UNIVERSIDADE ESTADUAL DO PIAUÍ
CENTRO DE TECNOLOGIA E URBANISMO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Antônio Rodrigues dos Santos Júnior

MoviesChain - Avaliação de filmes com Blockchain

TERESINA
2025

Antônio Rodrigues dos Santos Júnior

MoviesChain - Avaliação de filmes com Blockchain

Monografia de Trabalho de Conclusão de Curso apresentado na Universidade Estadual do Piauí – UESPI como parte dos requisitos para conclusão do Curso de Bacharelado em Ciência da Computação.

Orientador: Constantino Augusto Dias Neto

TERESINA

2025

S237m Santos Junior, Antonio Rodrigues dos.
Movieschain - avaliação de filmes com blockchain / Antonio Rodrigues Dos Santos Junior. - 2025.
75 f.: il.

Monografia (graduação) - Universidade Estadual do Piauí-UESPI, Bacharelado em Ciências da Computação, Campus Poeta Torquato Neto, Teresina-PI, 2025.
"Orientador: Prof. Dr. Constantino Augusto Dias Neto".

1. Blockchain. 2. Avaliação de filmes. 3. Contratos inteligentes. 4. Ethereum. 5. Sistemas de reputação. I. Dias Neto, Constantino Augusto . II. Título.

CDD 004.07

Ficha elaborada pelo Serviço de Catalogação da Biblioteca da UESPI
Nayla Kedma de Carvalho Santos (Bibliotecário) CRB-3ª/1188

MoviesChain - Avaliação de filmes com Blockchain

Antônio Rodrigues dos Santos Júnior

Monografia de Trabalho de Conclusão de Curso apresentado na Universidade Estadual do Piauí – UESPI como parte dos requisitos para conclusão do Curso de Bacharelado em Ciência da Computação.

Constantino Augusto Dias Neto, Dsc.
Orientador

Nota da Banca Examinadora: **9,0**

Banca Examinadora:

Constantino Augusto Dias Neto, Dsc.
Presidente

Liliam Barroso Leal, Dsc.
Membro

Maurício Rêgo da Mota Rocha, Dsc.
Membro

AGRADECIMENTOS

Agradeço, primeiramente, aos meus pais, Maria Aparecida e Antônio, por sempre acreditarem em mim e estarem ao meu lado nos momentos em que mais precisei.

Aos colegas e amigos que fiz ao longo do curso de Ciência da Computação na UESPI, levo comigo não apenas os conhecimentos adquiridos, mas também as experiências e amizades que marcaram essa jornada.

Aos professores do curso, pela contribuição fundamental na minha formação acadêmica e profissional.

E, por fim, ao meu orientador, Constantino, por estar sempre disponível quando necessário e pelo apoio contínuo durante todo o desenvolvimento deste e de outros trabalhos que realizei durante o curso.

“A dúvida é o princípio da sabedoria.“
(Aristóteles)

.

RESUMO

As plataformas de avaliação de filmes atuais possuem um papel importante nas tomadas de decisões dos usuários, porém elas enfrentam diversas dificuldades relacionadas à manipulação, falta de transparência e práticas de ódio direcionado, como o *review bombing*, que comprometem a credibilidade desses sistemas. Diante desse cenário, é proposto o desenvolvimento do MoviesChain, um sistema de avaliação de filmes baseado em *blockchain* e contratos inteligentes. O objetivo é garantir a unicidade e transparência das avaliações e possibilitar a auditoria pública realizada pelos próprios usuários. A pesquisa adota abordagem qualitativa e exploratória, com fundamentação teórica sobre plataformas de avaliação centralizadas, *blockchain* e sistemas correlatos. O desenvolvimento segue os princípios de engenharia de software, incluindo levantamento de requisitos, elaboração de casos de uso e modelagem de classes. A aplicação foi construída com *back-end* em Java, *front-end* em Vue.js e contratos inteligentes implantados na *blockchain* Ethereum Sepolia. Os resultados demonstram a viabilidade técnica de uma plataforma descentralizada para avaliação de filmes, evidenciam suas limitações e apontam melhorias para trabalhos futuros.

Palavras-chave: *Blockchain*, Avaliação de Filmes, Contratos Inteligentes, Ethereum, Sistemas de Reputação.

ABSTRACT

Current film review platforms face several challenges related to manipulation, lack of transparency, and targeted hate practices such as *review bombing*, which undermine the credibility of these systems. In response to this scenario, the development of MoviesChain is proposed — a film review system based on *blockchain* and smart contracts. The goal is to ensure the uniqueness and transparency of reviews while enabling public auditing carried out by the users themselves. The research adopts a qualitative and exploratory approach, with theoretical foundations on centralized review platforms, *blockchain*, and related systems. The development follows software engineering principles, including requirements gathering, use case design, and class modeling. The application was built with a Java-based *back-end*, Vue.js *front-end*, and smart contracts deployed on the Ethereum Sepolia *blockchain*. The results demonstrate the technical feasibility of a decentralized film review platform, highlight its limitations, and suggest improvements for future work.

Keywords: *Blockchain*, Movie Rating, Smart Contracts, Ethereum, Reputation Systems.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABI	Application Binary Interface
API	Application Programming Interface
CSS	Cascading Style Sheets
dApps	Decentralized Applications
ETH	Ether
EVM	Ethereum Virtual Machine
HTML	HyperText Markup Language
IMDb	Internet Movie Database
JSON	JavaScript Object Notation
JWT	JSON Web Token
NFT	Non-Fungible Tokens
POS	Proof of Stake
POW	Proof of Work
SFC	Single File Components
SPA	Single Page Application
UML	Unified Modeling Language
ZKP	Zero Knowledge Proofs

SUMÁRIO

1 INTRODUÇÃO

A avaliação de produtos da indústria de entretenimento, tais como filmes, séries e videogames, virou uma prática comum na era digital. Plataformas como Internet Movie Database (IMDb), Rotten Tomatoes e Metacritic possuem um alto impacto no consumo do público. As médias das avaliações deixadas nesses ambientes são comumente utilizadas para validar a qualidade dos filmes, o que afeta diretamente o desempenho comercial dos mesmos. De acordo com Watson *et al.* (2018, apud Cantone *et al.*, 2024), as avaliações na internet são uma das principais fontes de informação para o consumidor contemporâneo.

Apesar do alto uso dessas plataformas pelos consumidores, várias falhas comprometem a confiabilidade desses sistemas. A falta de tecnologias de autenticação mais rigorosas, combinada com a facilidade de criar múltiplas contas, permite que usuários realizem avaliações repetidas ou coordenadas, distorcendo os resultados. Hazim et al. (2018) mostram que avaliações falsas podem resultar em grandes prejuízos financeiros, além de afetar a confiança dos consumidores. Um exemplo é o fenômeno conhecido como *review bombing*, que ocorre quando são feitas avaliações negativas ou positivas em massa, com o objetivo de manipular o resultado da média das avaliações. No caso do jogo *The Last of Us Part II*, analisado por Cantone et al. (2024), 79% dos usuários envolvidos criaram contas exclusivamente para avaliar o jogo de forma negativa, sem qualquer histórico prévio na plataforma.

Neste contexto, torna-se necessário reavaliar a maneira como as avaliações online são efetuadas. A *blockchain* é um banco de dados digital distribuído que protege e interliga uma lista de blocos ordenados por meio de métodos criptográficos (Yang et al., 2023). Por não permitir alterações nos blocos já enviados para a rede, ela garante que essas informações não serão manipuladas. Esse atributo torna a *blockchain* uma tecnologia bastante apropriada para sistemas onde a confiança no conteúdo (como as avaliações) deve ser mantida de maneira independente, sem uma autoridade central.

Este trabalho apresenta o MoviesChain, um sistema de avaliação de filmes que utiliza a *blockchain* para armazenamento das avaliações. A proposta tem como objetivo minimizar as questões de manipulação e *review bombing* encontradas em plataformas convencionais, utilizando contratos inteligentes para garantir que cada usuário possa registrar apenas uma avaliação por obra. O estudo utiliza uma metodologia aplicada e exploratória, fundamentada em revisão de literatura e desenvolvimento experimental da solução. Durante o estudo, são discutidos os princípios técnicos, a execução do sistema e uma avaliação da aplicabilidade da *blockchain* em sistemas de avaliação.

1.1 Problemática e Justificativa

O aumento da popularidade das avaliações online estabeleceu essas plataformas como um meio direto para influenciar a tomada de decisão dos consumidores. Porém, o método de armazenamento tradicional dessas avaliações, como IMDb e Rotten Tomatoes, tem se mostrado cada vez mais suscetível a diversas manipulações. A falta de sistemas de autenticação mais robustos possibilita que usuários criem várias contas para realizar avaliações repetidas ou ataques coordenados, como o procedimento conhecido como *review bombing*. Isso pode afetar a imagem dos filmes de maneira proposital e injusta, colocando em risco a credibilidade do sistema.

Levando em conta o aumento do impacto das avaliações online na escolha dos consumidores, é crucial garantir que essas informações representem verdadeiramente a opinião dos avaliadores. A integridade dos sistemas de reputação é afetada por ações como a utilização de várias contas e ataques coordenados, que alteram os resultados e afetam a confiança dos usuários. Plataformas com sistemas de armazenamento centralizado não possuem mecanismos eficientes para prevenir esses ataques.

Nesse contexto, surge o questionamento: como garantir uma maior autenticidade, unicidade e transparência nas avaliações de filmes online? Por esse motivo, o uso de uma tecnologia descentralizada como a *blockchain* para garantir a integridade, transparência e auditoria dos sistemas de avaliação de filmes torna-se fundamental.

1.2 Objetivo Geral

Desenvolver uma plataforma de avaliação de filmes utilizando tecnologia *blockchain*, assegurando assim autenticidade, unicidade e transparência nas avaliações feitas pelos usuários.

1.3 Objetivos Específicos

- Avaliar as questões de confiabilidade existentes em plataformas de avaliação atuais.
- Analisar os princípios técnicos da *blockchain* e sua utilização em sistemas de avaliação.
- Identificar os requisitos funcionais e técnicos para a criação do sistema MoviesChain.
- Implementar um contrato inteligente para o registro único de avaliação por filme.

- Criar uma interface web para avaliação dos filmes com autenticação através do MetaMask.
- Configurar o sistema na rede de testes Ethereum Sepolia e mostrar sua operação.
- Avaliar os ganhos e as limitações trazidas pelo MoviesChain.

1.4 Estrutura do trabalho

O Capítulo 2 apresenta o Referencial Teórico sobre os temas relacionados à pesquisa. Serão explicados os principais conceitos sobre plataformas de avaliação online, problemas de confiabilidade como avaliações falsas e *review bombing*, a comparação entre os tipos de sistema de reputação, a tecnologia *blockchain*, contratos inteligentes e a análise de alguns trabalhos relacionados.

No Capítulo 3, a metodologia utilizada no desenvolvimento é detalhada. Nele, é definido o tipo de pesquisa e suas características, é feito um levantamento bibliográfico e é explicada a forma como será feita a modelagem da aplicação. Também são apresentadas as tecnologias utilizadas na implementação da solução.

O Capítulo 4 descreve o desenvolvimento completo do MoviesChain. Nele, é mostrada a modelagem da aplicação, os requisitos, casos de uso, diagramas de classes e arquitetura. Também é mostrado como foram implementados os componentes principais do MoviesChain: o contrato inteligente, o *back-end* e o *front-end*. Após a implementação, todo o funcionamento do sistema é demonstrado por meio de figuras.

Por fim, o Capítulo 5 apresenta as conclusões do trabalho, destacando as vantagens e limitações encontradas na implementação do sistema e as contribuições da pesquisa para a área.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta a fundamentação teórica necessária para compreender os conceitos, tecnologias e problemáticas que embasam o desenvolvimento do MoviesChain

2.1 Plataformas de avaliação online

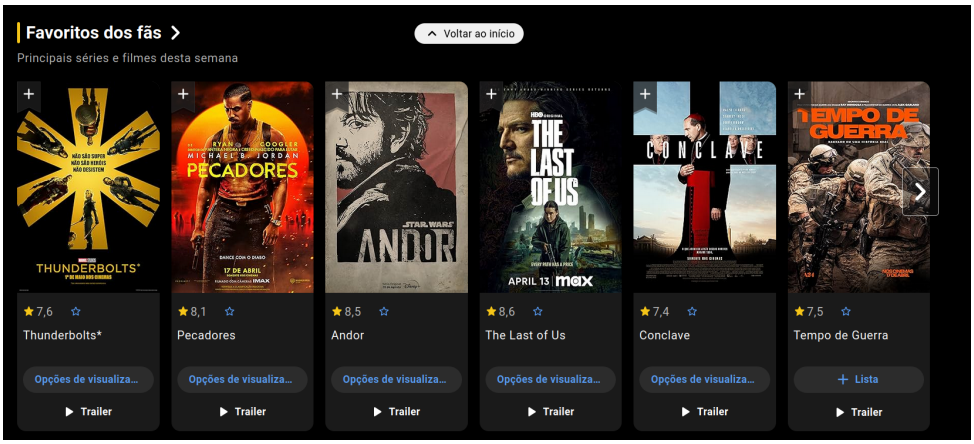
O aumento do uso da internet levou ao surgimento de plataformas online voltadas para a avaliação de filmes. Sites como o IMDb criaram métodos de avaliação compartilhada focando nas opiniões dos consumidores. Com o passar do tempo, plataformas como Rotten Tomatoes e Metacritic também se tornaram populares, aumentando a variedade e desenvolvendo novas formas de avaliação. Essas plataformas são exemplos comuns do que Tadelis (2016) caracteriza como sistemas de reputação online: ferramentas que recolhem e consolidam opiniões de usuários para criar confiança e orientar decisões.

Essas plataformas ficaram ainda mais populares após o boom das redes sociais e ferramentas de pesquisa. As avaliações deixadas nessas plataformas, além de serem utilizadas para registro pessoal, também passaram a ter valor cultural e comercial para o item avaliado. Aridor, Che e Salz (2022) indicam que as sugestões online afetam de forma significativa a atenção e o comportamento do consumidor, orientando suas decisões mesmo antes de terem contato com o conteúdo analisado.

Esse efeito nas escolhas dos usuários é facilitado pelos próprios métodos de avaliação utilizados por essas plataformas. Os usuários deixam suas avaliações nos produtos através de notas e comentários, utilizando valores numéricos ou componentes visuais como quantidades de estrelas ou tomates. Então, o sistema se vale das avaliações armazenadas para calcular uma média global para o filme, representando a avaliação da maioria sobre o item avaliado. De acordo com Sharkey, Hsu e Kovács (2022), os agregadores de críticas atuam como mediadores de informação, condensando diversos julgamentos individuais para formar a visão coletiva sobre o produto.

A interface do IMDb oferece uma maneira prática de visualizar a utilização dos sistemas de reputação. A plataforma apresenta a média das notas dadas pelos usuários, normalmente representada por uma escala de 0 a 10, com uma estrela amarela realçando a pontuação final. Conforme demonstrado na Figura 1, esse modelo exibe a visão geral do público acerca de cada obra.

Figura 1 – Média geral dos filmes/séries baseada nas avaliações dos usuários no IMDb.



Fonte: Captura de tela do IMDb, 2025.

Para que esse tipo de informação seja realmente confiável e útil, é crucial que as avaliações atendam a padrões mínimos de autenticidade e qualidade. A moderação de conteúdo passa a ser relevante para garantir a integridade das informações. Conforme Jiang, Ravichandran e Kuruzovich (2023), a clareza nos processos de moderação pode ter um impacto considerável no comportamento dos usuários, impactando tanto a quantidade quanto a qualidade das avaliações enviadas.

Assim que a confiabilidade das avaliações é garantida, elas começam a impactar diretamente as decisões dos usuários. É comum que uma média alta para um filme influencie na avaliação positiva final de um usuário, ao passo que críticas negativas podem provocar rejeição antes mesmo de ter contato com o mesmo.

Segundo Cantone et al. (2024), as plataformas de reputação online começaram a influenciar significativamente as decisões dos consumidores, podendo influenciar sua visão sobre as obras mesmo antes de serem consumidas. Neste contexto, é possível observar que os sistemas de reputação online passaram a ter papel importante para o consumo de alguns tipos de mídias na era digital, funcionando como uma espécie de filtragem de qualidade. Porém, mesmo tendo grande importância, as plataformas de avaliação ainda apresentam problemas estruturais que colocam em xeque a eficiência delas, em pontos essenciais como a transparência e integridade dos processos de avaliação.

2.2 Problemas de confiabilidade: Avaliações falsas e *review bombing*

A falta de métodos mais robustos de autenticação acaba contribuindo para o aumento da fraude nas plataformas. Sistemas com autenticação fraca possibilitam que usuários mal-intencionados divulguem avaliações falsas de forma frequente, o que acaba prejudicando a imagem dos itens avaliados. Redes coordenadas de *bots* ou geração de contas falsas por meio de *scripts* são problemas reais. Isso cria um cenário onde avaliações criadas por usuários reais se misturam com avaliações geradas por contas falsas, afetando não só a imagem do filme, mas também a credibilidade da plataforma. Hazim et al. (2018, p. 6), explicam que:

> "Avaliações falsas podem induzir o consumidor a tomar decisões de compra equivocadas, o que pode, em última instância, danificar a imagem dos produtos e dos vendedores."[1]

Esse problema já foi amplamente observado em sites de comércio eletrônico, nos quais o impacto das avaliações sobre a reputação e o consumo é igualmente relevante. Como afirmam Wu et al. (2020, p. 1), :

> "Avaliações falsas on-line no comércio eletrônico afetam significativamente os consumidores on-line, os comerciantes e, como resultado, a eficiência do mercado."[2]

O *review bombing* é um método coordenado de manipulação da reputação. Ele ocorre quando uma grande quantidade de avaliações são realizadas por perfis em um curto intervalo de tempo, com o objetivo de prejudicar a imagem do item avaliado. Cantone et al. (2024) estudaram o caso do jogo *The Last of Us Part II* e identificaram que 79% das contas envolvidas foram criadas exclusivamente para fazer avaliações negativas do jogo. Isso prejudica a confiança nos sistemas, uma vez que a média de avaliações não representa a opinião real dos usuários. Wu et al. (2020) advertem que iniciativas como essas prejudicam o mercado, ao produzir informações distorcidas que impactam a tomada de decisões do usuário. Portanto, o *review bombing* é um perigo não só técnico, mas também social.

Em face de manipulações em grande escala, como o *review bombing*, as táticas de moderação das plataformas centralizadas geralmente se mostram ineficientes diante de avaliações falsas. Wu et al. (2020, p. 10), afirmam que:

---

[1] No original: "Fake reviews can mislead the consumer to make wrong purchase decisions which can ultimately damage the image of the products and sellers."

[2] No original: "Fake online reviews in e-commerce significantly affect online consumers, merchants, and, as a result, market efficiency."

> "O desenvolvimento de algoritmos computacionais para identificar avaliações falsas enfatiza predominantemente o 'tratamento' do 'sintoma'. [...] Muitas avaliações falsas ainda persistem, mesmo com algoritmos que apresentam altas taxas de detecção."[3]

Adicionalmente, esses sistemas podem ser fraudados por meio de textos elaborados para simular avaliações. É também importante observar que o *review bombing* provoca um aumento no engajamento das plataformas — ainda que artificial —, o que, de certa maneira, pode incentivar o não combate rigoroso a essa prática.

Com base nas limitações apresentadas — desde os problemas com a autenticação fraca até as falhas na moderação —, é possível afirmar que as plataformas de avaliações atuais não possuem mecanismos eficientes para garantir transparência e imparcialidade nas avaliações. Os ataques coordenados, como o *review bombing*, expõem não apenas problemas técnicos, mas também questões éticas relacionadas à forma como essas plataformas são geridas. Neste contexto, surge a demanda por uma mudança na maneira como as avaliações são registradas e auditadas.

2.3 Sistemas de reputação centralizados vs descentralizados

Para compreender as vantagens e limitações das diferentes abordagens para sistemas de avaliação, é necessário a comparação entre os modelos centralizados e descentralizados. Cada modelo possui características próprias e distintas em termos de controle, transparência, escalabilidade e confiabilidade, o que impacta diretamente na experiência do usuário e na confiança em relação as avaliações. Esta seção tem como objetivo examinar os dois modelos e destacar os benefícios e desafios que ambos possuem atualmente.

2.3.1 Sistemas de reputação centralizados

Os sistemas de reputação centralizados operam sob uma estrutura na qual uma única plataforma é encarregada de armazenar, filtrar e tratar todas as avaliações. Conforme destacam Liu e Munro (2012, p. 2), "em sistemas de reputação centralizados, o servidor principal controla a coleta, agregação e exibição de todas as avaliações, sendo a única fonte confiável para o processo."[4]

[3] No original: "Developing computerized algorithms to identify fake reviews predominantly emphasizes 'treatment' of the 'symptom.' [...] Many fake reviews still exist regardless of algorithms that have high detection rates."

[4] No original: "In centralized reputation systems, the central server controls the collection, aggregation, and display of all ratings, acting as the sole trusted source in the process." (Liu; Munro, 2012, p. 2).

Esta centralização de tarefas proporciona uma certa eficácia operacional, uma vez que centraliza a gestão de dados e diminui os gastos com infraestrutura. Contudo, também estabelece restrições à transparência e à capacidade de auditoria externa, deixando os usuários à mercê da integridade e imparcialidade da entidade controladora.

Liu e Munro (2012, p. 443), afirmam que:

> "A maior parte dos sistemas centralizados emprega algoritmos bastante básicos para calcular as avaliações, e [...] todos têm uma complexidade relativamente baixa."[5]

No entanto, essa baixa complexidade existe justamente pela falta de mecanismos eficientes anti-fraude. Muitas vezes as plataformas utilizam filtros e sistemas de moderação automáticos que possuem vieses imperceptíveis, e influenciam a maneira como as avaliações são apresentadas. Em situações onde essas avaliações impactam as decisões dos clientes, a falta de mecanismos de verificação transparentes ao público pode prejudicar a confiabilidade do sistema como um todo. Na Tabela 1, são apresentados os atributos de um sistema de avaliação centralizado ideal e o que normalmente acontece na prática.

Tabela 1 – Diferenças entre o funcionamento ideal e prático de sistemas centralizados.

Parâmetro de Comparação	**Ideal**	**Prática**
Acesso aos algoritmos	Público e documentado	Privado, sem descrição dos processos utilizados
Processo de moderação	Claros e auditáveis	Automatizado e com pouca ou nenhuma explicação
Remoção de avaliações	Visível e com suporte para contestação	Sem justificativa para o usuário
Verificação externa	Verificável, com justificativa razoável	Inexistente para o público
Auditoria	Possibilidade de auditoria independente	Sem mecanismo público de verificação

Fonte: Elaborado pelo autor.

2.3.2 Sistemas de reputação descentralizados

Os sistemas de reputação descentralizados surgem como uma opção aos modelos centralizados, propondo uma nova estrutura para a coleta e validação de avaliações. Ao contrário do modelo centralizado, nesses sistemas os dados não são guardados ou administrados por um único órgão, mas sim espalhados por uma rede. Arshad

[5] No original: "Most centralized systems employ fairly basic algorithms for computing reputation scores and [...] all have relatively low complexity."

et al. (2022) mostram que esse sistema pode utilizar tecnologias como a *blockchain* para documentar avaliações de maneira inalterável, rastreável e disponível para todos os envolvidos. No modelo descentralizado, normalmente é utilizado um contrato inteligente no lugar da figura controladora.

Embora os sistemas descentralizados proporcionem benefícios como a transparência e a inalterabilidade dos dados, sua implementação ainda apresenta desafios técnicos consideráveis. Um dos principais problemas é a escalabilidade: o armazenamento direto de avaliações na *blockchain* pode gerar custos elevados e comprometer a performance da rede. Arshad et al. (2022) destacam que blockchains atuais enfrentam dificuldades relacionadas à escalabilidade, ao tempo de processamento das transações e ao custo financeiro do armazenamento de dados, especialmente em ambientes com muitos usuários. Desse modo, ainda que a *blockchain* represente um avanço em termos técnicos em alguns aspectos como transparência, auditoria e descentralização, o problema da escalabilidade é real e possui grande impacto.

2.3.3 Comparativo entre os modelos de sistemas de reputação

Tanto os sistemas centralizados quanto os descentralizados possuem benefícios e limitações relevantes. Apesar dos sistemas que usam estrutura centralizada serem mais fáceis de controlar, a ausência de transparência e a sua vulnerabilidade a ataques e manipulação prejudicam a confiança nas avaliações. Por outro lado, os sistemas descentralizados, ao dar prioridade à autenticidade e imutabilidade, representam um progresso nessa direção. No entanto, ainda encontram problemas relacionados à escalabilidade e à velocidade no processamento.

Por isso, a decisão entre o modelo com estrutura centralizada e o modelo com estrutura descentralizada deve levar em conta esse *trade-off* entre confiança e custo. O desafio maior é criar uma solução que una o melhor dos dois mundos.

Esses pontos podem ser melhor visualizados na Tabela 2.

Tabela 2 – Comparação entre sistemas centralizados e descentralizados de reputação.

Critério	**Centralizado**	**Descentralizado**
Controle de dados	Plataforma única	Distribuído entre os nós da rede
Transparência	Limitada	Alta (registros imutáveis)
Auditabilidade	Dependente da empresa	Pública e verificável
Risco de viés/moderação	Alto	Reduzido, se regras forem públicas e imutáveis
Escalabilidade	Alta (dependente da infraestrutura)	Limitada pelo custo e velocidade da blockchain
Custo operacional	Moderado	Elevado (dependendo do projeto)

Fonte: Elaborado pelo autor.

2.4 Tecnologia Blockchain

Esta seção apresenta os conceitos essenciais da *blockchain*, explorando desde sua definição e origem até os mecanismos de consenso que garantem sua segurança e integridade. Também são abordados os diferentes tipos de *blockchain* existentes e os seus recursos criptográficos fornecendo a base teórica necessária para compreender como essa tecnologia pode ser aplicada ao desenvolvimento do MoviesChain.

2.4.1 Definição e origem da blockchain

A *blockchain* foi pensada para permitir transações digitais seguras e sem intermediários. Ela funciona como um banco de dados distribuído que atua como um livro-razão público e inalterável, onde as informações são organizadas em blocos e interligadas através de algoritmos criptográficos. O conceito foi apresentado em 2008, no artigo de Satoshi Nakamoto, que sugeriu a *blockchain* como base para uma nova moeda digital, o *Bitcoin*, com a finalidade de viabilizar transações de forma segura. O termo *blockchain* não aparece no artigo original, apesar do texto utilizar as palavras *block* (bloco) e *chain* (cadeia) diversas vezes. O nome *blockchain*, como uma só palavra, só passou a ser utilizado posteriormente.

A substituição de uma entidade central controladora por um mecanismo que utiliza apenas a verificabilidade pelos usuários participantes foi uma revolução nos modelos de confiança digital. Ao invés de confiar em um intermediário terceiro para validar uma transação, os participantes da rede *blockchain* depositam confiança no protocolo e na implementação de regras codificadas no software. Para Werbach (2018), a *blockchain* não remove a exigência de confiança, mas a transfere das instituições

humanas para o funcionamento técnico da rede, que passa a funcionar como um novo sistema confiável.

2.4.2 Funcionamento técnico da blockchain

A *blockchain* organiza os dados em blocos, que funcionam como pequenos arquivos contendo várias transações aprovadas. Cada bloco inclui três partes principais: as informações das transações, a data e hora em que foram registradas (*timestamp*) e um *hash*. Além disso, cada bloco guarda o *hash* do bloco anterior, formando uma cadeia contínua. Essa ligação entre os blocos é o que impede alterações: se alguém tenta mudar qualquer dado em um bloco, o *hash* muda, e isso quebra toda a sequência.

O *hash* é gerado a partir de todas as informações contidas no bloco, como se fosse uma impressão digital única daquele conteúdo. Ele é criado por uma função matemática que transforma os dados em uma sequência de letras e números. Se qualquer parte do bloco for alterada, o *hash* muda completamente. Werbach (2018) destaca que essa característica torna a *blockchain* segura por padrão, porque os próprios participantes da rede podem verificar se os blocos foram alterados, apenas conferindo se os *hashes* continuam válidos.

A Figura 2 exibe o encadeamento dos blocos em uma *blockchain*. Cada bloco contém três informações principais: o seu próprio *hash* (gerado com base nos dados internos), o *hash* do bloco anterior e o *timestamp* da criação. O primeiro bloco de uma *blockchain*, chamado de bloco gênese, tem como referência anterior um valor nulo, já que é o ponto inicial da cadeia. Os blocos seguintes armazenam o *hash* do bloco anterior, formando uma sequência lógica. Essa organização garante que qualquer modificação em um bloco invalide todos os que vierem depois, já que quebraria o encadeamento porque o *hash* do bloco modificado também seria alterado automaticamente, colapsando todos os outros.

Figura 2 – Estrutura sequencial de blocos em uma *blockchain*.

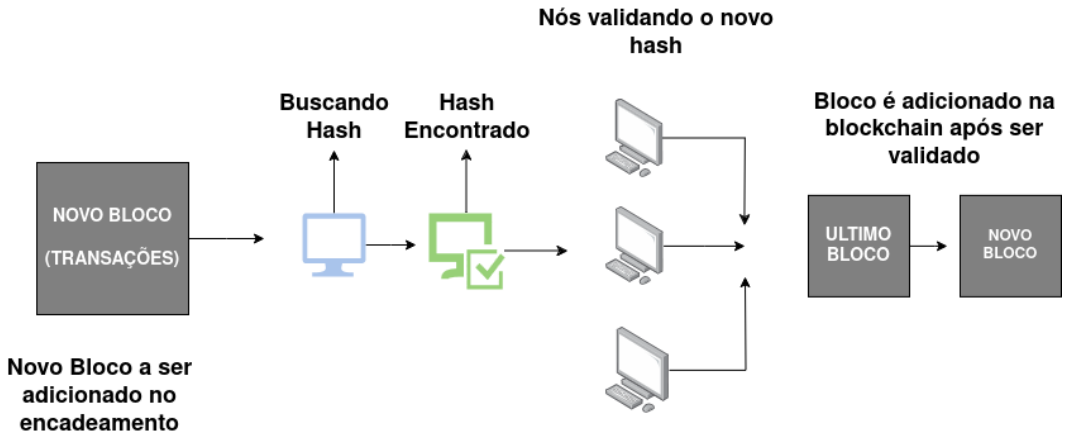
Fonte: Elaborado pelo autor (2025).

2.4.3 *Proof of Work* (POW)

O encadeamento dos blocos deixa a *blockchain* resistente a alterações, porém não garante que usuários mal-intencionados manipulem os dados antes da criação de um novo bloco e o envio para a rede. Com isso, torna-se necessário um mecanismo eficiente de consenso que define como os blocos são validados e aceitos na cadeia. Um desses mecanismos é o *Proof of Work* (POW), que é utilizado pelo *Bitcoin*. No POW, os participantes da rede, chamados de mineradores, competem para resolver um desafio matemático complexo. O primeiro minerador que encontrar uma solução consegue adicionar um novo bloco na *blockchain*. Em seguida, o bloco é propagado e aceito caso o conteúdo esteja de acordo com as regras definidas. Werbach (2018) explica que esse modelo reduz a necessidade de confiança entre os participantes, pois todos seguem o mesmo conjunto de regras e qualquer tentativa de manipulação pode ser verificada por qualquer outro membro da rede.

Para encontrar um *hash* válido, os mineradores precisam fazer diversas tentativas até que o resultado obedeça às regras da rede, como começar com uma certa quantidade de zeros. Para isso, a cada iteração é ajustado o *nonce* para alterar o resultado do *hash*. Os mineradores testam diferentes valores de *nonce* até que a função matemática retorne um *hash* aceito. Esse esforço de tentativa e erro é o trabalho que dá o nome ao POW. Nakamoto (2008) descreve esse processo como uma forma de provar que trabalho real foi feito antes que um bloco seja aceito pela rede.

Figura 3 – Funcionamento do POW com validação e encadeamento do bloco.



Fonte: Elaborado pelo autor (2025).

A Figura 3 mostra, de forma simplificada, como funciona o POW. O novo bloco, com suas transações, é submetido ao processo de mineração. Os mineradores competem tentando diferentes valores de *nonce* até gerar um *hash* válido. Assim que isso

acontece, o bloco é transmitido para os outros nós, que verificam se o resultado é correto. Se tudo estiver de acordo e a maioria dos nós aprovar, o bloco é adicionado à cadeia.

O funcionamento do POW torna a *blockchain* segura porque exige um grande esforço para validar novos blocos. No entanto, esse modelo tem um custo alto: ele consome grandes quantidades de energia elétrica e exige equipamentos potentes, o que limita a participação de usuários comuns. Além disso, a competição entre mineradores pode concentrar o poder nas mãos de poucos grupos com maior capacidade computacional, o que vai contra o ideal de descentralização. Esses problemas levaram ao surgimento de alternativas ao POW, sendo o *Proof of Stake* (POS) uma das mais populares.

2.4.4 *Proof of Stake* (POS)

O POS foi desenvolvido como uma alternativa ao POW, buscando reduzir o consumo de energia e os custos da mineração. Ao invés de utilizar o poder computacional como no POW, o POS seleciona um validador com base na quantidade de moedas que ele está disposto a manter bloqueada na rede — como o validador escolhido "aposta" essas moedas para ser escolhido, o processo ficou conhecido como *stake*. Quanto maior a aposta, mais chances de ser escolhido para validar (inserir) um novo bloco. Caso o validador esteja envolvido em algum tipo de trapaça, ele pode ser punido pelos *atestors* e perder o valor da aposta. Segundo Saleh (2021), esse modelo substitui o gasto de energia por um risco econômico: se o validador agir de forma desonesta, pode perder parte ou todo o valor apostado, o que desestimula comportamentos maliciosos e protege a integridade da *blockchain*.

Ao contrário do que ocorre no POW, onde a segurança da rede depende de grandes investimentos em máquinas e energia elétrica, o POS oferece uma alternativa mais econômica e sustentável. Como não há necessidade de resolver cálculos complexos, o consumo de energia é menor, o que torna essa abordagem mais acessível a diferentes perfis de usuários. Além disso, o POS reduz o risco de centralização, já que não favorece quem possui mais capacidade computacional, mas sim quem demonstra maior comprometimento com a rede. Esse modelo já vem sendo adotado por diversas criptomoedas modernas, como a Ethereum, que iniciou sua transição do POW para o POS a partir de 2022.

2.4.5 Tipos de *blockchain*

Segundo Zheng et al. (2017), a *blockchain* pode ser classificada em três tipos principais:

- ***Blockchain* pública**: É o modelo mais utilizado atualmente. Qualquer pessoa pode participar da rede, validar, consultar e registrar dados. Utiliza mecanismos de consenso como o POW ou POS para garantir a segurança e imutabilidade dos registros na rede. Por ser uma rede de acesso público, é altamente transparente e descentralizada, porém apresenta limitações em termos de velocidade e escalabilidade.
- ***Blockchain* privada**: O acesso à rede é controlado por uma organização. É mais rápida que o modelo público, mas os usuários precisam confiar na entidade que controla a rede. Nesse modelo, os participantes possuem mais permissões para realizar alterações e acessar os dados.
- ***Blockchain* permissionada**: É um modelo intermediário onde várias organizações ou participantes são responsáveis pelo controle da rede. Permite um equilíbrio entre eficiência, velocidade e descentralização. Esse modelo é muito utilizado em ambientes corporativos, onde a confiança é distribuída entre os membros autorizados.

A Tabela 3 apresenta as principais diferenças entre os modelos.

Tabela 3 – Diferenças entre os tipos de *blockchain*.

Propriedade	**Pública**	**Permissionada**	**Privada**
Determinação do consenso	Todos os mineradores	Conjunto selecionado de nós	Uma única organização
Permissão de leitura	Pública	Pode ser pública ou restrita	Pode ser pública ou restrita
Imutabilidade	Praticamente impossível de alterar	Pode ser alterada	Pode ser alterada
Eficiência	Baixa	Alta	Alta
Centralização	Não	Parcial	Sim
Processo de consenso	Sem permissão	Com permissão	Com permissão

Fonte: Adaptado de Zheng et al. (2017).

As *blockchains* públicas são mais utilizadas quando a aplicação requer descentralização e não limitação de acesso, como ocorre com criptomoedas e no mercado de *Non-Fungible Tokens* (NFT). As privadas são comuns em ambientes corporativos, onde é necessário maior controle dos dados, privacidade e eficiência. Já as permissionadas são usadas quando há colaboração entre organizações diferentes, mas dentro de um ambiente controlado, com regras bem definidas sobre acesso e validação dos dados. A tabela 4 apresenta algumas aplicações que utilizam a blockchain e qual o tipo de rede utilizado.

Tabela 4 – Aplicações da *blockchain* e seus respectivos tipos.

Aplicação	**Exemplos**	**Tipo de *Blockchain***
Criptomoedas	Bitcoin, Ethereum, Binance Coin	Pública
Finanças descentralizadas (DeFi)	Uniswap, Aave, Compound, Curve	Pública
Identidade digital	ID2020, Sovrin	Permissionada / Híbrida
Propriedade intelectual (NFTs)	OpenSea, Rarible, Axie Infinity	Pública
Governança descentralizada (DAOs)	MakerDAO, Aave DAO, Uniswap DAO	Pública

Fonte: Elaborado pelo autor (2025).

2.4.6 Criptografia

Ainda que os participantes de uma rede não se conheçam e não possuam confiança entre si, na *blockchain* basta confiar nos modelos criptográficos utilizados nos algoritmos da rede. A matemática se torna o intermediário que garante a autenticidade e a integridade das informações. Sem esses modelos, não seria possível assegurar que uma *blockchain* fosse legítima, imutável e transparente. Werbach (2018, p. 40), afirma:

> "O poder da criptografia está no fato de que ela é uma forma de matemática aplicada. Suas afirmações podem ser formalmente provadas e seus algoritmos implementados por computadores, cujo poder aumenta a cada ano."[6]

O principal recurso criptográfico utilizado em uma *blockchain* são as funções *hash*, que garantem a organização e a integridade dos blocos. Segundo Narayanan et al. (2016), as funções *hash* possuem três propriedades fundamentais que as tornam adequadas para uso em *blockchain*:

- **Entrada de qualquer tamanho**: A função aceita como entrada uma sequência de dados de qualquer comprimento, seja um pequeno texto ou um bloco inteiro.
- **Saída de comprimento fixo**: Independentemente do tamanho da entrada, o *hash* gerado terá sempre o mesmo tamanho, funcionando como uma impressão digital daquele conjunto de dados.

[6] No original: "The power of cryptography is that it is a form of applied mathematics. Its claims can be proven formally and its algorithms implemented through computers whose power increases every year." (Werbach, 2018, p. 41)

- **Eficiência computacional**: A função é projetada para ser rápida, permitindo que seu cálculo seja feito de forma viável mesmo para grandes volumes de dados.

Outro recurso essencial é a criptografia assimétrica, que permite que um usuário possua um par de chaves: uma privada e uma pública. A chave privada é usada para assinar transações digitalmente, comprovando a autoria do titular da chave. Com a chave pública, qualquer usuário consegue verificar a autenticidade de uma assinatura. Antonopoulos (2017, p. 63) , explica o funcionamento no *Bitcoin*:

> "Uma carteira de Bitcoin contém um conjunto de pares de chaves, cada um composto por uma chave privada e uma chave pública. A chave privada é usada para assinar transações, fornecendo uma prova matemática de que a transação foi criada pelo proprietário da carteira. A chave pública correspondente permite que qualquer pessoa no mundo verifique a assinatura, autorizando assim a transação sem revelar a chave privada."[7]

A combinação desses três elementos — funções *hash*, criptografia assimétrica e assinaturas digitais — é o que permite a segurança da *blockchain*. Os *hashes* garantem a integridade, a chave privada garante a autoria e a assinatura digital permite a verificação pública e transparente.

2.5 Contratos inteligentes

Atualmente, contratos inteligentes são programas de computador armazenados e executados em uma *blockchain*. Eles permitem a automação de acordos, regras e transações, sem a necessidade de um intermediário. O termo foi cunhado por Nick Szabo nos anos 1990, que definiu contratos inteligentes como:

> "Um conjunto de promessas, especificadas em formato digital, incluindo protocolos dentro dos quais as partes cumprem essas promessas." (Szabo, 1997, p. n.p.)[8]

Na prática, contratos inteligentes são códigos que rodam em uma rede *blockchain* e são acionados quando determinada condição programada é atendida. Isso permite a execução automática do acordo, eliminando a necessidade de terceiros para garantir o cumprimento.

---

[7] No original: "A bitcoin wallet contains a collection of key pairs, each consisting of a private key and a public key. The private key is used to sign transactions, providing mathematical proof that the transaction came from the owner of the wallet. The corresponding public key allows anyone in the world to verify the signature, thereby authorizing the transaction without revealing the private key." (Antonopoulos, 2017, p. 63)

[8] O documento não possui paginação.

Para que o contrato inteligente seja integrado à *blockchain*, sua lógica precisa ser escrita em uma linguagem de programação específica daquela rede. No Ethereum, por exemplo, é utilizada a linguagem Solidity. Após codificado, o contrato é convertido em *bytecode* para ser interpretado pela *Ethereum Virtual Machine* (EVM), que executa o contrato em todos os nós da rede. Uma vez implantado, o contrato não pode mais ser alterado.

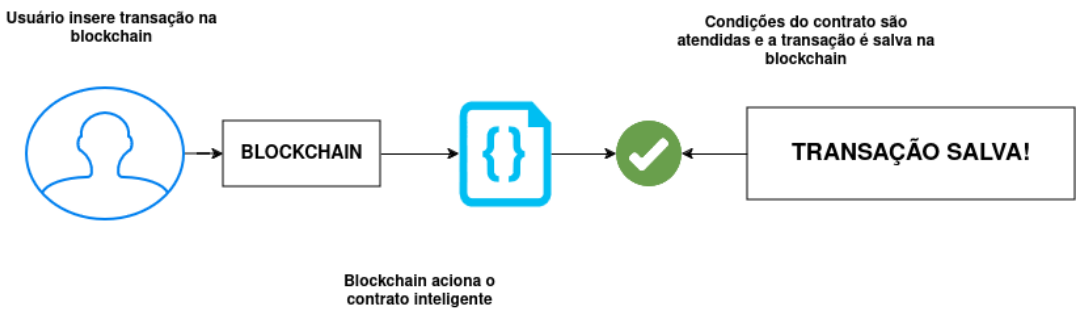
Segundo Narayanan et al. (2016), no Ethereum, um contrato é tratado como um programa que vive na *blockchain*, podendo ser criado por qualquer usuário da rede que envie seu código por meio de uma transação. Esse contrato, após implantado, possui endereço próprio na rede, podendo interagir com usuários e com outros contratos.

Entre as principais vantagens dos contratos inteligentes estão a segurança, a transparência e a eficiência. Segundo Zheng et al. (2017), os contratos inteligentes proporcionam maior confiabilidade, criando um ambiente onde as regras são autoexecutáveis e as obrigações são cumpridas automaticamente, sem depender de intermediários.

Apesar das vantagens, os contratos inteligentes também apresentam limitações. Uma vez implantado na rede, qualquer erro no código não pode ser corrigido, o que pode gerar prejuízos. Isso ficou evidente no caso do ataque à organização *The DAO* em 2016, que resultou na perda de aproximadamente 3,6 milhões de ethers devido a uma falha no contrato inteligente, explorada por meio de um ataque de reentrância (Chainlink, 2022; Gemini, 2022).

Além disso, traduzir as relações do mundo real para linhas de código não é uma tarefa trivial. Nem tudo pode ser antecipado durante o desenvolvimento do contrato, e isso levanta discussões sobre a validade legal desses contratos. A figura 4 ilustra de forma simplificada como funciona esse processo a partir da interação de um usuário em uma blockchain.

Figura 4 – Funcionamento de um contrato inteligente na *blockchain*.



Fonte: Elaborado pelo autor (2025).

Os contratos inteligentes revolucionaram a forma com que acordos podem ser executados digitalmente. Agora é possível programar regras que são cumpridas automaticamente, sem depender de uma entidade central. Apesar dos desafios, os contratos inteligentes já desempenham um papel relevante na *blockchain*, sendo utilizados em aplicações que vão desde transações financeiras até sistemas autônomos. É provável que os contratos inteligentes sejam protagonistas na transformação das relações contratuais no futuro.

2.6 Trabalhos Relacionados

Esta seção realiza uma análise de trabalhos anteriores que exploraram a aplicação da tecnologia blockchain em sistemas de avaliação e reputação.

2.6.1 *Blockchain-based online movie rating system*

O trabalho de Saveetha e Maragatham (2021) propõe a criação de um sistema de avaliação de filmes baseado na *blockchain*, também com o objetivo de resolver problemas comuns em plataformas centralizadas, como avaliações falsas, spam e uso de *bots*. É criado um contrato inteligente na linguagem Solidity, executado no Remix IDE e implantado na *blockchain* Ethereum via Ganache. As avaliações dos usuários são registradas diretamente na blockchain, impedindo a alteração ou remoção após terem sido criadas.

O sistema não possui uma interface elegante ou integração com *back-end* e *front-end* avançado. Saveetha e Maragatham se limitam a demonstrar o funcionamento do contrato em uma rede Ethereum local. Também não há nenhum tipo de autenticação de usuários, permitindo que qualquer endereço de carteira possa votar, sem mecanismo para impedir que o mesmo usuário faça várias avaliações do mesmo filme.

2.6.2 *Movie recommendation and classification system using blockchain*

O trabalho de Abdulmunim et al. (2024) propõe um sistema de recomendação e classificação de filmes baseado em *blockchain*, com foco na privacidade dos usuários. Também são utilizadas técnicas de aprendizado de máquina no sistema. Os contratos inteligentes são implantados na rede Ethereum e são responsáveis pela lógica de recomendação.

A aplicação roda na Ethereum, registra avaliações, preferências e interações dos usuários. O sistema também gera recomendações a partir de um modelo de filtragem colaborativa, onde é calculada a similaridade entre os usuários para sugerir filmes. São usados algoritmos de aprendizado de máquina como o *Dual-LightGCN* para

melhorar a eficiência das recomendações e das classificações dos filmes. A solução demonstra a integração de uma aplicação baseada em *blockchain* com tecnologias de inteligência artificial.

2.6.3 *A Blockchain-Based E-Commerce Reputation System Built With Verifiable Credentials*

A proposta de Dogan e Can (2023) é um sistema descentralizado para plataformas de *e-commerce*, com foco na privacidade dos usuários e na prevenção de fraudes. A arquitetura utiliza as blockchains Hyperledger Indy, que possui suporte a credenciais verificáveis com *Zero Knowledge Proofs* (ZKP), e Hyperledger Fabric, que implementa contratos inteligentes para as regras de negócios e avaliações.

O sistema gera tokens de compra após cada transação, como forma de garantir que apenas compradores reais possam avaliar, e também tokens de desconto quando um usuário envia um feedback. Por utilizar *blockchains* permissionadas, a solução possui uma camada de centralização, já que depende de uma entidade para validar os participantes. Isso limita a transparência do sistema.

2.6.4 Comparação entre os trabalhos relacionados

A análise dos trabalhos relacionados evidencia que o uso da tecnologia blockchain tem sido aplicado em diferentes contextos relacionados à indústria do entretenimento e aos sistemas de reputação.

Embora todos os trabalhos utilizem *blockchain*, cada um se limita a resolver um problema específico. Diferente dos sistemas analisados, que focam em classificação, recomendações ou reputação de *e-commerce*, este trabalho propõe um sistema que une aspectos de descentralização, transparência e integridade, permitindo que as avaliações sejam públicas e possam ser auditadas, garantindo assim imunidade a manipulações. Além disso, também se preocupa em manter a unicidade das avaliações, evitando que o mesmo usuário avalie o mesmo item mais de uma vez. A Tabela 5 faz um comparativo entre todos os trabalhos apresentados.

Tabela 5 – Comparação entre os trabalhos relacionados

Fator	**Trabalho 1**	**Trabalho 2**	**Trabalho 3**	**MoviesChain**
Tema	Sistema de avaliação de filmes	Sistema de recomendação e classificação de filmes	Sistema de reputação para *e-commerce*	Sistema de avaliação de filmes

Fator	**Trabalho 1**	**Trabalho 2**	**Trabalho 3**	**MoviesChain**
Tecnologia Blockchain	Ethereum	Ethereum	Hyperledger Indy + Hyperledger Fabric	Ethereum (Sepolia)
Tipo de Blockchain	Pública	Pública	Permissionada	Pública
Imutabilidade	Sim	Sim	Sim	Sim
Foco em Privacidade	Não	Não	Sim (ZKP)	Não
Prevenção de fraudes	Sim (porém, não verifica duplicidade de avaliações)	Sim (manipulação de dados)	Sim (identidade falsa e avaliações maliciosas)	Sim (review bombing e manipulação)
Uso de Tokens	Não	Não	Tokens de compra e de desconto	Não
Autenticidade dos usuários	Não	Não	Sim (credenciais verificáveis)	Associado à carteira pública

Fonte: Elaborado pelo autor (2025).

3 METODOLOGIA

Esse capítulo tem como objetivo descrever os procedimentos adotados durante a realização da pesquisa, bem como as etapas que foram seguidas durante o desenvolvimento do sistema proposto, além de apresentar as tecnologias utilizadas.

3.1 Tipo de Pesquisa

O presente trabalho pode ser caracterizado como uma pesquisa aplicada, pois gera conhecimento para a solução de um problema específico: a falta de confiabilidade e transparência em avaliações online. Segundo Lakatos e Marconi (2003), a pesquisa aplicada tem como objetivo gerar conhecimento para uma aplicação prática, com esse conhecimento sendo direcionado à resolução de questões concretas. Nesse sentido, o MoviesChain demonstra, por meio de um protótipo funcional, como a tecnologia *blockchain* e contratos inteligentes podem ser aplicados na criação de sistemas de reputação descentralizados.

Além disso, a pesquisa também possui caráter exploratório e experimental, pois busca aprofundar o entendimento sobre as possíveis formas de utilização da *blockchain* em sistemas de reputação, que atualmente, utilizam em sua maioria, sistemas centralizados. E como o trabalho tem como objetivo o desenvolvimento prático de uma aplicação, possui também característica experimental. As duas abordagens servem para construir uma base teórica mais sólida sobre o tema e, ao mesmo tempo, evidenciar os desafios e potencialidades da nova abordagem tecnológica.

3.2 Levantamento Bibliográfico

O levantamento bibliográfico foi realizado com o objetivo de fundamentar este trabalho, a partir do levantamento dos trabalhos, foi possível aprofundar sobre os conceitos relacionados à tecnologia *blockchain*, contratos inteligentes, sistemas de reputação atuais, e aplicações descentralizadas. A pesquisa bibliográfica incluiu livros, artigos acadêmicos e conteúdo jornalístico, além das documentações técnicas das ferramentas utilizadas no desenvolvimento da solução.

Os principais trabalhos utilizados foram agrupados na Tabela 6, nele é possível relacionar os autores, temas abordados e as contribuições para o desenvolvimento da proposta deste trabalho.

Tabela 6 – Lista de trabalhos utilizados na fundamentação teórica e suas contribuições

Autor/Ano	**Tema**	**Contribuição na Fundamentação**
Tadelis (2016)	Sistemas de reputação e feedback	Define o conceito de sistemas de reputação online e sua importância para plataformas digitais.
Dellarocas (2003)	Mecanismos de feedback online	Discute desafios enfrentados pelos sistemas de feedback online.
Aridor, Che e Salz (2022)	Sistemas de recomendação	Mostra como as avaliações online influenciam decisões de consumo.
Sharkey, Hsu e Kovács (2022)	Agregadores de avaliações online	Analisa o impacto dos agregadores de avaliações na percepção dos consumidores.
Cantone et al. (2024)	*Review bombing*	Estudo de caso sobre o *review bombing* ocorrido contra o jogo *The Last Of Us Part II* no Metacritic.
Hazim et al. (2018)	Avaliações falsas	Discute os impactos das *fake reviews* em plataformas digitais.
Wu et al. (2020)	Avaliações falsas, confiabilidade	Revisão sobre impactos de avaliações falsas nos mercados digitais.
Liu e Munro (2012)	Modelos centralizados de reputação	Mostra as limitações dos sistemas de reputação centralizados.
Arshad et al. (2022)	*Blockchain* e sistemas de reputação	Discute a efetividade e os desafios dos modelos descentralizados baseados em *blockchain*.
Yang et al. (2023)	*Blockchain*, arquitetura	Conceitos técnicos da *blockchain*, funcionamento e desafios.
Nakamoto (2008)	*Blockchain*, Bitcoin	Conceito que deu origem à *blockchain*.
Werbach (2018)	Confiança, *blockchain* e criptografia	Explica a *blockchain* como modelo de confiança algorítmica, além dos princípios de segurança criptográfica.
Zheng et al. (2017)	Tipos de *blockchain*, aplicações	Classifica os tipos de *blockchain* e exemplifica suas aplicações.
Narayanan et al. (2016)	*Blockchain*, criptografia, contratos inteligentes	Explica funcionamento técnico da *blockchain*, funções hash, segurança e contratos inteligentes.
Antonopoulos (2017)	Criptografia, Bitcoin	Detalha o funcionamento da criptografia de chave pública e assinaturas digitais.
Szabo (1997)	Contratos inteligentes	Define o conceito de contratos inteligentes, origem e funcionamento.

Fonte: Elaborado pelo autor (2025).

3.3 Modelagem do Trabalho

Com foco em obter uma definição clara das funcionalidades, regras de negócio e componentes técnicos, o processo de modelagem do sistema utilizou os princípios da engenharia de software.

Conforme destaca Pressman (2010, p. 1), “a engenharia de software abrange um processo, um conjunto de métodos (práticas) e um conjunto de ferramentas que permitem aos profissionais construir software de alta qualidade.”[1] Neste sentido, os procedimentos de engenharia de software adotados foram essenciais para garantir a qualidade e a eficiência do sistema.

Para o MoviesChain foram utilizadas técnicas de modelagem orientada a objetos, utilizando a *Unified Modeling Language* (UML) como padrão para documentar a aplicação. Os modelos utilizados no desenvolvimento são os seguintes:

- **Diagrama de Casos de Uso**: utilizado para representar as principais funcionalidades do sistema sob a perspectiva dos usuários, incluindo ações como cadastrar avaliações, visualizar avaliações e conectar carteira. Exemplo: fluxos de cadastro de avaliações, login e registro no sistema.
- **Diagrama de Classes**: desenvolvido para mapear as principais entidades envolvidas no sistema, as suas propriedades, métodos e os relacionamentos entre elas.
- **Diagrama de Sequência**: aplicado para ilustrar a interação sequencial entre os atores (usuários, contratos inteligentes e *front-end*) e os componentes do sistema.
- **Diagrama de Arquitetura**: utilizado para representar a visão macro do sistema.

Além disso, foram definidos os requisitos funcionais e não funcionais do sistema, que estabeleceram os critérios da aplicação, como a integridade e imutabilidade dos registros, a autenticação via MetaMask, qual rede *blockchain* seria utilizada e as condições inseridas no contrato inteligente.

A utilização desses modelos garantiu uma visão global do funcionamento da aplicação, facilitando assim o desenvolvimento e validação dos requisitos, seguindo as boas práticas recomendadas na engenharia de software.

3.4 Tecnologias Utilizadas

Esta seção apresenta detalhadamente as principais ferramentas, *frameworks* e tecnologias utilizadas no desenvolvimento do sistema, explicando o papel de cada uma na arquitetura geral e justificando as escolhas técnicas realizadas.

[1] No original: “Software engineering encompasses a process, a collection of methods (practice), and an array of tools that allow professionals to build high-quality computer software.” (Pressman, 2010, p. 1).

3.4.1 Solidity

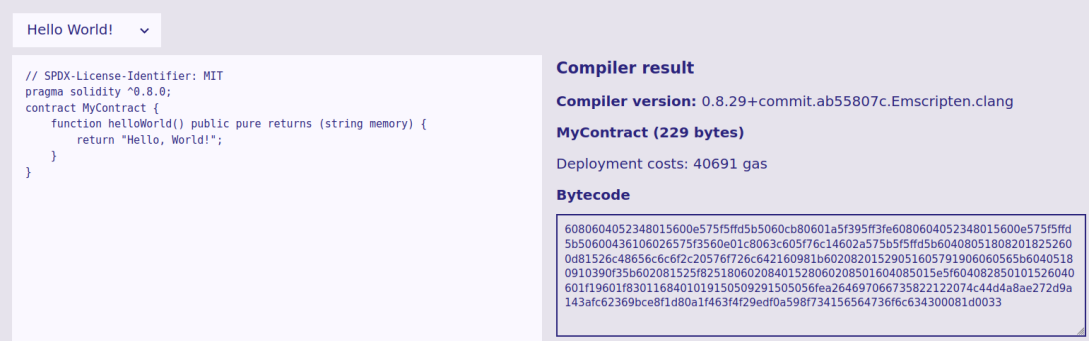
A Solidity é uma linguagem de programação orientada a contratos e projetada para aplicações que rodam na EVM. Foi desenvolvida especificamente para lidar com requisitos da programação de contratos inteligentes, permitindo implementação de regras de negócio na *blockchain* Ethereum (Solidity, 2025).

A linguagem possui uma sintaxe inspirada em outras linguagens como JavaScript, C++ e Python (Solidity, 2025). Estão entre suas principais características:

- Tipagem estática;
- Herança;
- Uso de bibliotecas;
- Definição de tipos criados pelo usuário (ou classes na programação orientada a objetos).

A Figura 5 mostra o exemplo de como um contrato escrito em Solidity é convertido em *bytecode*, que é o código executável pela EVM. Na imagem, é possível observar o código-fonte de um contrato simples, sua compilação, o custo estimado de *gas* para implantação na rede e o *bytecode* gerado, que representa a versão binária do contrato.

Figura 5 – Exemplo de compilação de um contrato Solidity e geração de *bytecode*



Fonte: Solidity Documentation (2025).

No contexto deste trabalho, a Solidity teve papel na construção do contrato inteligente utilizado pela aplicação. O contrato tem como objetivo executar uma lógica que garante o registro único de avaliações para o mesmo filme.

3.4.2 Hardhat

O ambiente de desenvolvimento utilizado para compilar, implantar e debugar o contrato inteligente criado foi o Hardhat. Ele é um framework voltado para aplicações Ethereum e permite a compilação, teste, depuração e implantação de contratos inteligentes na rede (Hardhat, 2025).

O Hardhat pode ser instalado na máquina utilizando o Node.js com o comando `npm`. No contexto deste trabalho, ele foi utilizado para compilar o contrato e gerar os artefatos necessários para o funcionamento do contrato inteligente na rede.

Esses artefatos são, respectivamente:

- ***Bytecode***: código binário com as instruções do contrato, que será armazenado na blockchain;
- ***Application Binary Interface* (ABI)**: interface pública do contrato inteligente, gerada no formato *JavaScript Object Notation* (JSON). É por meio dela que o *front-end* da aplicação consegue interagir com o contrato, pois nela estão descritas as funções, eventos, tipos de dados e os parâmetros disponíveis no contrato, de forma padronizada.

3.4.3 MetaMask

A MetaMask é uma carteira digital de criptomoedas que funciona como uma extensão do navegador. Ela permite que os usuários consigam interagir com *Decentralized Applications* (dApps) na *blockchain* Ethereum (MetaMask, 2025).

A carteira serve como uma ponte entre o usuário e a *blockchain*, permitindo gerenciar chaves e endereços, além de assinar transações e interações com contratos inteligentes.

No contexto do MoviesChain, a MetaMask foi utilizada com duas finalidades, são elas:

- **Autenticação**: Para registro e login, o usuário precisa estar conectado à sua carteira. A identificação no sistema é feita por meio do endereço público da carteira.
- **Assinatura de transações**: Toda vez que o usuário registra uma avaliação, é necessária uma assinatura digital utilizando sua chave privada. Essa assinatura serve para validar e autorizar a operação na blockchain, garantindo que a ação foi realmente realizada pelo titular daquela carteira.

3.4.4 Ethers.js

A Ethers.js é uma biblioteca compacta escrita em JavaScript, desenvolvida para permitir que aplicações web interajam diretamente com a *blockchain* Ethereum (Ethers.js, 2025).

A biblioteca oferece uma *Application Programming Interface* (API) que permite abstrair um contrato inteligente implantado na *blockchain* como um objeto no código, com seus métodos e regras de negócio definidos.

No contexto deste trabalho, a Ethers.js foi utilizada tanto no processo de autenticação quanto no registro das avaliações. Durante o login, ela é responsável por estabelecer a conexão com a carteira MetaMask e gerar uma assinatura digital, que é utilizada para validar a identidade do usuário no sistema. No processo de avaliação, a biblioteca instancia um objeto do contrato inteligente a partir do ABI e do endereço na *blockchain*, permitindo acessar e executar suas funções.

3.4.5 Vue.js

O Vue.js é um framework progressivo para a construção de interfaces web, que permite a criação de aplicações do tipo *Single Page Application* (SPA). Ele oferece uma arquitetura reativa, baseada em componentes, com sintaxe declarativa e de fácil aprendizado (Vue.js, 2025).

Uma das principais características do Vue é a utilização de *Single File Components* (SFC). Este modelo permite que cada componente da aplicação seja desenvolvido em um único arquivo, reunindo tanto a lógica em JavaScript (ou TypeScript), quanto as marcações *HyperText Markup Language* (HTML) e os estilos *Cascading Style Sheets* (CSS). Um SFC possui todas as partes de um componente, facilitando tanto o desenvolvimento quanto a manutenção de código (Vue.js, 2025).

A Figura 6 ilustra um exemplo de SFC, onde é possível observar no mesmo arquivo a definição do comportamento (*script*), da interface visual (*template*) e dos estilos (*style*) de um botão contador.

Figura 6 – Exemplo de um SFC no Vue.js

```vue
<script setup>
import { ref } from 'vue'
const count = ref(0)
</script>

<template>
  <button @click="count++">Count is: {{ count }}</button>
</template>

<style scoped>
button {
  font-weight: bold;
}
</style>
```

Fonte: Vue.js Documentation (2025).

Para o processo de build da aplicação, foi utilizado o *Vite*, uma ferramenta criada pela própria equipe do *Vue*. O *Vite* oferece um ambiente extremamente rápido, com *hot-reloads*, e gera arquivos otimizados e leves para *deploy* (Vite, 2025).

Além disso, para a estilização, foi adotado o *Tailwind CSS*, um framework baseado em classes utilitárias. O *Tailwind* permite construir interfaces modernas de forma rápida e consistente, aplicando estilos diretamente no *HTML* e eliminando a necessidade de escrever em arquivos *CSS* tradicionais (Tailwind, 2025).

No contexto deste trabalho, o *Vue.js* é responsável por todo o desenvolvimento da camada de *front-end*, incluindo as telas de login, registro, listagem de filmes e avaliações. A ferramenta também gerencia toda a interação com a blockchain Ethereum, utilizando a biblio-

teca *Ethers.js* para comunicação com a *MetaMask* e com os contratos inteligentes implantados na rede.

3.4.6 Spring Boot

O Spring Boot é um framework para desenvolvimento de aplicações usando Java. Ele possui um extenso ecossistema e tem como objetivo simplificar a configuração e desenvolvimento de aplicações baseadas em Spring (Boot, 2025).

O Spring Boot oferece uma estrutura robusta para construir APIs, possui suporte a arquiteturas complexas de desenvolvimento como a de microsserviços e, por utilizar Java, a escrita do código é orientada a objetos.

No contexto deste trabalho, o Spring Boot foi utilizado para desenvolvimento do *back-end* da aplicação. Ele é responsável pelo gerenciamento de dados que não serão armazenados na *blockchain*, como as informações dos filmes (títulos, descrições e imagens) e dados auxiliares dos usuários. Essa abordagem se mostrou necessária, pois armazenar esse tipo de dado em uma *blockchain* pública demandaria um alto custo das taxas de transações, o que limita a escalabilidade e o desempenho da aplicação.

Além disso, o *back-end* é responsável pela geração dos tokens de autenticação *JSON Web Token* (JWT). Após a autenticação via MetaMask, é gerado um token para o acesso das funcionalidades do sistema web. Dessa forma, o *back-end* é um ponto intermediário entre a aplicação *front-end* e a *blockchain*.

É importante destacar que esses dados armazenados fora da *blockchain*, apesar de criarem um ponto de centralização, não eliminam as propriedades de transparência, autenticidade e imutabilidade das avaliações, pois o registro delas ainda é armazenado direto na *blockchain*, com endereços e transações verificáveis publicamente por todos os usuários da rede.

3.4.7 Ethereum Sepolia

A Ethereum Sepolia é uma rede de testes oficial da *blockchain* Ethereum. Ela é desenvolvida para possibilitar o teste de aplicações descentralizadas e contratos inteligentes em um ambiente que simula o funcionamento da rede principal, sem custos financeiros e sem riscos (Ethereum, 2025).

A rede Sepolia opera com os mesmos princípios da rede principal: descentralização, imutabilidade e criptografia. Ela utiliza tokens de testes chamados *Ether* (ETH) Sepolia, que não possuem valor real. Ela é amplamente utilizada por desenvolvedores *blockchain* para validação de soluções descentralizadas.

No contexto deste trabalho, a Ethereum Sepolia foi usada para simular um *deploy* da aplicação em uma rede pública. Tanto o contrato inteligente quanto as transações são armazenadas na Sepolia, de forma pública, imutável e auditável. Os ETH Sepolia podem ser geren-

ciados através da carteira MetaMask, e são necessários para interagir com a rede ao realizar funcionalidades do sistema, como o login, registro e avaliações. Isso ocorre porque, assim como na rede principal Ethereum, toda transação demanda uma taxa de transação, conhecida como *gas fee*, que é paga em ETH. Porém, na Sepolia, esses ETH podem ser obtidos de forma gratuita em plataformas conhecidas como *faucets*, como, por exemplo, a Alchemy Sepolia Faucet.

4 DESENVOLVIMENTO

Este capítulo descreve o processo de desenvolvimento do MoviesChain desde a concepção e modelagem até a implementação e demonstração de funcionamento do sistema.

4.1 Visão Geral da Solução

O MoviesChain possui o objetivo de se tornar uma alternativa descentralizada para as plataformas de avaliações de filmes atuais. Ele foi criado utilizando a tecnologia *blockchain* para garantir autenticidade, integridade e transparência dos dados registrados. A solução é composta por três principais camadas: um contrato inteligente, implantado na rede Ethereum Sepolia, que é responsável por armazenar as avaliações de forma imutável e única por usuário; uma API desenvolvida em Spring Boot, que realiza a mediação entre o *front-end* e a *blockchain*, fornecendo serviços como a persistência complementar de dados menos sensíveis, como as informações técnicas dos filmes; e um *front-end* em Vue.js, que oferece uma interface moderna onde os usuários conseguem se conectar via carteira MetaMask, visualizar os filmes cadastrados e registrar as avaliações.

4.2 Modelagem da Aplicação

Para modelagem do sistema, foram utilizados conceitos da engenharia de software, buscando representar as funcionalidades, entidades e interações entre os sistemas de forma clara e organizada. Essa etapa foi primordial para alinhar os requisitos levantados para a implementação da solução, diminuindo assim as dúvidas sobre o que iria compor a aplicação.

Para garantir a organização, foram utilizados modelos UML, que possibilitam a representação visual das funcionalidades e estrutura do sistema. As especificações escolhidas nesta fase foram:

- **Requisitos Funcionais e Não Funcionais**: descrição textual das funcionalidades esperadas e dos critérios a serem atendidos.
- **Diagrama de Casos de Uso**: utilizado para representar as funcionalidades do sistema do ponto de vista dos usuários.
- **Diagrama de Classes**: descreve as principais entidades do sistema e os relacionamentos entre elas.
- **Diagrama de Arquitetura**: mostra a divisão em camadas e a interação entre os módulos do sistema.

A escolha dessas especificações se deu pela necessidade de documentar os fluxos do sistema e também sua estrutura. Isso oferece uma base tanto para as decisões durante a implementação, quanto para pesquisadores interessados em uma visão mais detalhada e organizada da aplicação.

Sanches (2023) destaca a relevância dessa etapa no processo de desenvolvimento:

> ”a modelagem é uma parte central de todas as atividades que levam à implantação de um bom software”(BOOCH; RUMBAUGH; JACOBSON, 2006, apud SANCHES, 2023, p. 45).

4.2.1 Requisitos Funcionais e Não Funcionais

A definição clara dos requisitos do sistema é uma etapa muito importante no desenvolvimento. Segundo Sommerville (2011, p. 57), “os requisitos de um sistema são as descrições do que o sistema deve fazer, os serviços que oferece e as restrições a seu funcionamento”. Para o MoviesChain, os requisitos foram divididos em funcionais e não funcionais: os requisitos funcionais descrevem funcionalidades específicas do sistema; os não funcionais estabelecem critérios de qualidade ou restrições relevantes.

4.2.1.1 Requisitos Funcionais

Os requisitos funcionais mostram as funcionalidades que o sistema deve prover para os usuários. A Tabela 7 apresenta os requisitos funcionais mapeados para o MoviesChain.

Tabela 7 – Requisitos funcionais do MoviesChain.

Número	**Requisito**	**Descrição**
RF001	Registrar usuário	O sistema deve permitir que novos usuários se registrem com nome, CPF, senha e vinculando uma carteira MetaMask.
RF002	Autenticar usuário	O sistema deve permitir login através da carteira MetaMask.
RF003	Listar filmes	O sistema exibirá todos os filmes cadastrados com suas informações e média de avaliações.
RF004	Visualizar detalhes do filme	O sistema deve apresentar informações de um filme, incluindo todas as avaliações e comentários a respeito do mesmo.
RF005	Avaliar filme	O sistema deve permitir que usuários autenticados avaliem filmes com nota de 1 a 5 (representadas por quantidade de estrelas marcadas) e comentário opcional.
RF006	Garantir unicidade de avaliação	O sistema deve impedir que o mesmo usuário avalie o mesmo filme mais de uma vez.
RF007	Registrar avaliação na *blockchain*	O sistema deve gravar cada avaliação como uma transação imutável na *blockchain*.
RF008	Verificar transação	O sistema deve fornecer link para verificação da transação.
RF009	Calcular média de avaliações	O sistema deve calcular e exibir a média das notas de cada filme.

Fonte: Elaborado pelo autor (2025).

4.2.1.2 Requisitos Não Funcionais

Os requisitos não funcionais definem regras de qualidade e restrições relevantes para o sistema. A Tabela 8 apresenta os requisitos não funcionais do MoviesChain divididos por categoria.

Tabela 8 – Requisitos não funcionais do MoviesChain.

Número	**Categoria**	**Requisito**	**Descrição**
RNF001	Segurança	Autenticação segura	Toda autenticação deve ser realizada através de assinatura criptográfica via MetaMask.
RNF002	Segurança	Integridade dos dados	As avaliações devem ser imutáveis e salvas em uma *blockchain* Ethereum.
RNF003	Usabilidade	Interface simples	A interface do sistema deve ser simples e parecida com a de sistemas de avaliações de filmes populares, como o IMDb e o Rotten Tomatoes.
RNF004	Rastreabilidade	Auditoria	Todas as avaliações devem ser publicamente verificáveis na *blockchain* através do Etherscan.

Fonte: Elaborado pelo autor (2025).

4.2.2 Diagramas de Casos de Uso

Os diagramas de caso de uso são utilizados para proporcionar uma visão mais simples e gráfica de um fluxo específico para o usuário, descrevendo as interações entre os atores e o sistema. Segundo Guedes (2018, p. 52) “o diagrama de casos de uso procura, por meio de uma linguagem simples, possibilitar a compreensão do comportamento externo do sistema por qualquer pessoa, tentando apresentar o sistema através de uma perspectiva do usuário”.

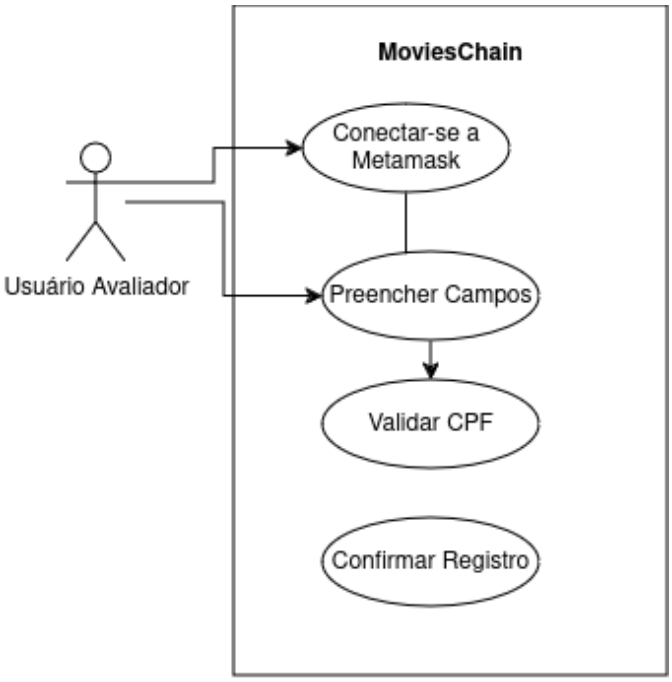
Para Valente (2020), os casos de uso são relevantes pois permitem uma documentação clara e simples das funcionalidades, de forma que tanto desenvolvedores quanto *stakeholders* possam compreender claramente o escopo de um sistema.

O ator principal do MoviesChain é o usuário que avalia os filmes; nos diagramas, ele é chamado de **usuário avaliador**. Os casos de uso foram organizados para representar as principais funcionalidades: autenticação, gerenciamento de filmes e avaliações.

4.2.2.1 Caso de Uso 01 - Registro no Sistema

O primeiro caso de uso especifica o processo de registro de um usuário no sistema. Nesse fluxo, o usuário estabelece o vínculo entre sua conta no sistema e a carteira MetaMask, garantindo que suas ações sejam rastreadas e autenticadas. A Figura 7 ilustra a interação entre o usuário e o sistema durante o registro, e a Tabela 9 apresenta a especificação detalhada do caso de uso.

Figura 7 – Diagrama de caso de uso UC001



Fonte: Elaborado pelo autor (2025).

Tabela 9 – Especificação do caso de uso UC001

Campo	**Descrição**
Identificador	UC001
Nome	Registro no Sistema
Descrição	O usuário cria uma conta no sistema vinculando seus dados pessoais à sua carteira MetaMask.
Ator Principal	Usuário Avaliador
Pré-condições	• Usuário deve possuir MetaMask instalada; • Usuário não pode estar autenticado no sistema.
Pós-condições	• Conta criada com sucesso; • Dados do usuário armazenados no banco; • Carteira MetaMask vinculada ao cadastro.

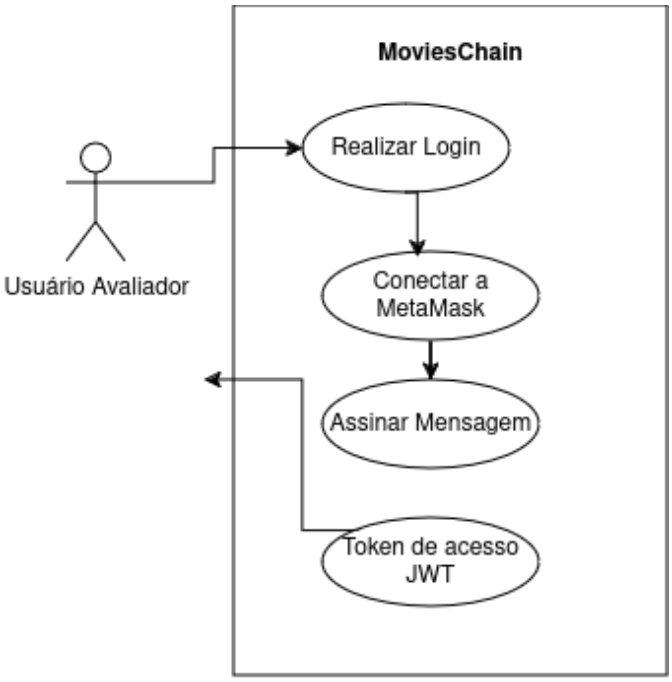
Campo	**Descrição**
Fluxo Principal	1. Usuário acessa a página de registro; 2. Usuário realiza conexão com MetaMask; 3. Sistema captura endereço da carteira conectada; 4. Usuário preenche campos obrigatórios: nome, CPF e senha; 5. Sistema executa validação do CPF; 6. Sistema valida formato e dígitos verificadores do CPF; 7. Usuário clica em confirmar registro; 8. Sistema verifica unicidade de CPF e carteira no banco de dados; 9. Sistema cria conta e gera autenticação; 10. Sistema redireciona usuário autenticado para página inicial.
Fluxo Alternativo A	**CPF já cadastrado:** 8a. Sistema detecta que CPF já existe no banco; 9a. Sistema exibe mensagem: “CPF já cadastrado”; 10a. Usuário corrige CPF ou cancela operação.
Fluxo Alternativo B	**Carteira já cadastrada:** 8b. Sistema detecta que carteira já está vinculada a outro usuário; 9b. Sistema exibe mensagem: “Wallet já cadastrada”.

Fonte: Elaborado pelo autor (2025).

4.2.2.2 Caso de Uso 02 - Realizar Login

A autenticação no sistema também é realizada utilizando a carteira MetaMask. Como a ligação com a carteira envolve assinaturas criptográficas, o método se torna ainda mais seguro, além de garantir que o usuário que está fazendo o login é realmente o proprietário da carteira. A Figura 8 ilustra o diagrama de uso para o login, e a Tabela 10 apresenta as especificações.

Figura 8 – Diagrama de caso de uso UC002



Fonte: Elaborado pelo autor (2025).

Tabela 10 – Especificação do caso de uso UC002

Campo	**Descrição**
Identificador	UC002
Nome	Realizar Login
Descrição	O usuário registrado no sistema realiza login através de sua carteira MetaMask.
Ator Principal	Usuário Avaliador
Pré-condições	• Usuário deve estar registrado no sistema; • MetaMask deve estar instalada no navegador.
Pós-condições	• Usuário autenticado com sucesso; • Token JWT gerado e retornado ao usuário.

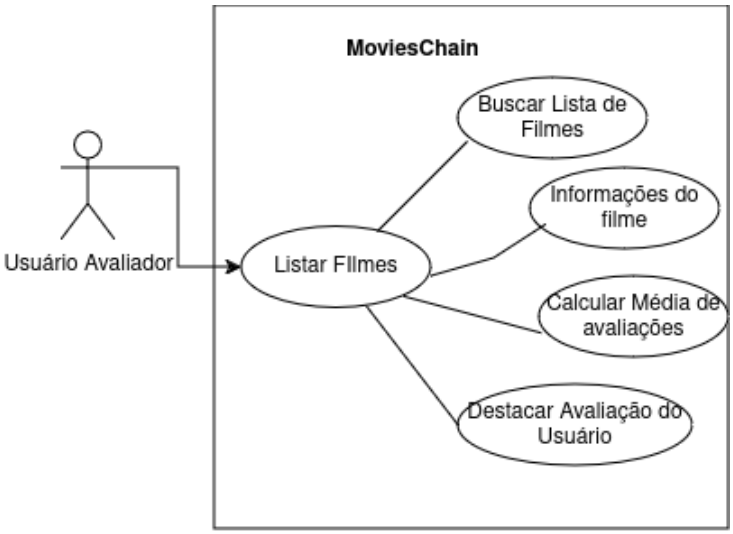
Campo	**Descrição**
Fluxo Principal	1. Usuário acessa a página de login; 2. Usuário clica no botão “Conectar Carteira”; 3. Sistema inicia processo de “Realizar Login”; 4. Sistema executa “Conectar à MetaMask”; 5. Usuário autoriza conexão no MetaMask; 6. Sistema captura endereço da carteira conectada; 7. Sistema gera mensagem com timestamp; 8. Sistema executa “Assinar Mensagem”; 9. Usuário assina mensagem no MetaMask; 10. Sistema verifica a assinatura e valida a carteira no banco; 11. Sistema gera Token de acesso JWT; 12. Sistema retorna token ao usuário; 13. Usuário é redirecionado para página inicial autenticado.
Fluxo Alternativo	**Carteira não cadastrada:** 10a. Sistema não encontra carteira no banco de dados; 11a. Sistema exibe mensagem: “Usuário não encontrado para essa carteira”.
Fluxo de Exceção A	**Usuário cancela conexão:** 5e. Usuário rejeita conexão no MetaMask; 6e. Sistema exibe mensagem: “Erro ao conectar a carteira”.

Fonte: Elaborado pelo autor (2025).

4.2.2.3 Caso de Uso 03 - Listar Filmes

A funcionalidade de listar os filmes é o ponto inicial do sistema após o login. Nessa tela é mostrado todo o catálogo do MoviesChain. Esse caso de uso permite visualizar todos os filmes disponíveis, as médias de avaliações dos usuários e, para filmes já avaliados pelo usuário, exibe uma indicação com a nota que ele inseriu. A Figura 9 ilustra as interações dessa funcionalidade, e a Tabela 11 detalha sua especificação.

Figura 9 – Diagrama de caso de uso UC003



Fonte: Elaborado pelo autor (2025).

Tabela 11 – Especificação do caso de uso UC003

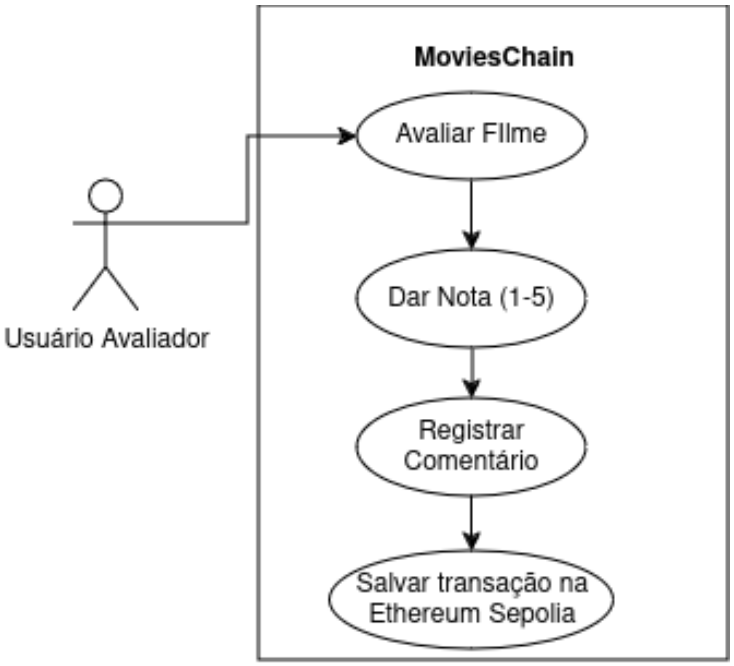
Campo	**Descrição**
Identificador	UC003
Nome	Listar Filmes
Descrição	O sistema exibe o catálogo completo de filmes do *MoviesChain* com suas respectivas médias e status de avaliação do usuário.
Ator Principal	Usuário Avaliador
Pré-condições	Usuário deve estar autenticado no sistema.
Pós-condições	• Lista de filmes exibida com sucesso; • Informações de avaliação apresentadas.
Fluxo Principal	1. Usuário acessa a página inicial autenticado; 2. Sistema busca todos os filmes cadastrados; 3. Sistema verifica quais filmes o usuário já avaliou; 4. Sistema calcula média de avaliações para cada filme; 5. Sistema monta cards com: título, imagem, descrição e média; 6. Sistema destaca filmes já avaliados pelo usuário; 7. Usuário visualiza catálogo e pode acessar detalhes de qualquer filme.

Fonte: Elaborado pelo autor (2025).

4.2.2.4 Caso de Uso 04 - Avaliar Filme

O caso de uso de avaliar filme especifica a funcionalidade principal do MoviesChain, pois é nela que ocorre a interação com a *blockchain* Ethereum. O processo garante que as avaliações serão registradas de forma imutável na rede e com total transparência, podendo posteriormente ser visualizadas através do *hash* na Etherscan. A Figura 10 ilustra o diagrama com as etapas desse processo, e a Tabela 12 detalha de forma descritiva os fluxos relacionados.

Figura 10 – Diagrama de caso de uso UC004



Fonte: Elaborado pelo autor (2025).

Tabela 12 – Especificação do caso de uso UC004

Campo	**Descrição**
Identificador	UC004
Nome	Avaliar Filme
Descrição	O usuário registra uma avaliação única e imutável para um filme na *blockchain* Ethereum Sepolia.
Ator Principal	Usuário Avaliador
Pré-condições	• Usuário deve estar autenticado; • Usuário não pode ter avaliado o filme anteriormente; • Usuário deve possuir ETH Sepolia para taxas; • MetaMask deve estar conectada.

Campo	**Descrição**
Pós-condições	• Avaliação registrada na *blockchain*; • Transação confirmada na Ethereum Sepolia; • *Hash* da transação disponível.
Fluxo Principal	1. Usuário seleciona filme e acessa “Avaliar Filme”; 2. Sistema exibe interface de avaliação; 3. Sistema verifica se usuário já avaliou; 4. Usuário executa “Dar Nota (1-5)” selecionando estrelas; 5. Usuário pode “Registrar Comentário” (opcional); 6. Usuário confirma avaliação; 7. Sistema prepara dados para *blockchain*; 8. Sistema solicita assinatura via MetaMask; 9. Usuário assina e confirma taxa de *gas*; 10. Sistema executa “Salvar transação na Ethereum Sepolia”; 11. *Blockchain* processa e confirma transação; 12. Sistema recebe *hash* de confirmação; 13. Sistema armazena referência no banco de dados; 14. Sistema exibe sucesso com link para Etherscan.
Fluxo Alternativo	**Filme já avaliado:** 3a. Sistema detecta avaliação existente do usuário; 4a. Sistema exibe nota e comentário anteriores; 5a. Sistema desabilita opção de nova avaliação.
Fluxo de Exceção A	**Saldo insuficiente:** 9e. *MetaMask* indica saldo ETH insuficiente; 10e. Sistema exibe: “Saldo insuficiente para realizar a avaliação”.
Observações	• O contrato inteligente garante unicidade por usuário/filme; • Todas as transações são públicas na rede Sepolia.

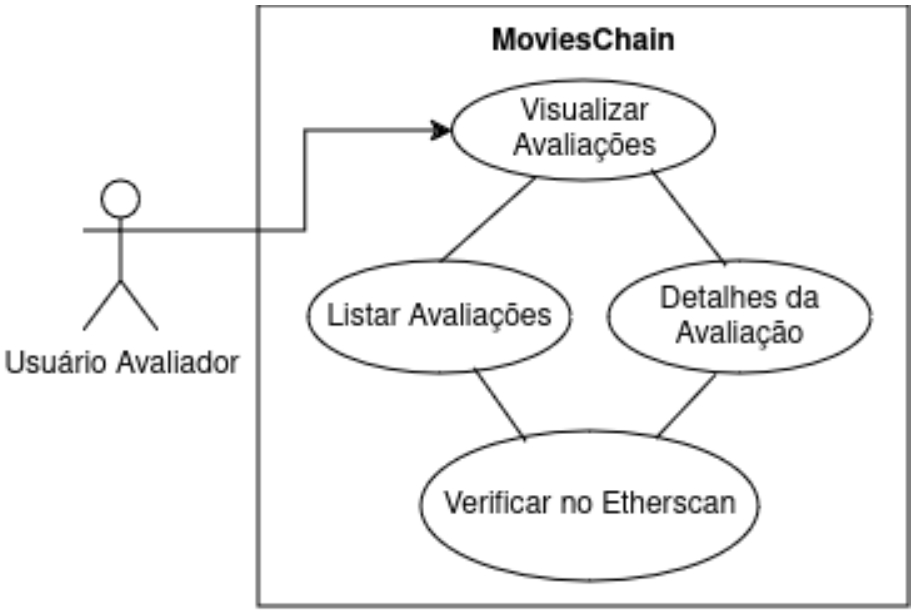
Fonte: Elaborado pelo autor (2025).

4.2.2.5 Caso de Uso 05 - Visualizar Avaliações

O caso de uso visualizar avaliações é responsável por permitir que os usuários do MoviesChain possam consultar todas as avaliações realizadas para um filme específico, promovendo a transparência do sistema. A funcionalidade também permite que as avaliações sejam

verificadas diretamente na *blockchain* através da Etherscan. A Figura 11 ilustra a funcionalidade, e a Tabela 13 detalha como o sistema recupera e exibe as transações.

Figura 11 – Diagrama de caso de uso UC005



Fonte: Elaborado pelo autor (2025).

Tabela 13 – Especificação do caso de uso UC005

Campo	**Descrição**
Identificador	UC005
Nome	Visualizar Avaliações
Descrição	O usuário visualiza todas as avaliações de um filme e pode verificar a transação direto na *blockchain*.
Ator Principal	Usuário Avaliador
Pré-condições	• Usuário deve estar autenticado; • Filme deve possuir avaliações registradas na rede.
Pós-condições	• Avaliações exibidas com sucesso; • Links para verificação disponibilizados.

Campo	**Descrição**
Fluxo Principal	1. Usuário acessa página de detalhes do filme; 2. Sistema inicia “Visualizar Avaliações”; 3. Sistema executa “Listar Todas Avaliações” do filme; 4. Sistema recupera dados do banco: nome, nota, comentário, data; 5. Sistema recupera *hash* de cada transação; 6. Sistema monta lista ordenada por data; 7. Para cada avaliação, sistema “Exibir Detalhes da Avaliação”: • Nome do avaliador; • Nota em estrelas; • Comentário (se houver); • Data e hora. 8. Sistema gera link “Verificar no Etherscan” para cada avaliação; 9. Usuário visualiza lista completa de avaliações; 10. Usuário pode clicar no link para verificar transação.
Observações	• Link do Etherscan abre em nova aba; • Avaliações são exibidas da mais recente para mais antiga.

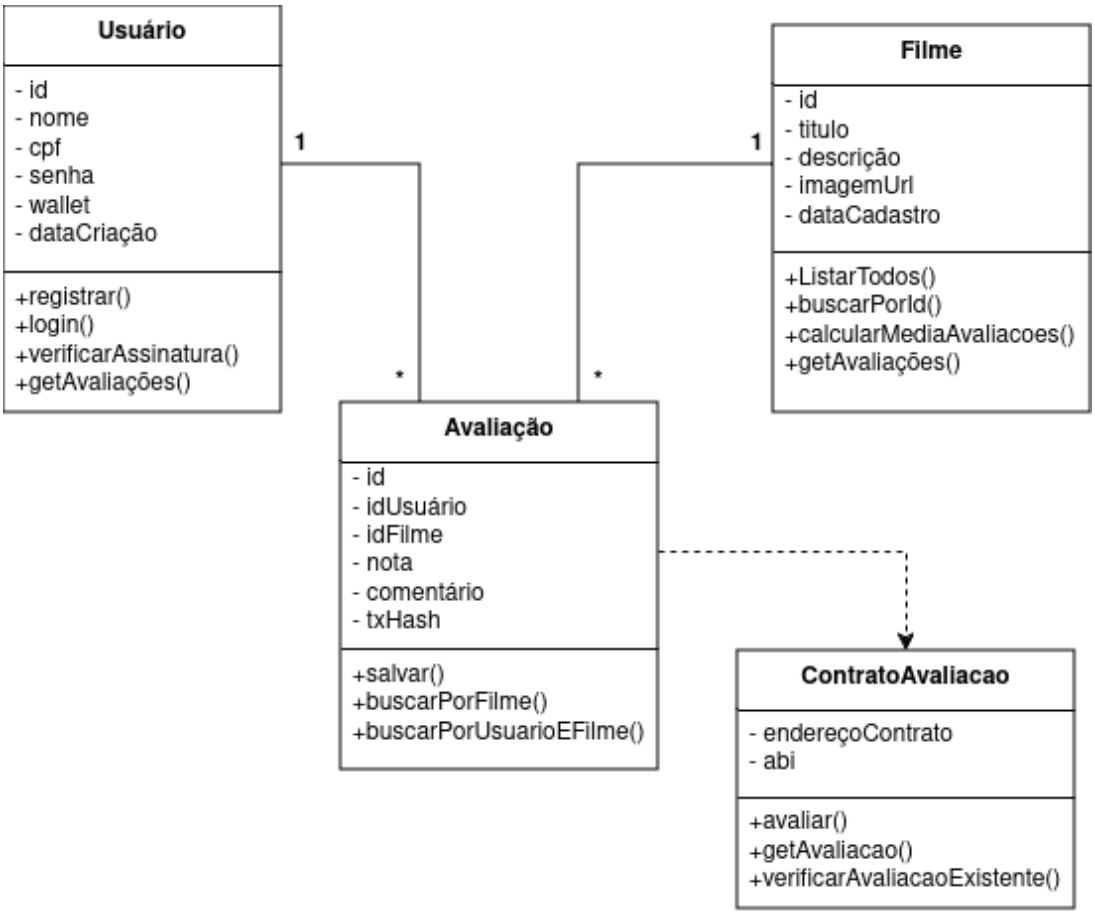
Fonte: Elaborado pelo autor (2025).

4.2.3 Diagrama de Classes

O diagrama de classes proporciona uma visão sobre as classes, seus atributos, métodos e os relacionamentos entre elas. De acordo com Guedes (2018, p. 31), “o diagrama de classes é um dos mais importantes da UML, sendo utilizado como base para o desenvolvimento de outros diagramas e para a implementação do sistema.”

Para o MoviesChain, o diagrama de classes representa a estrutura das entidades do domínio da aplicação e suas interações, incluindo também a integração com o contrato inteligente na *blockchain*. A Figura 12 ilustra o diagrama de classes elaborado.

Figura 12 – Diagrama de classes do MoviesChain



Fonte: Elaborado pelo autor (2025).

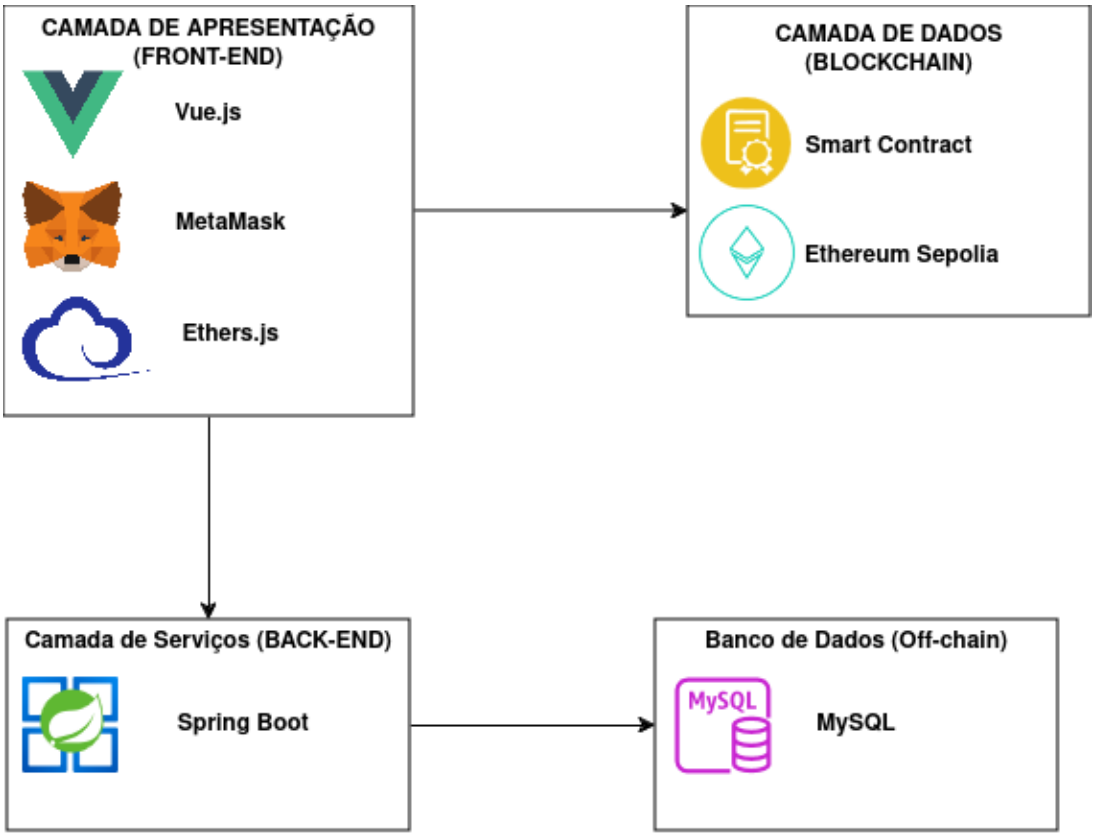
As regras de negócio relevantes do sistema estão representadas pelo diagrama de classes: um usuário pode realizar várias avaliações, porém apenas uma avaliação por filme; cada filme pode receber várias avaliações de usuários diferentes; e a avaliação é dependente do contrato, pois só é possível criá-la após o registro passar pelas regras do contrato e ser salvo de forma imutável na *blockchain*.

4.2.4 Diagrama de Arquitetura

A arquitetura do MoviesChain foi dividida seguindo o padrão de camadas, separando responsabilidades para garantir que cada componente consiga ser implementado de forma independente. Como existe a integração com uma *blockchain* pública, foi importante pensar com cuidado em quais dados seriam armazenados na rede descentralizada e quais poderiam ser mantidos em bancos de dados tradicionais, garantindo assim menor custo e maior desempenho.

A Figura 13 apresenta o diagrama de arquitetura do sistema, ilustrando a separação entre as camadas e suas interações.

Figura 13 – Diagrama de arquitetura do MoviesChain



Fonte: Elaborado pelo autor (2025).

A camada de apresentação (*front-end*) se comunica com a *blockchain* de forma direta, utilizando a MetaMask para autenticação e a biblioteca Ethers.js para o registro das avaliações. Em paralelo, a camada de serviços (*back-end*) em Spring Boot gerencia os dados complementares (como informações de filmes e usuários), armazenando-os em um banco de dados tradicional MySQL. Essa separação permite que o sistema aproveite a rapidez de um banco de dados relacional em operações menos sensíveis, enquanto utiliza a *blockchain* Ethereum Sepolia para registrar as avaliações de forma permanente, imutável e transparente.

4.3 Visão Geral da Implementação

Conforme mostrado no diagrama de arquitetura, o MoviesChain foi organizado em três camadas: a de apresentação, a de serviços e a de dados (*on-chain* e *off-chain*). Diante disso, foram estruturados três projetos independentes, facilitando a manutenção de cada parte e separando as responsabilidades de forma clara. O código-fonte completo, com os três projetos, está disponível publicamente no repositório GitHub, em <https://github.com/antonioroddev/movieschain>.

O repositório foi organizado em três diretórios principais:

- `movieschain-contracts`: contém o projeto Hardhat com o contrato inteligente e o script para *deploy*;
- `movieschain-api`: projeto Spring Boot responsável pelo *back-end*;
- `movieschain-web`: aplicação Vue.js, que cuida da interface de usuário.

Durante a implementação dos projetos, foi priorizada a clareza de código e a separação de responsabilidades. O contrato inteligente foi projetado para conter regras diretas e de fácil entendimento, focando no essencial: o registro único de avaliações por usuário.

No *backend*, o projeto foi estruturado com base em uma arquitetura amplamente utilizada no mercado, o padrão MVC. Os componentes foram separados em três camadas principais:

- **Controle**: onde são expostos os serviços para o *front-end*;
- **Serviço**: onde foram implementadas regras de negócio, como o cálculo da média das avaliações;
- **Dados**: camada que interage com o banco de dados MySQL *(off-chain)* e é responsável pela persistência.

Já no *front-end*, a implementação se baseou na lógica de componentização, característica central da arquitetura do Vue.js.

As próximas seções detalham a implementação de cada projeto e suas especificidades.

4.3.1 Contrato Inteligente

O contrato inteligente é o principal componente do MoviesChain, já que ele é o responsável por garantir a integridade, unicidade e transparência das avaliações. Foi desenvolvido utilizando a linguagem Solidity, na versão 0.8.18. O contrato possui o nome `AvaliacaoFilme` e implementa as regras necessárias para que o sistema mantenha essas propriedades fundamentais.

Na Figura 14 é possível visualizar o contrato completo, incluindo sua estrutura, mapeamentos, funções e eventos.

Figura 14 – Código-fonte do contrato inteligente

```
1  // SPDX-License-Identifier: MIT
2  pragma solidity ^0.8.18;
3
4  contract AvaliacaoFilme {
5      struct Avaliacao {
6          uint8 nota;
7          string comentario;
8          address autor;
9      }
10
11     mapping(uint256 => mapping(address => Avaliacao)) public avaliacoes;
12
13     event Avaliado(uint256 filmeId, address autor, uint8 nota, string comentario);
14
15     function avaliar(uint256 filmeId, uint8 nota, string memory comentario) public {
16         require(nota >= 1 && nota <= 5, "Nota invalida");
17         require(avaliacoes[filmeId][msg.sender].nota == 0, "Ja avaliou esse filme");
18
19         avaliacoes[filmeId][msg.sender] = Avaliacao(nota, comentario, msg.sender);
20         emit Avaliado(filmeId, msg.sender, nota, comentario);
21     }
22
23     function getAvaliacao(uint256 filmeId, address autor) public view returns (uint8, string memory) {
24         Avaliacao memory a = avaliacoes[filmeId][autor];
25         return (a.nota, a.comentario);
26     }
27 }
```

Fonte: Elaborado pelo autor (2025).

O contrato define uma estrutura chamada `Avaliacao`, com os campos `nota`, `comentario` e `autor`. As avaliações são armazenadas em um *mapping* duplo, onde o ID do filme referencia um outro *mapping* que associa um endereço de carteira a uma avaliação.

A função `avaliar` é responsável por registrar as avaliações na *blockchain*. Ela executa duas validações antes de atribuir uma nota ao filme:

- **Validação da nota:** garante que a pontuação esteja entre 1 e 5, conforme os requisitos do MoviesChain;
- **Unicidade da avaliação:** impede que o mesmo usuário avalie o mesmo filme mais de uma vez. Em Solidity, o valor padrão de uma `uint8` é zero. Portanto, a linha 11 da Figura 14 verifica se a nota registrada para o `msg.sender` naquele filme ainda é zero, o que indica que ele ainda não realizou uma avaliação (já que a menor nota válida é 1).

Por fim, a função emite um `event` chamado `Avaliado`, que registra a avaliação na *blockchain* e armazena os detalhes no log, permitindo o rastreamento da interação.

Também foi implementada a função de consulta `getAvaliacao`, que permite recuperar uma avaliação a partir do endereço da carteira do usuário e do ID do filme.

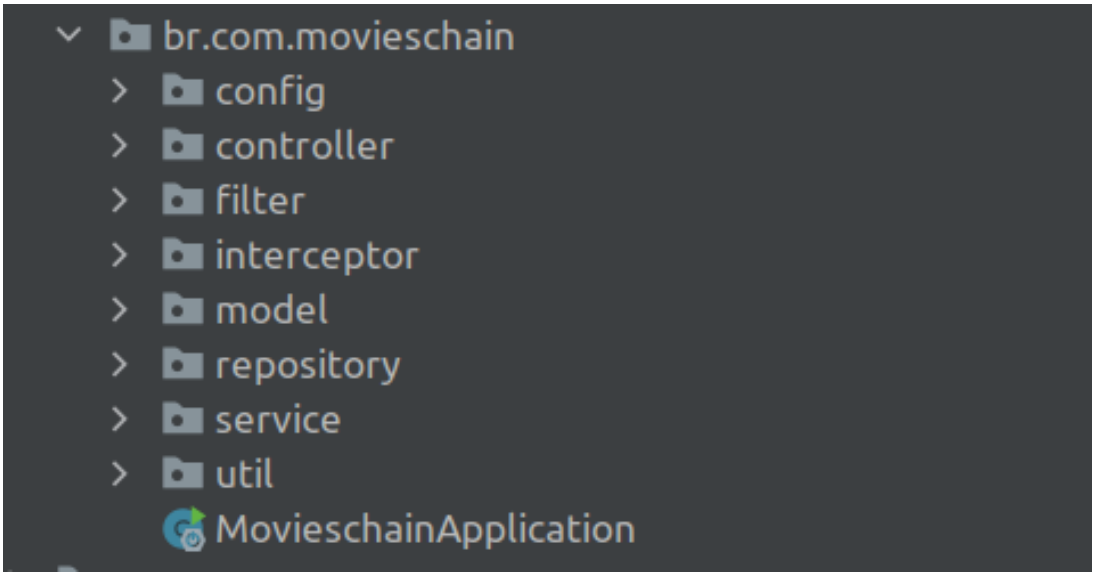
Após sua criação, o contrato foi implantado na rede Ethereum Sepolia por meio do provedor Alchemy, que oferece infraestrutura para envio e gestão de contratos inteligentes. Após o *deploy*, foi gerado um endereço único para o contrato, o qual foi utilizado no *front-end* para o registro e consulta das avaliações.

4.3.2 *Back-end* com Spring Boot

O *back-end* do MoviesChain foi desenvolvido em Java, utilizando o *framework* Spring Boot. O projeto é responsável por expor uma API que se comunica com o *front-end*. A API gerencia os dados auxiliares do sistema, como informações dos filmes e dos usuários, e também gera os tokens de acesso JWT para manutenção da sessão no navegador.

A estrutura de diretórios do projeto segue as convenções padrão do Spring Boot, com separação clara entre controladores, entidades de banco de dados, repositórios, serviços e componentes de configuração. Essa organização é ilustrada na Figura 15.

Figura 15 – Estrutura de diretórios do MoviesChain (*back-end*)



Fonte: Elaborado pelo autor (2025).

Os dados do sistema são representados por três entidades principais: `User`, `Filme` e `Avaliacao`. A Tabela 14 apresenta os atributos e a função de cada uma dessas entidades.

Tabela 14 – Entidades do *back-end*

Entidade	**Atributos**	**Descrição**
`User`	id, nome, cpf, senha, wallet	Representa o usuário do sistema, vinculado a uma carteira MetaMask.
`Filme`	id, titulo, descricao, imagemUrl	Armazena os dados dos filmes disponíveis para avaliação.
`Avaliacao`	idUsuario, idFilme, nota, comentario, txHash, dataCriacao	Representa a avaliação registrada na *blockchain*.

Fonte: Elaborado pelo autor (2025).

Os endpoints da API são agrupados em três tipos principais de funcionalidade: autenticação, gerenciamento de filmes e gerenciamento de avaliações. A Tabela 15 apresenta um resumo das rotas disponíveis.

Tabela 15 – Endpoints do *back-end*

Endpoint	**Verbo HTTP**	**Descrição**
`/api/auth/wallet-login`	POST	Realiza o login do usuário pela carteira, via assinatura digital.
`/api/auth/register-wallet`	POST	Cadastra novo usuário associando CPF, nome e carteira.
`/api/filmes`	GET	Lista todos os filmes, com informações de média de avaliações e dados do usuário.
`/api/filmes/{id}`	GET	Retorna os detalhes de um filme, incluindo avaliação do usuário logado.
`/api/filmes`	POST	Cadastra novos filmes (restrito ao administrador).
`/api/avaliacoes`	POST	Salva uma avaliação realizada na *blockchain*, com referência do *hash*.
`/api/avaliacoes/{filmeId}`	GET	Lista todas as avaliações de um filme, com nome, comentário e *hash*.

Fonte: Elaborado pelo autor (2025).

Para exemplificar o funcionamento da API, foram selecionados dois dos principais endpoints implementados. A Figura 16 mostra o código do endpoint de login via carteira MetaMask. O método, escrito em Java, recebe uma mensagem assinada (gerada no *front-end*) e valida sua autenticidade com a função `verifySignature`. Se a assinatura for válida, o sistema localiza o usuário associado à carteira e retorna um objeto com o token JWT e o nome do usuário, mantendo a sessão ativa no navegador.

Figura 16 – Código-fonte do endpoint de login com carteira

```
@PostMapping("/wallet-login")
public ResponseEntity<?> walletLogin(@RequestBody WalletLoginRequestDTO request) {
    String walletAddress = request.getWallet();
    String signedMessage = request.getSignedMessage();
    String originalMessage = request.getOriginalMessage();

    if (walletAddress == null || signedMessage == null || originalMessage == null) {
        return ResponseEntity.badRequest().body("Dados inválidos.");
    }

        boolean isValid = WalletUtils.verifySignature(walletAddress, signedMessage, originalMessage);

        if (!isValid) {
            return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("Assinatura inválida.");
        }

    User user = userRepository.findByWallet(walletAddress.toLowerCase())
            .orElse( other: null);

    if (user == null) {
        return ResponseEntity.status(HttpStatus.UNAUTHORIZED).body("Usuário não encontrado para essa carteira.");
    }

    LoginResponseDTO response = authService.loginWithWallet(user);

    return ResponseEntity.ok(response);
}
```

Fonte: Elaborado pelo autor (2025).

O objeto retornado é do tipo `LoginResponseDTO`, contendo dois atributos: o token JWT e o nome do usuário armazenado *off-chain*.

A lógica interna da função `verifySignature` é ilustrada na Figura 17. Essa função reconstrói o endereço da carteira a partir da assinatura fornecida. Para isso, aplica o prefixo padrão da Ethereum, gera o *hash* da mensagem, extrai os dados e recupera a chave pública. Por fim, compara o endereço reconstruído com o endereço fornecido pelo usuário.

Figura 17 – Função de verificação de assinatura Ethereum

```
public static boolean verifySignature(String address, String signedMessage, String originalMessage) {
    try {
        String prefix = "\u0019Ethereum Signed Message:\n" + originalMessage.length();
        byte[] messageHash = Hash.sha3((prefix + originalMessage).getBytes());

        Sign.SignatureData signatureData = extractSignatureData(signedMessage);

        BigInteger publicKeyRecovered = Sign.signedMessageHashToKey(messageHash, signatureData);

        String recoveredAddress = "0x" + Keys.getAddress(publicKeyRecovered);

        return recoveredAddress.equalsIgnoreCase(address);
    } catch (Exception e) {
        e.printStackTrace();
        return false;
    }
}
```

Fonte: Elaborado pelo autor (2025).

A Figura 18 mostra o endpoint responsável por registrar a avaliação no banco de dados. Mesmo após a validação pela *blockchain* feita no *front-end*, o *back-end* também realiza checagens para garantir a consistência dos dados armazenados *off-chain*. O método identifica o usuário pela sessão e associa a avaliação aos dados recebidos (nota, comentário, identificador do filme e *hash*). Se o usuário já tiver avaliado o filme, uma exceção é lançada.

Figura 18 – Código-fonte do endpoint de registro de avaliação

```
@PostMapping
public ResponseEntity<?> avaliar(@RequestBody Avaliacao avaliacao,
                                 @AuthenticationPrincipal User userAutenticado) {

    Long idUsuario = userAutenticado.getId();

    Optional<Avaliacao> existente = avaliacaoRepository
            .findByIdUsuarioAndIdFilme(idUsuario, avaliacao.getIdFilme());

    if (existente.isPresent()) {
        return ResponseEntity.badRequest().body("Usuário já avaliou esse filme.");
    }

    avaliacao.setIdUsuario(idUsuario);
    Avaliacao nova = avaliacaoRepository.save(avaliacao);
    return ResponseEntity.ok(nova);
}
```

Fonte: Elaborado pelo autor (2025).

4.3.3 Front-end com Vue.js e Ethers.js

Para o *front-end* do MoviesChain, a tecnologia escolhida foi o Vue.js 3, que proporcionou uma interface reativa baseada em componentes. Como o *front-end* se comunica diretamente com a *blockchain*, também foi utilizada a biblioteca Ethers.js. Para a estilização das telas, foi adotado o Tailwind CSS.

A estrutura de diretórios do projeto *front-end* é organizada em módulos, separando responsabilidades entre componentes, páginas, serviços e configurações. Para o controle de navegação, foi utilizado o Vue Router, que também permite proteger rotas com verificação de autenticação. O gerenciamento de estado foi implementado com Pinia Store, sendo o token de sessão armazenado no *Local Storage* do navegador.

O sistema possui quatro páginas principais: Login, Registro, Home e Avaliar Filme. Todas foram desenvolvidas seguindo o princípio de componentização, separando a lógica de negócio da interface. A integração com a *blockchain* acontece durante as etapas de autenticação e avaliação.

A autenticação é realizada por meio da interação com a carteira MetaMask. O processo é iniciado pelo usuário: a biblioteca Ethers.js estabelece a conexão e solicita ao usuário que

assine uma mensagem. A Figura 19 apresenta a função `loginComWallet`, responsável por esse procedimento. Após a assinatura, a mensagem é enviada ao *back-end* para verificação da identidade do usuário.

Figura 19 – Código-fonte da função de login no *front-end*

```
40  async function loginComWallet() {
41    try {
42      erro.value = ''
43
44      if (!windowTyped.ethereum) {
45        erro.value = 'MetaMask não encontrada.'
46        return
47      }
48
49      const provider = new ethers.BrowserProvider(windowTyped.ethereum)
50      const accounts = await provider.send('eth_requestAccounts', [])
51      const address = accounts[0]
52
53      const message = `Login no MoviesChain - ${new Date().toISOString()}`
54
55      const signer = await provider.getSigner()
56      const signature = await signer.signMessage(message)
57
58      const response = await api.post('/auth/wallet-login', {
59        wallet: address,
60        signedMessage: signature,
61        originalMessage: message
62      })
63
64      auth.login(response.data.token, response.data.user || 'Usuário')
65
66      router.push('/')
67    } catch (err) {
68      console.error(err)
69      erro.value = 'Erro ao conectar a carteira.'
70    }
71  }
```

Fonte: Elaborado pelo autor (2025).

Se a assinatura for validada, o sistema armazena o token JWT na sessão por meio do Pinia Store, permitindo o acesso às demais páginas da aplicação, como a tela inicial e a de avaliação.

A interação com o contrato inteligente ocorre na etapa de avaliação. Quando o usuário insere a nota e o comentário (opcional), o *front-end* invoca diretamente a função `avaliar` do contrato por meio da Ethers.js. A Figura 20 apresenta o trecho de código responsável por esse procedimento.

Figura 20 – Código-fonte da função de avaliação no *front-end*

```
async function avaliar() {
  try {
    const provider = new ethers.BrowserProvider(window.ethereum)
    const signer = await provider.getSigner()
    const contract = new ethers.Contract(
      AVALIACAO_CONTRACT_ADDRESS,
      abi.abi,
      signer
    )
    const tx = await contract.avaliar(filmeId, nota.value, comentario.value)
    const receipt = await tx.wait()
    txHash.value = receipt.hash
    await api.post('/avaliacoes', {
      idFilme: filmeId,
      nota: nota.value,
      comentario: comentario.value,
      txHash: receipt.hash
    })
    await fetchFilme()
    success.value = 'Avaliação registrada com sucesso!'
  } catch (err: any) {
    if (err.message?.includes('MetaMask')) {
      error.value = 'MetaMask não encontrada. Instale a extensão para continuar.'
    } else if (err.message?.includes('user rejected')) {
      error.value = 'Transação recusada pelo usuário.'
    } else if (err.message?.includes('insufficient funds')) {
      error.value = 'Saldo insuficiente para realizar a avaliação.'
    } else if (err.message?.includes('execution reverted')) {
      error.value = 'Não foi possível realizar a avaliação. Verifique se já avaliou esse filme.'
    } else {
      error.value = 'Erro inesperado ao enviar a avaliação.'
    }
  } finally {
    isLoading.value = false
  }
}
```

Fonte: Elaborado pelo autor (2025).

A função interage com o contrato implantado na rede a partir do endereço e do ABI gerado pelo projeto `movieschain-contracts`. Caso ocorra um erro durante a transação — como falta de *gas* ou tentativa de avaliação duplicada — a mensagem de erro é capturada e exibida na tela.

Para garantir transparência ao usuário, foi adicionado um link na tela de avaliações que direciona diretamente ao Etherscan, permitindo a verificação pública da transação na *blockchain*. A Figura 21 mostra o código-fonte dessa funcionalidade.

Figura 21 – Código-fonte do link para verificação da transação no Etherscan

```
<a :href="`https://sepolia.etherscan.io/tx/${avaliacao.txHash}`" target="_blank"
  class="text-primary hover:text-orange-500 font-semibold underline text-sm transition-all">
  Ver transação
</a>
```

Fonte: Elaborado pelo autor (2025).

Outro detalhe importante é a exibição da nota dos filmes em formato visual de estrelas. Apesar de armazenadas numericamente de 1 a 5, a interface utiliza um componente gráfico com cinco estrelas, preenchendo conforme a nota atribuída pelo usuário. A Figura 22 apresenta o código-fonte deste componente.

Figura 22 – Código-fonte do componente de notas em formato de estrelas

```
<div class="flex justify-center mb-6">
  <div class="flex space-x-1">
    <template v-for="n in 5" :key="n">
      <svg v-if="filme.notaUsuario !== null" :class="[ …
      </svg>
      <svg v-else @click="nota = n" @mouseover="hoverNota = n" @mouseleave="hoverNota = 0" :class="[ …
      </svg>
    </template>
  </div>
</div>
```

Fonte: Elaborado pelo autor (2025).

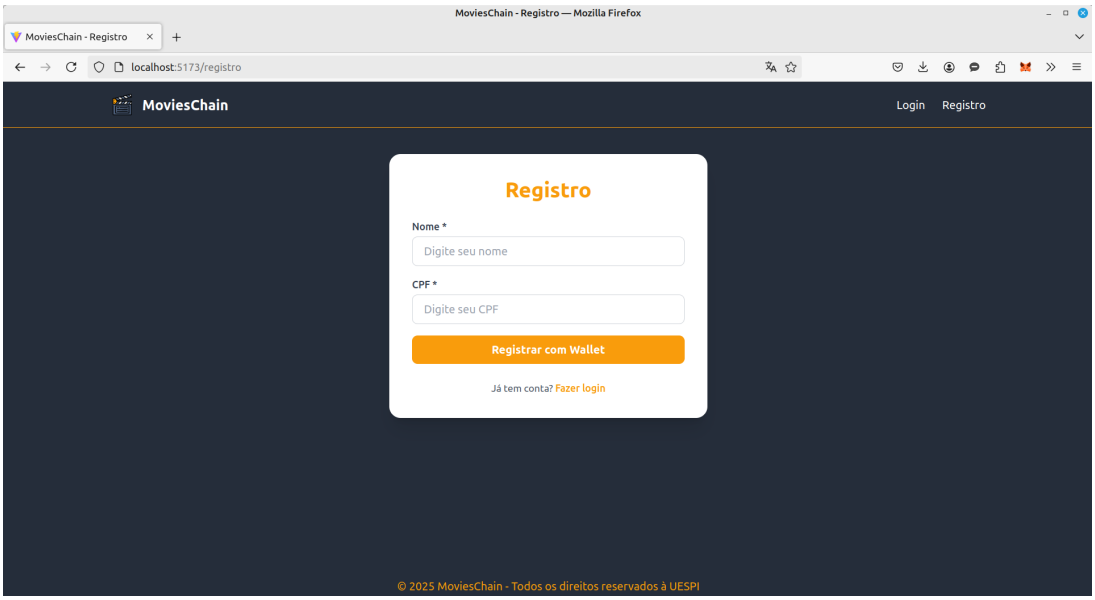
4.4 Funcionamento da Solução

Esta seção demonstra, de forma prática e por meio de imagens, como funciona o MoviesChain. É apresentado todo o fluxo de uso do sistema, desde o registro do usuário até a auditoria das transações diretamente no Etherscan.

4.4.1 Registro e Login

O registro no MoviesChain é iniciado na tela apresentada na Figura 23. O usuário preenche o nome e o CPF e, em seguida, clica no botão “Registrar com Wallet”. O botão aciona a extensão do MetaMask no navegador. Caso ela esteja instalada, o sistema solicita permissão para acessar a carteira conectada.

Figura 23 – Tela de registro do MoviesChain

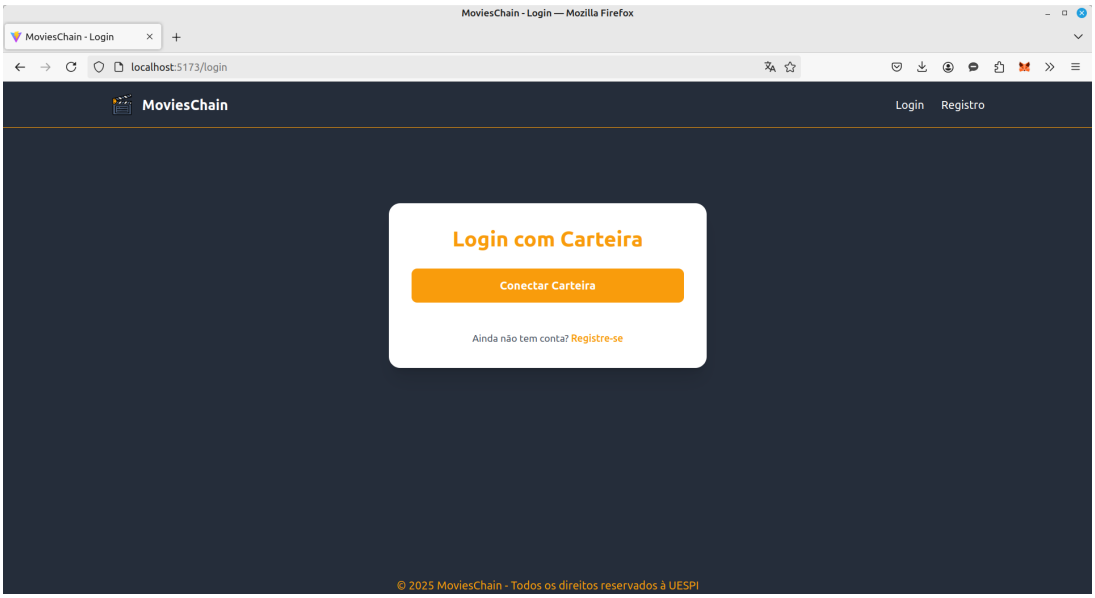


Fonte: Elaborado pelo autor (2025).

Finalizado o registro, a carteira é vinculada ao usuário. O login no sistema é feito exclusivamente via MetaMask. A Figura 24 mostra a tela de login, que é simples e direta, contendo

apenas um botão que inicia o processo de autenticação.

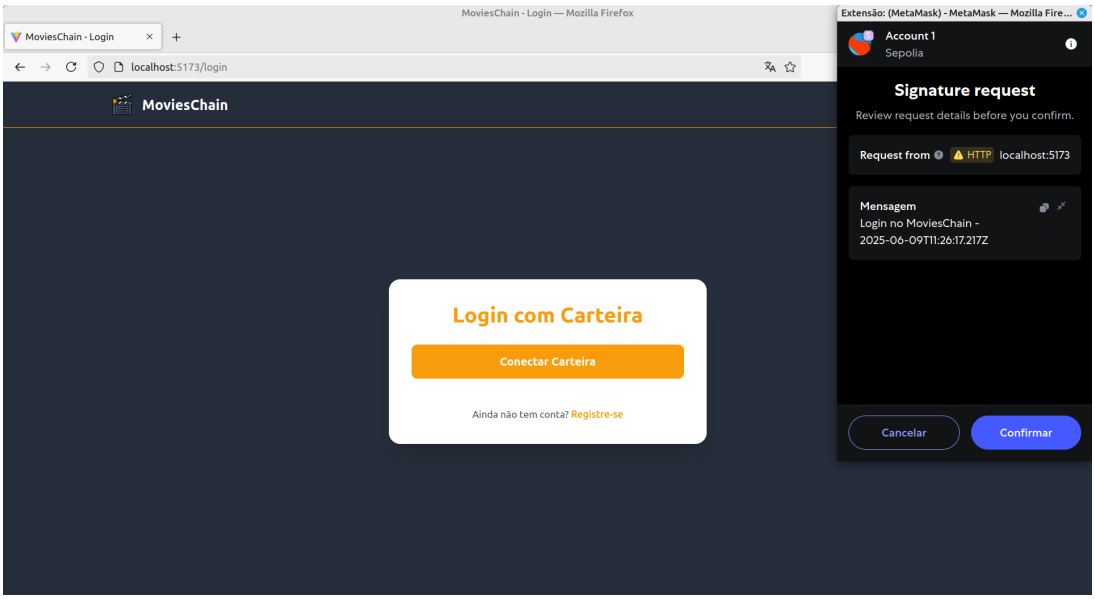
Figura 24 – Tela de login do MoviesChain



Fonte: Elaborado pelo autor (2025).

Após o início do processo de autenticação, o sistema gera automaticamente uma mensagem personalizada com o *timestamp*, que deve ser assinada pelo usuário via MetaMask. A solicitação da assinatura é ilustrada na Figura 25. Essa assinatura é uma prova de que o usuário que está tentando se logar possui a carteira. Após a confirmação, o sistema envia a mensagem assinada para o back-end, que faz a validação da assinatura e retorna o token de acesso para criação de uma sessão no navegador.

Figura 25 – Solicitação de assinatura da mensagem pela MetaMask

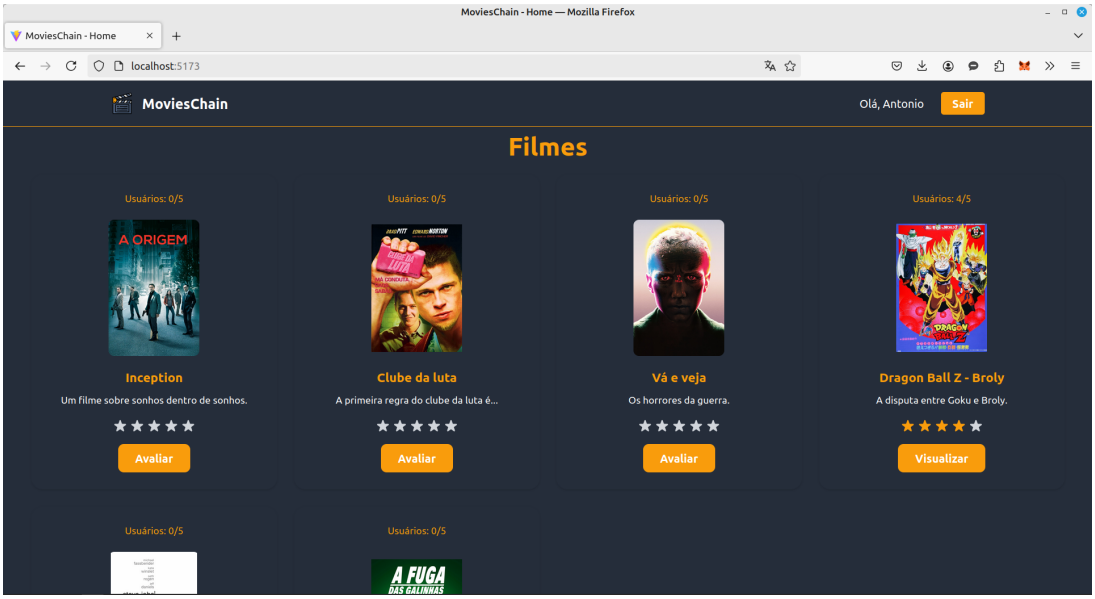


Fonte: Elaborado pelo autor (2025).

4.4.2 Página Inicial

Após a finalização do login, o usuário é redirecionado para a tela inicial do MoviesChain. Como é possível observar na Figura 26, essa é a tela onde o catálogo de filmes a serem avaliados é disponibilizado. Os filmes são exibidos em cards em um grid. Cada card contém a imagem do pôster do filme, o título, uma breve descrição e uma média das avaliações feitas pelos usuários do sistema. O card também conta com um botão chamado “Avaliar”, que redireciona para a tela de detalhes do filme, onde é possível fazer a avaliação. Caso o usuário já tenha avaliado o filme, o botão é renderizado com o nome “Visualizar”, e a pontuação atribuída pelo usuário é exibida.

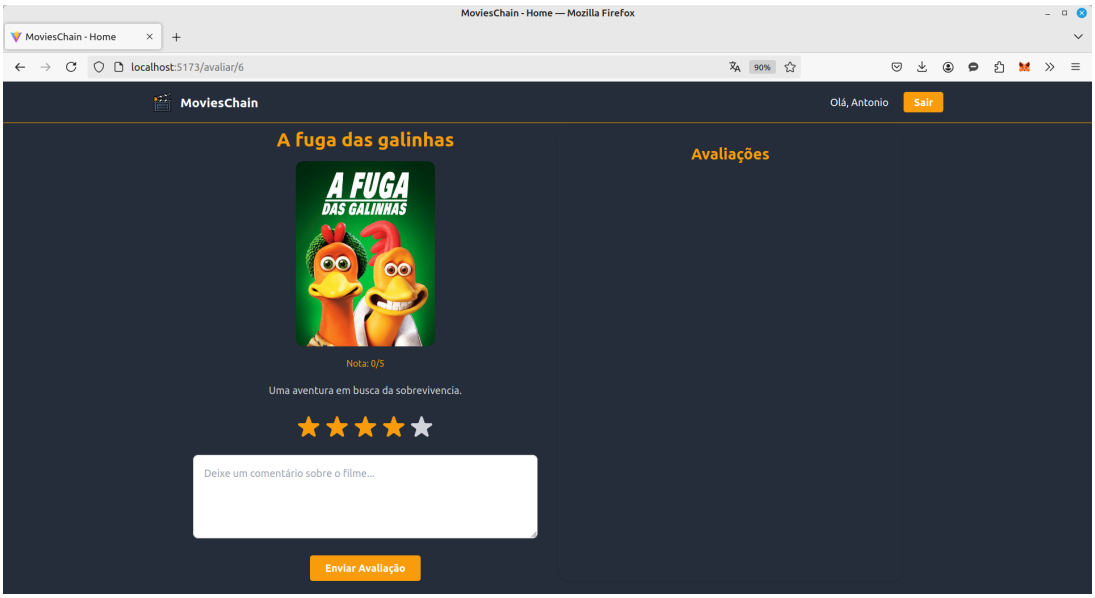
Figura 26 – Tela inicial do MoviesChain



Fonte: Elaborado pelo autor (2025).

Ao ser redirecionado para a tela de detalhes do filme, o usuário encontra as mesmas informações exibidas no card da tela anterior, de forma expandida. Também pode selecionar a quantidade de estrelas que representa sua nota para o filme, podendo marcar de uma até cinco estrelas, e adicionar um comentário opcional. A Figura 27 apresenta a tela de detalhes do filme com uma avaliação em andamento.

Figura 27 – Tela de avaliação de um filme

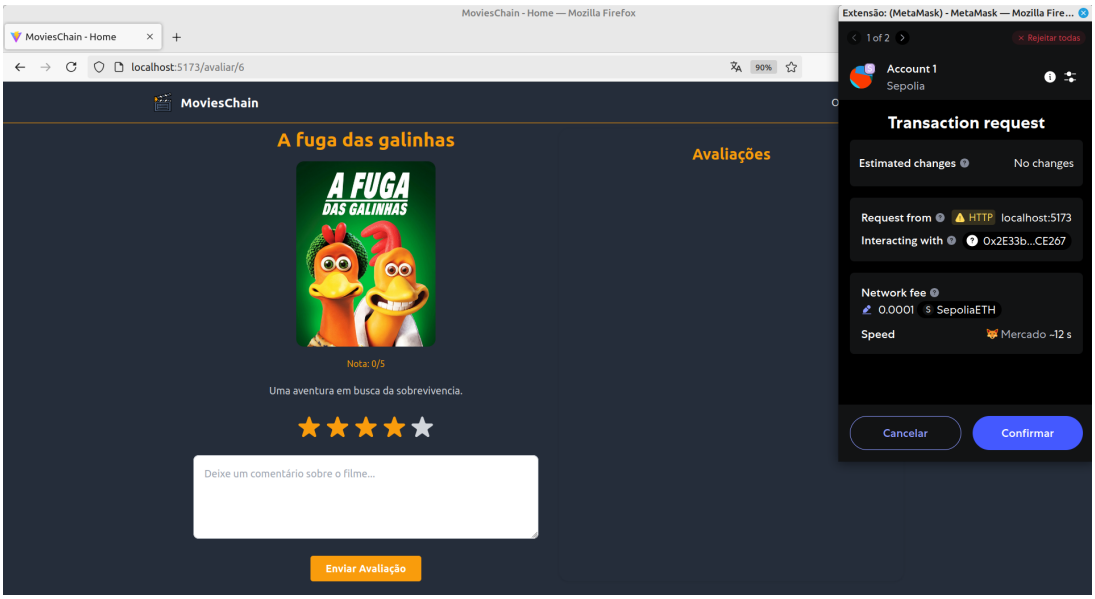


Fonte: Elaborado pelo autor (2025).

4.4.3 Avaliação

No envio da avaliação, é aberta novamente a extensão da carteira MetaMask no navegador. O usuário precisa autorizar a transação com os ETH Sepolia necessários de taxa (*gas fee*) para armazenar a transação na rede. Esse processo é ilustrado na Figura 28.

Figura 28 – Confirmação da transação de avaliação pela MetaMask

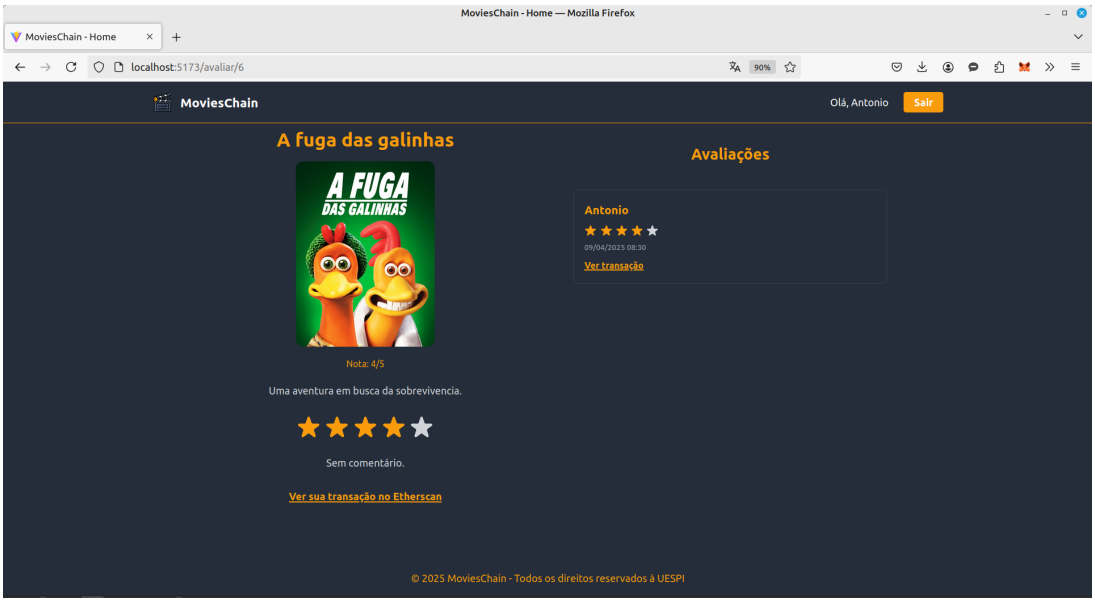


Fonte: Elaborado pelo autor (2025).

Após a confirmação, os dados da avaliação são registrados no contrato inteligente e

uma referência com o *hash* gerado da transação é enviada para a API, para ser armazenada no banco de dados. A Figura 29 exibe a tela de avaliação após o processo ter sido concluído. Nela é possível ver a nota do usuário, o botão de envio de avaliação não é renderizado e, no lugar dele, é exibido um link para o Etherscan. Esse link é utilizado para verificar a transação diretamente na rede, garantindo assim a transparência. Como é possível observar na figura, a avaliação realizada também aparece na seção de “Avaliações”. As avaliações de todos os usuários podem ser acessadas a partir dessa seção, e, para cada avaliação, também é possível acessar a transação no Etherscan.

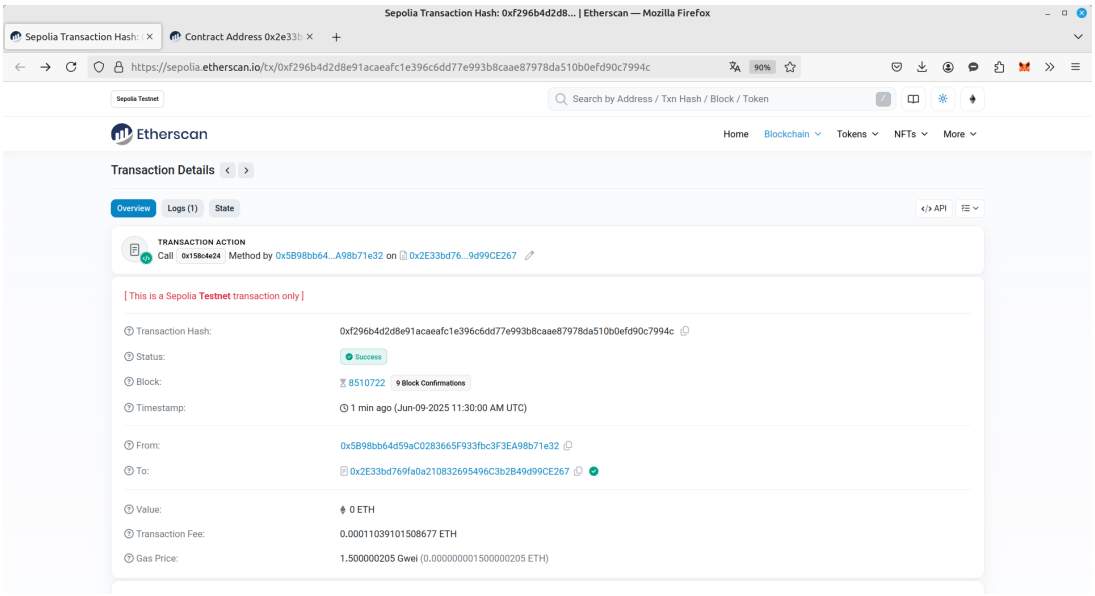
Figura 29 – Avaliação registrada com link para o Etherscan



Fonte: Elaborado pelo autor (2025).

Ao acessar o link da transação, o usuário é redirecionado para o Etherscan. A Figura 30 exibe a página da transação realizada. Nela é possível visualizar informações sobre o status da transação, horário, o bloco em que a transação foi inserida, o endereço da carteira do usuário e também o endereço do contrato que recebeu a chamada.

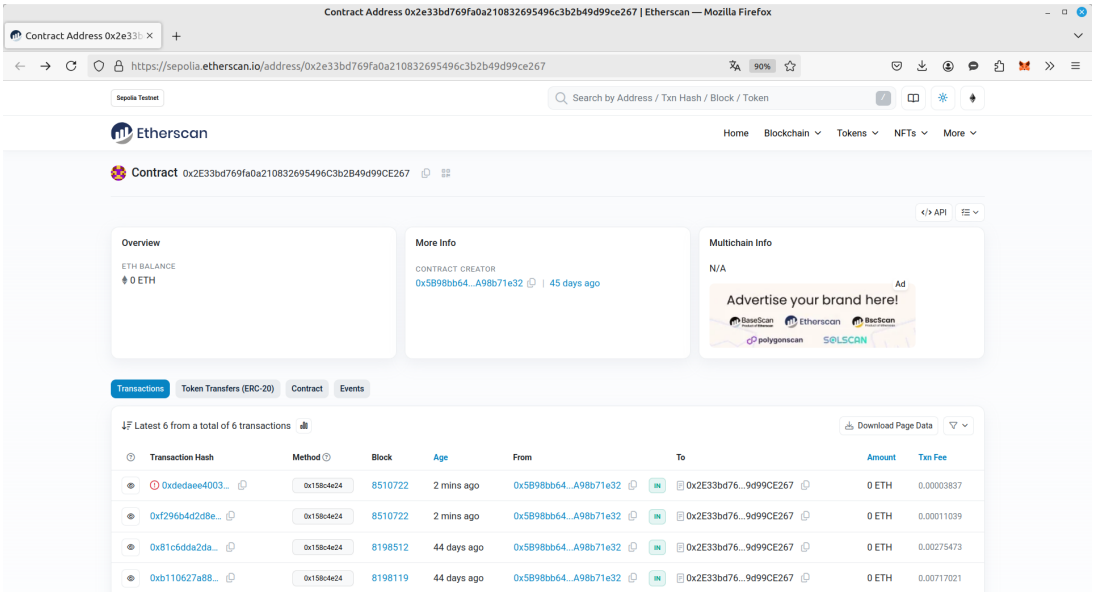
Figura 30 – Transação na Etherscan



Fonte: Elaborado pelo autor (2025).

Por último, a Figura 31 mostra a página do contrato inteligente AvaliacaoFilme no Etherscan. A partir dessa página é possível visualizar todo o histórico de transações realizadas que passaram pelo contrato. São exibidas também informações sobre o endereço do contrato, qual endereço o criou, o próprio contrato em Solidity e o saldo atual do contrato em ETH Sepolia. Com essas informações disponíveis de forma pública e mantidas pela própria rede Ethereum, o MoviesChain consegue alcançar seu objetivo de ser transparente e auditável.

Figura 31 – Página do contrato AvaliacaoFilme no Etherscan



Fonte: Elaborado pelo autor (2025).

5 CONCLUSÕES

5.1 Conclusão

O MoviesChain conseguiu demonstrar que a tecnologia blockchain pode ser utilizada de forma efetiva para resolver problemas relevantes dos sistemas de avaliação centralizados. Este trabalho foi responsável por viabilizar tecnicamente a criação desse sistema, que possui como principais características a imutabilidade, transparência e resistência à manipulação das avaliações — propriedades essenciais para garantir a confiança do usuário na plataforma.

A principal conquista foi a garantia de transparência e auditabilidade das avaliações. A blockchain Ethereum Sepolia garantiu a imutabilidade dos dados: as avaliações no sistema não podem ser alteradas ou removidas, criando um histórico permanente e seguro. Ataques como o review bombing ou duplicidade de avaliações também são evitados pelo contrato inteligente. A validação automática no momento da criação da transação impede que múltiplas avaliações do mesmo usuário sejam associadas ao mesmo filme.

Além disso, o trabalho oferece uma base teórica sólida e uma metodologia que pode ser replicada em futuros trabalhos da área. Embora limitações como o custo de transação e complexidades relacionadas à descentralização devam ser levadas em consideração ao avaliar a criação de sistemas como o MoviesChain, o potencial de transformação demonstrado por ele é relevante e promissor.

Dessa forma, o trabalho contribui para o avanço do conhecimento científico sobre sistemas descentralizados e, ao mesmo tempo, oferece uma solução prática para um problema real e comum atualmente, permitindo assim que o estudo seja utilizado como base para o desenvolvimento de novos sistemas de reputação mais confiáveis e transparentes.

5.2 Contribuições

As principais contribuições deste trabalho são destacadas a seguir:

- Análise e documentação detalhada sobre desafios e vantagens na implementação de sistemas descentralizados de avaliação.
- Demonstração da integração entre tecnologias descentralizadas, como a *blockchain*, contratos inteligentes e carteiras digitais, com ferramentas de desenvolvimento tradicionais, como o Vue.js.
- Implementação e deploy de um contrato inteligente que garante a unicidade de avaliações, podendo ser utilizado em trabalhos similares.

- Análise comparativa entre sistemas centralizados e descentralizados de reputação, podendo servir como base para pesquisas futuras.
- Código-fonte aberto disponibilizado no GitHub, permitindo que outros pesquisadores possam reproduzir a solução.
- Demonstração prática de como combater desafios como o review bombing e a manipulação de avaliações utilizando tecnologias descentralizadas.

5.3 Limitações Encontradas

Apesar das vantagens apontadas, o MoviesChain apresenta algumas limitações importantes que precisam ser consideradas. Uma delas é o custo das transações (*gas fees*) na rede Ethereum. Na rede de testes Sepolia, o gasto não é real, pois a moeda ETH Sepolia é simulada e fácil de minerar pela internet, sem nenhum custo. Porém, se o sistema fosse implantado na rede principal Ethereum, o usuário teria que utilizar Ethers reais para pagar a taxa cada vez que uma avaliação fosse registrada.

Arshad et al. (2022) destacam que as blockchains enfrentam desafios críticos relacionados à eficiência no processamento rápido de transações. Apesar da abordagem híbrida do MoviesChain, a funcionalidade de envio e armazenamento das avaliações ainda é lenta se comparada com plataformas como o IMDb.

A própria arquitetura híbrida pode ser considerada uma limitação, já que introduz um ponto de centralização no sistema. O processo de avaliação é descentralizado; porém, ao armazenar detalhes dos filmes e informações complementares do usuário, o sistema mantém a dependência de um servidor tradicional para esses dados. No MoviesChain, essa foi uma decisão técnica, escolhida pela economia de dados e melhor experiência do usuário.

5.4 Trabalhos Futuros

Considerando as melhorias e limitações observadas ao longo do trabalho, é possível sugerir diversos caminhos para trabalhos futuros. Alguns deles são listados a seguir:

- Análise da viabilidade de implementação de uma plataforma de avaliação de filmes na rede principal Ethereum.
- Investigar formas de reduzir os custos de transação e aumentar a velocidade de processamento da solução.
- Expandir o foco do MoviesChain para outros tipos de mídia, como livros, jogos ou música.
- Melhorias na interface do usuário, com organização dos filmes em seções e mecanismos de busca, como é comum em plataformas centralizadas.
- Desenvolver mecanismos para reputação dos próprios usuários da plataforma.

- Estudo sobre a aceitação e usabilidade do MoviesChain com usuários reais.

REFERÊNCIAS

ABDULMUNIM, M. et al. Movie recommendation and classification system using blockchain. *Web Intelligence*, IOS Press, p. 567–579, 2024. Disponível em: <https://doi.org/10.3233/WEB-230346>. Citado na página 30.

ANTONOPOULOS, A. M. *Mastering Bitcoin: Programming the Open Blockchain*. 2. ed. Sebastopol, CA: O'Reilly Media, 2017. ISBN 9781491954386. Citado na página 28.

ARIDOR, G.; CHE, Y.-K.; SALZ, T. *The Informational Role of Online Recommendations: Evidence from a Field Experiment*. 2022. Available at SSRN and arXiv. Disponível em: <https://arxiv.org/abs/2211.14219>. Citado na página 16.

ARSHAD, J. et al. Decentralized reputation management using blockchain technology: A survey. *IEEE Access*, IEEE, 2022. Disponível em: <https://ieeexplore.ieee.org/document/9840359>. Citado 2 vezes nas páginas 21 e 71.

BOOT, S. *Spring Boot Documentation*. 2025. Disponível em: <https://spring.io/projects/spring-boot>. Citado na página 39.

CANTONE, D. et al. Review bombing: ideology-driven polarisation in online ratings. *Quality & Quantity*, Springer, 2024. Disponível em: <https://doi.org/10.1007/s11135-024-01981-z>. Citado 3 vezes nas páginas 13, 17 e 18.

CHAINLINK. *Reentrancy Attacks and The DAO Hack*. 2022. Acesso em: 20 março 2025. Disponível em: <https://blog.chain.link/reentrancy-attacks-and-the-dao-hack/>. Citado na página 29.

DOGAN, □.; CAN, H. K. A blockchain-based e-commerce reputation system built with verifiable credentials. *IEEE Access*, IEEE, 2023. Disponível em: <https://doi.org/10.1109/ACCESS.2023.3274707>. Citado na página 31.

ETHEREUM. *Networks: Sepolia*. 2025. Disponível em: <https://ethereum.org/en/developers/docs/networks/#sepolia>. Citado na página 39.

ETHERS.JS. *Ethers.js Documentation*. 2025. Disponível em: <https://docs.ethers.org/v5/>. Citado na página 37.

GEMINI. *The DAO Hack*. 2022. Acesso em: 20 março 2025. Disponível em: <https://www.gemini.com/cryptopedia/the-dao-hack-makerdao>. Citado na página 29.

GUEDES, G. T. *UML 2: Uma abordagem prática*. 3. ed. São Paulo: Novatec Editora, 2018. Citado 2 vezes nas páginas 44 e 53.

HARDHAT. *Hardhat Documentation*. 2025. Acesso em: 10 maio 2025. Disponível em: <https://hardhat.org/>. Citado na página 36.

HAZIM, M. et al. Detecting opinion spams through supervised boosting approach. *PLOS ONE*, Public Library of Science, 2018. Disponível em: <https://doi.org/10.1371/journal.pone.0198884>. Citado 2 vezes nas páginas 13 e 18.

JIANG, Z.; RAVICHANDRAN, T.; KURUZOVICH, J. Review moderation transparency and online reviews: Evidence from a natural experiment. *MIS Quarterly*, 2023. Forthcoming. Disponível em: <https://misq.umn.edu/review-moderation-transparency-and-online-reviews-evidence-from-a-natural-experiment.html>. Citado na página 17.

LAKATOS, E. M.; MARCONI, M. d. A. *Fundamentos de metodologia científica*. 5. ed. São Paulo: Atlas, 2003. Citado na página 33.

LIU, X.; MUNRO, M. Systematic analysis of centralized online reputation systems. In: *Proceedings of the International Conference on Software Engineering and Knowledge Engineering (SEKE)*. Knowledge Systems Institute, 2012. Disponível em: <https://www.researchgate.net/publication/220196101>. Citado 2 vezes nas páginas 19 e 20.

METAMASK. *MetaMask Documentation*. 2025. Acesso em: 11 maio 2025. Disponível em: <https://support.metamask.io/start/getting-started-with-metamask#what-is-metamask>. Citado na página 37.

NAKAMOTO, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. Available at: <https://bitcoin.org/bitcoin.pdf>. Disponível em: <https://bitcoin.org/bitcoin.pdf>. Citado na página 24.

NARAYANAN, A. et al. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton, NJ: Princeton University Press, 2016. ISBN 9780691171692. Citado 2 vezes nas páginas 27 e 29.

PRESSMAN, R. S. *Software Engineering: A Practitioner's Approach*. 7. ed. New York: McGraw-Hill, 2010. ISBN 978-0-07-337597-7. Citado na página 35.

SALEH, F. Blockchain without waste: Proof-of-stake. *The Review of Financial Studies*, Oxford University Press, 2021. Disponível em: <https://doi.org/10.1093/rfs/hhaa075>. Citado na página 25.

SANCHES, H. M. O uso da unified modeling language (uml) na modelagem de software. In: *Innovate: Engenharia de computação*. Ponta Grossa, PR: Atena Editora, 2023. Disponível em: <https://www.researchgate.net/publication/373295090_O_USO_DA_UNIFIED_MODELING_LANGUAGE_UML_NA_MODELAGEM_DE_SOFTWARE>. Citado na página 42.

SAVEETHA, D.; MARAGATHAM, G. Movie rating system based on blockchain. In: *2021 International Conference on Computer Communication and Informatics (ICCCI)*. Coimbatore, India: IEEE, 2021. Disponível em: <https://doi.org/10.1109/ICCCI50826.2021.9402381>. Citado na página 30.

SHARKEY, A.; HSU, G.; KOVáCS, B. *Expert Critics, Rankings, and Review Aggregators: The Changing Nature of Intermediation and the Rise of Markets with Multiple Intermediaries*. 2022. Working paper. Disponível em: <https://www.researchgate.net/publication/359792741>. Citado na página 16.

SOLIDITY. *Solidity Documentation*. 2025. Acesso em: 10 maio 2025. Disponível em: <https://soliditylanguage.org/>. Citado na página 36.

SOMMERVILLE, I. *Engenharia de Software*. 9. ed. São Paulo: Pearson Prentice Hall, 2011. Tradução da 9ª edição original em inglês, com revisão técnica de Prof. Dr. Kechi Hirama. ISBN 9788579361081. Citado na página 42.

SZABO, N. *Formalizing and Securing Relationships on Public Networks*. 1997. Acesso em: 25 maio 2025. Disponível em: <https://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/formalize.html>. Citado na página 28.

TADELIS, S. Reputation and feedback systems in online platform markets. *Annual Review of Economics*, v. 8, n. 1, p. 321–340, 2016. Disponível em: <https://doi.org/10.1146/annurev-economics-080614-115222>. Citado na página 16.

TAILWIND. *Tailwind CSS Documentation*. 2025. Disponível em: <https://tailwindcss.com/docs>. Citado na página 38.

VALENTE, M. T. *Engenharia de Software Moderna: Princípios e Práticas para Desenvolvimento de Software com Produtividade*. Belo Horizonte: Publicação Independente, 2020. Disponível em: <https://engsoftmoderna.info>. Citado na página 44.

VITE. *Vite Documentation*. 2025. Disponível em: <https://vitejs.dev/>. Citado na página 38.

VUE.JS. *Vue.js Documentation*. 2025. Disponível em: <https://vuejs.org/>. Citado na página 38.

WERBACH, K. *The Blockchain and the New Architecture of Trust*. Cambridge, MA: MIT Press, 2018. ISBN 9780262038935. Disponível em: <https://mitpress.mit.edu/9780262038935/>. Citado 4 vezes nas páginas 22, 23, 24 e 27.

WU, D. et al. Fake online reviews: Literature review, synthesis, and directions for future research. *Decision Support Systems*, Elsevier, 2020. Disponível em: <https://www.sciencedirect.com/science/article/pii/S016792362030035X>. Citado na página 18.

YANG, L. et al. A comprehensive review of blockchain technology: Underlying principles, applications, and challenges. *Digital Communications and Networks*, Elsevier, 2023. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2772662223001844>. Citado na página 13.

ZHENG, Z. et al. An overview of blockchain technology: Architecture, consensus, and future trends. In: *2017 IEEE International Congress on Big Data (BigData Congress)*. IEEE, 2017. Disponível em: <https://doi.org/10.1109/BigDataCongress.2017.85>. Citado 2 vezes nas páginas 25 e 29.