INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
CAMPUS CAJAZEIRAS
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

MANA CLUB: DESENVOLVIMENTO DE UMA APLICAÇÃO WEB PARA GERENCIAMENTO DE COLEÇÕES DE MAGIC: THE GATHERING

EDUARDO DÊNIS PAIVA WHITEHURST
LÚCIO WESLLEY DE SOUZA LEITE

Cajazeiras
2026

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA

LUCIO WESLLEY DE SOUZA LEITE

E

EDUARDO DÊNIS PAIVA WHITEHURST

MANACLUB:DESENVOLVIMENTO DEUMAAPLICAÇÃOWEB PARA GERENCIAMENTO DE COLEÇÕES DE MAGIC:THE GATHERING

Trabalho de Conclusão de Curso apresentado junto ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - Campus Cajazeiras, como requisito à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador

Prof. Dr. Francisco Daladier Marques Júnior

Aprovada em: **14 de agosto de 2026.**

Prof. Dr. Francisco Daladier Marques Júnior - Orientador

IFPB - Campus Cajazeiras

Prof. Me. Francisco Paulo de Freitas Neto - Avaliador

IFPB - Campus Cajazeiras

IFSertãoPB / Campus Cajazeiras
Coordenação de Biblioteca
Biblioteca Prof. Ribamar da Silva
Catalogação na fonte: Cícero Luciano Félix CRB-15/750

W593m

Whitehurst, Eduardo Dênis Paiva.

Mana club : desenvolvimento de uma aplicação web Para gerenciamento de coleções de Magic : The Gathering / Eduardo Dênis Paiva Whitehurst, Lúcio Weslley De Souza Leite.– 2026.

62f. : il.

Trabalho de Conclusão de Curso (Tecnólogo em Análise e Desenvolvimento de Sistemas) - Instituto Federal de Educação, Ciência e Tecnologia do Sertão Paraibano, Cajazeiras, 2026.

Orientador(a): Prof. Dr. Francisco Daladier Marques Júnior.

1. Desenvolvimento de sistemas. 2. Engenharia de Software. 3. Aplicação web. 4. Gerenciamento de coleções. I. Leite, Lúcio Weslley De Souza. II. Instituto Federal de Educação, Ciência e Tecnologia da Paraíba. III. Título.

IFSertão PB/CZ

CDU: 004.4 (043.2)

Prof. Me. Diogo Dantas Moreira - Avaliador

IFPB - Campus Cajazeiras

Documento assinado eletronicamente por:

- **Francisco Daladier Marques Junior**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 15/08/2026 10:55:53.
- **Francisco Paulo de Freitas Neto**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 15/08/2026 11:30:59.
- **Diogo Dantas Moreira**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 15/08/2026 14:19:41.

Este documento foi emitido pelo SUAP em 15/08/2026. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código 923731
Verificador: ae1438e3ac
Código de Autenticação:

EDUARDO DÊNIS PAIVA WHITEHURST
LÚCIO WESLLEY DE SOUZA LEITE

MANA CLUB: DESENVOLVIMENTO DE UMA APLICAÇÃO WEB PARA GERENCIAMENTO DE COLEÇÕES DE MAGIC: THE GATHERING

Trabalho de Conclusão de Curso apresentado junto ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - Campus Cajazeiras, como requisito à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador

Prof. Dr. Francisco Daladier Marques Júnior.

Cajazeiras
2026

Dedicamos este projeto aos amigos que compartilharam conosco mesas de estudo, partidas de Magic e incontáveis conversas que, muitas vezes, começaram sobre cartas e terminaram inspirando ideias.

AGRADECIMENTOS

Concluir esta etapa representa mais do que finalizar um Trabalho de Conclusão de Curso. Representa o encerramento de uma jornada marcada por desafios, aprendizados, noites de estudo, prazos apertados e inúmeras linhas de código.

Agradecemos, primeiramente, às nossas famílias, pelo apoio incondicional, pela paciência nos momentos de ausência e pelo incentivo para que nunca desistíssemos dos nossos objetivos.

Agradecemos aos professores do Curso de Análise e Desenvolvimento de Sistemas do Instituto Federal da Paraíba, Campus Cajazeiras, que compartilharam conhecimento, experiência e dedicação ao longo de toda a graduação. Cada disciplina contribuiu, de alguma forma, para a construção deste projeto.

Ao nosso orientador, pela confiança depositada, pelas sugestões, pelas críticas construtivas e pelo acompanhamento durante o desenvolvimento deste trabalho.

Aos colegas de curso, que dividiram dúvidas, trabalhos, desafios e conquistas ao longo desses anos, tornando a caminhada mais leve e enriquecedora.

Por fim, agradecemos um ao outro. Este trabalho nasceu muito antes da escrita deste documento. Nasceu entre uma aula e outra, em conversas sobre desenvolvimento de software, arquitetura de sistemas e, principalmente, em inúmeras partidas de *Magic: The Gathering*. O que começou como um interesse em comum tornou-se uma parceria acadêmica e resultou no Mana Club, um projeto que reúne parte daquilo que aprendemos durante a graduação com um universo que sempre fez parte da nossa amizade.

Esperamos que este trabalho represente não apenas a conclusão de um curso, mas também o início de muitos outros projetos desenvolvidos com a mesma parceria, curiosidade e entusiasmo que nos trouxeram até aqui.

"Plans are nothing. Planning is everything."

Dwight D. Eisenhower

RESUMO

Jogadores de *Magic: The Gathering* frequentemente enfrentam dificuldades para organizar suas coleções físicas, registrar cartas desejadas e identificar oportunidades de troca. Este trabalho apresenta o desenvolvimento do *Mana Club*, uma aplicação web que integra essas funcionalidades em uma única plataforma. A solução foi desenvolvida com base em princípios da Engenharia de Software, abrangendo levantamento de requisitos, modelagem, implementação e verificação. A aplicação utiliza arquitetura cliente-servidor, com interface desenvolvida em Next.js, React, TypeScript e Tailwind CSS, API REST implementada em Node.js, Express e TypeScript, banco de dados PostgreSQL e Redis para armazenamento temporário dos dados das coleções. A API Scryfall fornece os dados catalográficos das cartas. Como resultado, o sistema permite o cadastro e a autenticação de usuários, o gerenciamento de coleções e listas de desejos, a consulta a cartas disponibilizadas por outros jogadores e o gerenciamento de propostas de troca. Os principais fluxos foram executados manualmente conforme os requisitos e critérios de aceitação estabelecidos. A verificação indicou que as funcionalidades previstas para a primeira versão estavam disponíveis. Conclui-se que o *Mana Club* atende ao escopo funcional proposto e oferece uma solução integrada para organizar coleções, planejar aquisições e apoiar a negociação de cartas.

Palavras-chave: Engenharia de Software. Aplicação Web. Magic: The Gathering. Trading Card Games. API REST. Gerenciamento de Coleções.

ABSTRACT

The growth of Trading Card Games (TCGs) has fostered increasingly active communities and collections composed of hundreds or even thousands of cards. In this context, *Magic: The Gathering* players often face difficulties in managing their physical collections, tracking desired cards, avoiding duplicate purchases, and organizing card trades with other players. Although several applications have been developed to support collection management, many of them present limitations regarding web availability, trading support, or the integration of essential features into a single platform. This work presents the development of *Mana Club*, a web application designed to manage *Magic: The Gathering* card collections. The proposed solution was developed following Software Engineering principles, encompassing requirements elicitation, system modeling, software architecture definition, implementation, and verification. The application adopts a client–server architecture composed of a web interface developed with Next.js, React, and TypeScript, a REST API implemented in Node.js, Express, and TypeScript, and PostgreSQL as the relational database management system. Redis temporarily stores data about the cards in each user's collection, whereas the Scryfall API is used separately as the source of card catalog data. As a result, the developed application enables users to manage personal collections, create wishlists, browse cards made available for trade by other users, and manage trade proposals. The main functional flows were executed manually according to the requirements and acceptance criteria defined in the Software Requirements Specification. This verification indicated that the features planned for the first version were available, but it did not include usability studies with external participants or quantitative performance tests. The results indicate that Mana Club fulfills its proposed functional scope by providing an integrated solution for managing *Magic: The Gathering* collections and supporting acquisition planning and card trading among players.

Keywords: Software Engineering. Web Application. Magic: The Gathering. Trading Card Games. REST API. Collection Management.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
ADS	Análise e Desenvolvimento de Sistemas
API	Interface de Programação de Aplicações
CRUD	Create, Read, Update e Delete
ERS	Especificação de Requisitos de Software
HTTP	Hypertext Transfer Protocol
IFPB	Instituto Federal de Educação, Ciência e Tecnologia da Paraíba
JSON	JavaScript Object Notation
JWT	JSON Web Token
DER	Diagrama Entidade-Relacionamento
ORM	Object-Relational Mapping
REST	Representational State Transfer
RF	Requisito Funcional
RNF	Requisito Não Funcional
SPA	Single Page Application
SQL	Structured Query Language
SSR	Server-Side Rendering
TCG	Trading Card Game
UML	Unified Modeling Language

SUMÁRIO

1 INTRODUÇÃO

Os Jogos de Cartas Colecionáveis (*Trading Card Games – TCGs*) consolidaram-se como um dos segmentos mais relevantes da indústria mundial de entretenimento. Diferentemente dos jogos tradicionais, nos quais todos os participantes utilizam o mesmo conjunto de componentes, os TCGs baseiam-se na construção progressiva de coleções individuais de cartas, permitindo que cada jogador desenvolva estratégias próprias a partir dos recursos que possui. Essa característica faz com que a experiência do jogador extrapole a própria partida, envolvendo atividades de organização da coleção, montagem de decks, aquisição de novos exemplares e negociação entre participantes da comunidade (HUIZINGA, 2019; SALEN; ZIMMERMAN, 2004).

Entre os jogos desse gênero, *Magic: The Gathering* (MTG) ocupa posição de destaque por ter estabelecido o modelo dos jogos de cartas colecionáveis modernos desde seu lançamento, em 1993. Ao longo de mais de três décadas, o jogo expandiu continuamente seu universo por meio da publicação de novas coleções, formatos competitivos e eventos oficiais, resultando em um catálogo composto por dezenas de milhares de impressões de cartas. Esse crescimento, aliado à diversidade de perfis de jogadores — casuais, colecionadores e competidores — tornou o gerenciamento das coleções pessoais uma atividade cada vez mais complexa (Wizards of the Coast, 2026).

A administração de uma coleção de *Magic: The Gathering* envolve muito mais do que o simples registro das cartas possuídas. Um mesmo jogador frequentemente precisa identificar quais exemplares possui, quais deseja adquirir, quais estão disponíveis para negociação e quais já foram utilizados em diferentes estratégias ou formatos de jogo. Além disso, uma mesma carta pode existir em diferentes idiomas, edições, acabamentos e estados de conservação, características que influenciam tanto seu valor quanto seu interesse para colecionadores e jogadores.

Nos últimos anos surgiram diversas aplicações destinadas ao gerenciamento de coleções e construção de decks. Entretanto, observa-se que muitas dessas soluções concentram-se em funcionalidades específicas, exigindo que os usuários recorram a diferentes plataformas para controlar sua coleção, registrar cartas desejadas e organizar negociações. Essa fragmentação dificulta a administração das informações e reduz a integração entre atividades que fazem parte do cotidiano dos jogadores.

Diante desse contexto, este trabalho apresenta o Mana Club, uma aplicação

web desenvolvida para apoiar o gerenciamento de coleções físicas de Magic: The Gathering. A proposta consiste em reunir, em uma única plataforma, funcionalidades relacionadas ao cadastro de exemplares, gerenciamento de listas de desejos (wishlist), disponibilização de cartas para negociação e controle das propostas de troca realizadas entre usuários.

Do ponto de vista tecnológico, o desenvolvimento do Mana Club proporcionou a aplicação integrada de conceitos estudados ao longo do curso de Análise e Desenvolvimento de Sistemas, abrangendo Engenharia de Software, Levantamento de Requisitos, arquitetura de aplicações web, desenvolvimento de APIs REST, modelagem de bancos de dados relacionais, integração com serviços externos e mecanismos de cache. Dessa forma, além de atender a uma necessidade identificada na comunidade de jogadores, o projeto constitui um estudo aplicado sobre o desenvolvimento de sistemas web modernos fundamentados em princípios consolidados da Engenharia de Software.

1.1 CONTEXTUALIZAÇÃO

Os Jogos de Cartas Colecionáveis distinguem-se de outros jogos por atribuírem aos participantes a responsabilidade pela construção e administração de suas próprias coleções. Cada jogador seleciona quais cartas deseja adquirir, manter ou negociar, organizando seu acervo de acordo com seus objetivos pessoais, sejam eles competitivos, recreativos ou voltados ao colecionismo. Nesse cenário, a coleção deixa de ser apenas um conjunto de cartas e passa a representar um patrimônio informacional que precisa ser constantemente atualizado e consultado.

Essa característica torna-se particularmente evidente em *Magic: The Gathering*, cuja expansão contínua resulta na publicação recorrente de novas coleções, mecânicas e impressões de cartas. Como consequência, uma coleção pessoal pode reunir centenas ou milhares de exemplares com diferentes edições, idiomas, acabamentos e estados de conservação, exigindo do jogador um controle cada vez mais detalhado sobre seu acervo.

Embora existam ferramentas voltadas para consulta de cartas, construção de decks ou gerenciamento de coleções, muitas delas abordam apenas parte desse processo. Em consequência, torna-se comum que jogadores utilizem diferentes aplicações para desempenhar tarefas distintas, como registrar cartas desejadas, controlar exemplares disponíveis para troca ou negociar diretamente com outros usuários. Essa fragmentação reduz a praticidade do gerenciamento das coleções e dificulta a centralização das informações necessárias ao processo de aquisição e negociação de

cartas.

Nesse contexto, o desenvolvimento de uma aplicação capaz de integrar essas funcionalidades em um único ambiente representa uma alternativa para simplificar a administração das coleções e apoiar as interações existentes entre os membros da comunidade.

1.2 PROBLEMA DE PESQUISA

À medida que uma coleção cresce, aumenta também a dificuldade para manter informações atualizadas sobre os exemplares disponíveis, as cartas desejadas e os itens passíveis de negociação. A ausência de um controle centralizado favorece problemas como compras duplicadas, dificuldade para localizar cartas específicas durante a montagem de decks e perda de oportunidades de troca entre jogadores.

Embora existam aplicações destinadas ao gerenciamento de coleções ou à construção de decks, observa-se que muitas delas tratam essas funcionalidades de forma isolada, exigindo que o usuário utilize diferentes plataformas para atender às suas necessidades.

Diante desse cenário, este trabalho busca responder à seguinte questão de pesquisa:

> Como uma aplicação web pode auxiliar jogadores de *Magic: The Gathering* no gerenciamento de suas coleções de cartas, no controle de cartas desejadas e na organização de negociações entre usuários?

1.3 JUSTIFICATIVA

O desenvolvimento do Mana Club justifica-se pela necessidade de oferecer aos jogadores de *Magic: The Gathering* uma ferramenta capaz de centralizar informações relacionadas às suas coleções físicas. Ao reunir gerenciamento da coleção, lista de desejos, marketplace e propostas de troca, a aplicação reduz a fragmentação entre diferentes ferramentas e facilita a administração do acervo pelo usuário.

Sob o ponto de vista tecnológico, o projeto possibilitou a aplicação integrada de diferentes conceitos estudados durante a graduação, incluindo arquitetura cliente-servidor, desenvolvimento de APIs REST (FIELDING, 2000), persistência de dados relacionais, utilização de mecanismos de cache e integração com serviços externos. Dessa forma, o Mana Club representa um estudo de caso sobre o desenvolvimento

de aplicações web modernas utilizando tecnologias amplamente empregadas pela indústria de software.

No âmbito acadêmico, este trabalho contribui para demonstrar a aplicação prática dos princípios da Engenharia de Software durante o desenvolvimento de um produto completo. A utilização de técnicas de levantamento de requisitos, modelagem UML, desenvolvimento incremental e arquitetura em camadas permitiu estruturar o projeto de maneira organizada, favorecendo sua manutenção e evolução (PRESSMAN; MAXIM, 2021; SOMMERVILLE, 2019).

Além disso, o trabalho aproxima conceitos teóricos da realidade vivenciada por uma comunidade específica de usuários, evidenciando como decisões de projeto, integração de APIs e modelagem de dados podem ser empregadas na solução de problemas concretos.

1.4 OBJETIVOS

1.4.1 Objetivo Geral

Desenvolver uma aplicação web integrada para gerenciamento de coleções físicas de cartas de *Magic: The Gathering*, permitindo o controle dos exemplares pertencentes ao usuário, o gerenciamento de cartas desejadas e a organização de negociações entre jogadores.

1.4.2 Objetivos Específicos

- desenvolver funcionalidades de cadastro, autenticação e gerenciamento de usuários;
- implementar o cadastro, consulta, atualização e remoção de exemplares da coleção do usuário;
- integrar os dados das cartas por meio da API pública Scryfall;
- implementar funcionalidades para gerenciamento de *Wishlist*;
- permitir a consulta a exemplares disponibilizados para troca por outros usuários;
- permitir a criação, consulta, resposta e cancelamento de propostas de troca;
- avaliar a aplicação desenvolvida quanto ao atendimento dos requisitos estabelecidos.

1.5 ORGANIZAÇÃO DO TRABALHO

Este trabalho está estruturado em cinco capítulos. O Capítulo 2 apresenta a revisão de literatura, abordando conceitos relacionados aos Jogos de Cartas Colecionáveis, ao universo de *Magic: The Gathering*, às tecnologias empregadas no desenvolvimento da aplicação e aos fundamentos de Engenharia de Software que sustentam a proposta.

O Capítulo 3 descreve a metodologia adotada durante o desenvolvimento do Mana Club, apresentando o processo de desenvolvimento, a arquitetura da solução e os principais artefatos de modelagem utilizados.

O Capítulo 4 apresenta os resultados obtidos com a implementação da aplicação, descrevendo suas funcionalidades, as principais decisões técnicas adotadas e a avaliação da solução desenvolvida.

Por fim, o Capítulo 5 apresenta as considerações finais do trabalho, discutindo as contribuições alcançadas, as limitações identificadas e as possibilidades de evolução do Mana Club em trabalhos futuros.

2 REVISÃO DE LITERATURA

O desenvolvimento de um sistema para gerenciamento de coleções de *Magic: The Gathering* exige compreender não apenas os aspectos tecnológicos envolvidos em sua implementação, mas também o domínio ao qual a aplicação se destina. Dessa forma, este capítulo apresenta os fundamentos que sustentam a proposta do Mana Club, iniciando pela caracterização dos Jogos de Cartas Colecionáveis e de *Magic: The Gathering*, discutindo a representação digital de coleções físicas e analisando soluções já existentes para esse público. A partir dessa fundamentação, torna-se possível compreender as decisões adotadas durante a modelagem e implementação da aplicação.

2.1 JOGOS DE CARTAS COLECIONÁVEIS

Os jogos constituem uma manifestação cultural presente em diferentes sociedades e períodos históricos, desempenhando funções que extrapolam o entretenimento. Para Huizinga (2019), o jogo representa uma atividade voluntária, desenvolvida dentro de limites próprios de tempo e espaço, regida por regras específicas e dotada de significado para seus participantes. Essa perspectiva evidencia que a experiência lúdica envolve interação social, construção de significados e organização coletiva.

Complementando essa visão, Salen e Zimmerman (2004) descrevem os jogos como sistemas estruturados por regras, objetivos, recursos e decisões, nos quais os participantes enfrentam conflitos artificiais que conduzem a resultados mensuráveis. Sob essa abordagem, o interesse do jogador não está apenas na execução das regras, mas também nas escolhas realizadas ao longo da experiência.

Os Jogos de Cartas Colecionáveis (*Trading Card Games – TCGs*) ampliam essa dinâmica ao introduzirem um elemento inexistente na maioria dos jogos tradicionais: a propriedade individual dos componentes utilizados durante a partida. Em vez de todos os participantes utilizarem um conjunto previamente definido de cartas, cada jogador constrói sua própria coleção ao longo do tempo, adquirindo novos exemplares, organizando-os e selecionando aqueles que serão utilizados em diferentes estratégias.

Essa característica faz com que a experiência do jogador deixe de se restringir às partidas propriamente ditas. A construção de decks, o gerenciamento da coleção, a aquisição de novas cartas e a negociação com outros participantes tornam-se atividades permanentes e igualmente relevantes para o funcionamento do jogo.

Do ponto de vista computacional, essa dinâmica amplia significativamente a complexidade do domínio. Uma coleção pode reunir centenas ou milhares de cartas, distribuídas entre diferentes edições, idiomas, acabamentos e estados de conservação. Consequentemente, sistemas destinados ao gerenciamento desse tipo de acervo precisam representar não apenas a existência de uma determinada carta, mas também as características específicas de cada exemplar pertencente ao usuário.

Assim, o desenvolvimento de aplicações para TCGs envolve simultaneamente aspectos relacionados à organização da informação, persistência de dados e apoio às atividades realizadas pelos jogadores fora do ambiente das partidas.

2.2 MAGIC: THE GATHERING

Lançado em 1993 pela Wizards of the Coast, *Magic: The Gathering* consolidou o modelo contemporâneo dos Jogos de Cartas Colecionáveis e permanece como uma das principais referências do gênero. Sua longevidade decorre da publicação contínua de novas coleções, da realização de eventos competitivos e da existência de uma comunidade formada por jogadores casuais, colecionadores e competidores profissionais (Wizards of the Coast, 2026).

Ao longo de mais de três décadas de evolução, o jogo acumulou um catálogo composto por milhares de cartas e dezenas de milhares de impressões diferentes. Embora muitas compartilhem o mesmo conjunto de regras, uma carta pode existir em diferentes idiomas, ilustrações, acabamentos, coleções e estados de conservação, atributos que influenciam tanto seu valor comercial quanto seu interesse para diferentes perfis de usuários.

Essa distinção possui impacto direto sobre a modelagem de sistemas voltados ao gerenciamento de coleções. Sob a perspectiva computacional, é necessário diferenciar o conceito abstrato da carta — correspondente ao conjunto de informações catalográficas disponibilizadas oficialmente — do exemplar físico pertencente ao jogador. Enquanto os dados catalográficos permanecem comuns a todos os usuários, características como condição de conservação, idioma, acabamento, localização e disponibilidade para troca pertencem exclusivamente ao exemplar armazenado na coleção individual.

Essa separação reduz redundâncias no armazenamento de dados e permite que informações públicas sejam obtidas por meio de serviços especializados. No Mana Club, essa função é desempenhada pela API Scryfall, responsável por fornecer dados catalográficos estruturados, incluindo identificadores, imagens, textos de regras,

legalidades e demais atributos das cartas (Scryfall, 2026).

A utilização de um serviço externo elimina a necessidade de manter localmente um catálogo constantemente atualizado. Entretanto, essa decisão arquitetural também introduz dependência em relação à disponibilidade e estabilidade da API, exigindo que a aplicação trate adequadamente situações de indisponibilidade temporária ou alterações futuras na interface do serviço.

2.3 REPRESENTAÇÃO E GERENCIAMENTO DE COLEÇÕES DIGITAIS

O gerenciamento digital de coleções exige que conceitos frequentemente tratados como equivalentes pelos jogadores sejam representados como entidades distintas dentro do sistema.

Neste trabalho, a “carta” corresponde ao elemento catalográfico disponibilizado pela API Scryfall, enquanto o “exemplar” representa a unidade física pertencente a um usuário específico. Essa distinção permite que diferentes exemplares da mesma carta possuam atributos independentes, como idioma, acabamento, condição de conservação ou disponibilidade para negociação.

A adoção dessa abordagem influencia diretamente a modelagem dos dados e das regras de negócio. Operações como atualização da coleção, criação de listas de desejos e realização de propostas de troca deixam de manipular apenas cartas e passam a operar sobre exemplares específicos, preservando a individualidade de cada item pertencente aos usuários.

Além da representação dos exemplares, o sistema precisa manter relacionamentos consistentes entre usuários, coleções, listas de desejos e negociações. Em uma proposta de troca, por exemplo, um exemplar não pode pertencer simultaneamente a dois usuários nem participar de negociações incompatíveis. A manutenção dessas restrições justifica a utilização de um modelo relacional, no qual mecanismos de integridade e transações contribuem para preservar a consistência dos dados(PostgreSQL Global Development Group, 2026).

Outro aspecto relevante refere-se à recuperação eficiente das informações. Coleções extensas tornam inviável a consulta manual dos registros, tornando indispensáveis mecanismos de pesquisa, filtragem e ordenação capazes de localizar rapidamente cartas específicas segundo critérios como nome, edição, raridade ou disponibilidade para troca.

Da mesma forma, listas de desejos (*wishlists*) e listas de troca representam

estados distintos do relacionamento entre o usuário e seu acervo. Enquanto a primeira expressa interesse na aquisição de uma carta, a segunda indica a intenção de disponibilizar determinado exemplar para negociação. A proposta de troca surge como mecanismo responsável por relacionar essas informações, registrando participantes, exemplares envolvidos e o estado atual da negociação.

Essa modelagem permite representar de maneira consistente o ciclo de vida das trocas realizadas pelos usuários e constitui um dos principais diferenciais funcionais do Mana Club.

2.4 SOLUÇÕES EXISTENTES

O crescimento da comunidade de *Magic: The Gathering* estimulou o surgimento de diversas ferramentas destinadas ao gerenciamento de coleções, construção de decks e negociação entre jogadores. Embora compartilhem parte dessas funcionalidades, essas aplicações apresentam diferentes objetivos e prioridades de desenvolvimento.

Com o propósito de contextualizar o escopo do Mana Club, foram analisadas quatro soluções amplamente utilizadas pela comunidade: ManaBox, Moxfield, Archidekt e Deckbox. O levantamento foi realizado a partir das funcionalidades publicamente documentadas por seus respectivos desenvolvedores, considerando páginas institucionais, documentação e guias de utilização disponíveis em agosto de 2026. Dessa forma, a comparação restringe-se às informações oficialmente divulgadas, não constituindo uma avaliação experimental ou um estudo comparativo de desempenho.

O Quadro 1 sintetiza as funcionalidades identificadas durante essa análise.

Quadro 1 – Comparativo entre as Ferramentas Analisadas

Ferramenta	Acesso principal	Coleção	Lista de desejos	Decks	Negociação entre usuários
ManaBox	Aplicativo móvel	Sim	Sim, por meio de listas	Sim	Apoio à comparação de trocas; propostas entre contas não identificadas
Moxfield	Aplicação web	Não identificada	Não identificada	Sim	Não identificada
Archidekt	Aplicação web responsiva	Sim	Não identificada como módulo próprio	Sim	Não identificada
Deckbox	Aplicação web	Sim	Sim	Sim	Tradelist, propostas e fluxo de trocas
Mana Club	Aplicação web responsiva	Sim	Sim	Não contemplado	Marketplace e propostas entre usuários

Fonte: Elaborado pelos autores (2026)

A comparação evidencia que não existe uma solução única capaz de atender plenamente todas as necessidades relacionadas ao gerenciamento de coleções físicas.

O ManaBox apresenta-se como aplicativo para dispositivos móveis e divulga digitalização de exemplares físicos, pesquisa de cartas, gerenciamento de coleção, listas, construção e simulação de decks, acompanhamento de preços e ferramenta para comparação de trocas. Seus guias distinguem *binders*, destinados às cartas possuídas, de listas usadas para itens que não necessariamente pertencem ao usuário, incluindo listas de desejos (ManaBox, 2026b; ManaBox, 2026a). Essa organização aproxima coleção física e planejamento de decks, mas a documentação pública consultada não descreve um fluxo equivalente ao do Mana Club para envio, resposta e persistência de propostas entre duas contas.

O Moxfield apresenta-se publicamente como uma ferramenta web de construção de decks. Sua comunicação institucional enfatiza criação, organização, compartilhamento e análise de listas. Na página pública consultada, não foram localizados módulos documentados de propostas de troca entre usuários. Por essa razão, o quadro registra apenas o que pôde ser confirmado publicamente, sem concluir que recursos não documentados sejam necessariamente inexistentes (Moxfield, 2026).

O Archidekt descreve-se como um construtor visual de decks e divulga recursos de criação, organização e teste de listas. A página institucional também apresenta ferramentas para importação, visualização, ordenação e filtragem de coleções pessoais.

Entretanto, não foi identificado, na documentação pública examinada, um módulo próprio de propostas de troca entre contas (Archidekt, 2026).

O Deckbox é a solução mais próxima do fluxo tratado neste trabalho. Sua documentação diferencia inventário, *tradelist* e *wishlist*, oferece construção de decks e descreve trocas locais ou por envio, propostas, histórico e reputação. Dessa forma, a plataforma não apenas registra itens disponíveis, mas também apoia a interação entre participantes durante a negociação (Deckbox, 2026a; Deckbox, 2026b).

O comparativo indica que a simples reunião de coleção e lista de desejos não é exclusiva do Mana Club. Sua delimitação está na implementação acadêmica de um fluxo integrado, no qual exemplares físicos individualizados podem ser publicados para troca e incluídos em propostas persistidas entre usuários. O propósito da comparação não é demonstrar superioridade geral, mas identificar diferentes coberturas funcionais e situar o escopo escolhido para a primeira versão da aplicação.

2.5 ENGENHARIA DE SOFTWARE E DESENVOLVIMENTO INCREMENTAL

A Engenharia de Software reúne processos, métodos e práticas destinados à especificação, ao desenvolvimento, à validação e à evolução de sistemas de software. Para Sommerville (2019), o processo de software compreende atividades que permitem transformar necessidades em um produto que possa ser desenvolvido, avaliado e posteriormente evoluído. Essa perspectiva é particularmente relevante em projetos nos quais o sistema é construído de forma progressiva, pois permite relacionar as decisões de desenvolvimento aos requisitos que deram origem ao produto.

Entre essas atividades, a Engenharia de Requisitos ocupa papel fundamental. Requisitos descrevem os serviços que o sistema deve oferecer e as restrições às quais ele está submetido. Os requisitos funcionais estão relacionados aos comportamentos e operações que devem ser disponibilizados, enquanto os requisitos não funcionais representam propriedades ou restrições associadas, entre outros aspectos, à segurança, ao desempenho, à disponibilidade e à usabilidade (SOMMERVILLE, 2019; PRESSMAN; MAXIM, 2021).

A ISO/IEC/IEEE 29148 (ISO/IEC/IEEE, 2018) destaca características relacionadas à qualidade dos requisitos e à rastreabilidade ao longo do processo de Engenharia de Requisitos. A rastreabilidade permite relacionar uma necessidade aos requisitos correspondentes e, posteriormente, aos elementos de projeto, implementação e verificação que contribuem para atendê-la. No Mana Club, essa relação foi representada pelos identificadores dos requisitos funcionais e não funcionais, permitindo relacionar a

especificação às demais etapas do desenvolvimento.

O desenvolvimento do Mana Club foi apoiado pelo uso do GitHub para controle de versão e acompanhamento das atividades. As demandas foram registradas como *issues* e organizadas no GitHub Projects em um quadro de trabalho, no qual as tarefas eram movimentadas entre diferentes estados conforme sua evolução. Essa organização permitiu visualizar o fluxo das atividades e acompanhar o trabalho em progresso, incorporando práticas características do método Kanban (ANDERSON, 2010; GitHub, 2026).

A utilização dessas práticas não significa que o projeto tenha adotado formalmente todas as prescrições do método Kanban. O quadro foi utilizado principalmente como mecanismo de visualização e acompanhamento do fluxo de trabalho.

2.6 ARQUITETURA DE APLICAÇÕES WEB

A arquitetura de software representa a organização dos principais elementos de um sistema, suas responsabilidades e as relações estabelecidas entre eles. Bass, Clements e Kazman (BASS et al., 2021) destacam que as decisões arquiteturais exercem influência sobre atributos como modificabilidade, desempenho, disponibilidade e segurança. Entretanto, a simples adoção de determinada tecnologia não garante a obtenção desses atributos, uma vez que eles dependem das decisões de projeto e precisam ser avaliados em cenários concretos.

O Mana Club foi estruturado segundo o modelo cliente-servidor, separando a interface responsável pela interação com o usuário (frontend) dos serviços responsáveis pelas regras de negócio e pelo acesso aos dados (backend). Essa separação permite concentrar as regras do domínio no backend e estabelecer uma interface de comunicação entre as diferentes partes da aplicação.

A comunicação entre frontend e backend foi implementada por meio de uma API baseada em HTTP e no formato JSON. Essa abordagem permite que a interface envie solicitações ao servidor e receba representações estruturadas dos dados necessários para atualizar a aplicação.

A API segue princípios do estilo arquitetural REST (Representational State Transfer). Fielding (FIELDING, 2000) define REST a partir de um conjunto de restrições arquiteturais aplicáveis a sistemas distribuídos, incluindo separação entre cliente e servidor, ausência de estado de sessão armazenado pelo servidor, possibilidade de utilização de cache, interface uniforme e organização em camadas.

Essa distinção é relevante porque a utilização de HTTP e JSON, isoladamente, não caracteriza uma API como REST. A aderência ao estilo depende da forma como os recursos são identificados, das representações utilizadas, dos métodos empregados e das demais restrições arquiteturais. No Mana Club, esses princípios orientam a comunicação entre a aplicação cliente e os serviços disponibilizados pelo backend.

O protocolo HTTP estabelece a semântica das mensagens utilizadas na comunicação entre os componentes da aplicação, incluindo métodos, códigos de estado e cabeçalhos (Internet Engineering Task Force, 2022). O formato JSON, por sua vez, fornece uma representação textual estruturada para o intercâmbio de dados (JSON, 2025). Em conjunto, esses mecanismos constituem o contrato de comunicação utilizado entre o frontend e a API do Mana Club.

2.6.1 Frontend com React, Next.js, TypeScript e Tailwind CSS

O frontend é responsável pela interação direta com o usuário, apresentando informações, recebendo entradas e encaminhando ações para os serviços disponibilizados pelo backend. Para a construção da interface do Mana Club foram utilizadas as tecnologias React, Next.js, TypeScript e Tailwind CSS.

React organiza a interface por meio de componentes, permitindo dividir a aplicação em unidades reutilizáveis e atualizar sua apresentação conforme mudanças de estado (React, 2026). Essa característica é adequada a uma aplicação como o Mana Club, na qual diferentes telas compartilham elementos de interface e comportamentos relacionados à apresentação e manipulação de dados.

Next.js fornece uma estrutura de desenvolvimento baseada em React, incorporando mecanismos de roteamento e recursos relacionados à construção e entrega de aplicações web (Vercel, 2026). Sua utilização no projeto permitiu organizar as diferentes páginas e fluxos da interface dentro de uma estrutura única de aplicação.

TypeScript foi empregado como linguagem de programação tanto na interface quanto na API. A linguagem adiciona tipagem estática ao JavaScript, possibilitando que determinadas incompatibilidades entre tipos sejam identificadas durante o desenvolvimento, antes da execução da aplicação (Microsoft, 2026). Esse recurso é particularmente útil em sistemas que manipulam estruturas de dados compartilhadas entre diferentes componentes, embora a tipagem estática não elimine a necessidade de validação de dados recebidos em tempo de execução.

Para a estilização da interface foi utilizado Tailwind CSS, que disponibiliza classes utilitárias para a composição dos elementos visuais. No Mana Club, sua utilização

contribuiu para a padronização dos componentes e para a construção de interfaces adaptáveis a diferentes dimensões de tela. Entretanto, a utilização de uma biblioteca de estilos não constitui, por si só, garantia de responsividade; essa característica depende da implementação e precisa ser verificada nos ambientes considerados relevantes para a aplicação.(Tailwind Labs Inc., 2026).

A escolha conjunta dessas tecnologias permitiu estabelecer uma camada de apresentação componentizada, com tipagem estática e uma estrutura organizada para as diferentes páginas e fluxos do sistema

2.6.2 Backend com Node.js, Express e TypeScript

O backend concentra as responsabilidades que não devem depender exclusivamente da interface apresentada ao usuário. No Mana Club, essa camada é responsável pela autenticação, autorização, aplicação das regras de negócio, integração com serviços externos e coordenação das operações de persistência.

Node.js fornece um ambiente de execução para JavaScript fora do navegador, enquanto Express disponibiliza mecanismos para definição de rotas e composição de middlewares utilizados no tratamento das requisições HTTP (OpenJS Foundation, 2026b; OpenJS Foundation, 2026a).

A utilização dessas tecnologias permitiu estruturar a API responsável pela comunicação com o frontend. Entretanto, a adoção de uma arquitetura modular não garante automaticamente características como manutenibilidade ou baixo acoplamento. Esses atributos dependem também da definição clara das responsabilidades, da organização das interfaces entre módulos e da disciplina empregada durante a evolução do sistema

No Mana Club, a API atua como intermediária entre a interface e os recursos persistentes ou externos. As solicitações recebidas são processadas de acordo com o contexto do usuário e com as regras do domínio antes que operações sejam realizadas no PostgreSQL, no Redis ou na API Scryfall.

Essa centralização é especialmente importante para as operações que envolvem propriedade e negociação de exemplares. Regras como a verificação de quem pode alterar uma proposta de troca não devem depender apenas de controles implementados na interface, uma vez que o backend constitui a camada responsável por aplicar essas restrições sobre os dados efetivamente persistidos.

2.6.3 Persistência com PostgreSQL

O PostgreSQL é um sistema gerenciador de banco de dados objeto-relacional que oferece recursos como transações, índices, restrições e integridade referencial (PostgreSQL Global Development Group, 2026). Essas características são compatíveis com o domínio do Mana Club, no qual usuários, exemplares, listas e negociações possuem relações que precisam permanecer consistentes.

A utilização de um banco de dados relacional é particularmente relevante nas operações de troca. Uma negociação pode envolver exemplares pertencentes a usuários diferentes e, após sua aceitação, alterar a propriedade desses registros. Nesse contexto, a operação precisa ser tratada como uma unidade lógica para evitar estados intermediários em que apenas parte da transferência tenha sido concluída.

As transações fornecem mecanismos para agrupar operações relacionadas e preservar a consistência do banco de dados quando uma operação é concluída ou revertida. No entanto, a existência desse recurso no sistema gerenciador não significa que qualquer aplicação estará automaticamente protegida contra inconsistências: é necessário que a implementação utilize adequadamente as transações e as demais restrições pertinentes ao domínio.

No Mana Club, o PostgreSQL permanece como fonte de persistência permanente das informações da aplicação. Dados relacionados aos usuários, coleções e negociações são mantidos nessa camada, enquanto o Redis é utilizado como mecanismo complementar de armazenamento temporário.

2.6.4 Cache com Redis

Cache consiste na manutenção temporária de dados em uma camada destinada a reduzir o custo de acessos repetidos a uma fonte persistente. Quando utilizado adequadamente, esse mecanismo pode diminuir a quantidade de consultas realizadas ao armazenamento principal. Entretanto, a introdução de uma camada de cache também cria responsabilidades adicionais relacionadas à definição de chaves, expiração, invalidação e consistência dos dados.

O Redis é uma tecnologia de armazenamento em memória que disponibiliza diferentes estruturas de dados e pode ser utilizada para implementação de mecanismos de cache (Redis, 2026). No Mana Club, sua utilização foi direcionada aos dados das cartas que compõem as coleções dos usuários.

No Mana Club, o cache é aplicado aos dados das cartas que compõem a

coleção de cada usuário. O PostgreSQL permanece responsável pela persistência, enquanto o Redis mantém uma representação temporária desses dados para atender consultas recorrentes às coleções. O cache não é utilizado para armazenar respostas provenientes da API Scryfall. Alterações na coleção exigem que a informação temporária permaneça coerente com o estado persistido, por meio da atualização ou invalidação da entrada correspondente. A presença desse mecanismo fundamenta a expectativa de reduzir acessos repetidos ao banco de dados, mas qualquer alegação de ganho de desempenho requer medição em ambiente e cenário definidos.

2.7 VERIFICAÇÃO E AVALIAÇÃO DE SOFTWARE

A verificação e a validação constituem atividades relacionadas à avaliação de um produto de software, mas possuem objetivos distintos. A verificação procura determinar se o produto foi construído de acordo com as especificações estabelecidas, enquanto a validação está relacionada à adequação do produto às necessidades que motivaram sua construção (SOMMERVILLE, 2019; PRESSMAN; MAXIM, 2021).

Essa distinção é importante para a avaliação do Mana Club. A execução de um fluxo de cadastro, por exemplo, pode demonstrar que o sistema apresenta o comportamento especificado para aquele requisito. Entretanto, esse resultado não permite concluir, por si só, que a interface é adequada para todos os usuários ou que o sistema apresenta bom desempenho sob condições de carga elevada.

A verificação funcional pode ser estruturada a partir dos requisitos e de seus respectivos critérios de aceitação. Para cada cenário, podem ser definidos um estado inicial, as ações realizadas pelo usuário, os dados de entrada, o resultado esperado e o resultado observado. A comparação entre esses elementos permite verificar se o comportamento apresentado pela aplicação corresponde ao comportamento especificado.

A execução manual desses cenários é uma estratégia adequada para verificar os principais fluxos funcionais de uma primeira versão do sistema, especialmente quando o objetivo é confirmar a integração entre diferentes módulos. Entretanto, apresenta limitações em relação à repetibilidade, à cobertura e à capacidade de identificar determinados tipos de falha quando comparada à utilização de testes automatizados.

Além disso, diferentes requisitos exigem diferentes estratégias de avaliação. Um requisito relacionado a desempenho demanda medições de tempo e condições de carga; requisitos de compatibilidade exigem a definição dos ambientes avaliados; requisitos de segurança requerem procedimentos específicos; e requisitos de usabilidade podem demandar a participação de usuários representativos.

Por esse motivo, a avaliação do Mana Club deve distinguir aquilo que foi efetivamente verificado daquilo que permaneceu fora do escopo dos procedimentos realizados. Neste trabalho, a verificação concentrou-se nos fluxos funcionais definidos nos requisitos, não constituindo um estudo formal de usabilidade, desempenho, segurança ou disponibilidade.

Essa delimitação evita extrapolar os resultados obtidos. A verificação funcional permite afirmar que os cenários executados apresentaram o comportamento esperado nas condições em que foram avaliados, mas não permite generalizar esses resultados para diferentes perfis de usuários, grandes volumes de acesso ou situações de falha e ataque.

Essa abordagem também estabelece uma relação direta entre a fundamentação teórica apresentada neste capítulo e o procedimento de verificação descrito posteriormente na metodologia. Os requisitos e respectivos critérios de aceitação constituem a referência para os cenários executados, permitindo que os resultados apresentados sejam relacionados às especificações estabelecidas durante o desenvolvimento.

3 METODOLOGIA

Este capítulo apresenta os procedimentos adotados para o desenvolvimento e a avaliação do Mana Club. O trabalho caracteriza-se como uma pesquisa aplicada, uma vez que emprega conhecimentos de Engenharia de Software na construção de uma solução para um problema concreto relacionado ao gerenciamento de coleções de *Magic: The Gathering*. Quanto aos objetivos, possui caráter descritivo e tecnológico, pois envolve a caracterização do domínio, a especificação de uma solução computacional e a documentação de seu processo de desenvolvimento e verificação.

Os procedimentos metodológicos compreenderam pesquisa bibliográfica, análise documental de soluções existentes, especificação de requisitos, modelagem do sistema, desenvolvimento incremental e verificação funcional da aplicação. Essas atividades foram organizadas de forma relacionada, permitindo que os requisitos identificados orientassem a modelagem e a implementação e, posteriormente, servissem como referência para a verificação do produto.

A pesquisa bibliográfica foi utilizada para fundamentar os conceitos relacionados aos Jogos de Cartas Colecionáveis, *Magic: The Gathering,* Engenharia de Software, Engenharia de Requisitos, arquitetura de aplicações web, persistência de dados e mecanismos de cache. A análise documental foi empregada na identificação das funcionalidades disponibilizadas por soluções existentes destinadas ao gerenciamento de coleções e à construção de decks. Foram consideradas informações publicamente disponíveis em páginas institucionais, documentação e guias de utilização dessas ferramentas.

A análise das soluções existentes teve caráter descritivo e serviu para contextualizar o escopo do Mana Club. Não foram realizados testes comparativos de desempenho, usabilidade ou segurança entre as aplicações analisadas. Essa delimitação é importante para que os resultados da comparação não sejam interpretados como uma avaliação experimental das ferramentas.

O desenvolvimento do sistema foi organizado em cinco etapas principais:

1. delimitação do problema e do escopo da aplicação;
2. elaboração do Quadro de Requisitos Funcionais e Requisitos Não-Funcionais;
3. modelagem da arquitetura, dos casos de uso e dos dados;

4. implementação incremental dos módulos;
5. verificação manual dos princípais fluxos funcionais com base nos requisitos e respectivos critérios de aceitação.

A relação entre essas etapas foi mantida por meio da identificação dos requisitos e dos registros produzidos durante o desenvolvimento. Dessa forma, a metodologia buscou estabelecer uma sequência entre a definição do problema, a especificação do comportamento esperado, a implementação e a verificação da solução.

3.1 PROCESSO DE DESENVOLVIMENTO

O desenvolvimento do Mana Club foi conduzido de forma incremental, utilizando práticas do método Kanban para organizar e acompanhar o fluxo das atividades. A escolha dessa abordagem esteve relacionada à necessidade de dividir o desenvolvimento em unidades de trabalho menores e permitir que as funcionalidades fossem implementadas, integradas e verificadas progressivamente.

As atividades foram registradas no GitHub Issues e organizadas por meio do GitHub Projects em um quadro visual. As issues representavam unidades de trabalho relacionadas a funcionalidades, correções, melhorias, tarefas técnicas e atividades de documentação. À medida que o trabalho avançava, as tarefas eram movimentadas entre os estados definidos no quadro, permitindo visualizar as atividades pendentes, em andamento e concluídas.

A utilização do quadro também permitiu acompanhar o trabalho em progresso e identificar as atividades que ainda aguardavam implementação ou verificação. Essa organização foi particularmente útil em um projeto desenvolvido por dois autores, pois possibilitou compartilhar uma visão comum das tarefas e reduzir a sobreposição de atividades. Embora essas práticas sejam associadas ao Kanban, o projeto não buscou implementar uma aplicação formal de todos os princípios e práticas descritos na literatura, concentrando-se na visualização e no acompanhamento do fluxo de trabalho.

Entre as atividades registradas estiveram a implementação do modelo de usuários, a estruturação das entidades do banco de dados, o desenvolvimento dos *endpoints* da API REST, a integração com a API Scryfall, a implementação da autenticação, o desenvolvimento da interface web e a utilização do Redis como mecanismo de cache.

A utilização de unidades de trabalho menores permitiu que as funcionalidades fossem desenvolvidas de forma progressiva. Em termos gerais, cada módulo passou

pelas etapas de especificação, modelagem, implementação e verificação antes de sua integração ao restante da aplicação. Essa organização não impediu alterações posteriores, mas permitiu que mudanças fossem incorporadas de maneira localizada durante o desenvolvimento.

O processo também utilizou controle de versão para manter o histórico das alterações realizadas no código. Essa prática possibilitou identificar a evolução da implementação e recuperar versões anteriores quando necessário, além de apoiar o trabalho colaborativo entre os dois desenvolvedores.

3.2 PROCEDIMENTO DE VERIFICAÇÃO

A verificação da aplicação foi orientada pelos requisitos funcionais e pelos respectivos critérios de aceitação registrados na Especificação de Requisitos de Software. Após a implementação e integração dos módulos, os autores executaram manualmente os principais fluxos da aplicação por meio da interface web, observando tanto o comportamento apresentado ao usuário quanto os efeitos produzidos nos dados persistidos.

O procedimento consistiu em iniciar cada fluxo em uma condição conhecida, executar as ações previstas e comparar o resultado observado com o comportamento esperado definido no requisito correspondente. Quando aplicável, também foram verificadas as informações persistidas e as restrições de acesso relacionadas à operação.

Entre os fluxos verificados estiveram:

1. criação de conta e autenticação;
2. pesquisa e cadastro de exemplares;
3. consulta, atualização e remoção de itens da coleção;
4. inclusão, alteração e remoção de itens da wishlist;
5. publicação e consulta de exemplares disponíveis para troca;
6. criação, aceitação, recusa e cancelamento de propostas;
7. consulta ao histórico de trocas.

Nos fluxos relacionados às negociações, a verificação considerou também as regras de autorização. Foi observado se usuários sem permissão não poderiam

modificar propostas pertencentes a outros participantes e se a aceitação de uma negociação produzia a alteração esperada na propriedade dos exemplares envolvidos.

O procedimento adotado corresponde a uma verificação funcional manual. Não foram realizados testes formais com usuários externos, testes quantitativos de desempenho, testes de carga, auditoria de segurança ou avaliação sistemática de disponibilidade. Dessa forma, os resultados obtidos permitem discutir o funcionamento dos fluxos implementados nas condições em que foram executados, mas não permitem generalizar conclusões sobre usabilidade, desempenho sob carga, segurança ou disponibilidade da aplicação.

A ausência desses procedimentos constitui uma limitação da avaliação realizada e é considerada posteriormente na discussão dos resultados. Essa delimitação também evita atribuir ao sistema características que não foram efetivamente medidas durante o desenvolvimento.

3.3 ARQUITETURA DA SOLUÇÃO

O Mana Club foi desenvolvido seguindo uma arquitetura cliente-servidor, composta por uma aplicação frontend responsável pela interação com o usuário e uma API REST responsável pelas regras de negócio e acesso aos dados.

A arquitetura do Mana Club foi definida a partir de uma separação entre a camada de apresentação, os serviços responsáveis pelas regras de negócio e os mecanismos de persistência e integração externa.

A aplicação segue o modelo cliente-servidor. O frontend é responsável pela interação com o usuário e pela apresentação das informações, enquanto o backend disponibiliza uma API responsável pelo processamento das solicitações, aplicação das regras de negócio e coordenação do acesso aos dados.

A Figura 1 apresenta uma visão geral da arquitetura utilizada no desenvolvimento.

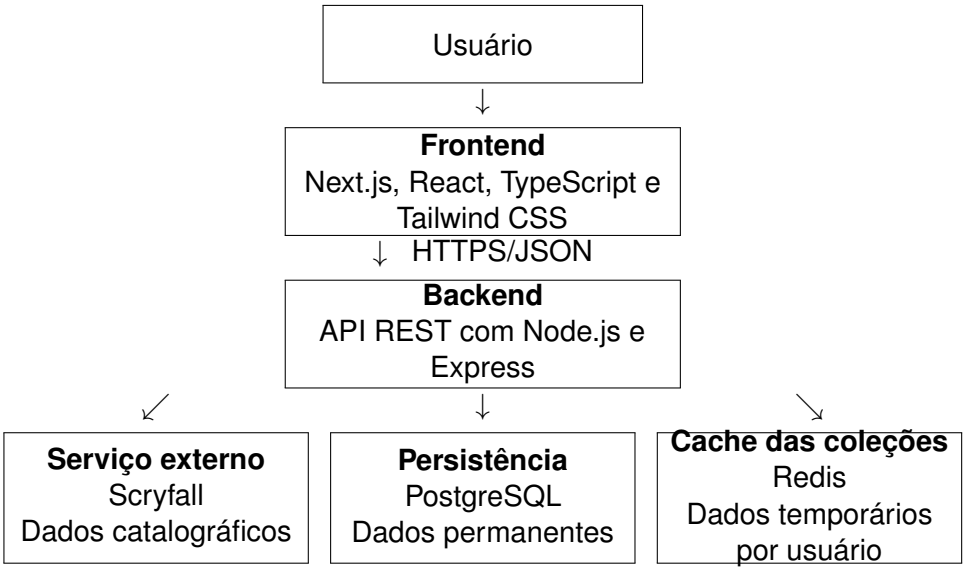

Figura 1 – Arquitetura utilizada no desenvolvimento do Mana Club

Fonte: Elaborado pelos autores (2026).

Na camada de apresentação foram utilizados Next.js, React, TypeScript e Tailwind CSS. Essa camada concentra as páginas e componentes responsáveis pela interação com os usuários e realiza as solicitações necessárias ao backend.

A comunicação entre frontend e backend ocorre por meio de requisições HTTP, utilizando JSON para representação dos dados. A comunicação é realizada sobre HTTPS no ambiente configurado para a aplicação.

O backend foi implementado com Node.js, Express e TypeScript. Essa camada concentra a autenticação, autorização, regras de negócio e comunicação com os demais serviços. Dessa forma, operações que envolvem propriedade de exemplares, gerenciamento de coleções e negociações são processadas no servidor, e não apenas na interface do usuário.

O PostgreSQL é utilizado como mecanismo de persistência permanente. Nele são armazenadas as informações necessárias ao funcionamento da aplicação, incluindo dados dos usuários, coleções, exemplares, wishlist e propostas de troca.

O Redis atua como camada de armazenamento temporário para os dados das cartas pertencentes às coleções dos usuários. Seu uso foi delimitado às consultas relacionadas às coleções, não sendo utilizado para armazenar as respostas provenientes da API Scryfall. Quando ocorre uma alteração nos dados da coleção, a informação correspondente no cache deve ser atualizada ou invalidada para preservar a coerência com os dados persistidos.

A aplicação também utiliza a API Scryfall como fonte externa de dados catalográficos das cartas. Essa integração permite consultar informações que não precisam ser mantidas integralmente na base de dados da aplicação, reduzindo a necessidade de replicação local do catálogo.

A arquitetura resultante pode ser resumida da seguinte forma:

Usuário → Frontend → API REST → PostgreSQL / Redis / Scryfall

Essa organização estabelece responsabilidades distintas entre os componentes e permite que a aplicação seja desenvolvida e modificada de maneira mais localizada. Entretanto, a arquitetura adotada não foi submetida a uma avaliação quantitativa de atributos como desempenho ou escalabilidade. Assim, essas características são apresentadas como aspectos arquiteturais considerados no projeto, e não como propriedades experimentalmente comprovadas.

3.4 TECNOLOGIAS UTILIZADAS

A Tabela 1 apresenta as principais tecnologias empregadas durante o desenvolvimento do sistema.

Tabela 1 – Tecnologias utilizadas no desenvolvimento do Mana Club

Tecnologia	Finalidade
Next.js	Interface web
React	Componentes da interface
Tailwind CSS	Estilização da interface responsiva
TypeScript	Linguagem de programação
Node.js	Ambiente de execução do backend
Express	Implementação da API REST
PostgreSQL	Persistência dos dados
Redis	Cache dos dados das coleções dos usuários
Scryfall API	Dados catalográficos das cartas
GitHub	Versionamento e gerenciamento das atividades

Fonte: Elaborado pelos autores (2026).

A tabela apresenta a função de cada tecnologia no contexto específico do Mana Club. As características individuais dessas ferramentas e os fundamentos que justificam sua utilização foram discutidos no Capítulo 2; nesta seção, a apresentação concentra-se na forma como elas foram empregadas na implementação.

3.5 MODELAGEM DO SISTEMA

Antes da implementação, os requisitos foram definidos e organizados em módulos para apoiar o desenvolvimento. A partir desses requisitos foram identificados os principais casos de uso, possibilitando a construção dos diagramas de casos de uso que representam as interações entre os usuários e o sistema.

Os requisitos funcionais foram organizados em quatro módulos, conforme apresentado no Quadro 2.

Quadro 2 – Resumo dos requisitos funcionais por módulo

Módulo	**Quantidade**	**Requisitos**
Gestão de Usuários	5	RF-01 a RF-05
Gerenciamento da Coleção	7	RF-06 a RF-12
Wishlist	4	RF-13 a RF-16
Gerenciamento de Trocas	7	RF-17 a RF-23
Total	**23**	**RF-01 a RF-23**

Fonte: Elaborado pelos autores (2026).

Em seguida foi elaborado o Diagrama Entidade-Relacionamento (DER), responsável por representar a estrutura lógica do banco de dados da aplicação e as relações entre usuários, coleções, exemplares, wishlist e propostas de troca.

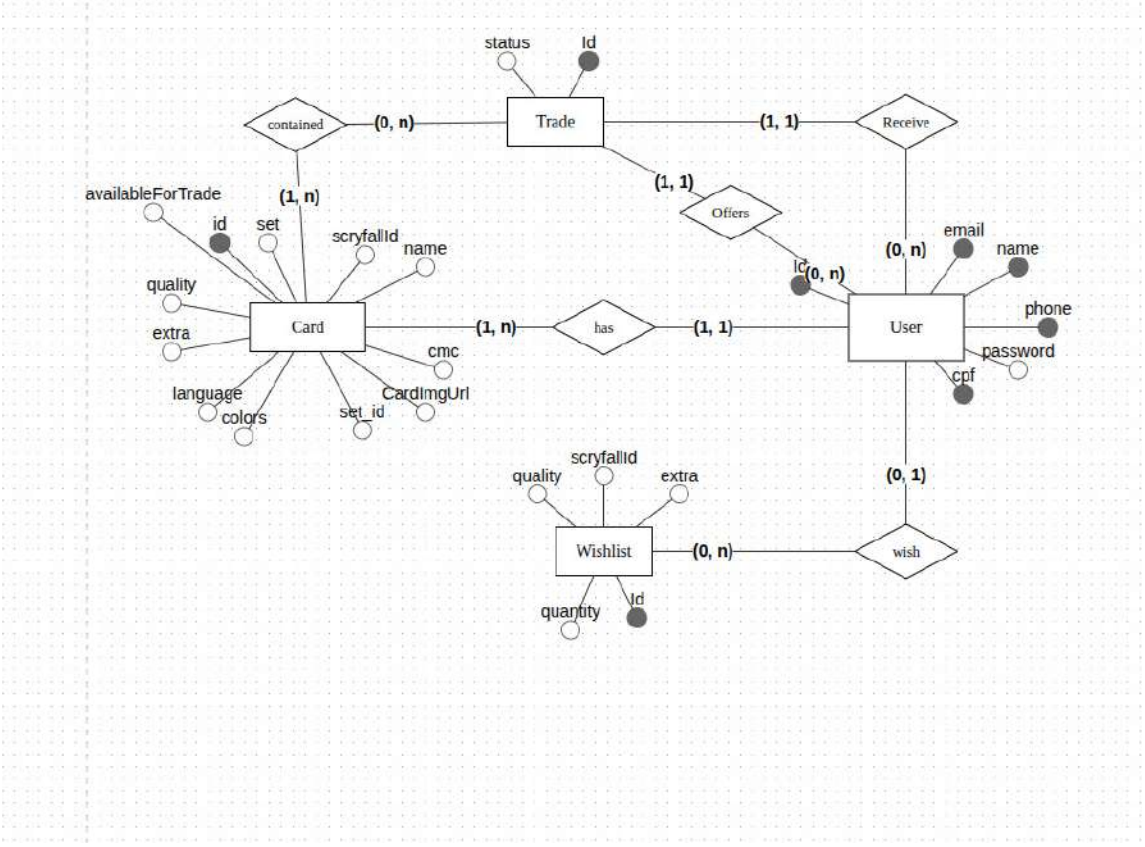

Figura 2 – Diagrama Entidade-Relacionamento do Mana Club

Fonte: Elaborado pelos autores (2026).

Durante a modelagem foram identificados dois atores principais: Visitante e Usuário. O Visitante representa a pessoa que ainda não possui sessão autenticada e pode realizar cadastro, login e recuperação de senha. O Usuário representa a pessoa autenticada, com acesso ao gerenciamento da coleção, wishlist, marketplace, propostas de troca, histórico de negociações e atualização de perfil.

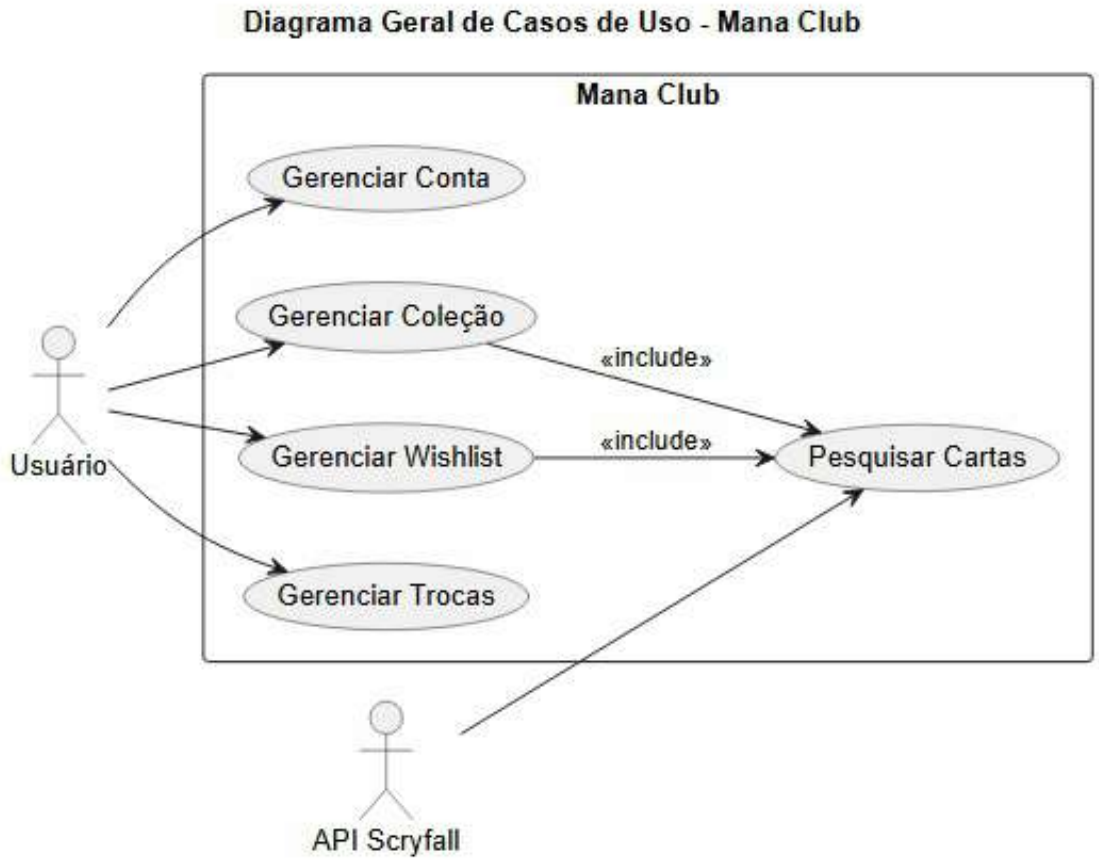

Figura 3 – Diagrama geral de casos de uso

Fonte: Elaborado pelos autores (2026).

Além da distribuição por módulo, os requisitos funcionais foram consolidados em uma visão analítica, contendo a funcionalidade esperada, uma descrição resumida e sua prioridade no desenvolvimento. Essa organização permitiu orientar a implementação incremental e serviu como referência para a avaliação apresentada no Capítulo 4.

Quadro 3 – Requisitos funcionais do Mana Club

Código	Funcionalidade	Descrição	Prioridade
RF-01	Cadastro de Usuário	Permitir que visitantes criem uma conta informando nome de exibição, e-mail e senha.	Essencial
RF-02	Autenticação de Usuário	Permitir acesso à plataforma mediante validação de e-mail e senha.	Essencial
RF-03	Recuperação de Senha	Permitir solicitação de redefinição de senha por meio do e-mail cadastrado.	Importante
RF-04	Atualização de Perfil	Permitir consulta e atualização dos dados cadastrais do usuário.	Importante
RF-05	Encerramento de Sessão	Permitir que o usuário encerre sua sessão de forma segura.	Essencial
RF-06	Pesquisar e Visualizar Cartas	Permitir pesquisa de cartas na API Scryfall e visualização de seus dados catalográficos.	Essencial
RF-07	Registrar Exemplar	Permitir cadastro individual de exemplares na coleção do usuário.	Essencial
RF-08	Atualizar Exemplar	Permitir alteração de dados específicos de exemplares pertencentes ao usuário.	Essencial
RF-09	Remover Exemplar	Permitir remoção de exemplares da coleção do usuário.	Essencial
RF-10	Consultar Coleção	Permitir visualização dos exemplares cadastrados na coleção do usuário.	Essencial
RF-11	Pesquisar e Filtrar Coleção	Permitir pesquisa e aplicação de filtros sobre os exemplares cadastrados.	Importante
RF-12	Visualizar Detalhes do Exemplar	Permitir consulta detalhada das informações de um exemplar específico.	Importante
RF-13	Adicionar Carta à Wishlist	Permitir inclusão de cartas desejadas na wishlist do usuário.	Essencial
RF-14	Atualizar Item da Wishlist	Permitir alteração da quantidade desejada de uma carta cadastrada na wishlist.	Importante
RF-15	Remover Carta da Wishlist	Permitir remoção de cartas cadastradas na wishlist do usuário.	Essencial
RF-16	Consultar, Pesquisar e Filtrar Wishlist	Permitir visualização, pesquisa e filtragem das cartas desejadas.	Essencial
RF-17	Visualizar Marketplace	Permitir acesso à área de usuários e exemplares disponíveis para negociação.	Essencial
RF-18	Visualizar Exemplares Disponíveis para Troca	Permitir consulta aos exemplares marcados como disponíveis por outros usuários.	Essencial
RF-19	Criar Proposta de Troca	Permitir criação de propostas envolvendo exemplares oferecidos e solicitados.	Essencial
RF-20	Consultar Propostas de Troca	Permitir visualização de propostas enviadas e recebidas pelo usuário.	Essencial
RF-21	Responder Proposta de Troca	Permitir que o destinatário aceite ou recuse propostas pendentes.	Essencial
RF-22	Cancelar Proposta de Troca	Permitir que o autor cancele uma proposta enquanto estiver pendente.	Importante
RF-23	Consultar Histórico de Trocas	Permitir consulta às negociações concluídas anteriormente.	Importante

Fonte: Elaborado pelos autores (2026).

Também foram definidos requisitos não funcionais relacionados à integração, disponibilidade, segurança, confiabilidade, usabilidade, compatibilidade, persistênciae manutenibilidade. Esses requisitos orientaram decisões técnicas como a adoção de arquitetura cliente-servidor, uso de autenticação para funcionalidades restritas e armazenamento relacional dos dados.

Quadro 4 – Requisitos não funcionais do Mana Club

Código	Categoria	Requisito	Descrição	Prioridade
RNF-01	Segurança	Autenticação por tokens	O sistema deve utilizar autenticação baseada em tokens JWT para controle de acesso às funcionalidades protegidas.	Essencial
RNF-02	Segurança	Controle de autorização	O sistema deve garantir que usuários somente possam acessar e modificar dados aos quais possuem permissão.	Essencial
RNF-03	Segurança	Proteção de senhas	As senhas dos usuários devem ser armazenadas de forma segura, utilizando mecanismo de proteção adequado para impedir seu armazenamento em texto puro.	Essencial
RNF-04	Manutenibilidade	Arquitetura modular	O backend deve ser estruturado de forma modular, permitindo a manutenção e a expansão futura da aplicação.	Importante
RNF-05	Manutenibilidade	Tipagem estática	O código-fonte deve ser desenvolvido utilizando linguagem de programação com recursos de tipagem estática para auxiliar na identificação de inconsistências durante o desenvolvimento.	Importante
RNF-06	Usabilidade	Feedback visual	O sistema deve apresentar feedback visual ao usuário após ações relevantes, permitindo identificar o resultado das operações realizadas.	Importante
RNF-07	Integração	API Scryfall	O sistema deve consumir dados catalográficos das cartas por meio da API pública da Scryfall.	Essencial
RNF-08	Integridade	Consistência dos dados	O sistema deve garantir a integridade dos dados armazenados no banco de dados, preservando as restrições e relacionamentos definidos para as entidades da aplicação.	Essencial

Fonte: Elaborado pelos autores (2026).

Esses artefatos serviram como base para o desenvolvimento do Mana Club e contribuíram para reduzir inconsistências entre os requisitos levantados e a implementação realizada.

4 RESULTADOS

Este capítulo apresenta os resultados obtidos com o desenvolvimento do Mana Club. Inicialmente é apresentada uma visão geral da solução desenvolvida, seguida pela descrição da arquitetura implementada e das principais funcionalidades disponibilizadas aos usuários. Em seguida são discutidas a integração com serviços externos, a utilização de mecanismos de cache e a validação da aplicação em relação aos requisitos estabelecidos durante a especificação do sistema.

4.1 VISÃO GERAL DA SOLUÇÃO

Como resultado deste trabalho foi desenvolvida uma aplicação web denominada *Mana Club*, destinada ao gerenciamento de coleções de cartas de *Magic: The Gathering*. A aplicação integra funcionalidades de cadastro de usuários, gerenciamento da coleção pessoal, listas de desejos, consulta às coleções públicas de outros jogadores e gerenciamento de propostas de troca.

Diferentemente de soluções que concentram esforços apenas na construção de decks ou no gerenciamento individual da coleção, o Mana Club foi concebido para centralizar, em uma única plataforma, as principais atividades relacionadas ao gerenciamento de cartas e à interação entre jogadores.

Observa-se que toda a navegação ocorre por meio de uma interface web. As principais telas desenvolvidas são apresentadas nas Figuras 5 a 17 e demonstram os fluxos implementados para cadastro, autenticação, consulta de cartas, gerenciamento da coleção, wishlist, marketplace e propostas de troca.

4.2 IMPLEMENTAÇÃO DA APLICAÇÃO

O Mana Club foi implementado segundo a arquitetura cliente-servidor apresentada no Capítulo 3. A solução é composta por uma interface web, responsável pela interação com o usuário, e uma API REST, responsável pelo processamento das solicitações e aplicação das regras de negócio.

A separação entre frontend e backend permitiu concentrar as regras relacionadas aos dados e às operações da aplicação no servidor, enquanto a interface ficou responsável pela apresentação das informações e pela interação com o usuário.

4.2.1 Backend

O backend foi desenvolvido utilizando Node.js, Express e TypeScript, concentrando as regras de negócio e a comunicação com o banco de dados, o cache e a API externa Scryfall.

A API REST disponibiliza endpoints responsáveis pelo gerenciamento de usuários, autenticação, coleção, wishlist, marketplace e propostas de troca. A organização em módulos permitiu separar responsabilidades e facilitar a manutenção da aplicação, conforme resumido no Quadro 5.

Quadro 5 – Principais módulos da API REST

Módulo	Responsabilidade
Usuários e autenticação	Cadastro, login, sessão e perfil
Coleção	Cadastro, consulta, atualização e remoção de exemplares
Wishlist	Controle das cartas desejadas pelo usuário
Marketplace	Consulta a usuários e exemplares disponíveis para troca
Trocas	Criação, consulta, resposta, cancelamento e histórico de propostas
Integração externa	Consulta aos dados catalográficos disponibilizados pela Scryfall
Cache de coleções	Armazenamento temporário dos dados das cartas das coleções dos usuários

Fonte: Elaborado pelos autores (2026).

As informações persistentes são armazenadas em banco PostgreSQL. O Redis mantém temporariamente dados das cartas pertencentes às coleções dos usuários, permitindo que consultas recorrentes a essas coleções utilizem o cache quando houver uma entrada válida. Separadamente, os dados catalográficos das cartas são consultados em uma fonte externa especializada, a API Scryfall; as respostas desse serviço não constituem o conteúdo armazenado no cache Redis.

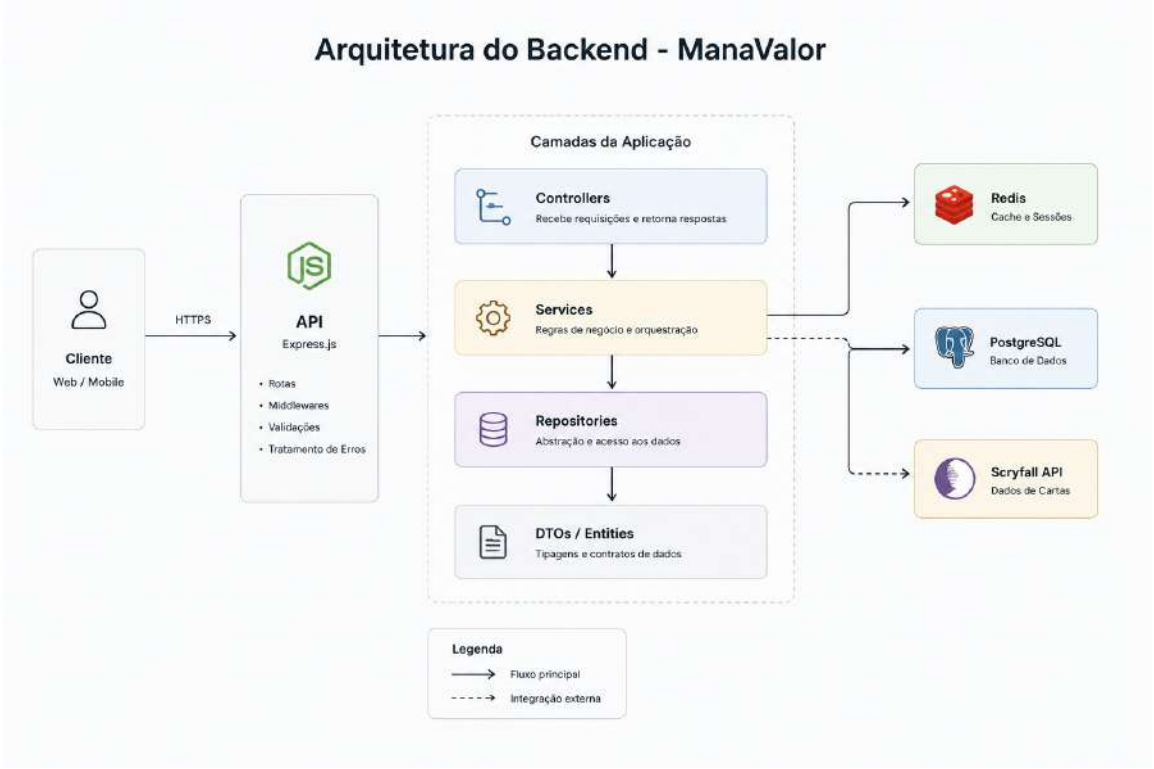

Figura 4 – Arquitetura lógica do backend do Mana Club

Fonte: Elaborado pelos autores (2026).

Conforme a Figura 4, o frontend comunica-se com a API REST por meio de requisições HTTPS e representações em JSON. A camada de controle recebe as solicitações, aplica autenticação, autorização e validação e encaminha cada operação ao módulo de negócio correspondente. Os módulos utilizam o PostgreSQL para persistência, o Redis para o cache temporário dos dados das cartas das coleções por usuário e a API Scryfall exclusivamente para consultar dados catalográficos. A figura representa a organização lógica das responsabilidades e não a estrutura física de diretórios do código-fonte.

4.2.2 Frontend

A interface foi desenvolvida utilizando React, Next.js, TypeScript e Tailwind CSS. O frontend consome os serviços disponibilizados pela API REST e apresenta ao usuário as funcionalidades de cadastro, autenticação, coleção, wishlist, marketplace e trocas.

A utilização de componentes reutilizáveis possibilitou manter padronização visual entre as páginas e facilitar futuras evoluções da aplicação. Entre os elementos reutilizados estão formulários, listas, botões de ação, campos de pesquisa, filtros e componentes de exibição das cartas.

Além disso, a utilização do Next.js proporcionou melhor organização do projeto e otimização do carregamento das páginas.

O Quadro 6 apresenta as principais telas da aplicação.

Quadro 6 – Principais telas da aplicação

Tela	Finalidade
Login	Permitir autenticação de usuários cadastrados
Cadastro	Permitir criação de novas contas de usuário
Home	Apresentar a tela inicial após autenticação
Meu perfil	Permitir a consulta e a atualização dos dados do usuário
Sets	Apresentar as coleções de cartas disponíveis no sistema
Pesquisa de carta	Permitir busca de cartas por meio da API Scryfall
Carta pesquisada	Apresentar os dados da carta retornada pela pesquisa
Coleção	Gerenciar exemplares pertencentes ao usuário
Carta da coleção detalhada	Exibir os dados completos de um exemplar cadastrado
Wishlist	Gerenciar as cartas desejadas pelo usuário
Marketplace	Consultar usuários disponíveis para negociação
Cartas disponíveis do usuário	Exibir exemplares disponíveis para troca do usuário selecionado
Troca	Permitir criação e acompanhamento de uma proposta de troca

Fonte: Elaborado pelos autores (2026).

4.3 FUNCIONALIDADES DESENVOLVIDAS

As funcionalidades implementadas foram diretamente derivadas dos requisitos funcionais definidos durante a fase de especificação do sistema.

4.3.1 Gerenciamento de Usuários e Autenticação

O módulo de gerenciamento de usuários reúne as funcionalidades relacionadas à criação e manutenção das contas, autenticação e controle de acesso à aplicação. Esse módulo constitui a porta de entrada para as funcionalidades que dependem de uma identidade autenticada, uma vez que as operações relacionadas à coleção, wishlist e negociações são vinculadas ao usuário responsável pelos dados.

O processo de criação de uma conta é realizado por meio da tela de cadastro, na qual o visitante informa os dados necessários para o registro. Após o cadastro, o usuário pode realizar a autenticação utilizando seu e-mail e senha, passando a ter acesso às funcionalidades protegidas da aplicação.

A Figura 5 apresenta a tela de login utilizada para autenticação dos usuários.

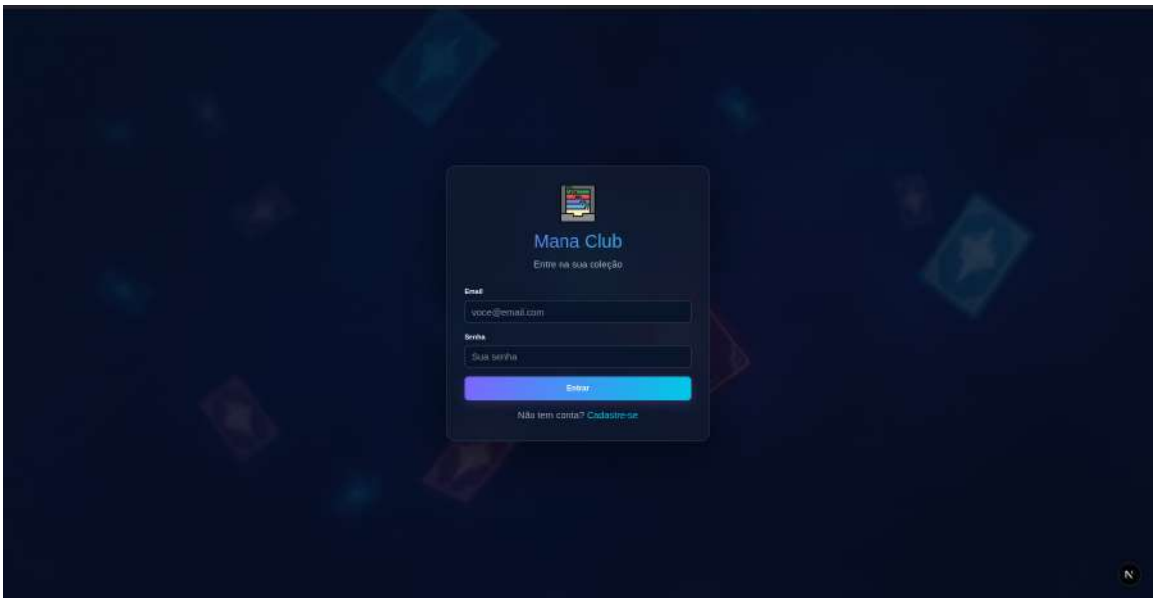

Figura 5 – Tela de login

Fonte: Elaborado pelos autores (2026).

A autenticação utiliza tokens JWT (JSON Web Token) para representar a sessão autenticada do usuário. Após a validação das credenciais, o sistema disponibiliza o token para que as requisições subsequentes às funcionalidades protegidas possam ser associadas ao usuário autenticado.

Além da autenticação, o backend realiza o controle de autorização das operações. Dessa forma, a identificação do usuário não é utilizada apenas para permitir ou negar o acesso à aplicação, mas também para verificar se ele possui permissão para realizar determinada operação sobre os dados solicitados.

Essa regra é particularmente importante no gerenciamento da coleção. Um usuário pode consultar e modificar os exemplares pertencentes à sua própria coleção, mas não deve utilizar os mesmos mecanismos para alterar os dados pertencentes a outro usuário. O mesmo princípio é aplicado às propostas de troca, nas quais determinadas operações dependem do papel do usuário na negociação.

O módulo também contempla a atualização dos dados cadastrais e o encerramento da sessão. A recuperação de senha permite que o usuário solicite a redefinição de sua credencial por meio do endereço de e-mail cadastrado.

A Figura 6 apresenta a tela de cadastro de usuário.

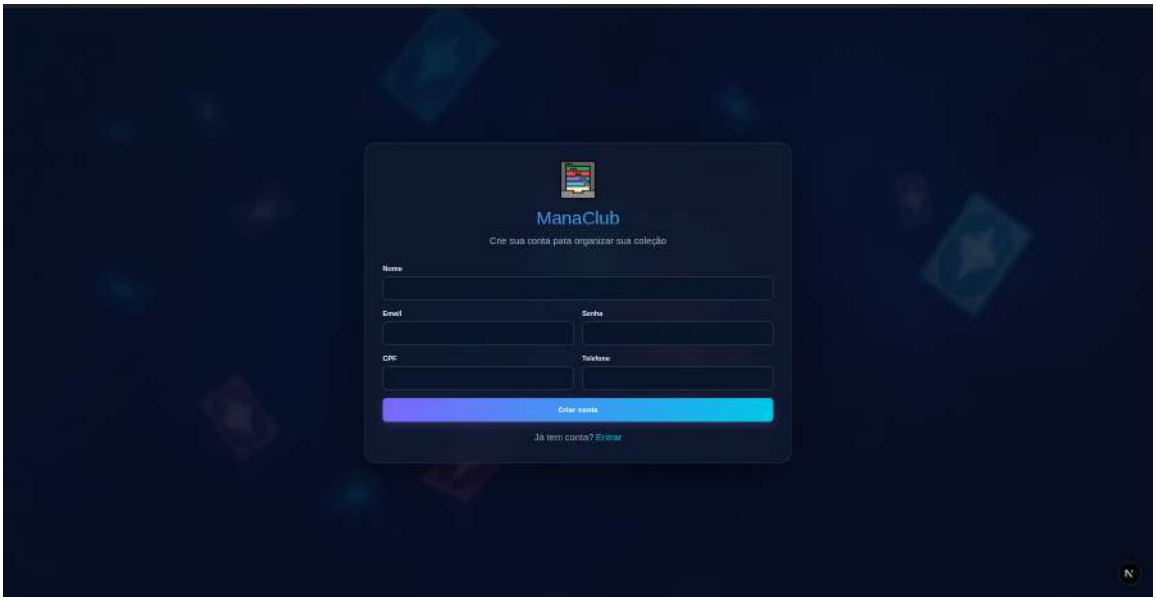

Figura 6 – Tela de cadastro de usuário

Fonte: Elaborado pelos autores (2026).

Após a autenticação, o usuário é direcionado à área principal da aplicação, apresentada na Figura 7. A partir desse ambiente, são disponibilizadas as funcionalidades relacionadas à coleção, wishlist, marketplace e negociações.

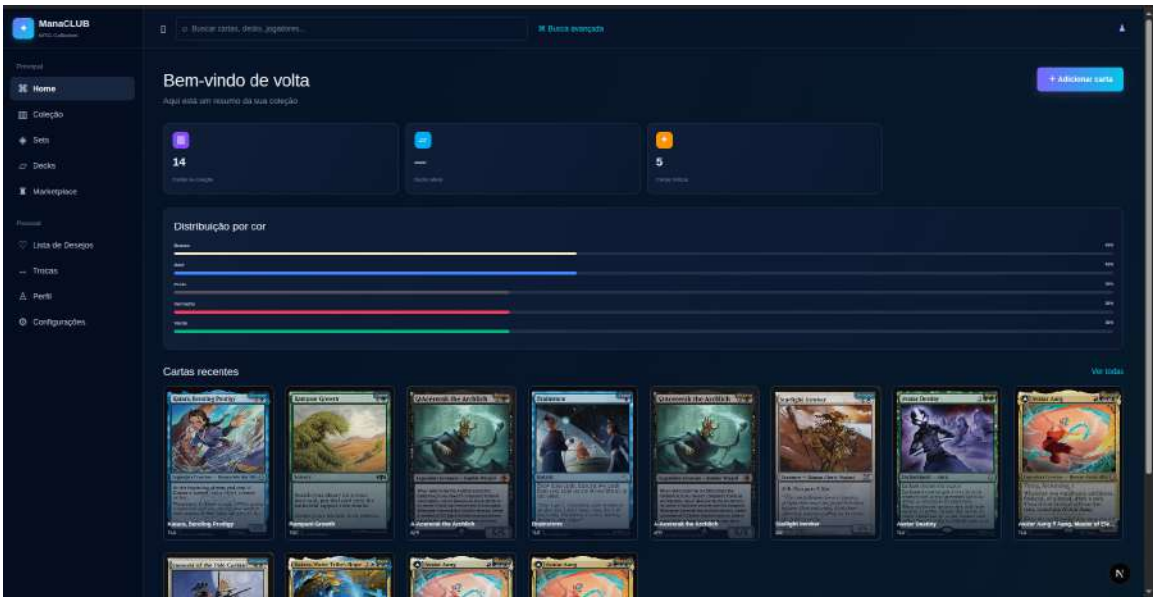

Figura 7 – Tela inicial da aplicação

Fonte: Elaborado pelos autores (2026).

O módulo também disponibiliza a tela de perfil, apresentada na Figura 8, por meio da qual o usuário pode consultar e atualizar seus dados cadastrais.

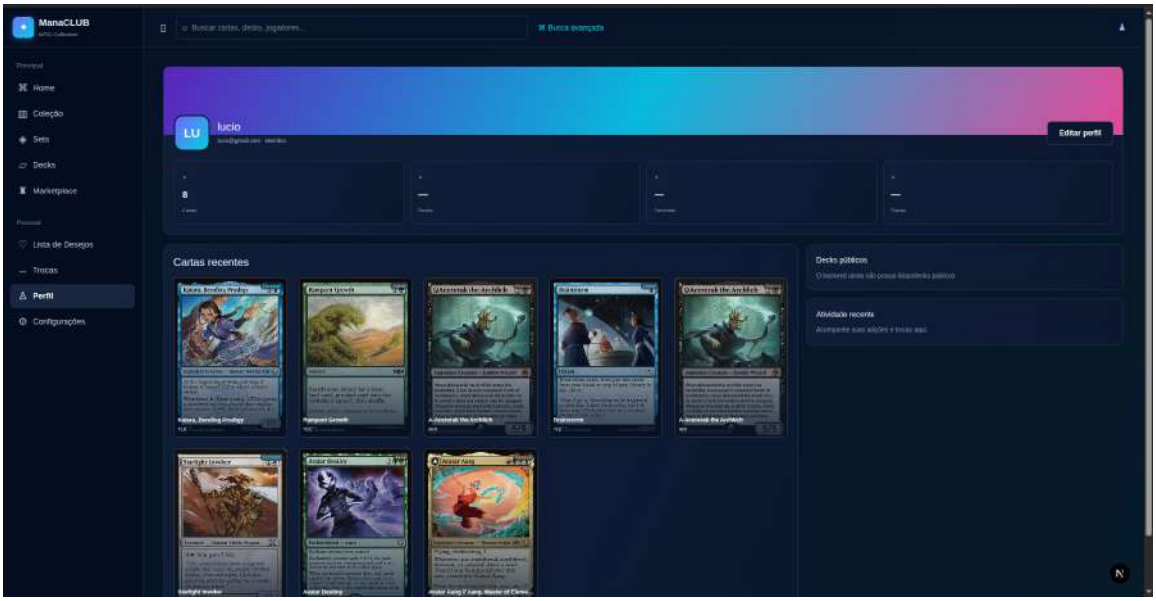

Figura 8 – Tela Meu Perfil

Fonte: Elaborado pelos autores (2026).

Dessa forma, as funcionalidades de cadastro, autenticação, recuperação de senha, atualização de perfil e encerramento da sessão atendem aos requisitos RF-01 a RF-05 definidos na Especificação de Requisitos de Software. O controle de autenticação e autorização também estabelece a base para a aplicação das regras de propriedade e acesso utilizadas nos demais módulos do sistema.

4.3.2 Gerenciamento da Coleção

O módulo de gerenciamento da coleção permite ao usuário cadastrar, consultar, atualizar e remover exemplares pertencentes à sua coleção. A inclusão de um exemplar é realizada a partir da identificação de uma carta obtida por meio da API Scryfall, à qual são associados os dados específicos do exemplar físico.

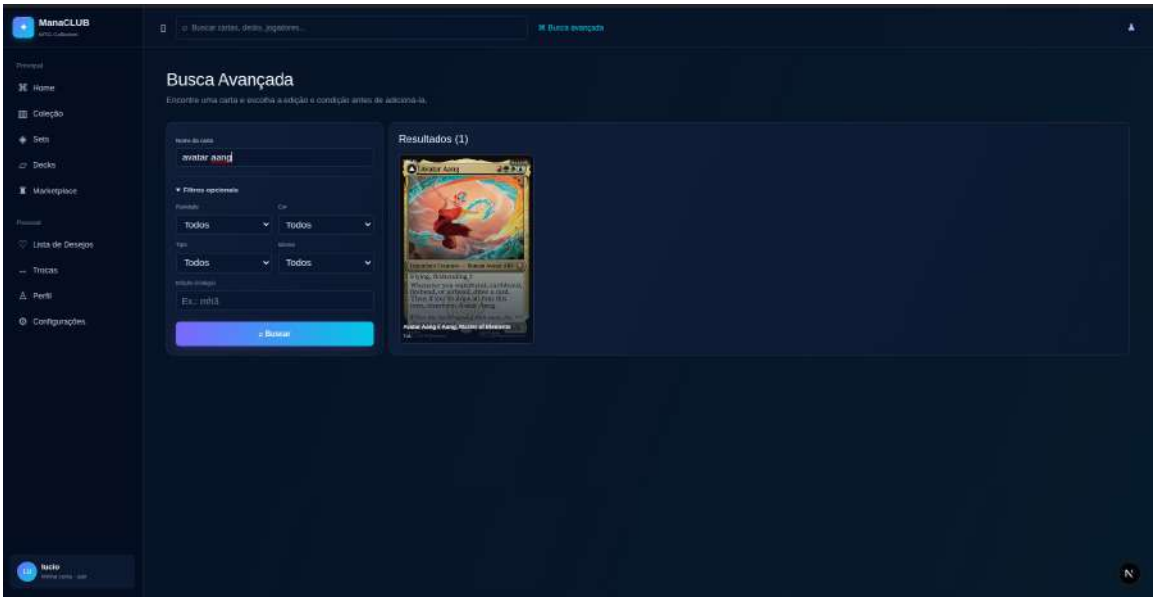

Figura 9 – Tela de pesquisa de carta

Fonte: Elaborado pelos autores (2026).

Entre as informações mantidas pelo Mana Club estão características como idioma, condição de conservação, acabamento e disponibilidade para troca. Essa separação permite que diferentes exemplares de uma mesma carta sejam registrados individualmente.

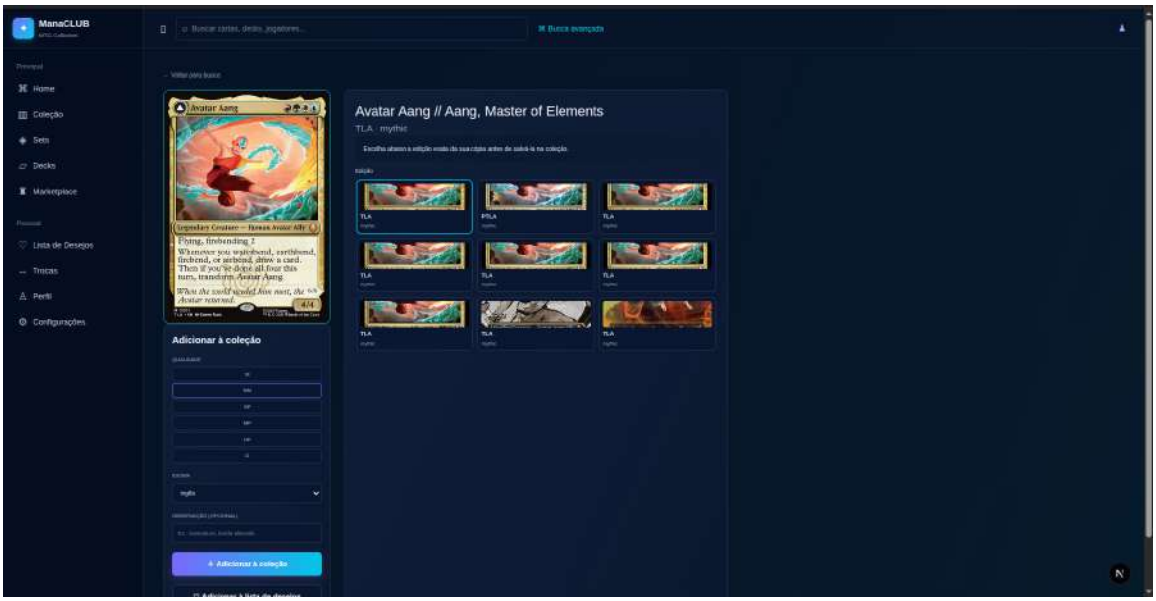

Figura 10 – Tela da carta pesquisada

Fonte: Elaborado pelos autores (2026).

A possibilidade de manter exemplares independentes é relevante para o do-

mínio da aplicação, pois duas cópias da mesma carta podem possuir características distintas. Um exemplar pode, por exemplo, estar disponível para negociação enquanto outro permanece exclusivamente na coleção do usuário.

A tela de gerenciamento da coleção permite consultar os exemplares cadastrados e realizar operações de inclusão, alteração e remoção.

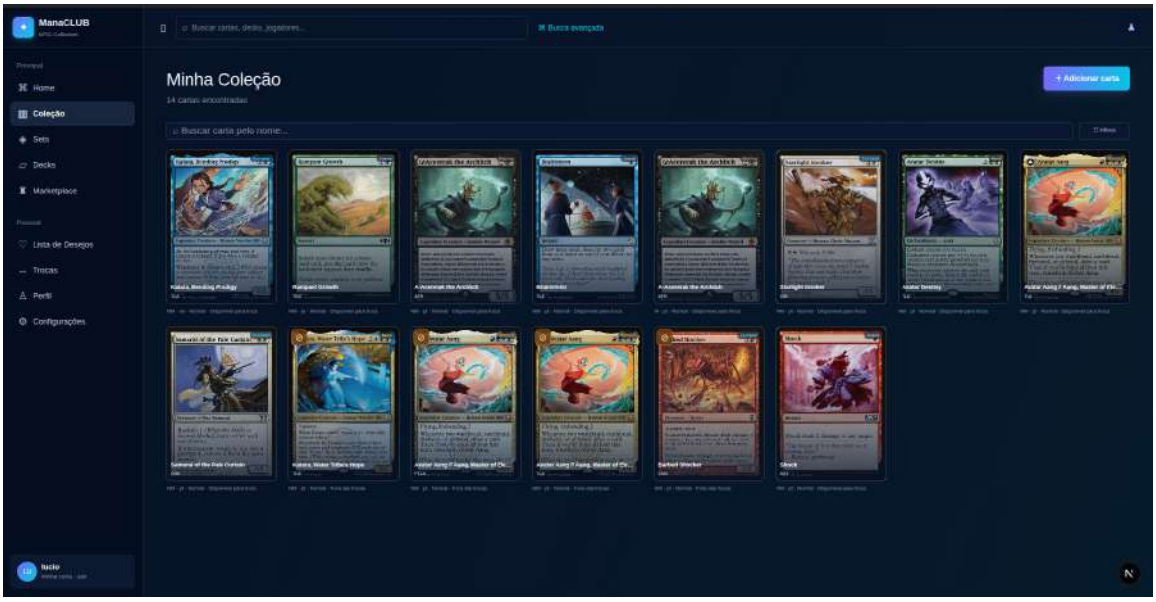

Figura 11 – Tela de gerenciamento da coleção

Fonte: Elaborado pelos autores (2026).

A partir da seleção de um exemplar, o sistema permite consultar suas informações detalhadas e realizar as alterações permitidas ao proprietário.

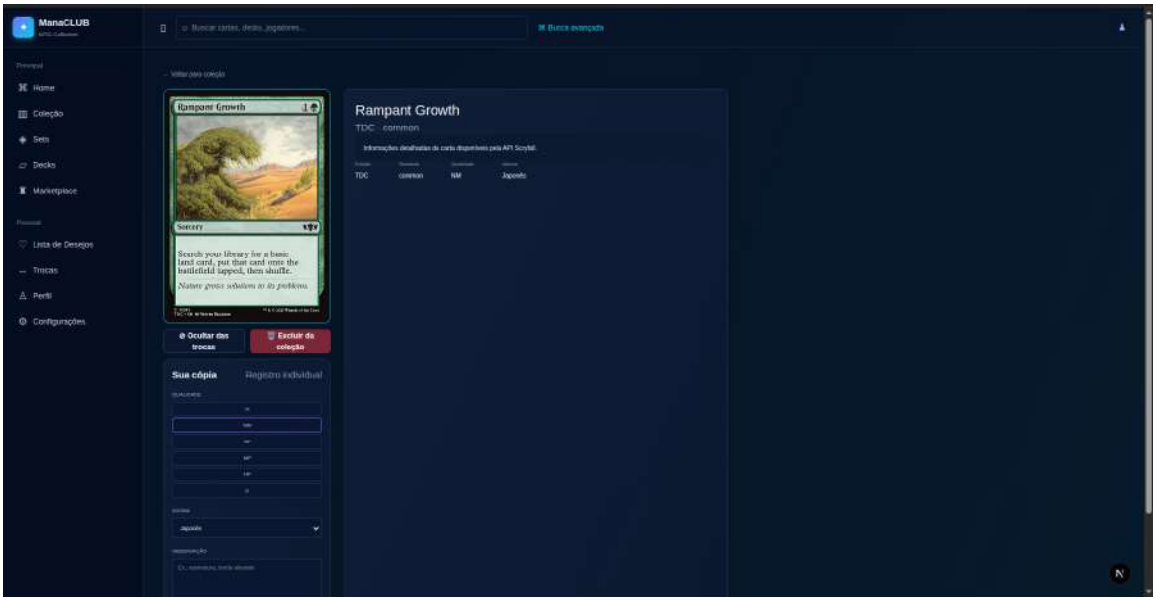

Figura 12 – Tela detalhada da carta na coleção

Fonte: Elaborado pelos autores (2026).

O módulo também disponibiliza mecanismos de pesquisa e filtragem, permitindo localizar exemplares dentro da coleção sem a necessidade de percorrer manualmente todos os registros.

As funcionalidades descritas nesta seção correspondem aos requisitos RF-06 a RF-12, relacionados à pesquisa de cartas, cadastro e gerenciamento de exemplares e consulta da coleção.

4.3.3 Wishlist

A wishlist permite ao usuário registrar cartas que pretende adquirir futuramente. Para cada carta desejada pode ser definida a quantidade pretendida.

Assim como ocorre no gerenciamento da coleção, os dados catalográficos das cartas são obtidos por meio da API Scryfall. O Mana Club mantém a relação entre o usuário e a carta desejada, sem necessidade de replicar integralmente os dados catalográficos fornecidos pelo serviço externo.

A tela da wishlist apresenta as cartas cadastradas e disponibiliza as operações de consulta, pesquisa, filtragem, alteração da quantidade desejada e remoção de itens.

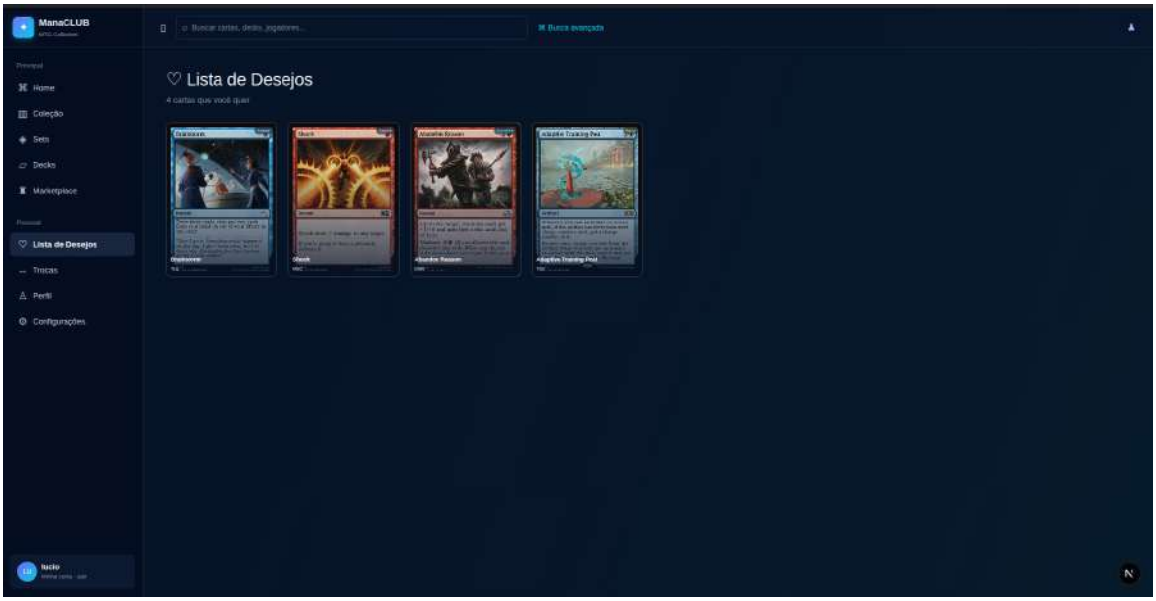

Figura 13 – Tela da wishlist

Fonte: Elaborado pelos autores (2026).

A funcionalidade permite, portanto, manter centralizadas as informações sobre cartas que o usuário pretende adquirir, podendo ser utilizada como referência para futuras aquisições ou negociações.

As funcionalidades descritas nesta seção correspondem aos requisitos RF-13 a RF-16, relacionados à inclusão, atualização, remoção, consulta, pesquisa e filtragem da wishlist.

4.3.4 Marketplace

O marketplace permite consultar usuários e exemplares disponibilizados para negociação. Para preservar as informações privadas das coleções, somente os exemplares marcados pelo proprietário como disponíveis para troca são apresentados aos demais usuários.

A tela do marketplace apresenta os usuários e as opções disponíveis para consulta.

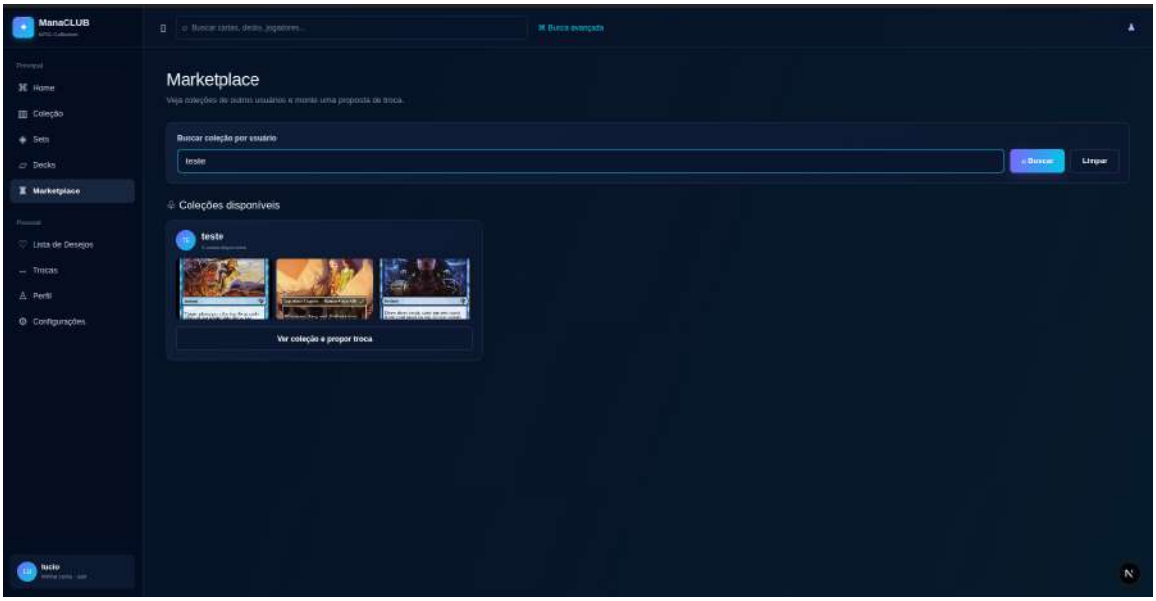

Figura 14 – Tela do marketplace

Fonte: Elaborado pelos autores (2026).

A partir da seleção de um usuário, é possível consultar os exemplares que foram disponibilizados para troca por aquele participante.

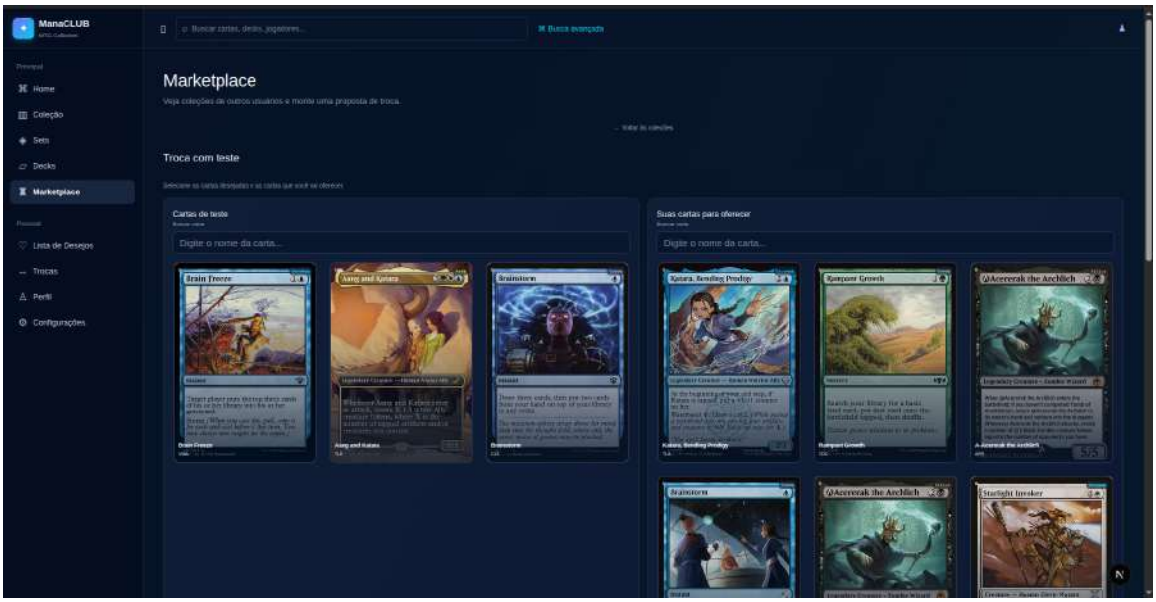

Figura 15 – Tela de cartas disponíveis do usuário selecionado

Fonte: Elaborado pelos autores (2026).

Essa organização estabelece a ligação entre o gerenciamento individual da coleção e o processo de negociação entre usuários. O proprietário continua controlando

sua coleção, enquanto apenas os exemplares explicitamente disponibilizados ficam visíveis no contexto do marketplace.

A consulta dos exemplares disponíveis permite que o usuário identifique itens de seu interesse e, posteriormente, utilize essas informações para elaborar uma proposta de troca.

As funcionalidades descritas nesta seção correspondem aos requisitos RF-17 e RF-18, relacionados à visualização do marketplace e dos exemplares disponibilizados para negociação.

4.3.5 Sistema de Trocas

O sistema de trocas permite que dois usuários estabeleçam uma proposta envolvendo exemplares de suas respectivas coleções.

Uma proposta registra o usuário que a criou, o destinatário, os exemplares oferecidos e os exemplares solicitados. Durante seu ciclo de vida, a negociação pode assumir os estados pendente, aceita, recusada ou cancelada.

A criação de uma proposta parte da seleção dos exemplares que serão oferecidos e dos exemplares que se pretende receber. Após o preenchimento das informações necessárias, a proposta é encaminhada ao usuário destinatário.

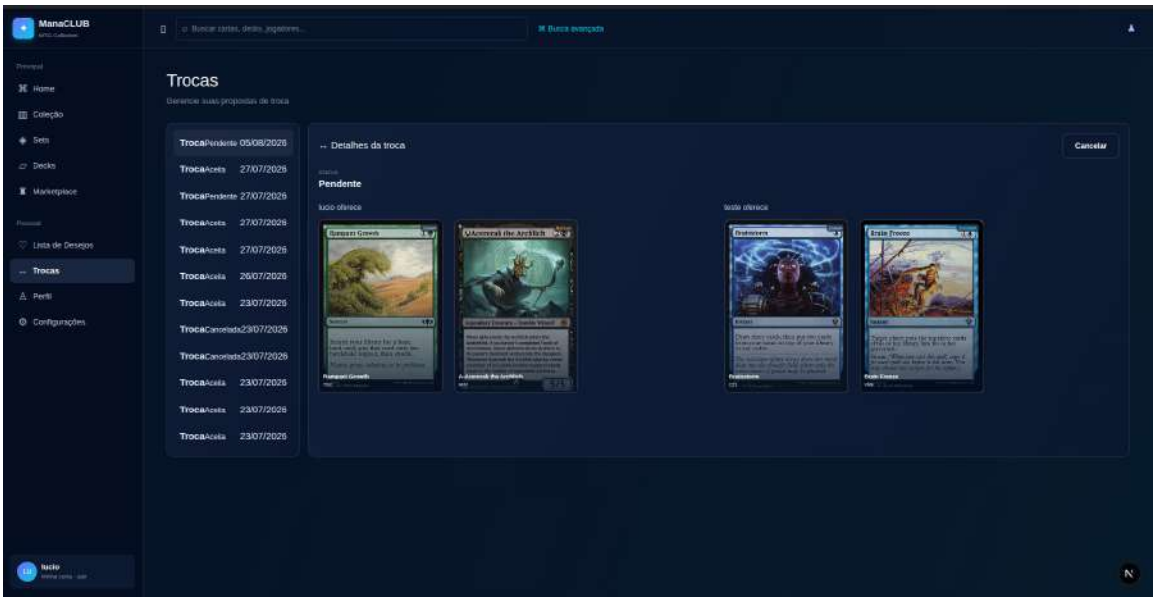

Figura 16 – Tela da proposta de troca

Fonte: Elaborado pelos autores (2026).

O usuário que recebe uma proposta pode consultá-la e decidir entre aceitá-la ou recusá-la. O autor da proposta pode cancelá-la enquanto ela permanecer pendente.

Quando uma proposta é aceita, os exemplares envolvidos são transferidos entre as coleções dos participantes e a negociação passa a integrar o histórico de trocas. Nos casos de recusa ou cancelamento, os exemplares permanecem associados às coleções originais.

Esse fluxo concentra uma das principais regras de negócio da aplicação, pois a negociação não representa apenas uma comunicação entre usuários, mas uma operação que pode alterar a propriedade dos exemplares registrados no sistema.

O sistema também permite consultar propostas enviadas e recebidas, acompanhar o estado das negociações e consultar o histórico das trocas concluídas.

A implementação desse fluxo atende aos requisitos RF-19 a RF-23, relacionados à criação, consulta, resposta, cancelamento e histórico das propostas de troca.

4.4 INTEGRAÇÃO COM SERVIÇOS EXTERNOS

As informações referentes às cartas são obtidas por meio da API pública Scryfall. Essa integração é utilizada durante pesquisas de cartas, cadastro de exemplares, inclusão de itens na wishlist e consulta a cartas disponíveis para troca.

Para reduzir consultas repetidas aos dados persistidos das coleções, foi utilizado Redis como mecanismo de cache.

Os registros permanentes da coleção permanecem no PostgreSQL, enquanto o Redis armazena temporariamente os dados das cartas agrupados pela coleção de cada usuário. Dessa forma, o backend pode utilizar o cache durante as consultas ao acervo sem confundi-lo com a integração externa. A política detalhada de preenchimento e sincronização do cache pertence à implementação; o aspecto relevante para esta arquitetura é que seu conteúdo representa as coleções dos usuários, e não respostas da API Scryfall.

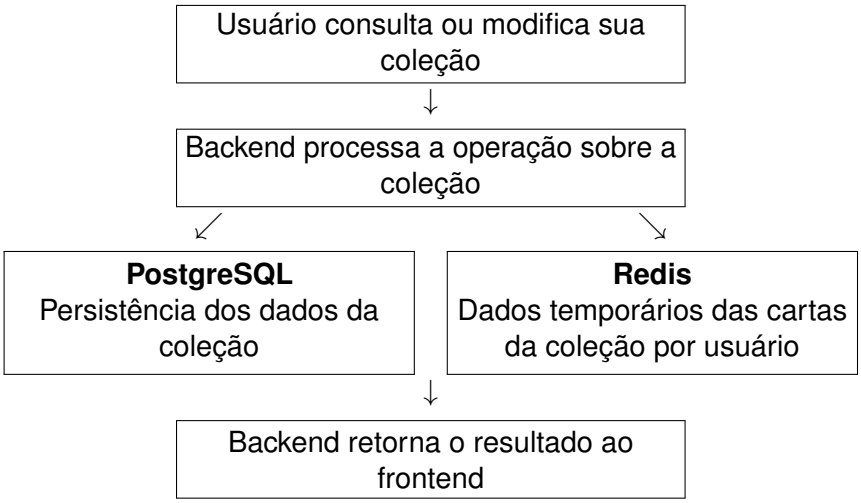

Figura 17 – Fluxo de utilização do cache Redis

Fonte: Elaborado pelos autores (2026).

4.5 VERIFICAÇÃO DOS REQUISITOS

A aplicação foi avaliada por meio da verificação dos requisitos definidos durante a especificação. Os autores executaram manualmente os principais fluxos pela interface e observaram o comportamento. Entre os cenários percorridos estiveram: cadastrar um usuário e autenticar-se; pesquisar uma carta e registrar, consultar, editar e remover um exemplar; incluir, atualizar e retirar um item da wishlist; disponibilizar um exemplar; e criar, aceitar, recusar ou cancelar uma proposta de troca.

Como exemplo, no fluxo de troca foram utilizadas duas contas. A primeira selecionou um exemplar disponibilizado pela segunda, acrescentou um exemplar próprio e enviou a proposta. Em seguida, pela conta destinatária, verificaram-se as ações de consulta e resposta. No cenário de aceitação, conferiu-se a alteração do estado da proposta, o registro no histórico e a transferência dos exemplares entre as coleções. Nos cenários alternativos, verificou-se que propostas recusadas ou canceladas não transferiam a propriedade dos itens.

Essa execução teve caráter de verificação funcional manual. Não foram conduzidos, nesta etapa, testes formais de usabilidade com usuários externos, testes de carga, ensaios de disponibilidade ou auditoria de segurança. Por isso, a análise distingue funcionalidades observadas de atributos de qualidade cuja avaliação quantitativa permanece como trabalho futuro.

Posteriormente, os requisitos não funcionais foram analisados segundo o tipo de evidência obtida durante o desenvolvimento. Requisitos arquiteturais e funcionais puderam ser inspecionados na implementação e nos fluxos manuais.

De forma geral, a execução manual indicou que os fluxos funcionais previstos para gerenciamento de coleções, listas de desejos e propostas de troca estavam disponíveis na versão examinada. A verificação não substitui uma suíte automatizada e sua repetição depende da execução dos mesmos cenários e dados de entrada.

A separação observada entre frontend, API, persistência e integração externa oferece pontos definidos para futuras evoluções. Entretanto, modificabilidade e escalabilidade não foram medidas e, portanto, são tratadas como expectativas arquiteturais, e não como resultados comprovados nesta avaliação.

5 CONSIDERAÇÕES FINAIS

O desenvolvimento do Mana Club teve como objetivo principal propor uma aplicação web capaz de auxiliar jogadores de *Magic: The Gathering* no gerenciamento de suas coleções de cartas, oferecendo recursos para organização da coleção pessoal, gerenciamento de listas de desejos e realização de negociações entre usuários.

A partir do levantamento dos requisitos, da modelagem do sistema e da implementação da aplicação, foi possível desenvolver uma solução integrada que reúne funcionalidades normalmente distribuídas entre diferentes plataformas. O sistema desenvolvido permite ao usuário cadastrar sua coleção, controlar cartas disponíveis para troca, registrar cartas desejadas, consultar coleções públicas e gerenciar propostas de negociação, atendendo aos objetivos definidos no início deste trabalho.

Sob a perspectiva da Engenharia de Software, o projeto possibilitou a aplicação prática dos conhecimentos adquiridos ao longo do curso de Análise e Desenvolvimento de Sistemas. Durante seu desenvolvimento foram empregados conceitos relacionados ao levantamento de requisitos, modelagem de dados, definição da arquitetura da aplicação, desenvolvimento de APIs REST, construção de interfaces web modernas, persistência de dados relacionais e utilização de mecanismos de cache para otimização de desempenho.

Os resultados obtidos indicam que a arquitetura adotada permitiu integrar as diferentes tecnologias de maneira organizada. A utilização da API Scryfall possibilitou incorporar dados catalográficos das cartas sem a manutenção de uma base local completa. O Redis, por sua vez, foi empregado para manter temporariamente os dados das cartas pertencentes às coleções dos usuários e reduzir consultas repetidas ao armazenamento persistente; ele não armazena as respostas provenientes da Scryfall. Como não foram realizados testes quantitativos de desempenho ou modificabilidade, eventuais ganhos nesses atributos permanecem como hipóteses a serem avaliadas.

Entretanto, algumas limitações devem ser consideradas. O sistema foi desenvolvido especificamente para o universo de *Magic: The Gathering*, utilizando a API Scryfall como principal fonte de dados catalográficos. Além disso, a avaliação foi baseada na execução manual dos fluxos e na verificação dos requisitos especificados, sem testes formais com usuários finais, testes de carga, auditoria de segurança ou avaliação sistemática de compatibilidade. Funcionalidades como sistema de mensagens entre usuários, notificações em tempo real, recomendações inteligentes de trocas, avaliação

de reputação e suporte a outros jogos de cartas colecionáveis também não foram contempladas nesta versão da aplicação.

Como trabalhos futuros, destaca-se inicialmente a implementação de um sistema de recomendação baseado em Inteligência Artificial capaz de sugerir trocas vantajosas entre usuários, identificar cartas compatíveis com a Wishlist e recomendar aquisições considerando a coleção existente. Também são perspectivas futuras a implementação de notificações em tempo real, mecanismos de reputação dos usuários, filtros avançados de pesquisa, integração com plataformas de comercialização de cartas e expansão da aplicação para suportar outros jogos de cartas colecionáveis.

Por fim, considera-se que os objetivos propostos foram alcançados e que o Mana Club constitui uma contribuição relevante tanto para a comunidade de jogadores de *Magic: The Gathering* quanto para a aplicação prática dos conhecimentos desenvolvidos durante a graduação em Análise e Desenvolvimento de Sistemas, evidenciando a importância da Engenharia de Software na construção de soluções capazes de atender necessidades reais de seus usuários.

REFERÊNCIAS

ANDERSON, D. J. **Kanban: Successful Evolutionary Change for Your Technology Business**. [S.l.]: Blue Hole Press, 2010.

Archidekt. **Archidekt**. 2026. Acesso em: 03 ago. 2026. Disponível em: <https://archidekt.com>.

BASS, L.; CLEMENTS, P.; KAZMAN, R. **Software Architecture in Practice**. 4. ed. Boston: Addison-Wesley, 2021.

Deckbox. **Getting Started**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://deckbox.org/help/start>.

____. **Trading Guidelines and FAQ**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://deckbox.org/help/trade_guidelines>.

FIELDING, R. T. **Architectural Styles and the Design of Network-based Software Architectures**. Tese (Doutorado) — University of California, Irvine, 2000.

GitHub. **GitHub Documentation**. 2026. Disponível em: <https://docs.github.com/>.

HUIZINGA, J. **Homo Ludens**. [S.l.]: Routledge, 2019.

Internet Engineering Task Force. **HTTP Semantics (RFC 9110)**. 2022. Acesso em: 03 ago. 2026. Disponível em: <https://www.rfc-editor.org/rfc/rfc9110>.

ISO/IEC/IEEE. **ISO/IEC/IEEE 29148:2018: Systems and Software Engineering – Life Cycle Processes – Requirements Engineering**. Geneva, 2018.

JSON. **Introducing JSON**. 2025. Acesso em: 03 ago. 2026. Disponível em: <https://www.json.org/json-en.html>.

ManaBox. **Collection: Getting Started**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://www.manabox.app/guides/collection/getting-started/>.

____. **ManaBox: MTG Card Collection**. 2026. Acesso em: 03 ago. 2026. Disponível em: <https://manabox.app>.

Microsoft. **The TypeScript Handbook**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://www.typescriptlang.org/docs/handbook/intro.html>.

Moxfield. **Moxfield**. 2026. Acesso em: 03 ago. 2026. Disponível em: <https://www.moxfield.com>.

OpenJS Foundation. **Express Documentation**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://expressjs.com/>.

____. **Node.js Documentation**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://nodejs.org/docs/latest/api/>.

PostgreSQL Global Development Group. **PostgreSQL Documentation**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://www.postgresql.org/docs/>.

PRESSMAN, R. S.; MAXIM, B. R. **Software Engineering: A Practitioner's Approach**. 9. ed. [S.l.]: McGraw-Hill, 2021.

React. **React Documentation**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://react.dev/>.

Redis. **Redis Documentation**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://redis.io/docs/latest/>.

SALEN, K.; ZIMMERMAN, E. **Rules of Play: Game Design Fundamentals**. Cambridge: MIT Press, 2004.

Scryfall. **Scryfall API Documentation**. 2026. Acesso em: 03 ago. 2026. Disponível em: <https://scryfall.com/docs/api>.

SOMMERVILLE, I. **Software Engineering**. 10. ed. [S.l.]: Pearson, 2019.

Tailwind Labs Inc. **Tailwind CSS Documentation**. 2026. Disponível em: <https://tailwindcss.com/docs>.

Vercel. **Next.js Documentation**. 2026. Acesso em: 05 ago. 2026. Disponível em: <https://nextjs.org/docs>.

Wizards of the Coast. **Magic: The Gathering**. 2026. Acesso em: 03 ago. 2026. Disponível em: <https://magic.wizards.com/>.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Cajazeiras - Código INEP: 25008978
Rua José Antônio da Silva, 300, Jardim Oásis, CEP 58.900-000, Cajazeiras (PB)
CNPJ: 10.783.898/0005-07 - Telefone: (83) 3532-4100

Documento Digitalizado Restrito

TRABALHO DE CONCLUSAO DE CURSO LUCIO E EDUARDO

Assunto:	TRABALHO DE CONCLUSAO DE CURSO LUCIO E EDUARDO
Assinado por:	Lucio Weslley
Tipo do Documento:	Anexo
Situação:	Finalizado
Nível de Acesso:	Restrito
Hipótese Legal:	Informação Pessoal (Art. 31 da Lei no 12.527/2011)
Tipo do Conferência:	Cópia Simples

Documento assinado eletronicamente por:

- **Lucio Weslley de Souza Leite**, **DISCENTE (202212010014) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 20/08/2026 14:52:23.

Este documento foi armazenado no SUAP em 20/08/2026. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 1959570
Código de Autenticação: 4a847a2cfb