Gênero -->
        <div class="mb-3">
            <label class="form-label">Gênero</label>
            <select name="genero" class="form-select common-input
text-black" required>
                <option value="" hidden selected aria-
invalid="true">Selecione</option> <option value="acao" class="text-
black">Ação</option>
                <option value="romance" class="text-
black">Romance</option> <option value="comedia" class="text-
black">Comédia</option>
                <option value="terror" class="text-
black">Terror</option> <option value="drama" class="text-
black">Drama</option>
                <option value="ficcao" class="text-black">Ficção
Científica</option> <option value="animado" class="text-
black">Animação</option>
                <option value="infantil" class="text-
black">Infantil</option>
            </select>
        </div>
        <!-- Tipo -->
        <div class="mb-3">
            <label class="form-label">Tipo</label>
            <select name="tipo" class="form-select common-input
text-black" required>
                <option value="" hidden selected aria-
invalid="true">Selecione</option>
                <option value="filme" class="text-
black">Filme</option>
                <option value="serie" class="text-
black">Série</option>
                <option value="novela" class="text-
black">Novela</option>
                <option value="desenho" class="text-
black">Desenho</option>
            </select>
        </div>
        <!-- Imagem -->
        <div class="mb-3">
            <label class="form-label">Imagem</label>
            <select name="imagem_existente" class="form-select
common-input text-black mb-2 " style="color: black;">
                <option value="" class="text-black">Selecione uma
imagem existente (opcional)</option>
            <?php
                $imagens = glob("../img/*.{jpg,jpeg,png,gif}",
GLOB_BRACE);
                foreach ($imagens as $imagem) {

```

```

        $nome = basename($imagem);
        echo "<option value='img/{ $nome }'>$nome</option>";
    }
    ?>
</select>
</div>
<button type="submit" class="btn common-btn w-100 mt-2"
name="adicionar">Adicionar</button>
    <?php if(isset($_POST['cadastrar']))){echo "<p>item
adicionado com sucesso</p>";}?>
</form>
</div>
</div>
<?php endif; ?>

<br>
<br>

<?php if (Auth::isUser()): ?>
<!-- Título -->
<div class="text-center my-4">
    <h2 class="title-font">
        Previsão
    </h2>
</div>
<?php endif; ?>

<!-- Formulário Calcular Preço -->
<div class="col-<?= Auth::isAdmin() ? 'md-6':'12' ?> col-lg-5 d-
flex justify-content-center">
    <div class="p-3 common-container bg-dark text-white w-100"
style="max-width: 400px;">
        <h4 class="text-center">Calcular Preço</h4>
        <form method="post" class="needs-validation mt-3"
novalidate>
            <!-- Tipo -->
            <div class="mb-3">
                <label class="form-label">Tipo</label>
                <select name="tipoCalculo" class="form-select common-
input text-black" required>
                    <option value="" selected hidden>Selecione</option>
                    <option value="filme" class="text-
black">Filme</option>
                    <option value="serie" class="text-
black">Série</option>
                    <option value="novela" class="text-
black">Novela</option>
                    <option value="desenho" class="text-
black">Documentário</option>

```

```

        </select>
    </div>
    <!-- Dias -->
    <div class="mb-3">
        <label class="form-label">Tempo em dias</label>
        <input type="number" name="diasCalculo" class="form-
control common-input text-black " min="1" required>
    </div>
    <button class="btn common-btn w-100 mt-2" type="submit"
name="calcular">Calcular</button>

    </form>
</div>
</div>
<br>
<br>
<?php if ($mensagem):?>
    <div class="alert alert-danger alert-dismissible fade show
text-danger" role="alert">
        <?= htmlspecialchars($mensagem) ?>
        <button type="button" class="btn-close" data-bs-
dismiss="alert" aria-label="Close"></button>
    </div>
<?php endif; ?>
<br>
<br>

<!-- Cards-->
<div class="row justify-content-center align-items-start gy-4 gx-4
mt-5">
    <!-- Itens cadastrados -->
    <?php foreach ($locadora->listarItens() as $item): ?>
        <div class="col-md-6 col-lg-5 d-flex justify-content-center">
            <div class="card bg-dark text-white w-100" style="max-width:
400px;">
                getTitulo()) ?>">
                <div class="card-body text-center">
                    <h4 class="card-title"><?= htmlspecialchars($item-
>getTitulo()) ?></h4>

                    <p><?= $item instanceof \Models\Filme ? 'Filme' : (
                    $item instanceof \Models\Serie ? 'Serie' : (
                    $item instanceof \Models\Novela ? 'Novela' : 'Desenho')
                    );?>
                    </p>
                </div>
            </div>
        </div>
    <!-- Formulário de ações -->

```



```

        <form method="POST" class="btn-group-actions d-flex flex-
column gap-2">
            <!-- Campo oculto para identificar o item -->
            <input type="hidden" name="titulo" value="<?=
htmlspecialchars($item->getTitulo()) ?>">
            <input type="hidden" name="genero" value="<?=
htmlspecialchars($item->getGenero()) ?>">
            <div class="d-flex gap-2 justify-content-center">
                <span class="badge bg-<?= $item->isDisponivel() ?
'success' : 'warning' ?>">
                    <?= $item->isDisponivel() ? 'Disponível' : 'Alugado'
?>
                </span>
            </div>
            <?php if (Auth::isAdmin()): ?>
            <?php if ($item->isDisponivel()): ?>
                <!-- Aluguel -->
                <div class="d-flex gap-2 justify-content-center">
                    <input type="number" name="dias" class="form-control
form-control-sm days-input" value="1" min="1" required style="width:
60px;">
                    <button class="btn btn-primary btn-sm" type="submit"
name="alugar">Alugar</button>
                </div>
            <?php else: ?>
                <!-- Devolução -->
                <div class="d-flex gap-2 justify-content-center">
                    <button type="submit" name="devolver" class="btn btn-
warning btn-sm">Devolver</button>
                </div>
            <?php endif; ?>
            <!-- Ações Admin -->
            <div class="d-flex gap-2 justify-content-center mt-2">
                <button class="btn btn-secondary btn-sm" type="submit"
name="editar">Editar</button>
                <button class="btn btn-danger btn-sm" type="submit"
name="deletar">Deletar</button>
            </div>
            <?php endif; ?>
            <?php if (Auth::isUser()): ?>
            <!-- Botão para abrir o modal -->
            <button type="button" class="btn btn-danger btn-sm mt-2"
data-bs-toggle="modal" data-bs-target="#modalSinopse<?= md5($item-
>getTitulo()) ?>">
                Ver mais
            </button>
            <?php endif; ?>

        </form>

```

```

        </div>
        <?php if (Auth::isUser()): ?>
            <!-- Modal de Sinopse -->
            <div class="modal fade" id="modalSinopse" ?= md5($item-
>getTitulo()) ?>" tabindex="-1" aria-labelledby="sinopseLabel" ?=
md5($item->getTitulo()) ?>" aria-hidden="true">
                <div class="modal-dialog">
                    <div class="modal-content bg-dark text-white">
                        <div class="modal-header">
                            <h5 class="modal-title" id="sinopseLabel" ?=
md5($item->getTitulo()) ?>">
                                <?= htmlspecialchars($item->getTitulo()) ?>
                            </h5>
                            <button type="button" class="btn-close btn-close-
white" data-bs-dismiss="modal" aria-label="Fechar"></button>
                        </div>
                        <div class="modal-body">
                            <p><?= nl2br(htmlspecialchars($item-
>getSinopse())) ?></p>
                        </div>
                    </div>
                </div>
            <?php endif; ?>
        </div>
    </div>

    <!-- ##### FIM ALUGUE ##### -->

</div>

<script>
    document.querySelector(".search-bar form").addEventListener("submit",
function (e) {
    e.preventDefault();
    const searchValue =
e.target.querySelector("input").value.toLowerCase();

    document.querySelectorAll(".carousel-inner .carousel-
item").forEach((item) => {
        const images = item.querySelectorAll("img");
        let hasMatch = false;

        images.forEach((img) => {
            const altText = img.alt.toLowerCase();

```

```

        const parent = img.parentElement;

        if (altText.includes(searchValue)) {
            img.style.display = "";
            hasMatch = true;
        } else {
            img.style.display = "none";
        }
    });

    item.style.display = hasMatch ? "block" : "none";
});
});
</script>

<!-- Rodapé -->
<footer class="d-flex flex-column align-items-center justify-content-center">
    <div class="nav-logo"></div>
    <p class="lh-1"><strong>Entre em contato:</strong>
contact@cinhome.com </p>
    <p class="lh-1" style="font-size: small;">©2025, Cinehome.com |
Direitos Reservados</p>
</footer>

<!-- js do bootstrap -->
<script
src="node_modules/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>

```

Fonte: Autoria própria (2025).

A Figura 33 mostra dois carrosséis diferentes — um de novelas e outro de desenhos animados — em uma única seção chamada "Novelas e Desenhos". O carrossel agora mostra imagens de ambos os tipos de conteúdo lado a lado, divididas em três slides. Cada slide contém um conjunto de imagens, com algumas novelas e alguns desenhos juntos. O layout usa o Bootstrap para exibir as imagens com espaçamento e adaptação ao tamanho da tela, garantindo que elas fiquem organizadas e responsivas. Com isso, o catálogo visualmente agrupa esses dois gêneros em uma apresentação contínua, mais compacta e fácil de navegar.

14 json.sprint

14.1 filmes.php

Imagem 34 – filmes.php

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Itens Json</title>
</head>
<body>
  <h2>Cadastro de Filmes</h2>
  <form action="salvar.php" method="post">
    Titulo: <input type="text" name="titulo" required><br><br>
    Tipo: <input type="text" name="tipo" required><br><br>
    Genero: <input type="text" name="genero" required><br><br>
    Status: <input type="text" name="status" required><br><br>
    Ano: <input type="date" name="ano">
    <button type="submit">Salvar</button>
  </form>

  <h2>Filmes cadastrados</h2>
  <ul>
    <?php
      $filmes = json_decode(file_get_contents("filmes.json"), true);

      if(!empty ($filmes)){
        foreach($filmes as $filme){
          echo "<li>{$filme['titulo']} - {$filme['tipo']} -
{$filme['genero']} - {$filme['status']}- {$filme['ano']}</li>";
        }
      } else {
        echo "<li>Nenhum item cadastrado</li>";
      }
    <?>
  </ul>
</body>
</html>
```

Fonte: Autoria própria (2025).

A Figura 34 mostra este código cria uma página web para cadastro e exibição de filmes. Ele contém um formulário onde o usuário pode inserir informações sobre um filme, como título, tipo, gênero, status e ano. Quando o

formulário é enviado, os dados são enviados para um arquivo PHP que deve salvar as informações em um arquivo JSON chamado filmes.json.

Na mesma página, há um trecho em PHP que lê o arquivo filmes.json, transforma seu conteúdo em uma lista de filmes e exibe essa lista para o usuário. Se não houver nenhum filme cadastrado, uma mensagem informando isso será mostrada.

Em resumo, o código permite adicionar novos filmes a um arquivo JSON e exibir os filmes já cadastrados, funcionando como um sistema simples de cadastro e listagem usando HTML, PHP e JSON.

14.2 salvar.php

Imagem 35 – salvar.php

```
<?php
// converte o json para um array em php
$filme = json_decode(file_get_contents("itens.json"),
true);

// verificacao da conversão de json para php
if(!is_array($filme)){
    $filme = [];
}

$novoFilme = [
    // pega as informações do formulario
    "titulo" => $_POST["titulo"],
    "tipo" => $_POST["tipo"],
    "genero" => $_POST["genero"],
    "status" => $_POST["status"],
    "ano" => $_POST["ano"]
];

//recebe e guarda as informacoes do formulario
$filme [] = $novoFilme;

//converte o php em json
file_put_contents("itens.json" ,json_encode($filme, JSON_PRETTY_PRINT));

echo "Dados salvos com sucesso ! <a href=filmes.php>Voltar</a>"
```

```

?>

<?php
// converte o json para um array em php
$dados = json_decode(file_get_contents("user.json"),
true);

// verificação da conversão de json para php
if(!is_array($dados)){
    $dados = [];
}

$novoDado = [
    // pega as informações do formulario
    "nome" => $_POST["nome"],
    "senha" => $_POST["senha"]
];

//recebe e guarda as informações do formulario
$dados [] = $novoDado;

//converte o php em json
file_put_contents("user.json" ,json_encode($dados, JSON_PRETTY_PRINT));

echo "Dados salvos com sucesso ! <a href=usuarios.php>Voltar</a>"

?>

```

Fonte: Autoria própria (2025)

A Figura 35 mostra os códigos PHP apresentados têm como objetivo cadastrar dados de filmes e usuários, armazenando essas informações em arquivos JSON. No primeiro trecho, os dados de um novo filme — incluindo título, tipo, gênero, status e ano — são capturados através de um formulário e adicionados a um array que representa o conteúdo do arquivo itens.json. Esse array é atualizado e salvo novamente no arquivo, garantindo que os dados sejam mantidos entre sessões.

Da mesma forma, o segundo trecho realiza o cadastro de usuários. Ele coleta o nome e a senha enviados por um formulário, adiciona essas informações ao conteúdo do arquivo user.json e salva os dados atualizados no formato JSON. Ambos os scripts usam `json_decode` e `json_encode` com `JSON_PRETTY_PRINT` para salvar os dados de forma legível.

Esses códigos oferecem uma solução simples de persistência de dados sem a necessidade de banco de dados, ideal para aplicações pequenas, testes ou protótipos.

14.3 usuarios.php

Imagem 36 – salvar.php

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>UserJson</title>
</head>
<body>
  <h2>Cadastro de usuarios</h2>
  <form action="salvar.php" method="post">
    Nome: <input type="text" name="nome" required><br><br>
    Senha: <input type="password" name="senha" required><br><br>
    <button type="submit">Salvar</button>
  </form>
</body>
</html>
```

Fonte: Autoria própria (2025).

A Figura 36 mostra a página HTML que exibe um formulário para o cadastro de usuários. Ela contém dois campos: um para inserir o nome e outro para a senha do usuário. O formulário envia os dados via método POST para o arquivo salvar.php, com `type="password"` se desejar ocultar os caracteres digitados e aceitar letras.

Este formulário é útil em sistemas simples onde os dados dos usuários são salvos sem a necessidade de um banco de dados relacional.