CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA
Faculdade de Tecnologia da Praia Grande Curso Superior
de Tecnologia em Análise e Desenvolvimento de
Sistemas

GABRIELA GOUVEA DA ROCHA NOVO
KAIK DOS SANTOS NOVAIS
LUANA PINHEIRO TEIXEIRA

CRITMEET: LOCALIZANDO E REUNINDO JOGADORES DE RPG

Praia Grande

2025

GABRIELA GOUVEA DA ROCHA NOVO
KAIK DOS SANTOS NOVAIS
LUANA PINHEIRO TEIXEIRA

CRITMEET: LOCALIZANDO E REUNINDO JOGADORES DE RPG

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia da Praia Grande, como exigência parcial para obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.
Orientador: Prof. Joseffe Barroso de Oliveira

Praia Grande
2025

AGRADECIMENTOS

Ao longo desses três anos de jornada acadêmica, vivemos experiências marcantes, enfrentamos desafios e alcançamos conquistas que, por vezes, pareciam distantes. Cada momento, seja grande ou pequeno, significativo ou aparentemente insignificante, contribuiu de forma essencial para nossa formação e nos guiou até a conclusão deste trabalho.

Agradecemos, primeiramente, a Deus, por nos conceder força, sabedoria e perseverança ao longo desta caminhada, aos professores, que compartilharam conosco seus conhecimentos e experiências, nos inspirando a buscar sempre o melhor de nós mesmos. Nosso reconhecimento também se estende aos funcionários da instituição, que com sua atenção e disponibilidade nos ajudaram em diversos momentos e que, com o tempo, acabaram tornando-se nossos amigos. Aos colegas de turma, com quem dividimos não apenas os momentos de estudo e superação, mas também risos e amizades que levaremos para além da faculdade.

Por fim, nosso mais sincero agradecimento às nossas famílias e a todas as pessoas próximas que, com suas experiências, palavras de incentivo, compreensão e apoio incondicional, nos deram a força necessária para continuar, mesmo nos momentos mais difíceis. Sem esse suporte, certamente esta etapa de nossas vidas teria sido muito mais desafiadora.

RESUMO

O presente trabalho de conclusão de curso apresenta um relatório técnico sobre o CritMeet, uma aplicação web desenvolvida com o objetivo de aproximar jogadores de RPG de mesa, facilitando a formação de grupos para campanhas e sessões de jogo. Inspirado no estilo de aplicativos de relacionamento, o sistema permite que os usuários encontrem outros jogadores com base em preferências de estilo de jogo, sistema de RPG, disponibilidade e localização. A plataforma é voltada tanto para iniciantes quanto para veteranos, que podem criar perfis detalhados e procurar por correspondências compatíveis com seus interesses no universo do RPG de mesa. O acesso é feito via navegador, com autenticação de usuários e funcionalidades como criação de perfil, sistema de curtidas e mensagens entre usuários conectados. O CritMeet busca promover a interação e o engajamento da comunidade de RPG, oferecendo um espaço seguro e focado para a formação de novos grupos de jogo.

Palavras-chave: rpg; campanha; geolocalização; matchmaking; social.

ABSTRACT

This final course project presents a technical report on CritMeet, a web application developed to bring together tabletop RPG players, making it easier to form groups for campaigns and game sessions. Inspired by dating app interfaces, the system allows users to find other players based on game style preferences, RPG systems, availability, and location. The platform is designed for both beginners and veterans, who can create detailed profiles and search for compatible matches according to their interests in the tabletop RPG universe. Access is done via browser, with user authentication and features such as profile creation, a like system, and messaging between connected users. CritMeet aims to promote interaction and engagement within the RPG community by offering a safe and focused space for forming new gaming groups.

Keywords: rpg; campaign; geolocation; matchmaking; social.

LISTA DE ABREVIAÇÕES E SIGLAS

ACID	Atomicidade, Consistência, Isolamento, Durabilidade
AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
APT	Advanced Package Tool
AWS	Amazon Web Services
CLI	Command Line Interface
CPU	Central Process Unit
CSS	Cascading Style Sheets
DBMS	Database Management System
D&D	Dungeons & Dragons
DOM	Document Object Model
GPS	Global Positioning System
HTML	HyperText Markup Language
HTTPS	HyperText Transfer Protocol Secure
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
LGPD	Lei Geral de Proteção de Dados
MER	Modelo Entidade-Relacionamento
PHP	HyperText Preprocessor
POO	Programação Orientada a Objetos
RDBMS	Relational Database Management System
RPG	Role-Playing Game
SCRUM	Sprint, Cycle, Review, Update, Meeting
SEO	Search Engine Optimization
SHA-256	Secure Hash Algorithm 256-bit
SSH	Secure Shell
SQL	Structured Page Application
UML	Unified Modeling Language
URL	Uniform Resource Locator
VPS	Virtual Private Server
XML	Extensible Markup Language

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O RPG de mesa é um estilo de jogo que teve início por volta de 1980, onde os jogadores são livres para criar e interpretar seus próprios personagens dentro de um mundo fictício, que muitas das vezes envolve elementos fantasiosos de diferentes culturas, aventuras voltadas para os feitos heroicos ou tragédias que se dão no seu decorrer com os personagens em questão. Além disso, é necessário mais um participante, cuja função principal será a de narrar a história para os jogadores, este participante é chamado de Mestre ou Narrador, ele será responsável também por ditar as regras que definirão o que os jogadores podem fazer durante o jogo, ou até mesmo desenvolver a própria aventura (RPGMAISBARATO, 2018).

Um dos primeiros desafios enfrentados pelos jogadores de RPG de mesa são os valores de mercado, os custos associados aos itens utilizados em jogos presenciais podem apresentar grande variação. É comum encontrar livros com preços entre R$70,00 e R$350,00[1] dependendo do tipo de sistema escolhido para jogo, sem considerar os demais acessórios, como conjuntos de dados, divisórias de mapa (*grids*), tabuleiros e miniaturas. Dessa forma, é possível afirmar que o investimento necessário para iniciar ou manter uma prática regular de RPG presencial tende a ser elevado. Outro fator que também pode afastar iniciantes são as despesas e tempo de deslocamento até os locais onde ocorrem as sessões de jogos. Tal realidade pode desestimular novos jogadores, limitando a expansão da comunidade de RPG presencial. Com isso, surge outro desafio enfrentado pelos jogadores, que é a dificuldade em encontrar indivíduos próximos e com gostos similares dispostos a iniciar uma rotina de jogos, não apenas em razão dos custos envolvidos, mas também devido à alta demanda de tempo que uma sessão exige, podendo durar, em média, quatro horas ou mais.

Diante desse cenário, foi desenvolvido um sistema voltado à mediação entre jogadores de RPG de mesa, com o objetivo de facilitar o encontro entre indivíduos geograficamente próximos que já possuam os materiais necessários para a prática do jogo. A proposta busca, simultaneamente, atender dois públicos: os jogadores

[1] Valores obtidos por meio de pesquisas em plataformas como Mercado Livre, Amazon e Jambô Editora.

experientes, que enfrentam dificuldades em formar grupos estáveis e compatíveis, e os iniciantes interessados em ingressar nesse universo, mas que se deparam com as limitações estruturais e de acesso. Ao proporcionar um ambiente digital que permita a filtragem por localização, sistemas de regras preferidos e disponibilidade, a aplicação se propõe a aproximar pessoas com interesses comuns, promovendo a integração da comunidade e incentivando a prática do RPG presencial de forma mais acessível e organizada.

De acordo com o site da AWS (2024), aplicações *web* são programas que são executados diretamente no navegador, permitindo que os usuários acessem serviços e funcionalidades sem a necessidade de instalar nada no computador ou no celular. Entre as principais vantagens das aplicações *web* estão a compatibilidade com diferentes dispositivos, a facilidade de acesso e a constante atualização automática, já que o usuário sempre utiliza a versão mais recente disponível. Além disso, esse modelo reduz custos com manutenção e desenvolvimento, pois uma única aplicação pode ser usada por diversos usuários, independentemente do sistema operacional (AWS, 2024). Por essas razões, as aplicações *web* têm se tornado uma escolha popular em diversas áreas, desde plataformas de comércio eletrônico até ferramentas de colaboração e comunicação.

Por se tratar de uma plataforma voltada à conexão entre jogadores de RPG de mesa, é fundamental que o sistema possa ser acessado de forma rápida e simples, independentemente do dispositivo utilizado. A escolha por uma aplicação *web* elimina a necessidade de instalação, permitindo que novos usuários testem e utilizem a plataforma com facilidade, o que favorece a adesão e a formação de uma comunidade ativa. Além disso, a compatibilidade com diferentes navegadores e sistemas operacionais garante que o CritMeet atinja um público mais amplo, sem exigir adaptações para diferentes plataformas, o que torna o processo de manutenção e atualização mais ágil e econômico.

2 PROBLEMATIZAÇÃO DE PESQUISA

A ausência de meios acessíveis para formar ou integrar grupos de RPG impõe uma barreira estrutural que restringe, de forma sistemática, o alcance dessa atividade ao público geral. Diferentemente de outros jogos recreativos, os RPGs apresentam obstáculos particulares que dificultam a entrada de novos participantes. Essa limitação não decorre apenas da complexidade das regras, mas principalmente da necessidade de uma estrutura social prévia, como um grupo estável e jogadores experientes.

Hawkes-Robinson (2020) aborda essa problemática de maneira sistemática, identificando quatro desafios principais enfrentados por novos jogadores de RPG. Segundo o autor, o primeiro problema enfrentado por jogadores iniciantes é aprender um novo sistema de jogo, visto que enquanto em jogos de tabuleiro ou cartas é possível iniciar uma partida e aprender rapidamente, os RPGs demandam um certo tempo até que todas (ou boa parte) das regras de um sistema sejam compreendidas. O segundo obstáculo listado em sua análise é justamente a dificuldade de encontrar outras pessoas interessadas em jogar — especialmente aquelas que já dominam as regras do sistema escolhido (HAWKES-ROBINSON, 2020, p. 3).

> O terceiro obstáculo é aprender a conduzir o jogo com outras pessoas como jogadores, sendo o primeiro jogador o mestre de jogo (GM). O quarto obstáculo está relacionado ao tempo, dedicação, mantendo o “equilíbrio do jogo”, dinâmicas sociais complexas, resolução de conflitos e pensamento de longo prazo (citação pendente), incluindo aprender a como manter e desenvolver uma campanha contínua procurando evoluir gradativamente os níveis dos personagens dos jogadores (PCs) de forma sustentável (HAWKES-ROBINSON, 2020, p. 3 - 4, tradução elaborada pelos autores)[2].

Essa barreira, de natureza simultaneamente social e estrutural, evidencia o caráter colaborativo do RPG. Por se tratar de uma atividade que depende da interação entre múltiplos jogadores com um certo nível de conhecimento, torna-se

[2] *The third obstacle is learning how to run the game with other people as players, and the first player as a game master (GM).The fourth obstacle is related to time, dedication, maintaining “game balance”, complex social dynamics, conflict resolution, and longitudinal thinking, (citation pending) including learning how to maintain and grow an ongoing campaign working up to higher player character (PC) power levels in a sustainable way* (HAWKES-ROBINSON, 2020, p. 3 - 4).

inviável para um indivíduo iniciar a prática sozinho. A situação se agrava no caso de iniciantes: mesmo que um novo jogador adquira os materiais necessários, como livros e dados, a ausência de um grupo ou mentor torna difícil, senão impossível, experimentar o jogo de maneira satisfatória.

Essa limitação não é circunstancial, mas estrutural. A dependência de redes sociais já consolidadas e a falta de mecanismos acessíveis para inclusão de novos jogadores dificultam a expansão da base de praticantes. Em outras palavras, o acesso ao RPG permanece restrito àqueles que já estão inseridos em comunidades específicas, reproduzindo um ciclo excludente que limita o crescimento da atividade.

Dessa forma, surge o desenvolvimento do CritMeet, uma plataforma criada com o propósito de facilitar o encontro entre jogadores de RPG de mesa. A proposta busca solucionar um dos principais obstáculos enfrentados por entusiastas desse estilo de jogo, que é a dificuldade em encontrar outras pessoas para formar grupos e iniciar campanhas.

3 SOLUÇÃO E HIPÓTESE

A proposta central da plataforma CritMeet é promover conexões entre usuários com afinidades, por meio de um sistema de pareamento de perfis, que visa sugerir combinações com base em critérios específicos como localização, experiência de jogo e sistemas preferidos. A inspiração para o projeto surgiu da observação de dificuldades enfrentadas por jogadores da própria comunidade, da qual os autores fazem parte, ao tentar formar ou ingressar em novos grupos de RPG presencial.

Desde o início da pandemia em 2019, os encontros presenciais de RPG de mesa tornaram-se raros, devido à necessidade de isolamento social. Como alternativa, muitos grupos migraram para plataformas online, e mesmo após o fim das restrições, essa forma remota de jogar permaneceu como preferência para boa parte dos jogadores. Esse novo cenário deixou de lado aqueles que valorizam a experiência presencial, tornando difícil encontrar novos grupos ativos em sua região. Em muitos casos, jogadores acabaram se distanciando da prática, por não conseguirem se reconectar com a comunidade local.

A fragmentação causada pela grande diversidade de sistemas de RPG também contribui para esse problema. A criação de novos sistemas e a renovação constante das edições dos já existentes, apesar de movimentar o mercado, dificultam a formação de grupos coesos. Jogadores veteranos que preferem sistemas mais antigos, por exemplo, têm dificuldade de encontrar outros com os mesmos gostos. Além disso, a socialização em torno do RPG de mesa pode ser um desafio, especialmente para iniciantes, já que o jogo exige comunicação, improviso e interação constante entre os participantes.

Dessa forma, o CritMeet propõe uma solução utilizando uma tecnologia de *matchmaking*, comumente encontrado em redes sociais e aplicativos de relacionamento, onde a plataforma conecta usuários com base em suas preferências e perfis de jogo, promovendo encontros mais compatíveis (ALMENARA, 2021). Esse sistema de pareamento permite sugerir combinações com maior probabilidade de sucesso ao cruzar dados de interesse, comportamento e localização.

3.1 OBJETIVO GERAL

Por meio dos conhecimentos adquiridos ao longo do curso, foi desenvolvido o CritMeet, uma aplicação *web* com o propósito de aproximar jogadores de RPG de mesa que compartilham interesses em sistemas de jogo semelhantes e desejam participar de sessões presenciais. A proposta surge da observação de uma dificuldade comum entre esses jogadores: a formação de grupos compatíveis, tanto em localização quanto em estilo de jogo. Ao oferecer uma plataforma acessível e direcionada, o CritMeet busca facilitar essa conexão, promovendo uma melhor experiência social dentro da comunidade de RPG de mesa.

A plataforma propõe colocar o jogador como figura central na busca por experiências personalizadas. Por meio de filtros como sistema de jogo preferido, localização e disponibilidade, o usuário pode encontrar outros participantes com perfis compatíveis para a formação de novos grupos ou integração em campanhas já existentes. Essa abordagem também visa incentivar a inclusão de novos jogadores, oferecendo um ambiente acessível e amigável, no qual iniciantes podem encontrar pessoas dispostas a recebê-los, reduzindo as barreiras de entrada no hobby.

De forma geral, o objetivo do CritMeet não é apenas conectar pessoas, mas fomentar a construção de uma comunidade mais engajada e ativa em torno do RPG de mesa presencial. Acredita-se que, ao proporcionar uma ferramenta intuitiva, funcional e gratuita, será possível revitalizar e fortalecer a prática do RPG fora do meio digital, ao mesmo tempo que se cria um espaço seguro para a troca de experiências, ideias e interesses em comum entre os usuários da plataforma.

3.2 OBJETIVOS ESPECÍFICOS

A aplicação CritMeet será estruturada como uma plataforma *web* unificada, mas com funcionalidades distintas para diferentes perfis de usuários. Uma das funcionalidades principais consistirá em um sistema de *matchmaking* inteligente, que permitirá aos jogadores se cadastrarem e fornecerem informações sobre suas

preferências de jogo, localização, experiência e disponibilidade. Paralelamente, a plataforma oferecerá ferramentas de comunicação e organização, possibilitando que os usuários interajam através de *chat*, formem grupos de jogo, agendem sessões e compartilhem experiências sobre diferentes sistemas de RPG.

O sistema também contemplará recursos específicos para diferentes níveis de experiência. Para jogadores veteranos, oferecerá filtros avançados de busca por sistemas específicos, estilos de campanha e níveis de complexidade desejados. Para iniciantes, proporcionará um ambiente acolhedor com indicadores de grupos receptivos a novos jogadores e recursos educacionais sobre diferentes sistemas de RPG. Adicionalmente, a plataforma incluirá funcionalidades de moderação e segurança, permitindo denúncias, bloqueios e controle de privacidade, garantindo que a comunidade mantenha um ambiente saudável e respeitoso para todos os participantes.

4 JUSTIFICATIVA

Como mencionado anteriormente, o CritMeet é uma aplicação *web* desenvolvida com o objetivo de facilitar a conexão entre jogadores de RPG de mesa, funcionando como uma plataforma de encontros mas voltada exclusivamente ao público interessado nesse tipo de jogo. A proposta é oferecer uma ferramenta prática e intuitiva que permita aos usuários criar perfis, definir preferências de jogo, localização e estilo de campanha, possibilitando o encontro com outros jogadores que compartilham interesses semelhantes.

A motivação para o desenvolvimento da plataforma surge da dificuldade que muitos jogadores encontram para formar grupos presenciais ou *online* com pessoas compatíveis em estilo de jogo, horários e objetivos narrativos. Atualmente, essa busca ocorre de maneira desorganizada em fóruns, redes sociais e grupos dispersos, o que torna o processo demorado e, muitas vezes, frustrante.

Inicialmente, o CritMeet será disponibilizado em formato *web* para acesso via navegador, visando atingir um público mais amplo com maior facilidade de manutenção e atualizações constantes. O foco principal está na criação de uma comunidade ativa e engajada, promovendo encontros, campanhas e novas amizades no universo do RPG.

A plataforma pretende não apenas facilitar a formação de grupos, mas também valorizar a experiência do usuário ao permitir a criação de perfis detalhados e recomendações baseadas em preferências. Com isso, o CritMeet se posiciona como uma solução inovadora e necessária para a comunidade de RPG de mesa, promovendo inclusão, diversão e fortalecimento das conexões entre jogadores.

4.1 RELEVÂNCIA SOCIAL

O estudo conduzido por Walsh e Linehan (2024) investigou os efeitos psicossociais do jogo de RPG de mesa *Dungeons & Dragons* sobre a saúde mental de seus jogadores. A pesquisa identificou cinco dimensões principais que

contribuem positivamente para o bem-estar psicológico dos jogadores: escapismo, exploração de si, expressão criativa, apoio social e rotina. Esses elementos demonstram como o D&D pode funcionar como um espaço terapêutico informal, promovendo autoconhecimento, pertencimento e estabilidade emocional (WALSH & LINEHAN, 2024).

Desta forma, é notável que o RPG de mesa se estabelece como uma atividade que vai além da diversão, sendo valorizada até mesmo em contextos como a saúde mental. Ao possibilitar a imersão em narrativas simbólicas e a interpretação de personagens com diferentes vivências, o RPG favorece o desenvolvimento da empatia, do pensamento crítico e da expressão emocional. Além disso, a dinâmica colaborativa entre os jogadores estimula habilidades sociais, como comunicação, escuta ativa e resolução de conflitos, especialmente encontrado em ambientes terapêuticos.

Com a expansão da internet e dispositivos móveis, a tecnologia também tornou-se uma ferramenta essencial para otimizar interações não somente em áreas como saúde, educação e meio corporativo, como também auxilia na conexão entre indivíduos, fazendo com que a comunicação entre eles seja mais rápida e acessível (MARTINS, 2023).

Visto que o RPG de mesa possui benefícios terapêuticos e que com a tecnologia é possível quebrar as barreiras de comunicação, desenvolvemos um sistema que influenciará positivamente na comunidade de RPG e auxiliará pessoas interessadas por este estilo de jogo. Dessa forma, estaremos contribuindo também para o desenvolvimento de áreas como pensamento estratégico, comunicação interpessoal e a própria criatividade dos usuários da aplicação.

5 METODOLOGIA

A principal questão abordada neste trabalho foi desenvolver uma solução tecnológica capaz de facilitar a formação de grupos presenciais de RPG de mesa, conectando jogadores com base em preferências, localização e disponibilidade. Para isso, foi adotada uma abordagem de pesquisa qualitativa, conforme descrito por Lucas Mathias (2022), que define esse tipo de pesquisa como voltado à compreensão de aspectos subjetivos como comportamentos, percepções e pontos de vista (MATHIAS, 2022). Essa abordagem se mostrou ideal para compreender as dificuldades enfrentadas pelos jogadores na busca por grupos compatíveis e para identificar oportunidades de melhoria na experiência de socialização do RPG.

A pesquisa teve início com uma extensa revisão de literatura e análise de referências técnicas, visando entender tanto o contexto social do RPG de mesa quanto os requisitos tecnológicos necessários para a construção de uma aplicação *web* funcional. As observações da própria vivência dos integrantes do grupo na comunidade de RPG também serviram como base empírica para a formulação dos principais requisitos da plataforma. Além disso, foram consideradas opiniões de usuários potenciais, coletadas informalmente ao longo do desenvolvimento, como forma de validar hipóteses e ajustar funcionalidades.

O foco da investigação foi identificar os principais desafios enfrentados por jogadores iniciantes e veteranos na hora de formar ou participar de grupos de RPG presencial, mapeando padrões de comportamento, preferências de uso e limitações encontradas em outras plataformas. A partir dessas observações, buscou-se propor soluções compatíveis com tecnologias atuais, como geolocalização e *matchmaking*, para promover conexões mais assertivas entre os usuários.

Ao longo do desenvolvimento do CritMeet, foi adotada uma abordagem fundamentada na metodologia ágil SCRUM, reconhecida e utilizada no setor de tecnologia da informação. Essa escolha se deve por permitir gerenciar projetos de forma iterativa e incremental, promovendo a entrega contínua de valor. Ele é especialmente eficaz em contextos complexos e dinâmicos, nos quais os requisitos podem mudar ao longo do tempo, como no desenvolvimento de *software*.

De acordo com a AWS (2024), o SCRUM se destaca especialmente em equipes de desenvolvimento de *software*, por possibilitar uma resposta ágil às mudanças nos requisitos, sem comprometer os prazos e o controle de custos (AWS, 2024). Dentro dessa estrutura, a equipe conduziu reuniões frequentes para alinhamento de ideias, acompanhamento de progresso e planejamento das próximas etapas. O trabalho foi dividido em sprints, períodos de execução com duração de duas a quatro semanas, definidos conforme a complexidade de cada tarefa.

A divisão do desenvolvimento em ciclos curtos e bem definidos foi essencial para manter o foco nas funcionalidades prioritárias ao longo do projeto. A cada novo ciclo, eram estabelecidos os requisitos a serem implementados, assim como as responsabilidades de cada integrante da equipe. Como nem todas as funcionalidades foram planejadas desde o início, a estrutura adotada permitiu a incorporação progressiva de novas demandas, sem comprometer o andamento e a organização do trabalho.

O trabalho colaborativo desempenhou um papel fundamental no sucesso do projeto. A constante troca de conhecimentos e o apoio entre os integrantes foram determinantes para superar desafios técnicos, promover o aprendizado coletivo e acelerar a entrega das funcionalidades. Os métodos utilizados foram adaptados conforme a realidade e o perfil dos membros da equipe, com o objetivo de manter um fluxo de trabalho flexível, produtivo e eficiente.

Assim, a combinação entre uma pesquisa qualitativa para embasar a compreensão do problema e a adoção de práticas ágeis, ajustadas à dinâmica do grupo, foi decisiva para a efetividade do desenvolvimento do CritMeet. Essa integração metodológica permitiu que o sistema fosse concebido com foco real nas necessidades dos usuários, garantindo a entrega de uma aplicação funcional, relevante e alinhada à proposta do projeto.

6 PLANEJAMENTO DO PROJETO

O desenvolvimento do CritMeet teve como principal objetivo criar uma aplicação *web* acessível, intuitiva e funcional, capaz de conectar jogadores de RPG de mesa de diferentes localidades. Para alcançar esse propósito, foi essencial não apenas a escolha criteriosa das ferramentas de desenvolvimento, mas também o domínio das tecnologias envolvidas, visando garantir o melhor desempenho e a qualidade da aplicação.

A decisão de desenvolver uma plataforma *web* foi motivada pela sua flexibilidade e facilidade de acesso, uma vez que esse tipo de aplicação pode ser utilizada em qualquer dispositivo com acesso à *internet*, sem necessidade de instalação local. Essa característica atende diretamente ao perfil do público-alvo, que inclui usuários com diferentes níveis de familiaridade com tecnologia e que utilizam diversos tipos de dispositivos, como computadores, *tablets* e *smartphones*.

Para o front-end, foram utilizadas tecnologias consolidadas no mercado, como HTML, CSS, JavaScript, PHP e o *framework* Bootstrap. O HTML foi responsável pela estruturação dos elementos da interface, enquanto o CSS cuidou da estilização visual, garantindo uma apresentação atrativa e responsiva. O Bootstrap foi integrado para agilizar o desenvolvimento de *layouts* adaptáveis, fornecendo componentes pré-configurados com possibilidade de personalização, o que otimizou o tempo de produção e assegurou a compatibilidade com diferentes tamanhos de tela.

No desenvolvimento da lógica de interação com o usuário, o JavaScript desempenhou um papel importante, proporcionando dinamismo e interatividade à interface. Já o PHP, além de ser utilizado no *front-end* para gerar páginas dinâmicas, foi também a principal linguagem responsável pelo desenvolvimento do *back-end*, gerenciando as operações de comunicação com o banco de dados, autenticação de usuários, e o processamento das regras de negócio.

A camada de persistência de dados foi implementada com o uso de um banco de dados relacional SQL, gerenciado por meio do MySQL, garantindo a segurança,

integridade e consistência das informações armazenadas, como perfis de usuários, listas de amigos, grupos de RPG e mensagens trocadas entre os participantes.

O sistema foi hospedado em um ambiente de servidor remoto, utilizando uma ferramenta de VPS fornecida pela DigitalOcean, com sistema operacional Debian, garantindo estabilidade, escalabilidade e controle total sobre o ambiente de produção. Esse modelo de hospedagem foi escolhido por oferecer melhor desempenho e liberdade na configuração do servidor.

Durante a etapa de planejamento, a equipe utilizou o Draw.io para a criação dos diagramas UML de caso de uso, sequência, e diagrama de classes, permitindo visualizar e avaliar as funcionalidades previstas de forma rápida e prática. Na sequência, o Figma foi a ferramenta escolhida para a elaboração do protótipo de baixa fidelidade, com foco em validar os fluxos de navegação, na usabilidade e experiência do usuário, tendo uma ideia aproximada da interface do produto final.

O ambiente de desenvolvimento principal foi o Visual Studio Code, uma IDE (Ambiente de Desenvolvimento Integrado) leve e versátil, que oferece suporte a múltiplas extensões, facilitando a codificação em diferentes linguagens e a integração entre as tecnologias adotadas. Para o controle de versão, foi utilizado o GitHub, ferramenta essencial para gerenciar o histórico de alterações, facilitar o trabalho colaborativo e garantir a rastreabilidade das modificações ao longo do ciclo de vida do projeto.

A combinação entre essas tecnologias e ferramentas foi cuidadosamente planejada para garantir que o CritMeet atendesse tanto aos requisitos técnicos quanto às expectativas dos usuários, entregando uma solução funcional, escalável e alinhada ao objetivo de promover a integração da comunidade de RPG de mesa.

6.1 PROTOTIPAÇÃO

Inicialmente, foi desenvolvido um protótipo de baixa fidelidade com o objetivo de estabelecer uma base visual e estrutural para o *design* do *site*. Esse tipo de prototipagem permitiu testar e ajustar a organização dos elementos da interface,

como menus, botões e fluxos de navegação, de forma rápida e sem a necessidade de investir tempo em detalhes visuais. Além disso, possibilitou avaliar as ideias iniciais, identificar melhorias e alinhar as expectativas quanto à usabilidade do sistema, servindo como um guia essencial para as próximas etapas do desenvolvimento.

6.2 PROTÓTIPO DE BAIXA FIDELIDADE

Os protótipos de baixa fidelidade são representações simples da interface de um sistema, criadas antes do desenvolvimento. Eles ajudam a testar ideias, validar funcionalidades e ajustar o design de forma rápida e econômica, conforme descrito no site Miro:

> Os protótipos de baixa fidelidade são mais úteis quando você precisa testar cada elemento de design: desde fluxos de trabalho ou caminhos de conversão até o posicionamento de elementos visuais ou interação com o site.
>
> [...]
>
> Os protótipos de baixa fidelidade são mais úteis quando você precisa testar cada elemento de design: desde fluxos de trabalho ou caminhos de conversão até o posicionamento de elementos visuais ou interação com o site (MIRO, 2025).

Para desenvolver o protótipo em questão, utilizamos a ferramenta Figma, que se trata de uma plataforma colaborativa de design de interface. Além disso, o Figma facilitou o compartilhamento e a validação das ideias entre os integrantes do projeto, contribuindo para um processo de criação mais ágil e alinhado:

Figura 1 - Protótipo de Baixa Fidelidade

Fonte: elaborado pelos autores.

6.3 PROTÓTIPO DE ALTA FIDELIDADE

Após avanços com o projeto e amadurecimento do conceito, utilizamos do protótipo de baixa fidelidade como base para então produzir o produzir as telas do protótipo de alta fidelidade do sistema, proporcionando uma maior aproximação com a versão final do projeto.

> Um protótipo de alta fidelidade é uma representação detalhada e precisa do produto, incluindo aparência, interatividade e funcionalidades.
>
> Diferentemente dos protótipos de baixa fidelidade – que são esboços mais básicos – os protótipos de alta fidelidade são mais elaborados e oferecem uma simulação mais realista da experiência do usuário (ROSA, 2024).

Com isso, foram elaboradas as telas principais do protótipo em questão, permitindo uma visualização clara da estrutura e do fluxo da aplicação. Essa etapa foi fundamental para guiar o desenvolvimento e garantir que as funcionalidades estivessem alinhadas com a proposta inicial:

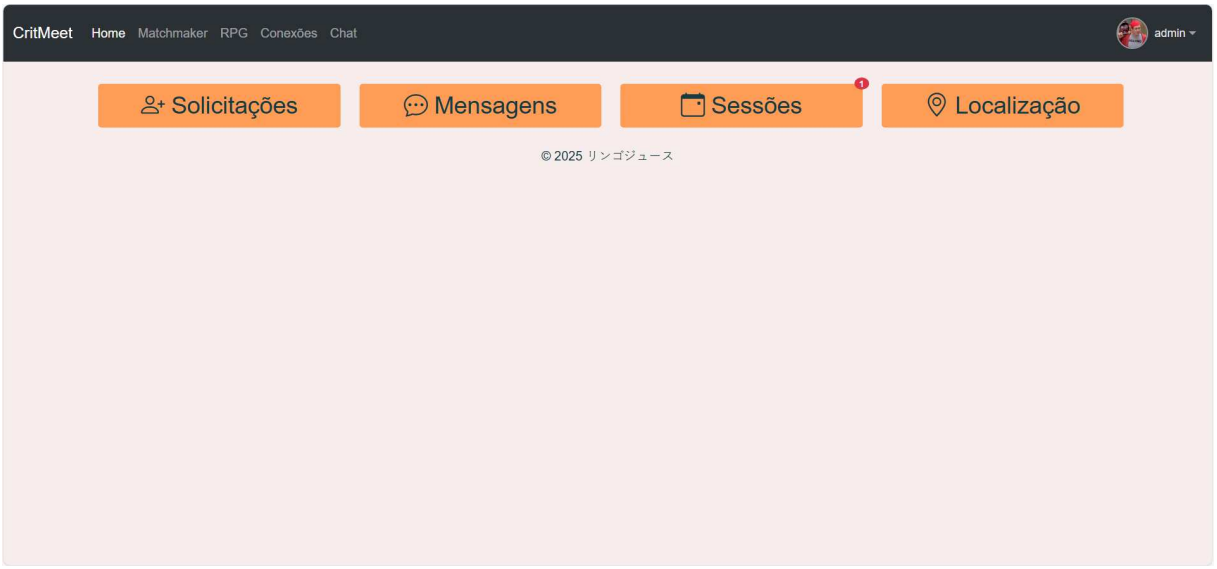
Figura 2 - Protótipo de alta fidelidade: Página de recepção

Fonte: elaborado pelos autores.

Figura 3 - Protótipo de alta fidelidade: Tela de login

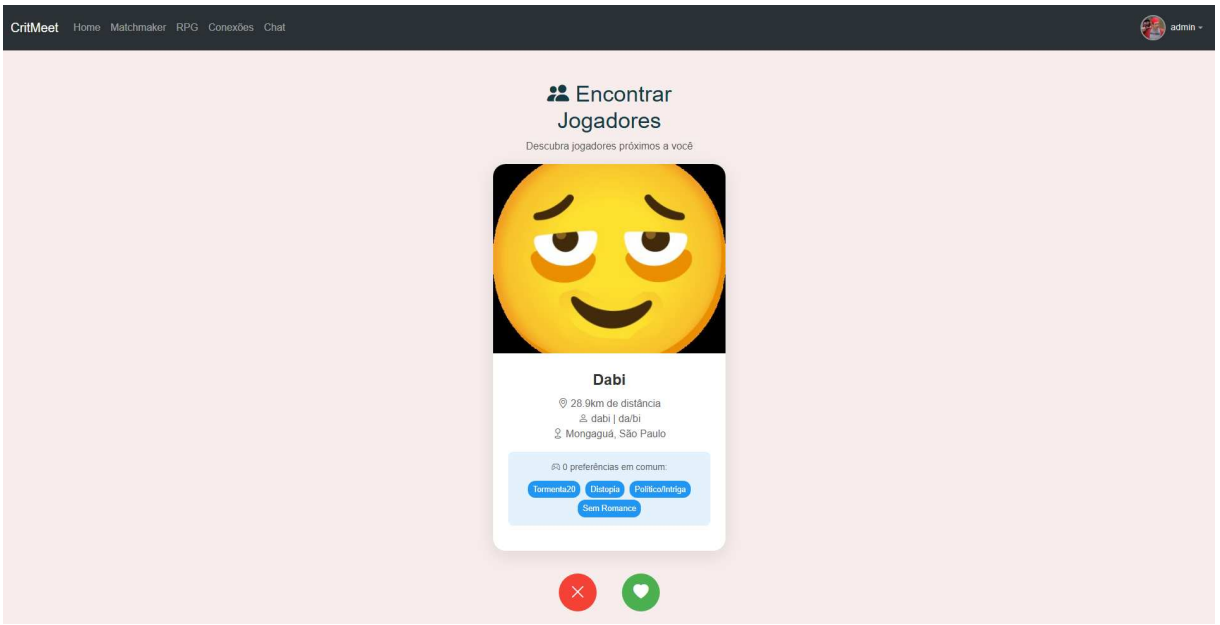
Fonte: elaborado pelos autores.

Figura 4 - Protótipo de alta fidelidade: Página inicial



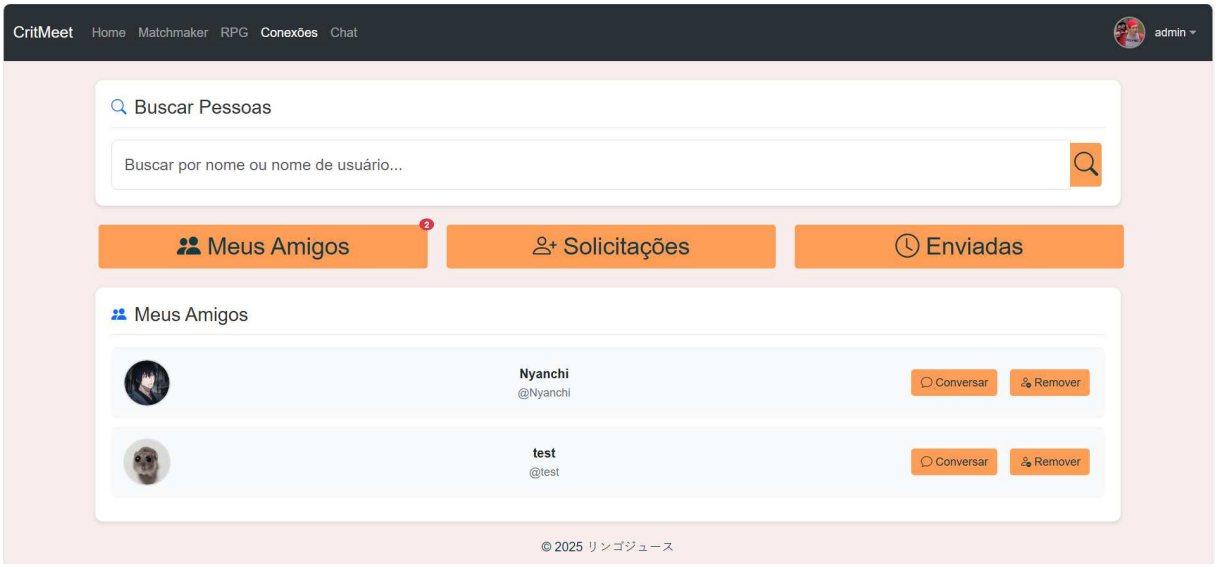
Fonte: elaborado pelos autores.

Figura 5 - Protótipo de alta fidelidade: Pesquisar jogadores próximos



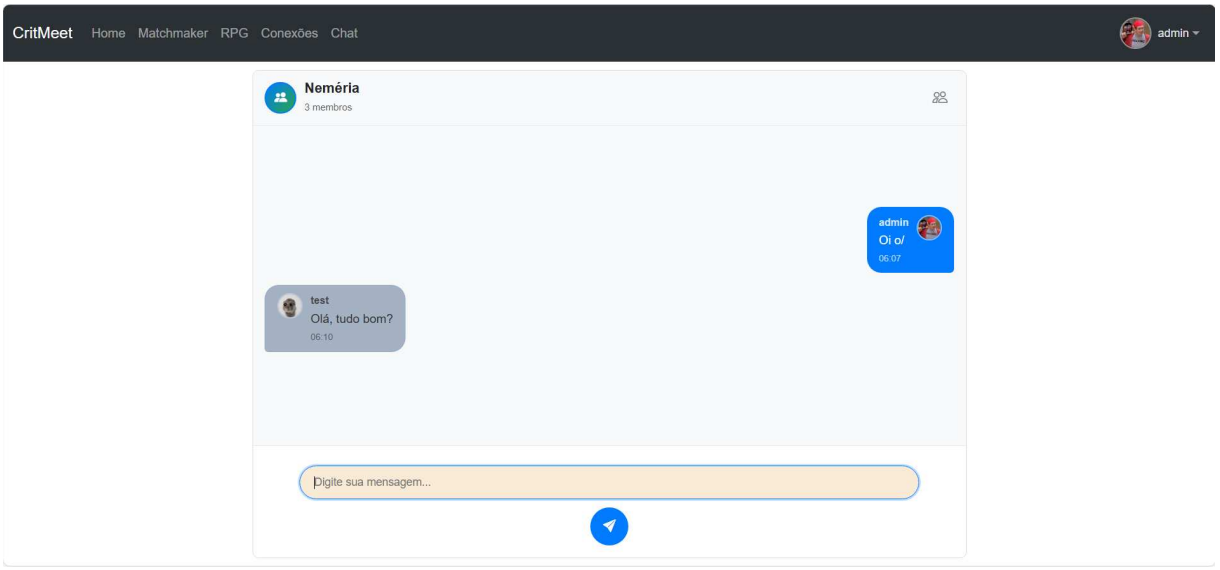
Fonte: elaborado pelos autores.

Figura 6 - Protótipo de alta fidelidade: Conexões



Fonte: elaborado pelos autores.

Figura 7 - Chat de grupo



Fonte: elaborado pelos autores.

6.4 ENTIDADES DO SISTEMA

Em modelagem de banco de dados, entidades representam os elementos fundamentais de um sistema, sendo compreendidas como abstrações de objetos ou conceitos do mundo real que possuem significado e sobre os quais se deseja armazenar informações. Cada entidade é caracterizada por um conjunto de

atributos, que descrevem suas propriedades ou características relevantes. Se um conceito envolve a necessidade de registrar informações específicas e estruturadas, como título e autor no caso de um livro, ele é reconhecido como uma entidade. Essas entidades, quando interligadas por referências, estabelecem relacionamentos, essenciais para garantir a integridade e coerência do modelo de dados (DEVMEDIA, 2007).

O site As profissões, afirma que “As entidades podem ser tangíveis, como um produto ou uma pessoa, ou intangíveis, como um evento ou uma transação” (ASPROFISSÕES, 2024). Dessa forma, ao identificar as entidades em um sistema, estamos definindo as representações de elementos concretos dentro de um sistema que podem se relacionar com a aplicação.

A definição precisa das entidades em um sistema é uma etapa crucial na modelagem de banco de dados, pois é a partir delas que se estrutura a base informacional da aplicação. Ao delimitar claramente quais elementos do mundo real serão abstraídos e representados no modelo, garante-se não apenas a organização lógica dos dados, mas também a capacidade do sistema de atender de forma eficaz às demandas funcionais e operacionais do domínio para o qual foi projetado. Além disso, uma identificação adequada das entidades promove uma maior consistência no relacionamento entre os dados e facilita a manutenção, a escalabilidade e a compreensão do sistema ao longo do tempo.

Para o *front-end*, as entidades exercem papel estruturante na definição das interfaces e dos componentes visuais. Elas orientam a construção das telas com base nos dados que precisam ser exibidos ou coletados, conforme os requisitos do sistema. Dessa forma, os elementos gráficos são desenvolvidos seguindo a estrutura lógica das entidades.

No *back-end*, as entidades se manifestam como estruturas que organizam os dados manipulados pelo sistema, funcionando como intermediárias entre o banco de dados e a lógica da aplicação. Assim, cada operação realizada pela aplicação, como o cadastro de um novo usuário ou a criação de uma nova sessão de jogo, envolve o uso de entidades como variáveis centrais que controlam o fluxo de dados e garantem a integridade da informação ao longo do sistema.

Após uma análise dos elementos do sistema, foram identificadas as entidades fundamentais que compõem a estrutura do CritMeet:

- Usuários (Users);
- Sessões (Sessions);
- Membros de Sessão (Session Members);
- Chats;
- Membros do Chat (Chat Members);
- Mensagens (Messages);
- Amigos (Friends);
- Localizações de Usuário (User Locations).

Tais entidades desempenham um papel central em diversos níveis do projeto, sendo empregadas de maneira integrada em todos os contextos da aplicação. Elas servirão de base para o desenvolvimento das interfaces no *front-end*, para a lógica de processamento no *back-end*, para a modelagem do banco de dados, bem como para a elaboração dos requisitos funcionais e dos diagramas UML que representam a arquitetura do sistema.

6.5 REQUISITOS FUNCIONAIS

Requisitos funcionais são as funcionalidades que o sistema deve realizar para atender às necessidades dos usuários. Eles descrevem o que o sistema deve fazer, como cadastrar usuários, realizar *login* ou emitir relatórios. Como descreve o site DevCommunity:

> São declarações das funcionalidades do sistema e de serviços que o sistema deve fornecer, como o sistema deve reagir a entradas específicas e como o sistema deve se comportar em determinadas situações. Também podem estabelecer explicitamente o que o sistema não deve fazer (FLORIANO, 2023).

A seguir, são listados os requisitos funcionais que definem as principais funcionalidades que o sistema deverá oferecer para atender às necessidades dos usuários e garantir seu correto funcionamento:

Tabela 1 - Requisitos funcionais

Nº Sequencial	**Identificação do Requisito**	**Descrição do Requisito**
RF001	Cadastrar perfil	Fundamental para que os usuários possam utilizar o sistema.
RF002	Editar perfil	Permite que o usuário edite seu próprio perfil, informações da conta e informações públicas.
RF003	Registro de permissões	Permite o controle de acesso e permissões dentro do sistema.
RF004	Deletar perfil	Permite que o usuário delete seu próprio perfil.
RF005	Localização de GPS	Permite que o sistema possa parear jogadores próximos com preferências similares.
RF006	Pesquisar jogadores próximos	Permite que o usuário pesquise jogadores próximos baseado em sua localização.
RF007	Filtrar pesquisa	Permite que o usuário busque por jogadores filtrando por categorias.
RF008	Adicionar à lista de amigos	Permite que o usuário adicione um perfil à sua lista de amigos.

Nº Sequencial	Identificação do Requisito	Descrição do Requisito
RF009	Desfazer amizade	Permite que o usuário delete um usuário da sua lista de amigos.
RF010	Bloquear usuário	Permite que o usuário bloqueie um perfil.
RF011	Chat interativo	Permite que os usuários se comuniquem e interajam entre si.
RF012	Criar grupo de jogo	Permite que o usuário crie um grupo de RPG e adicione outros jogadores.
RF013	Enviar convite de grupo	Permite que o usuário mestre envie um convite de grupo para um jogador.
RF014	Agendar sessão	Permite que o usuário mestre do grupo agende uma sessão de RPG.
RF015	Enviar mídia	Permite que o usuário envie imagens, links ou arquivos de mídia.
RF016	Denunciar usuário	Permite que o usuário denuncie um perfil que esteja infringindo leis e normas dentro da plataforma.

6.6 REQUISITOS NÃO FUNCIONAIS

Diferentemente dos requisitos funcionais, os requisitos não funcionais são características e restrições que o sistema deve atender, mas que não estão diretamente ligadas às suas funcionalidades. Eles definem como o sistema deve se comportar, abordando aspectos como desempenho, segurança, usabilidade,

confiabilidade, entre outros (FLORIANO, 2023).

Visto que o sistema em questão é uma plataforma *web*, os requisitos não funcionais do CritMeet devem garantir uma navegação eficiente, segura e acessível para todos os usuários. Esses requisitos estão descritos na tabela a seguir, servindo como base para orientar o desenvolvimento da aplicação:

Tabela 2 - Requisitos não funcionais

Nº Sequencial	**Identificação do Requisito**	**Descrição do Requisito**
RNF001	Tempo de resposta	O sistema deve responder a todas as solicitações realizadas pelo usuário em até 3 segundos para operações básicas (login, navegação, busca simples) e até 5 segundos para operações complexas (filtros avançados, processamento de localização), em condições normais de uso.
RNF002	Segurança de dados	O sistema deve utilizar o htpasswd para autenticação básica HTTP, garantindo que as senhas sejam armazenadas de forma criptografada usando algoritmos seguros (bcrypt e SHA-256). Todas as comunicações devem ser realizadas através de HTTPS.
RNF003	Design responsivo	O sistema deve ajustar automaticamente o layout e os elementos da interface para oferecer uma experiência consistente e otimizada em dispositivos com diferentes tamanhos de tela, incluindo smartphones (320px+), tablets (768px+) e desktops (1024px+).

Nº Sequencial	Identificação do Requisito	Descrição do Requisito
RNF004	Escalabilidade	O sistema deve suportar a adição de novos servidores para atender a um aumento na demanda de usuários simultâneos.
RNF005	Compatibilidade	O sistema deve ser compatível com os navegadores Chrome, Firefox, Safari e Edge, em suas versões lançadas nos últimos dois anos.
RNF006	Usabilidade	O sistema deve seguir os padrões de acessibilidade WCAG 2.1 nível AA, incluindo suporte a leitores de tela, navegação por teclado, contraste adequado de cores e textos alternativos para imagens, garantindo acesso inclusivo para pessoas com deficiência.
RNF007	Backup e recuperação	O sistema deve realizar backups automáticos diários do banco de dados e arquivos críticos, armazenando-os de forma segura por no mínimo 30 dias, com procedimentos de recuperação documentados e testados regularmente.
RNF008	Personalização de tema	O sistema deve permitir que os usuários alternem entre os temas claro e escuro em tempo real, sem necessidade de recarregar a página, mantendo a preferência persistente entre sessões.
RNF009	Performance de Geolocalização	O sistema deve processar consultas de localização e pareamento de jogadores próximos em até 2 segundos, utilizando algoritmos eficientes para cálculo de

Nº Sequencial	Identificação do Requisito	Descrição do Requisito
		distâncias geográficas e cache adequado para otimizar consultas repetitivas.

6.7 DIAGRAMAS UML

De acordo com as informações do *site* do Lucidchart, um diagrama UML (Linguagem de Modelagem Unificada) é uma representação visual padronizada usada para modelar, visualizar e documentar sistemas de *software*, especialmente orientados a objetos, facilitando a comunicação entre desenvolvedores, analistas e demais envolvidos. Ele inclui diagramas estruturais, que mostram a organização estática do sistema, e diagramas comportamentais, que representam seu funcionamento dinâmico, ajudando a entender e planejar o sistema durante todo o seu desenvolvimento.

Os diagramas UML usam a programação orientada a objetos (POO) para representar sistemas por meio de objetos que simulam entidades reais. Eles mostram classes, atributos, métodos, herança, encapsulamento e polimorfismo, facilitando a organização, o *design* e a comunicação do *software* seguindo os princípios da POO.

> Linguagens orientadas a objetos dominam o mundo da programação porque elas modelam objetos do mundo real. A UML é uma combinação de várias notações orientadas a objetos: design orientado a objetos, técnica de modelagem de objetos e engenharia de software orientada a objetos (LUCIDCHART, 2025).

Foram selecionados três tipos de diagramas UML para o projeto do CritMeet, cada um com um propósito específico:

- **Diagramas de Caso de Uso:** mostram as principais funcionalidades de um

sistema e como os usuários (atores) interagem com ele, focando no que o sistema é capaz de fazer dependendo do ator em questão;

- **Diagramas de Sequência:** apresentam como os objetos interagem entre si ao longo do tempo para realizar uma função no sistema, detalhando a ordem das mensagens trocadas entre os objetos e quando essas interações ocorrem, ajudando a entender o fluxo de execução de um processo ou funcionalidade específica;

- **Diagrama de Classes:** representa a estrutura estática de um sistema, mostrando as classes, seus atributos, métodos e os relacionamentos entre elas, auxiliando na visualização de como os elementos do sistema estão organizados.

Nos tópicos a seguir, serão apresentadas com mais detalhes as definições e funcionalidades de cada um dos diagramas selecionados para o projeto, complementando a breve discussão realizada anteriormente. Cada diagrama será abordado de forma individual, destacando seu propósito, a maneira como contribui para o desenvolvimento do CritMeet e sua relevância dentro da modelagem do sistema.

6.7.1 DIAGRAMAS DE CASO DE USO

De acordo com o site da IBM (2021), um diagrama de caso de uso é uma representação gráfica que mostra as funcionalidades de um sistema a partir da perspectiva do usuário, evidenciando as interações entre os usuários (atores) e o sistema para atingir determinados objetivos. Seu principal propósito é ilustrar as intenções dos usuários e os serviços que o sistema deve oferecer para satisfazê-las, sem entrar nos detalhes técnicos de como essas funcionalidades serão executadas.

> Outra vantagem dos casos de uso é que eles podem ser facilmente compreendidos por pessoas sem conhecimento técnico. Isso significa que eles podem ser usados para comunicação com *stakeholders* do projeto, incluindo gerentes de projeto, usuários finais e desenvolvedores (ALFF, 2024).

Este tipo de diagrama é essencial para apresentar de forma clara e simplificada como o sistema funcionaria para os usuários de modo geral. Para isso, ele utilizará três elementos para esta ilustração:

- **Atores:** são entidades externas que interagem com o sistema, como usuários ou outros sistemas. Eles representam quem utiliza ou se comunica com o sistema, ajudando a identificar as funcionalidades esperadas e os papéis envolvidos;

- **Caso de Uso:** representa uma função específica ou ação que o sistema realiza em resposta à interação de um ator. Ele descreve o que o software deve fazer para atender a uma necessidade do usuário, de forma simples e objetiva;

- **Relações:** mostram como os atores interagem com o sistema e como os casos de uso se conectam entre si, indicando dependências, inclusões («*include*»), extensões («*extends*») ou heranças de comportamento.

O primeiro passo para desenvolver um diagrama de caso de uso consistiu na identificação dos atores que interagem com o sistema. Com isso, pudemos identificar os seguintes atores:

- **Administrador:** responsável por gerenciar usuários, moderar conteúdos, garantir a segurança da plataforma e supervisionar o bom funcionamento do sistema;

- **Mestre de Jogo:** é um tipo de usuário capaz de criar, gerenciar e definir as regras de um grupo, organizar suas próprias sessões e interagir com jogadores interessados em participar das suas mesas. Ele é responsável pelo gerenciamento de sua própria comunidade criada dentro da plataforma;

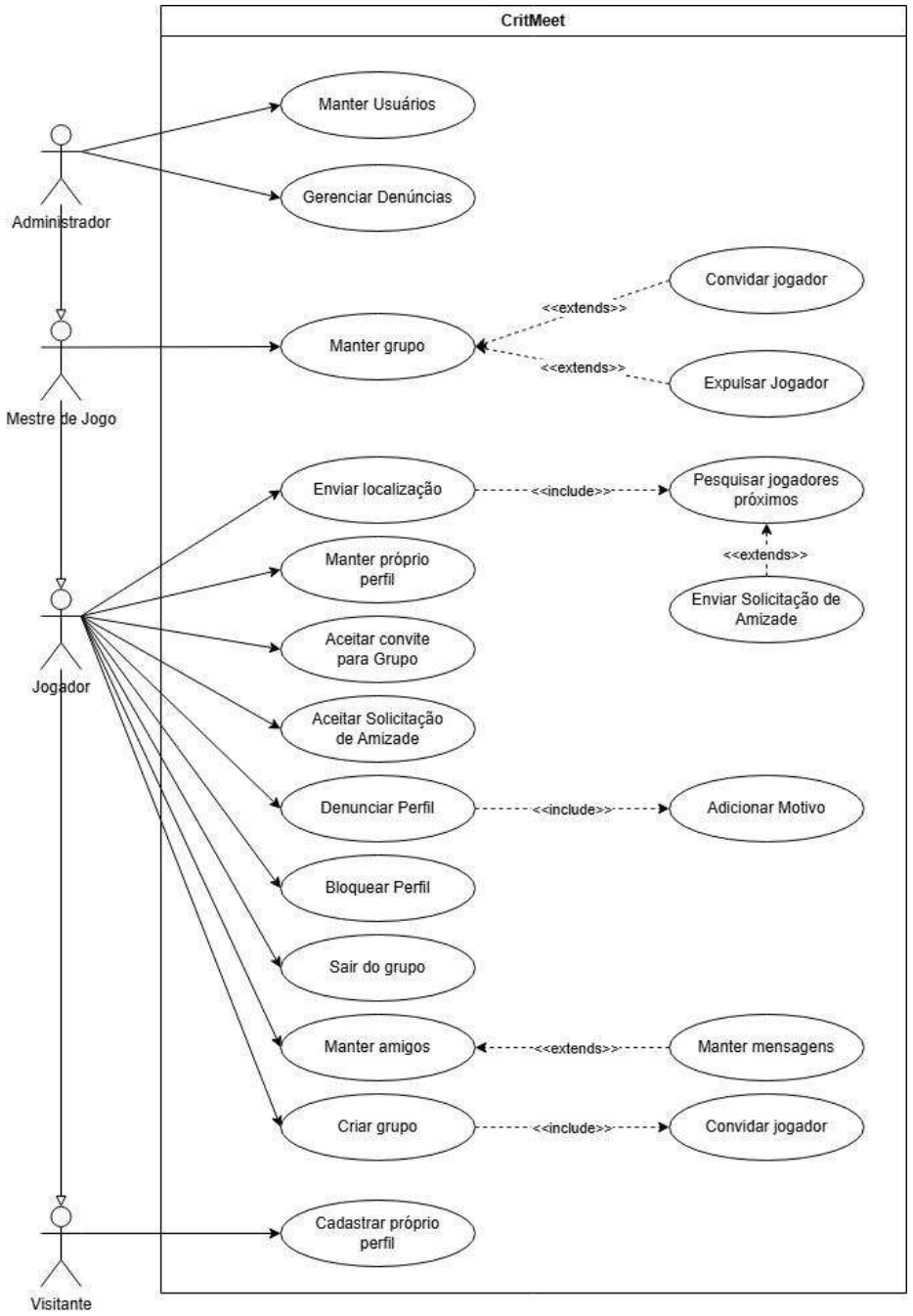
- **Jogador:** um usuário que busca e interage com outros jogadores com base em seus interesses, podendo participar de grupos e sessões no sistema. Qualquer jogador é capaz de se tornar um Mestre de Jogo ao criar um grupo;

- **Visitante:** é todo aquele que pode acessar a plataforma do CritMeet, mas não pode interagir, curtir ou enviar mensagens até se registrar ou entrar como

usuário.

Em seguida, foram listados os casos de uso do sistema, quais atores estão envolvidos e suas relações com as funcionalidades disponíveis. Após a associação dos atores com seus respectivos casos de uso, utilizamos da ferramenta Draw.io, muito conhecida e utilizada no desenvolvimento de diagramas UML.

Figura 8 - Diagrama de Caso de Uso

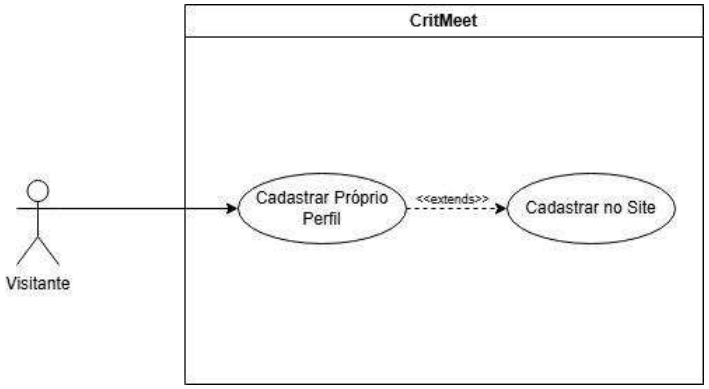


Fonte: elaborado pelos autores.

A figura 8 apresenta o diagrama de caso de uso geral do sistema, com o objetivo de ilustrar as principais funcionalidades oferecidas e os respectivos atores que interagem com cada uma delas. Observa-se uma hierarquia entre os atores, na qual os que estão posicionados no topo possuem acesso não apenas aos seus próprios casos de uso, mas também aos das camadas inferiores.

As figuras a seguir apresentam um detalhamento mais específico de cada caso de uso representado anteriormente na figura 8:

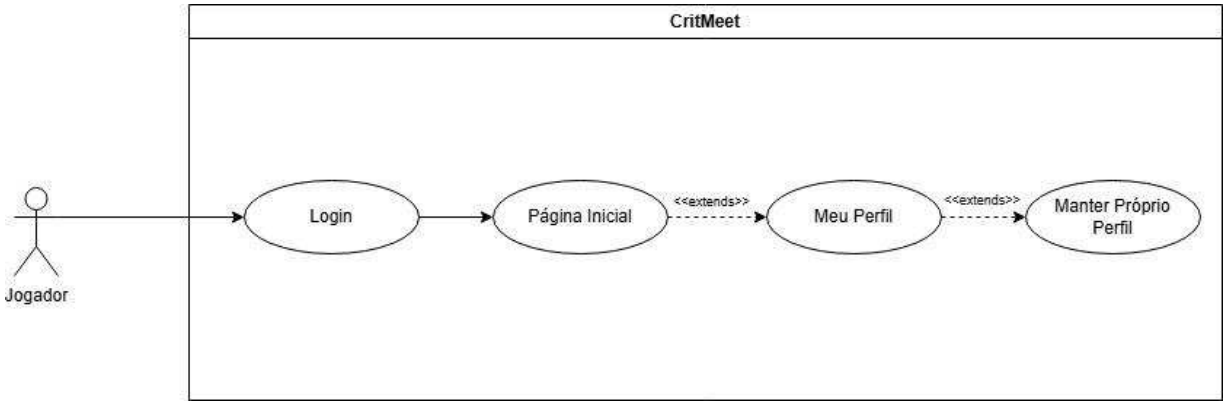
Figura 9 - Diagrama de Caso de Uso: Cadastrar usuário



Fonte: elaborado pelos autores.

A figura 9 retrata a interação de um visitante dentro da plataforma ao se cadastrar como usuário do CritMeet. Assim como apresentado no diagrama acima, a ação pode se estender para criar uma conta diretamente no próprio *site* da aplicação. Essa extensão é representada por uma linha pontilhada com a indicação «*extends*», mostrando que se trata de um comportamento complementar que é executado dependendo de como o ator interagiu no sistema.

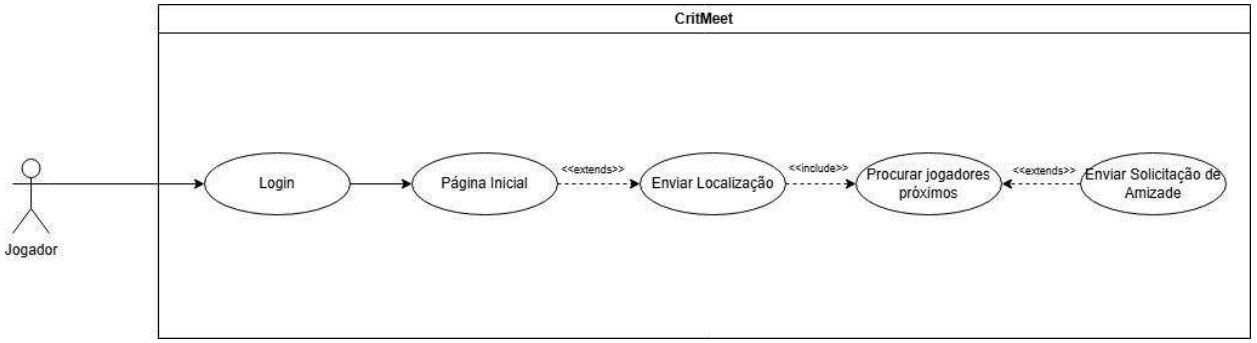
Figura 10 - Diagrama de Caso de Uso: Manter perfil



Fonte: elaborado pelos autores.

A figura 10 demonstra o diagrama de caso de uso do jogador relacionado à manutenção do próprio perfil. O uso do verbo “manter” neste caso de uso deve-se ao conjunto de funcionalidades envolvidas no gerenciamento do perfil, como edição, adição e exclusão de informações.

Figura 11 - Diagrama de Caso de Uso: Procurar jogadores próximos



Fonte: elaborado pelos autores.

A figura 11 apresenta o diagrama de caso de uso referente à funcionalidade em que um usuário do tipo jogador realiza a busca por outros jogadores próximos. Abaixo, está descrita a especificação detalhada dessa interação:

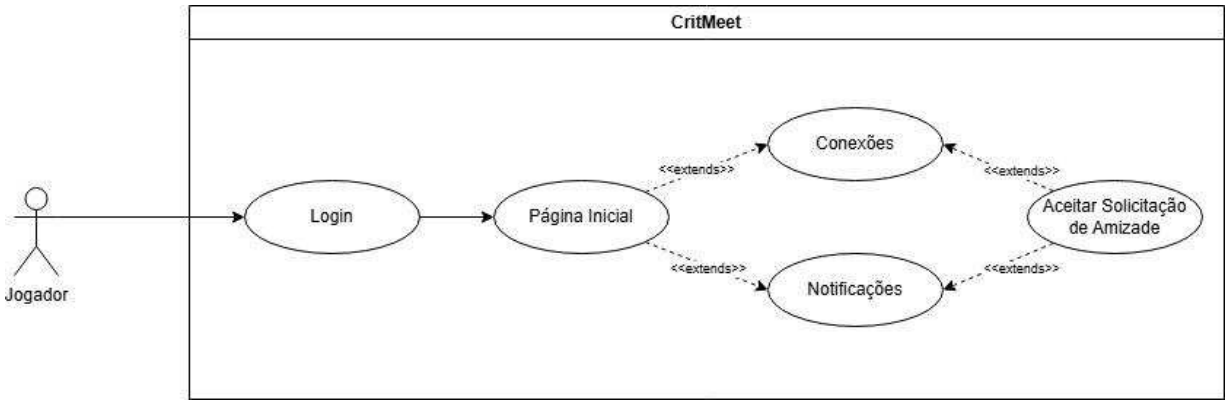
- **Enviar Localização:** esse caso de uso é uma extensão da página inicial e permite que o sistema identifique onde o usuário está, possibilitando encontrar e sugerir jogadores próximos para facilitar o pareamento e a criação de grupos locais;

- **Procurar jogadores próximos:** ao compartilhar a localização com o *site*, este caso de uso será incluído, permitindo que o ator pesquise por jogadores próximos de acordo com as preferências do usuário;

- **Enviar Solicitação de Amizade:** esse caso de uso estendido permite que ator envie solicitações de amizade para jogadores que aparecem como resultado de pesquisa.

Assim como as extensões, as relações de inclusão também são representadas por linhas pontilhadas no diagrama, mas com a indicação «*include*», que sinaliza a obrigatoriedade da execução do caso de uso incluído como parte integrante do caso de uso principal.

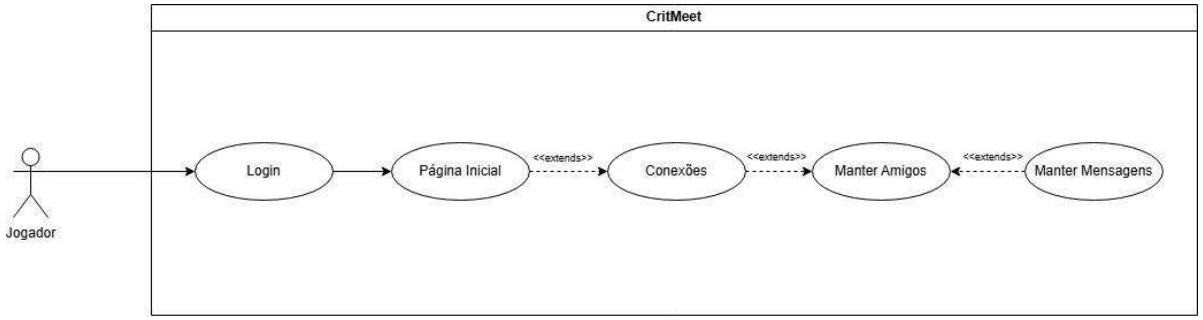
Figura 12 - Diagrama de Caso de Uso: Aceite de amizade



Fonte: elaborado pelos autores.

A figura 12 ilustra o caso de uso estendido de aceitação de amizade. Observa-se que o usuário pode aceitar uma solicitação de amizade por meio de dois caminhos diferentes dentro da aplicação: pela aba de conexões ou pela aba de notificações.

Figura 13 - Diagrama de Caso de Uso: Manter amigos

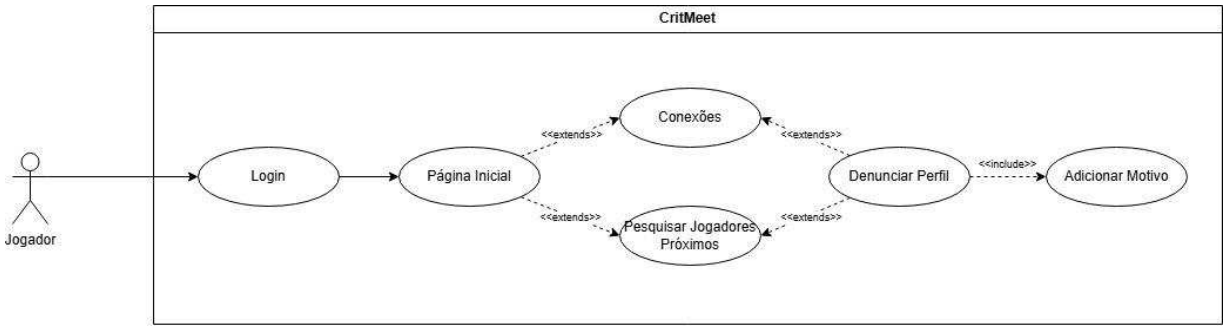


Fonte: elaborado pelos autores.

A figura 13 representa o caso de uso "Manter amigos", que engloba as ações relacionadas ao gerenciamento da lista de amizades do usuário. A seguir, detalha-se a especificação do diagrama em questão:

- **Manter amigos**: acessado por meio da aba de conexões, este caso de uso permite ao usuário gerenciar sua lista de amizades dentro da aplicação. Entre as ações possíveis estão a visualização do perfil de amigos e remoção de conexões existentes;
- **Manter mensagens**: esse caso de uso estendido permite ao usuário visualizar, enviar e excluir mensagens trocadas com outros jogadores da sua lista de conexões.

Figura 14 - Diagrama de Caso de Uso: Denunciar perfil

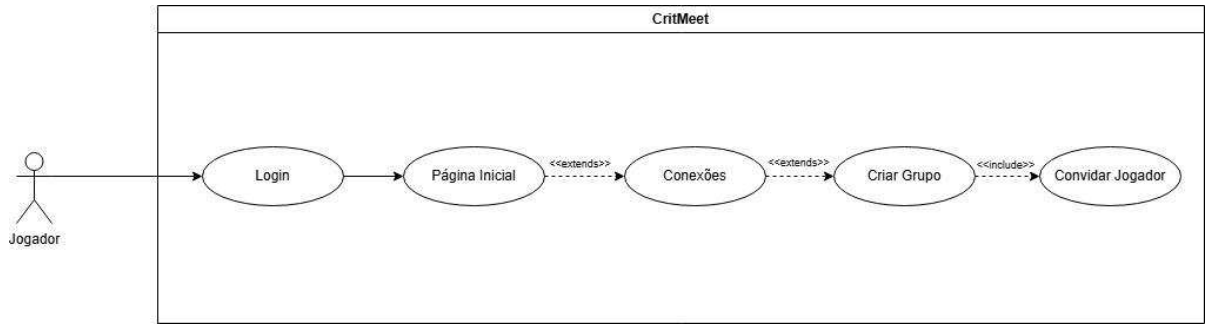


Fonte: elaborado pelos autores.

A figura 14 traz o diagrama de caso de uso referente à funcionalidade de denunciar perfil, de modo a evidenciar usuários infratores das leis e/ou regras da comunidade. A seguir, é descrita a especificação detalhada deste caso de uso:

- **Denunciar perfil:** este caso de uso estendido pode ser acessado tanto pela aba de pesquisa de jogadores quanto pela aba de conexões. Com ele, é possível permitir que os próprios usuários da plataforma delatem infratores para os administradores de modo que estes sejam punidos de acordo com suas ações;
- **Adicionar motivo:** um caso de uso incluído ao ato de denunciar um perfil que é acionado automaticamente sempre que um usuário opta por realizar uma denúncia, sendo necessário que o denunciante selecione e descreva o motivo pelo qual está reportando o perfil. Essa inclusão garante que todas as denúncias sejam registradas com uma justificativa clara, auxiliando os administradores na análise e no tratamento adequado de cada caso.

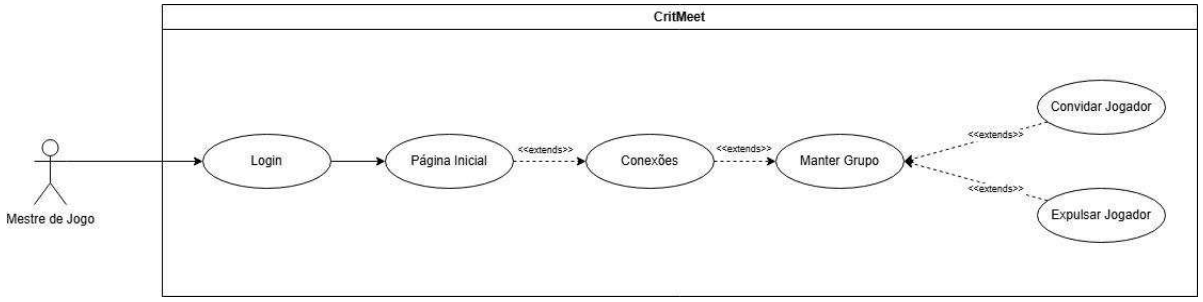
Figura 15 - Diagrama de Caso de Uso: Criar grupo



Fonte: elaborado pelos autores.

A figura 15 ilustra o caso de uso criar grupo, que permite ao jogador formar um grupo de jogo com outros usuários da plataforma. Em seguida, é acionado automaticamente o caso de uso incluído convidar jogador, que exibe uma lista dos amigos previamente adicionados, possibilitando o envio de convites para compor o grupo. Após a criação do grupo, o jogador responsável assume automaticamente o papel de Mestre de Jogo, passando a ter acesso às funcionalidades de gerenciamento do grupo formado.

Figura 16 - Diagrama de Caso de Uso: Manter grupo



Fonte: elaborado pelos autores.

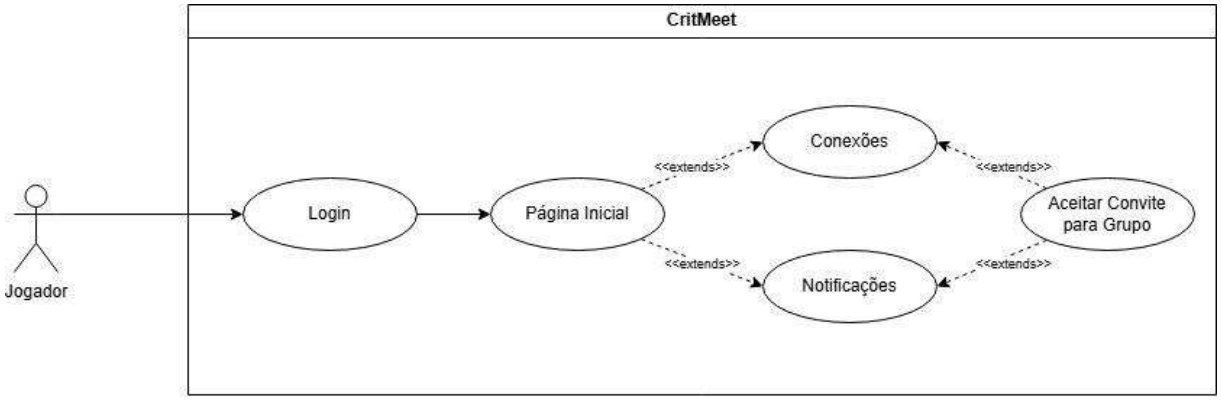
A figura 16 demonstra o diagrama de caso de uso manter grupo, associado ao ator Mestre de Jogo, detalhando as funcionalidades disponíveis para o gerenciamento de grupos já criados. Segue abaixo a especificação completa deste caso de uso:

- **Manter grupo:** permite o Mestre de Jogo gerenciar seu próprio grupo de RPG. As funcionalidades envolvem editar as informações do grupo, como nome, descrição, agendamento de sessões, e exclusão do grupo, sendo estas ações exclusivas do Mestre de Jogo;

- **Convidar jogador:** este caso de uso estendido representa a ação em que o Mestre de Jogo pode convidar outros usuários, previamente adicionados à sua lista de amigos, para participarem de um grupo de RPG já criado por ele. O convite é enviado por meio da plataforma, e os jogadores convidados podem aceitar ou recusar a solicitação;

- **Expulsar jogador:** este caso de uso estendido permite que o Mestre de Jogo remova um participante de um grupo de RPG que ele administra. Ao expulsar um jogador, sua participação é encerrada imediatamente, deixando de ter acesso às informações e interações do grupo.

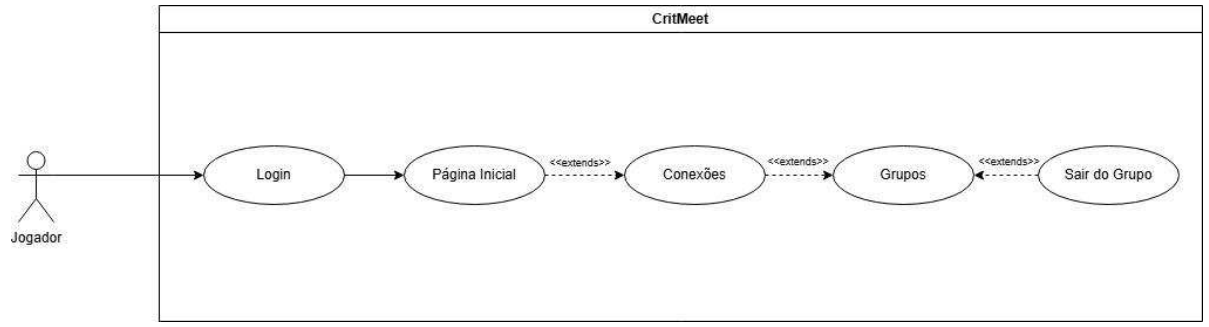
Figura 17 - Diagrama de Caso de Uso: Aceite de grupo



Fonte: elaborado pelos autores.

A figura 17 se trata do caso de uso estendido de aceite de convite para grupo. Esta funcionalidade pode ser acionada tanto pela aba de conexões como pela aba de notificações, tendo como objetivo permitir que um jogador aceite ou recuse um convite para participar de um grupo de RPG.

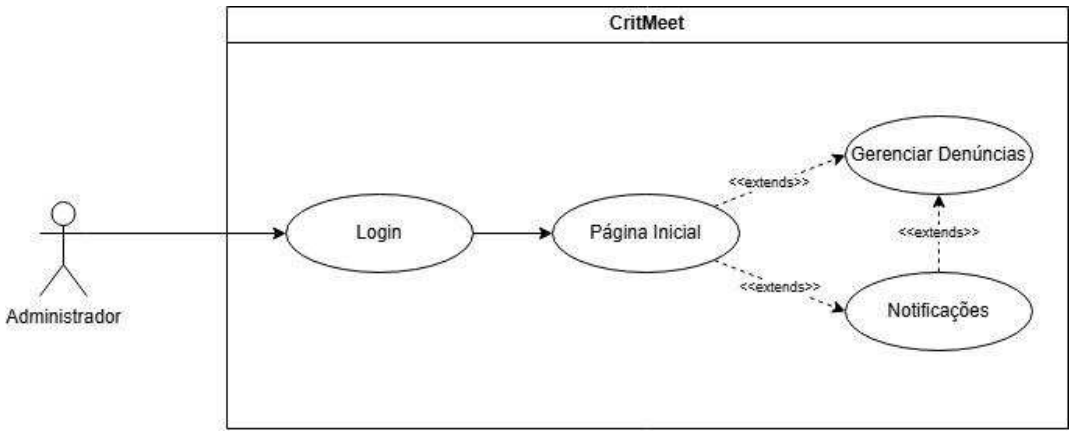
Figura 18 - Diagrama de Caso de Uso: Sair do grupo



Fonte: elaborado pelos autores.

A figura 18 apresenta o diagrama de caso de uso estendido de sair do grupo. Esse caso de uso permite que um jogador, enquanto participante de um grupo de RPG, se retire do grupo. Ao realizar essa ação, o jogador perde o acesso às interações e mensagens relacionadas ao grupo em questão.

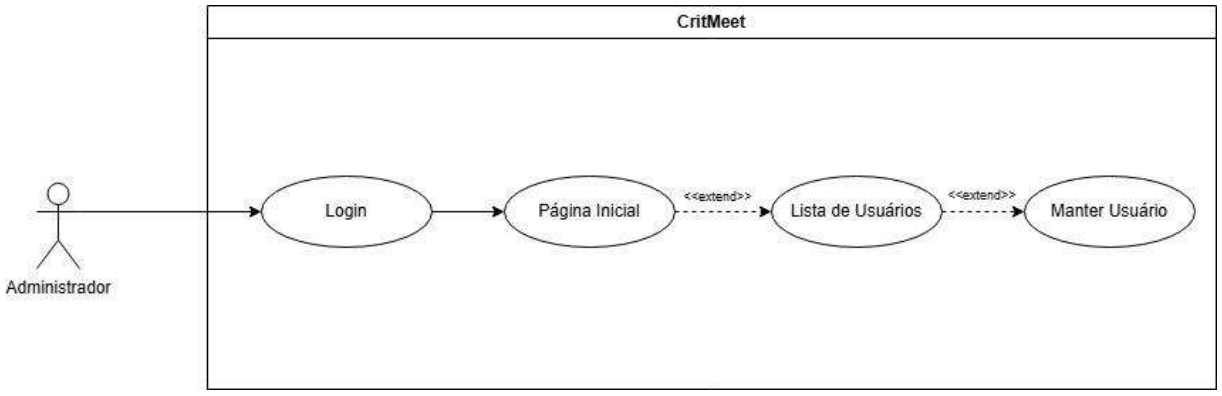
Figura 19 - Diagrama de Caso de Uso: Gerenciar denúncias



Fonte: elaborado pelos autores.

A figura 19 representa o caso de uso de gerenciar denúncias. Essa funcionalidade pode ser acessada tanto pela aba da página inicial quanto pela aba de notificações e permite que o administrador visualize e tome ações em relação às denúncias recebidas.

Figura 20 - Diagrama de Caso de Uso: Manter usuário



Fonte: elaborado pelos autores.

A figura 20 ilustra o caso de uso manter usuários, atribuído ao ator administrador. Esse caso de uso contempla as ações de gerenciamento dos perfis cadastrados na plataforma. Abaixo, encontra-se a especificação detalhada desse processo:

- **Lista de usuários:** funcionalidade que exibe todos os usuários cadastrados na plataforma, estando visível apenas para administradores da aplicação. Essa lista permite o administrador visualizar informações básicas dos perfis,

como nome, status e outras informações relevantes para o gerenciamento, facilitando a avaliação de perfis para execução de ações administrativas;

- **Manter usuários:** este caso de uso envolve as ações de gerenciamento dos perfis dos usuários realizadas pelo administrador do sistema. Inclui a capacidade de editar informações dos usuários, ativar ou desativar contas, redefinir senhas e excluir perfis, garantindo o controle e a manutenção adequada da base de usuários da plataforma CritMeet.

Essas descrições oferecem uma visão clara e detalhada das formas como os usuários podem interagir com o sistema, evidenciando as funcionalidades disponíveis para cada perfil e a dinâmica entre elas. Além disso, ao organizar e relacionar essas interações, é possível compreender melhor o fluxo operacional do CritMeet, garantindo que todas as ações estejam alinhadas aos objetivos da plataforma. Dessa forma, esse mapeamento contribui para um desenvolvimento mais estruturado e eficiente, além de facilitar futuras manutenções e aprimoramentos do sistema.

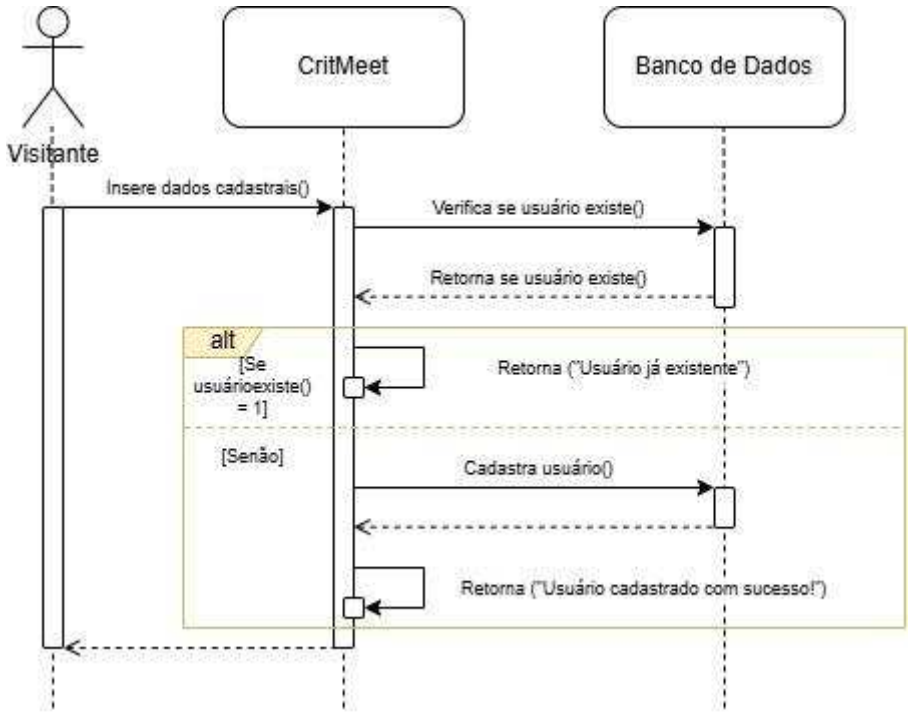
6.7.2 DIAGRAMAS DE SEQUÊNCIA

Conforme descrito no *site* do Lucidchart (2025), o diagrama de sequência é um dos tipos de diagramas UML utilizados para representar a interação entre objetos ou atores do sistema ao longo do tempo. Ele ilustra, de forma clara e sequencial, a troca de mensagens entre os elementos envolvidos durante a execução de um processo específico, funcionando como um roteiro visual que descreve o comportamento de um cenário dentro do sistema.

Esse tipo de diagrama é considerado dinâmico, pois foca na sequência de eventos que ocorrem dentro de um sistema. Cada linha vertical representa a interação de um objeto ou ator participante, e as setas horizontais indicam as mensagens ou chamadas de métodos trocadas entre eles. É muito útil para detalhar casos de uso, planejar fluxos de processos, identificar responsabilidades de objetos e compreender o funcionamento interno de sistemas complexos.

A seguir, estão listados os diagramas de sequência, que ilustram detalhadamente o fluxo das interações entre os elementos do sistema e usuários durante a execução dos processos, acompanhados de suas respectivas especificações:

Figura 21 - Diagrama de Sequência: Cadastrar usuário



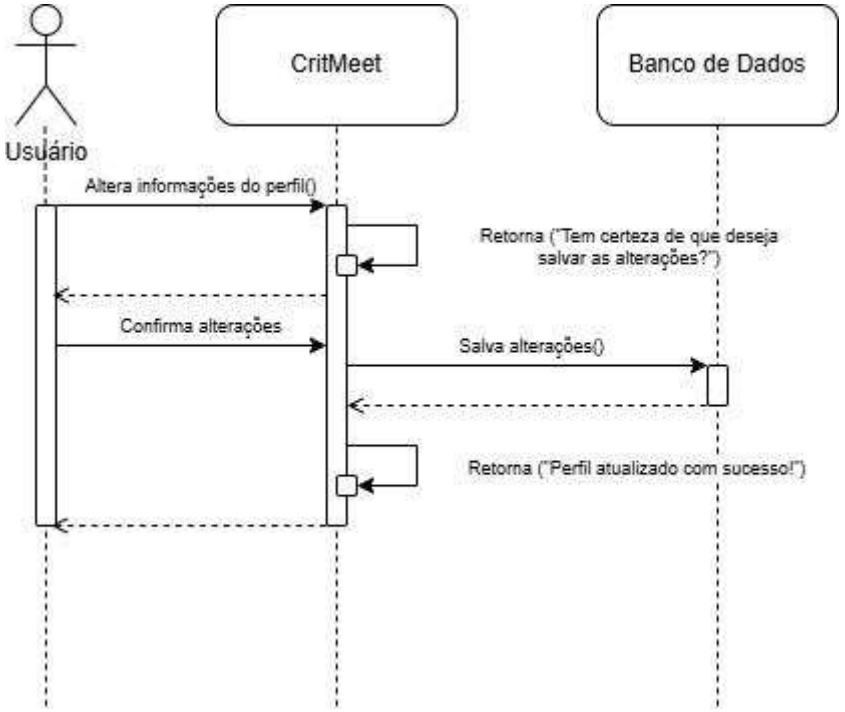
Fonte: elaborado pelos autores.

A figura 21 representa a interação de um visitante com a plataforma, onde ele insere seus dados cadastrais. Em seguida, o sistema verifica se informações como e-mail e nome de usuário já estão cadastrados. Caso exista um usuário com esses dados, a aplicação retorna a mensagem “Usuário já existente". Caso contrário, o sistema realiza o cadastro do usuário no banco de dados e retorna a confirmação “Usuário cadastrado com sucesso!”, notificando o visitante sobre a criação da sua conta no sistema.

O símbolo *alternative* (*alt*) do diagrama em questão costuma representar uma escolha entre duas ou mais sequências mutuamente exclusivas de mensagens. Ele é indicado por um retângulo com uma linha tracejada interna e rotulado como *alt* ou

alternative no canto superior, dividindo o espaço em áreas que correspondem a diferentes alternativas, cada uma com sua condição específica. Essa notação é usada para modelar decisões condicionais no fluxo de interação entre os elementos do sistema, mostrando que, dependendo da condição, apenas um dos caminhos será seguido.

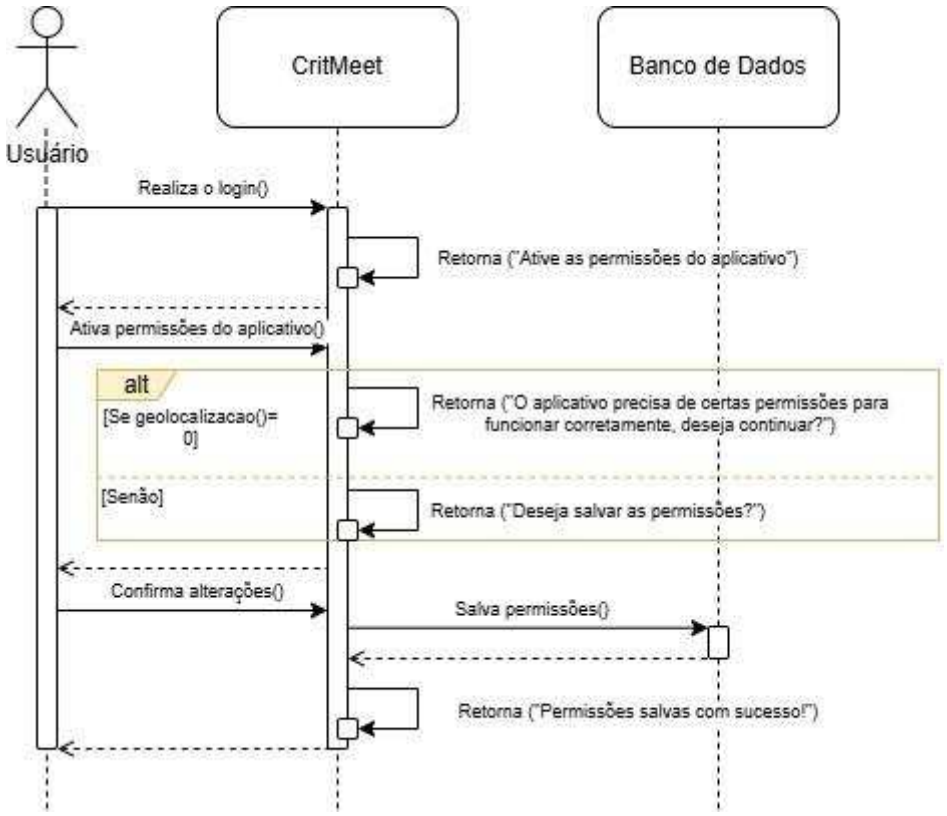
Figura 22 - Diagrama de Sequência: Editar perfil



Fonte: elaborado pelos autores.

A figura 22 apresenta um diagrama de sequência que representa a interação de um usuário ao editar seu próprio perfil. O usuário realiza as alterações desejadas em suas informações, e, ao finalizar, a plataforma exibe uma mensagem solicitando a confirmação para salvar as modificações. Após a confirmação, o sistema grava os novos dados no banco de dados e retorna uma notificação ao usuário informando que o perfil foi atualizado com sucesso.

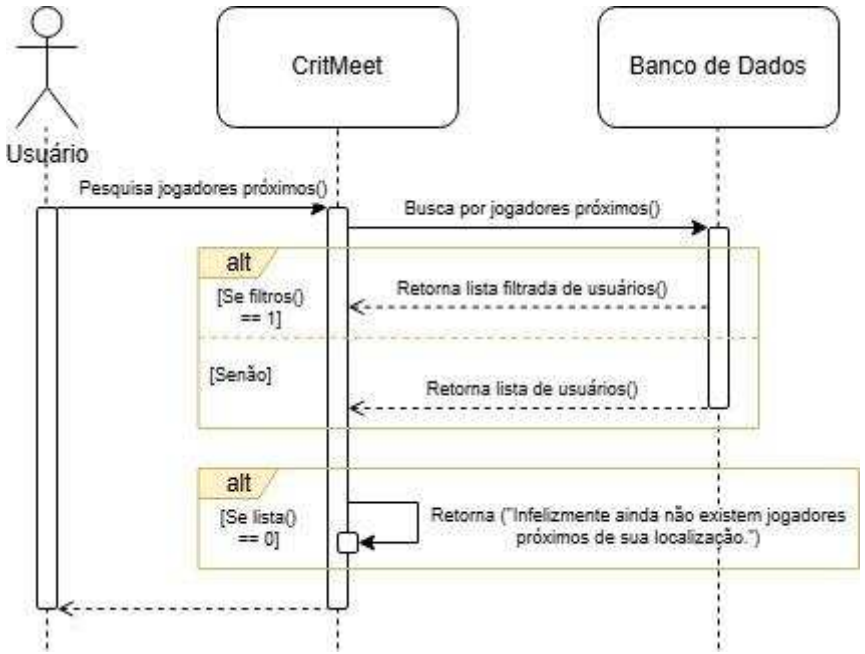
Figura 23 - Diagrama de Sequência: Registro de permissões



Fonte: elaborado pelos autores.

A figura 23 ilustra o processo de login de um usuário que não possui as permissões necessárias para o funcionamento adequado da plataforma. Após a tentativa de ativação das permissões, caso a permissão de geolocalização ainda esteja desativada, o sistema exibe uma mensagem alertando que certas permissões são indispensáveis para o uso correto do site. Quando todas as permissões são devidamente ativadas, o sistema retorna ao usuário se ele deseja salvar as alterações. Com a confirmação, as permissões são registradas no banco de dados, vinculadas ao respectivo usuário, e o sistema retorna uma notificação informando que as configurações foram salvas com sucesso.

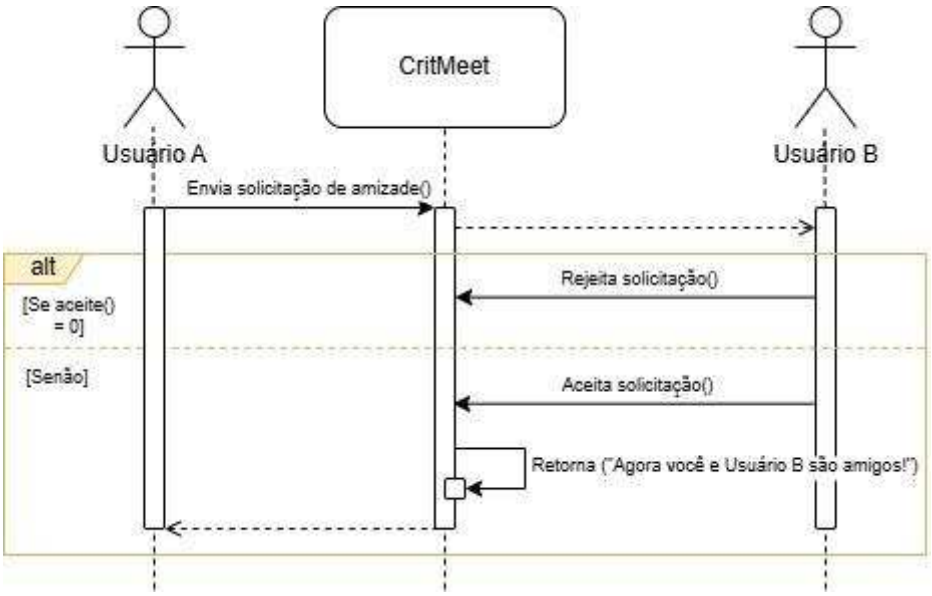
Figura 24 - Diagrama de Sequência: Pesquisar jogadores



Fonte: elaborado pelos autores.

A figura 24 traz o diagrama de sequência referente à busca de jogadores próximos, com ou sem a aplicação de filtros de pesquisa. O usuário inicia a busca, e o sistema verifica se há filtros ativos. Caso existam, a aplicação realiza a busca considerando os critérios definidos e retorna uma lista de jogadores que atendem aos filtros. Por outro lado, se não houver filtros ativos, a busca é realizada com base apenas na proximidade geográfica, retornando os jogadores mais próximos do usuário. Se nenhum jogador for encontrado, o sistema informa que ainda não existem usuários disponíveis na região. Observa-se que o diagrama utiliza dois blocos *alt*, um representando as decisões de busca com ou sem filtros, e outro para retorno caso não existam usuários nas proximidades.

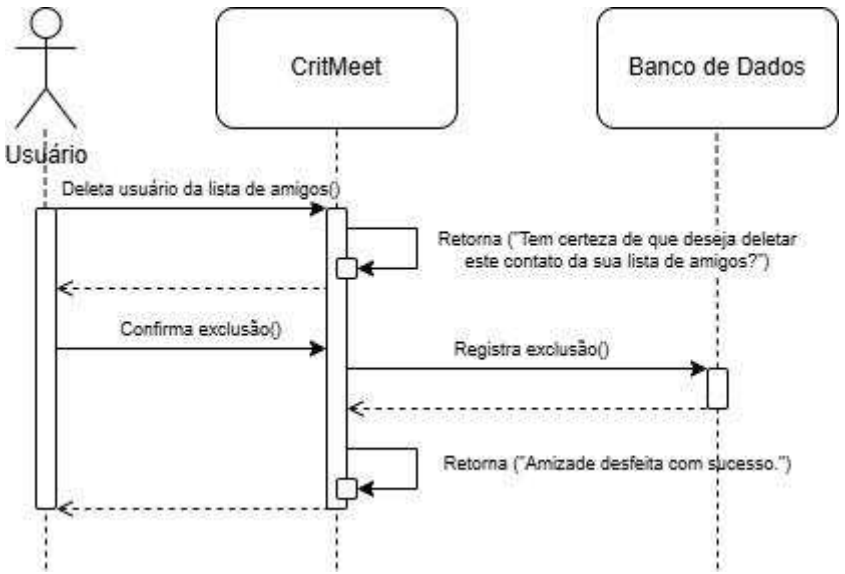
Figura 25 - Diagrama de Sequência: Adicionar à lista de amigos



Fonte: elaborado pelos autores.

A figura 25 demonstra o processo de adição de novas amizades à lista de amigos. Após o envio da solicitação por parte do usuário A, o usuário B pode optar por rejeitá-la, o que não exige um retorno de mensagem ao solicitante, ou aceitá-la. Caso a solicitação seja aceita, o sistema registra a nova amizade e notifica o usuário A de que a conexão foi estabelecida com sucesso, informando que ele e o usuário B agora são amigos.

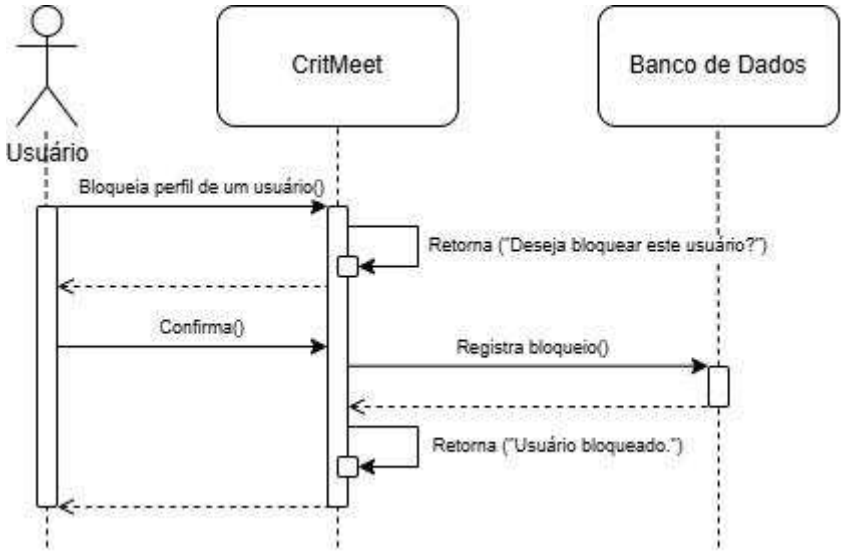
Figura 26 - Diagrama de Sequência: Desfazer amizade



Fonte: elaborado pelos autores.

A figura 26 se trata da remoção de um usuário da lista de amigos. Após selecionar a opção de deletar um contato, o sistema solicita a confirmação da ação por meio de uma mensagem. Com a confirmação, o sistema registra a exclusão no banco de dados e notifica o usuário informando que a amizade foi desfeita com sucesso.

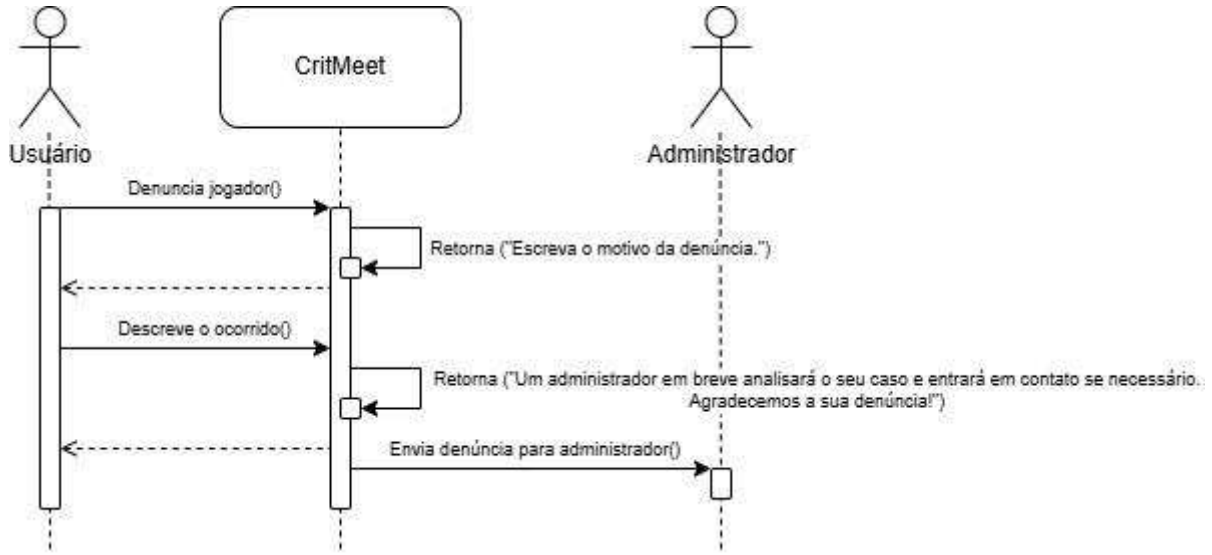
Figura 27 - Diagrama de Sequência: Bloquear perfil



Fonte: elaborado pelos autores.

A figura 27 apresenta a funcionalidade de bloqueio de um perfil na plataforma. Após selecionar a conta que deseja bloquear, o usuário recebe uma mensagem do sistema solicitando a confirmação da ação. Com a confirmação, o sistema registra o bloqueio no banco de dados e notifica o usuário de que o perfil foi bloqueado com sucesso. A partir desse momento, ambos os usuários ficam impedidos de trocar mensagens entre si.

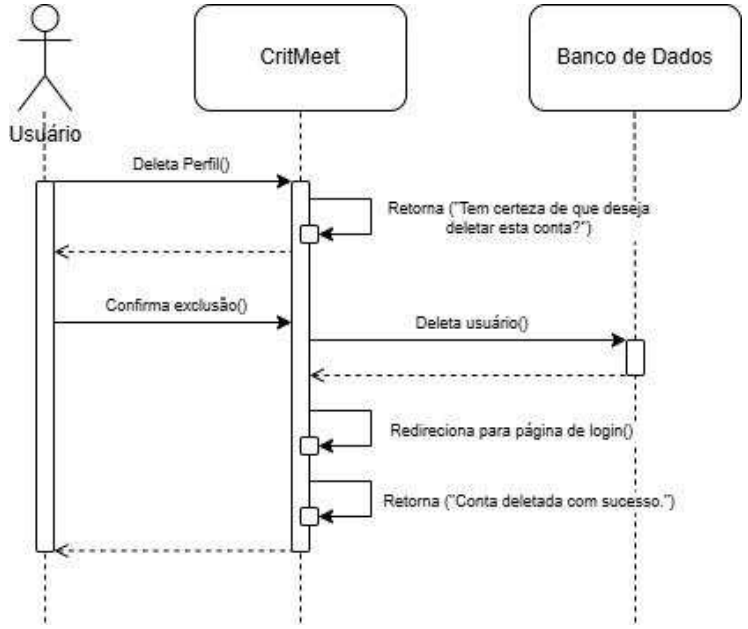
Figura 28 - Diagrama de Sequência: Denunciar perfil



Fonte: elaborado pelos autores.

A figura 28 representa o processo de denúncia de um jogador na plataforma. Ao realizar a denúncia, o sistema solicita que o usuário informe o motivo, descrevendo o ocorrido em um campo específico. Após o envio, a aplicação exibe uma mensagem informando que a denúncia será avaliada por um administrador, que entrará em contato caso necessário, e encaminha o relato para análise administrativa.

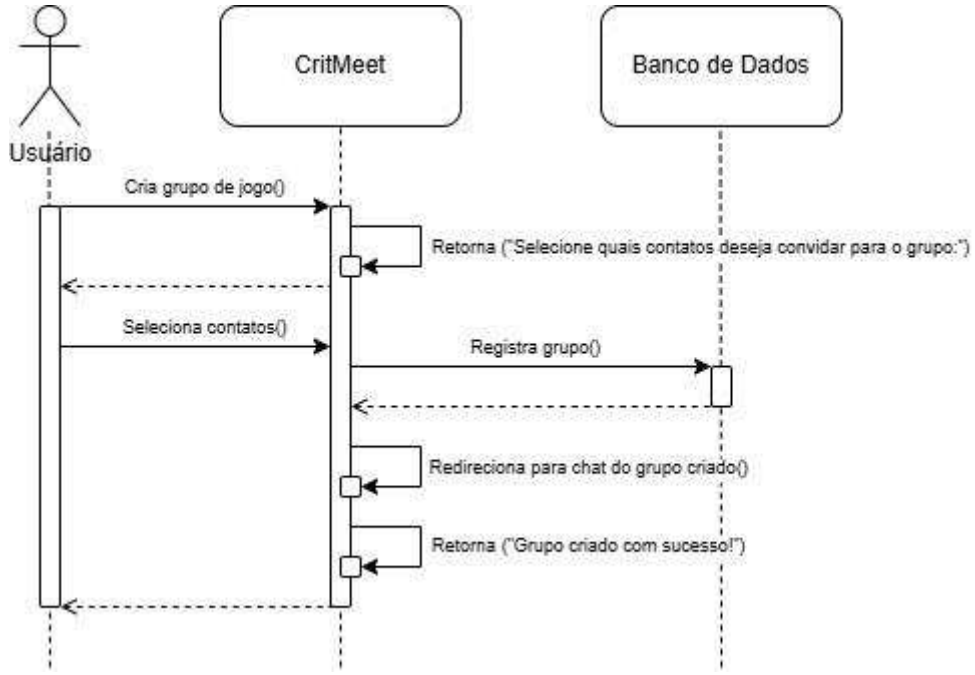
Figura 29 - Diagrama de Sequência: Deletar perfil



Fonte: elaborado pelos autores.

A figura 29 ilustra a ação de exclusão da própria conta na plataforma. Ao selecionar essa opção, o sistema solicita a confirmação do usuário, perguntando se ele realmente deseja deletar a conta. Após a confirmação, a conta é removida do banco de dados, e a plataforma redireciona o usuário para a página de *login*, exibindo uma mensagem de que a exclusão foi concluída com sucesso.

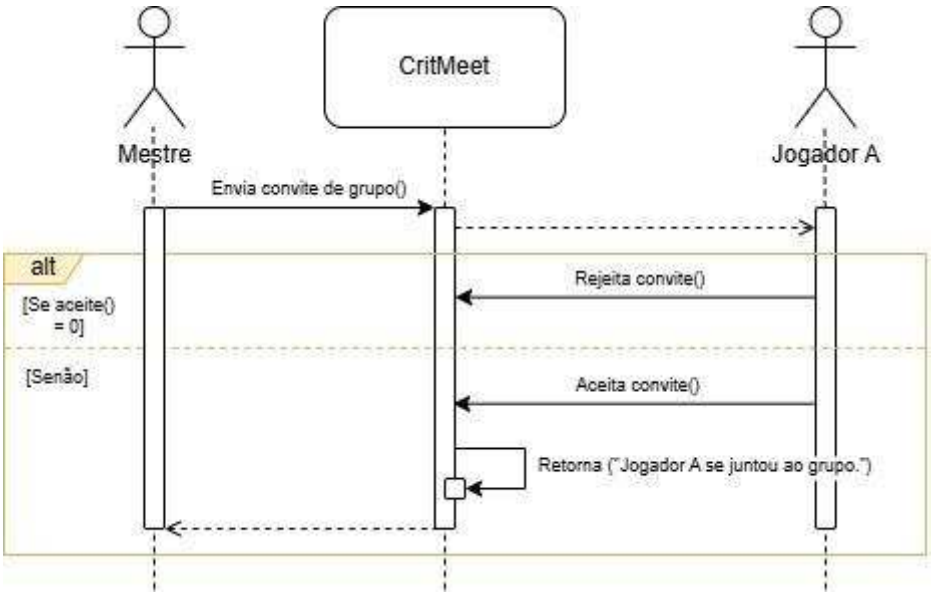
Figura 30 - Diagrama de Sequência: Criar grupo



Fonte: elaborado pelos autores.

A figura 30 demonstra a criação de um grupo de jogo na plataforma. Após iniciar o processo de criação, o usuário seleciona os contatos que deseja adicionar ao grupo. Com a seleção concluída, o sistema registra o novo grupo no banco de dados, redireciona o usuário para o *chat* correspondente e exibe uma mensagem confirmando que o grupo foi criado com sucesso.

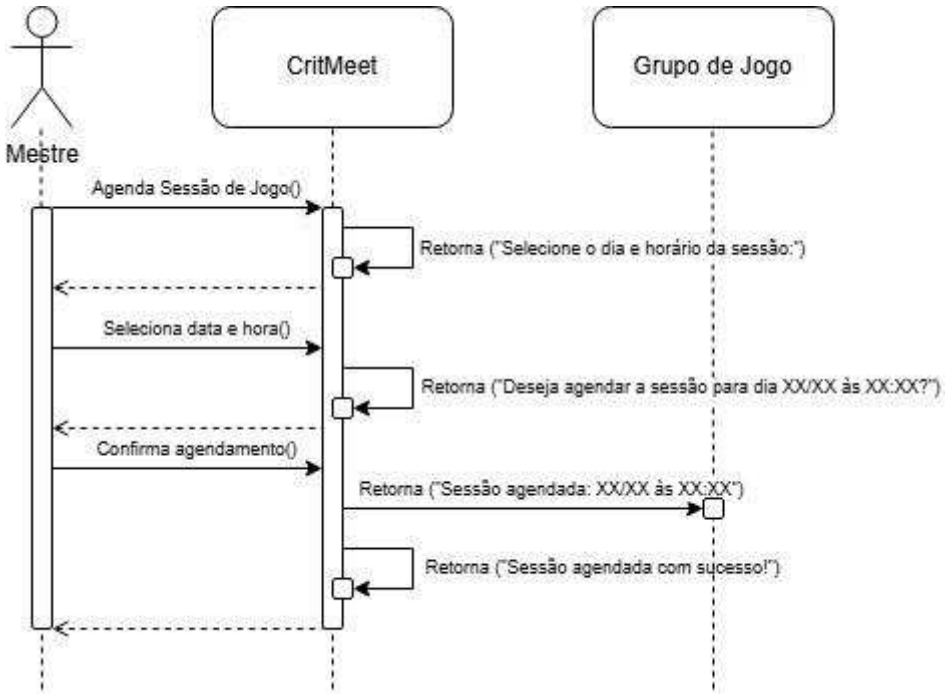
Figura 31 - Diagrama de Sequência: Convidar para grupo



Fonte: elaborado pelos autores.

A figura 31 se trata do diagrama de sequência referente ao convite para que um jogador participe de um grupo. Nesse processo, o mestre de jogo envia um convite ao jogador. Assim como na figura 24, caso o convite seja rejeitado, não há notificação ao mestre. No entanto, se o convite for aceito, o sistema registra a participação do jogador no grupo e informa ao mestre de jogo que o jogador se juntou com sucesso.

Figura 32 - Diagrama de Sequência: Agendar sessão



Fonte: elaborado pelos autores.

A figura 32 apresenta o diagrama de sequência relacionado ao agendamento de uma sessão de jogo. O mestre de jogo inicia o agendamento informando a data e o horário desejados. Em seguida, o sistema solicita a confirmação da ação. Após a confirmação, o sistema registra o agendamento, envia uma notificação ao grupo com as informações da sessão e informa ao mestre que a sessão foi agendada com sucesso.

Com isso, os diagramas de sequência descritos fornecem uma visão clara e detalhada das principais interações entre os usuários e o sistema da plataforma. Cada diagrama evidencia os fluxos de comunicação e as decisões envolvidas em diferentes funcionalidades, contribuindo para a compreensão do comportamento da aplicação em tempo de execução. Esses modelos são essenciais para validar o design do sistema, identificar possíveis melhorias e garantir que os requisitos funcionais sejam devidamente atendidos.

6.7.3 DIAGRAMA DE CLASSES

Com base no artigo do GoUML, um diagrama de classes é definido como um tipo de diagrama estrutural estático da UML, sendo amplamente utilizado no desenvolvimento de *software* para representar graficamente a estrutura de um sistema. Esse tipo de diagrama descreve as classes que compõem o sistema, incluindo seus atributos, métodos e os relacionamentos entre os objetos, sendo indispensável na modelagem de aplicações orientadas a objetos. Além disso, o diagrama pode incluir informações de multiplicidade, indicando quantas instâncias de uma classe podem estar vinculadas a outra (GOUML, 2025).

Dessa forma, os diagramas de classes são ferramentas essenciais para garantir uma melhor compreensão, documentação e implementação de sistemas complexos, oferecendo uma visão clara e estruturada da arquitetura interna do sistema, facilitando na visualização das entidades principais, suas características e as interações que ocorrem entre elas, o que contribui para um *design* mais organizado e coerente.

Para o desenvolvimento das classes do sistema CritMeet, foram considerados alguns dos elementos previamente definidos, os quais serviram como base para estruturar as responsabilidades e interações dos principais componentes da aplicação. No entanto, durante a transposição para o diagrama de classes, foi necessário realizar adaptações conceituais e adicionar novas classes que refletissem com maior precisão a lógica interna do sistema e suas particularidades:

- **Visitante:** representa o indivíduo que acessa a plataforma sem realizar autenticação. Essa classe é responsável por modelar as funções básicas que antecedem o registro do usuário;

- **Jogador:** representa um usuário autenticado dentro da plataforma. Seus atributos e funções refletem as ações específicas desse perfil. Essa é uma das classes principais do sistema, estabelecendo relações diretas com as demais classes relacionadas à experiência de jogo colaborativo;

- **Mestre:** esta classe herda os atributos da classe jogador, mas com permissões adicionais. Essa entidade representa o usuário que cria e gerencia sessões de RPG, sendo responsável por moderar, configurar e conduzir um grupo;

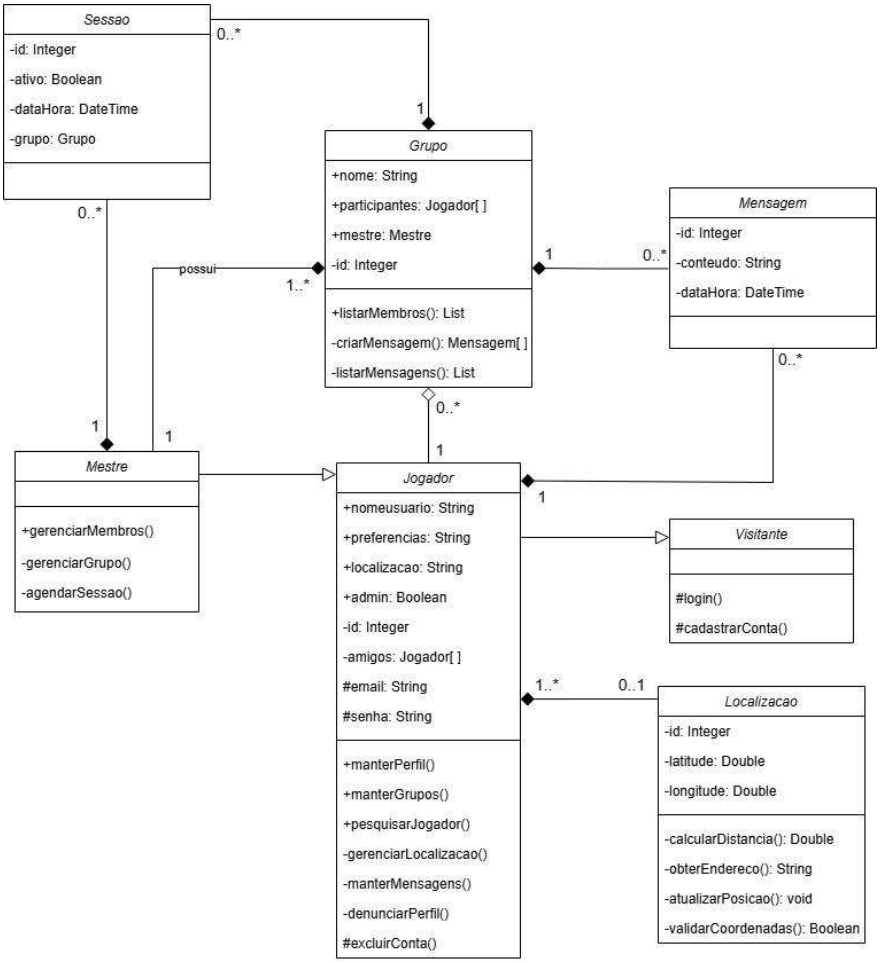
- **Grupo:** representa uma unidade social formada por jogadores. Ela abstrai a ideia de coletivo organizado, reunindo usuários com interesse comum em jogar RPG com determinada frequência, dentro de um mesmo sistema de jogo. Essa classe facilita a organização e a persistência de estruturas colaborativas dentro da plataforma;

- **Sessão:** toda sessão está associada a um único grupo e a múltiplos. A classe pode conter atributos como data, horário e status, sendo essencial para a dinâmica de um grupo, visto que a maior parte das interações entre os usuários ocorrem por meio dela;

- **Mensagem:** modela as comunicações textuais trocadas tanto entre os usuários quanto dentro de um grupo. Trata-se de uma classe fundamental para viabilizar a interação entre os usuários, registrando conteúdo textual, data e horário. Esta classe permite a flexibilidade no uso da comunicação escrita como parte da experiência da aplicação.

Assim como os diagramas de caso de uso e os diagramas de sequência, o diagrama de classes também foi elaborado por meio da ferramenta Draw.io. A representação gráfica a seguir, permite visualizar de forma estruturada a arquitetura lógica do sistema, evidenciando o papel e a responsabilidade de cada classe no contexto da aplicação:

Figura 33 - Diagrama de Classes



Fonte: elaborado pelos autores

6.8 MODELO ENTIDADE-RELACIONAMENTO

De acordo com o ScienceDirect (2025) os modelos entidade-relacionamento (MER) são comumente utilizados na modelagem conceitual de dados e tem como principal finalidade representar graficamente a estrutura lógica de um banco de dados. Esse modelo descreve o sistema por meio de entidades, atributos e relacionamentos, permitindo a identificação clara de objetos relevantes do domínio, suas propriedades e as interações entre eles. Essas classes são tradicionalmente representadas por caixas retangulares, com cantos retos ou arredondados, dependendo da variação do modelo. Os atributos, por sua vez, são identificadores ou propriedades dessas entidades e podem ser listados dentro das caixas das entidades ou separadamente, conforme a notação utilizada. Já os relacionamentos descrevem associações entre instâncias de classes de entidades e são

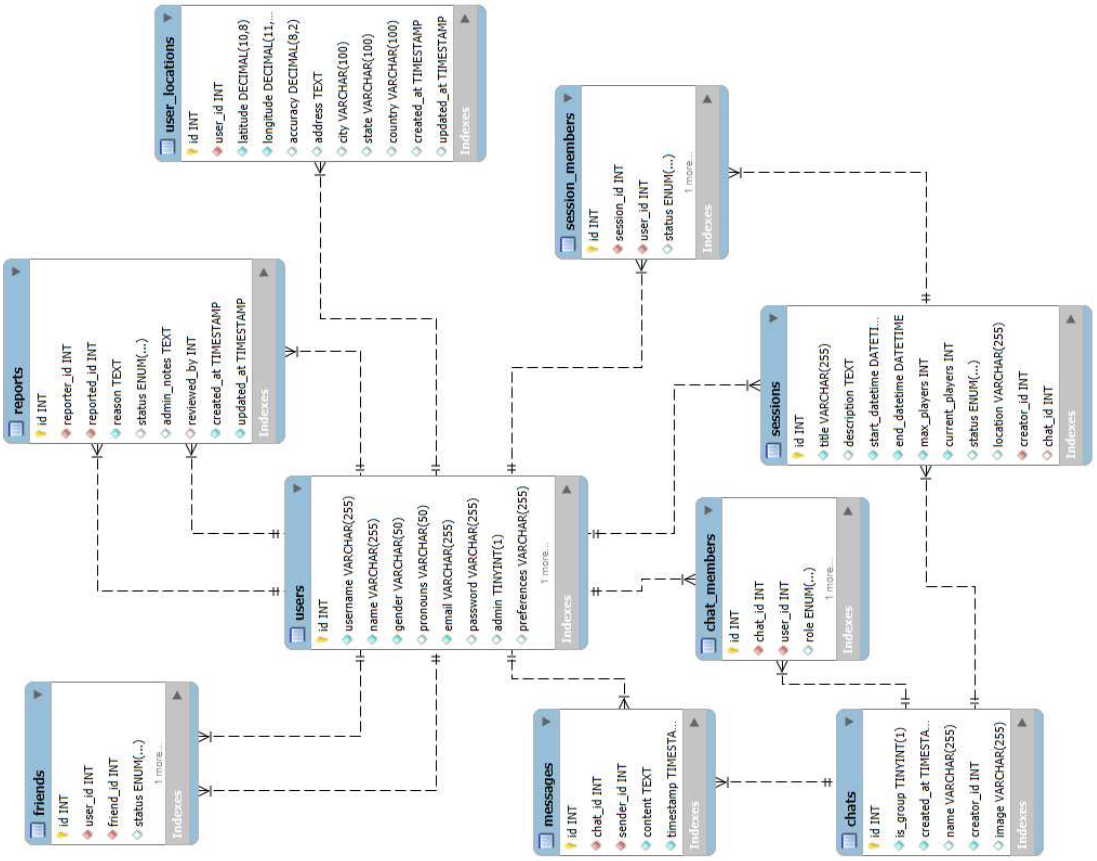
representados por linhas conectando as entidades envolvidas (SCIENCEDIRECT, 2025).

Segundo o site Escola DNC, a cardinalidade é um conceito essencial na modelagem de dados que define o número de instâncias de uma entidade que podem se relacionar com instâncias de outra entidade em um banco de dados relacional (ESCOLADNC, 2025). Os principais tipos de cardinalidade são:

- **Um-para-Um (1:1):** representa um relacionamento em que cada instância de uma entidade está associada a uma única instância de outra entidade, estabelecendo uma correspondência exclusiva entre elas;
- **Um-para-Muitos (1:N):** caracteriza um vínculo em que uma instância de uma entidade pode se relacionar com múltiplas instâncias de outra entidade, configurando uma relação hierárquica ou de agregação;
- **Muitos-para-Muitos (N:N):** define um relacionamento no qual múltiplas instâncias de uma entidade podem estar associadas a múltiplas instâncias de outra entidade, implicando uma associação bidirecional complexa.

Para o desenvolvimento do modelo, utilizou-se a ferramenta MySQL Workbench, que oferece recursos tanto para a modelagem de dados quanto para o desenvolvimento do banco de dados. Trata-se de um *software* amplamente utilizado no meio profissional por sua versatilidade e interface intuitiva. A seguir, é exibido o modelo entidade-relacionamento elaborado na ferramenta.

Figura 34 - Modelo Entidade-relacionamento



Fonte: elaborado pelos autores.

Segundo o *site* TechTarget, um dicionário de dados é uma coleção centralizada de descrições dos objetos ou itens de dados em um modelo, funcionando como um repositório de metadados para desenvolvedores e profissionais. Ele inclui nomes de objetos, tipos de dados, valores padrão possíveis e descrições resumidas. O dicionário facilita a compreensão dos dados e seus relacionamentos, sendo fundamental para a padronização e comunicação entre as equipes (CHAI, 2022). A seguir, apresentam-se os dicionários de dados utilizados no desenvolvimento do sistema.

Tabela 3 - Dicionário de dados da tabela de usuários

Tabela	users
Descrição	Armazena todos os usuários cadastrados

Observações	Possui chave estrangeira referenciada por múltiplas outras tabelas.O atributo **'password'** é criptografado usando bcrypt para segurança.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação único do usuário	Int	11	PK/Identity/Auto Increment
username	Login do usuário	Varchar	255	Not Null
name	Nome completo do usuário dentro da plataforma	Varchar	255	Not Null
gender	Gênero do usuário	Varchar	50	Not Null
pronouns	Pronomes do usuário	Varchar	50	Nullable
email	E-mail registrado pelo usuário	Varchar	255	Not Null/Unique
password	Senha criptografada do usuário (bcrypt)	Varchar	255	Not Null
admin	Indica se o usuário é administrador	Tinyint	1	Default 0
preferences	Preferências de jogo do usuário	Varchar	255	Nullable
image	Imagem de perfil do usuário	Text	-	Nullable

Tabela 4 - Dicionário de dados da tabela de chats

Tabela	chats			
Descrição	Armazena informações sobre as salas de chat/conversas			
Observações	Suporta tanto chats individuais quanto em grupo. O campo 'creator_id' não possui restrição de chave estrangeira definida no banco, mas referencia logicamente a tabela users.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação único do chat	Int	11	PK/Identity/Auto Increment
is_group	Indica se é um chat em grupo ou individual	TinyInt	1	Not Null/Default 0
created_at	Data e hora de criação do chat	Timestamp	-	Default CURRENT_TIMESTAMP
name	Nome do chat (usado para grupos)	Varchar	255	Nullable
creator_id	ID do usuário que criou o chat	Int	11	Not Null
Image	Imagem do chat/grupo	Varchar	255	Nullable

Tabela 5 - Dicionário de dados da tabela de membros de chats

Tabela	chat_members

Descrição	Relaciona usuários aos chats dos quais fazem parte			
Observações	Tabela de relacionamento N:N entre **‘users’** e **‘chats’**. Permite que um usuário participe de múltiplos chats e um chat tenha múltiplos participantes.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação único do membro	int	11	PK/Identity/Auto Increment
chat_id	ID do chat ao qual o usuário pertence	Int	11	FK para chats(id)/Not Null
user_id	ID do usuário membro do chat	ID do usuário membro do chat	11	FK para users(id)/Not Null
role	Papel do usuário no chat	Enum	-	Values: 'member', 'admin'/Default 'member'
last_seen	Última vez que o usuário visualizou o chat	Timestamp	-	Nullable

Tabela 6 - Dicionário de dados da tabela de amigos

Tabela	friends
Descrição	Gerencia as relações de amizade entre usuários
Observações	Permite relacionamento N:N entre usuários com controle de status da solicitação de amizade.

CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação único da amizade	Int	11	PK/Identity/Auto Increment
user_id	ID do usuário que iniciou a amizade	Int	11	FK para users(id)/Not Null
friend_id	ID do usuário que recebeu a solicitação	Int	11	FK para users(id)/Not Null
status	Status da solicitação de amizade	Enum	-	Values: 'pending', 'accepted'/Default 'pending'
created_at	Data e hora da solicitação	Timestamp	-	Default CURRENT_TIMESTAMP

Tabela 7 - Dicionário de dados da tabela de mensagens

Tabela	messages			
Descrição	Armazena todas as mensagens enviadas nos chats			
Observações	Relaciona-se com as tabelas **‘chats’** e **‘users’** para rastrear conversas e remetentes das mensagens.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação	Int	11	PK/Identity/Aut

	único da mensagem			o Increment
chat_id	ID do chat onde a mensagem foi enviada	Int	11	FK para users(id)/Not Null
sender_id	ID do usuário que enviou a mensagem	Int	11	FK para chats(id)/Not Null
content	Conteúdo da mensagem	Text	-	Not Null
timestamp	Data e hora do envio da mensagem	Timestamp	-	Default CURRENT_TIMESTAMP

Tabela 8 - Dicionário de dados da tabela de sessões

Tabela	sessions			
Descrição	Armazena informações sobre sessões de jogos/encontros organizadas pelos usuários			
Observações	Representa eventos ou sessões de RPG organizadas pelos usuários. Pode estar vinculada a um chat específico e possui controle de participantes.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação único da sessão	Int	11	PK/Identity/Auto Increment
title	Título da sessão	Varchar	255	Not Null

description	Descrição detalhada da sessão	Text	-	Nullable
start_datetime	Data e hora de início da sessão	Datetime	-	Not Null
end_datetime	Data e hora de fim da sessão	Datetime	-	Not Null
max_players	Número máximo de jogadores permitidos	Int	11	Not Null/Default 4
current_players	Número atual de jogadores confirmados	Int	11	Not Null/Default 1
status	Status atual da sessão	Enum	-	Values: 'active', 'full', 'cancelled', 'completed'/Default 'active'
location	Local onde a sessão será realizada	Varchar	255	Nullable
creator_id	ID do usuário que criou a sessão	Int	11	FK para users(id)/Not Null
chat_id	ID do chat associado à sessão	Int	11	FK para chats(id)/Not Null

Tabela 9 - Dicionário de dados da tabela de membros de sessões

Tabela	sessions_members
Descrição	Relaciona usuários às sessões das quais participam ou foram convidados

Observações	Tabela de relacionamento N:N entre 'users' e 'sessions'. Controla os participantes de cada sessão e seus status de participação.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação único do membro da sessão	Int	11	PK/Identity/Auto Increment
session_id	ID da sessão	Int	11	FK para sessions(id)/Not Null
user_id	ID do usuário participante	Int	11	FK para users(id)/Not Null
status	Status da participação do usuário	Enum	-	Values: 'pending', 'accepted', 'declined'/Default 'pending'
joined_at	Data e hora em que o usuário se juntou/foi convidado	Timestamp	-	Default CURRENT_TIMESTAMP

Tabela 10 - Dicionário de dados da tabela de localização do usuário

Tabela	user_locations
Descrição	Armazena informações de localização geográfica dos usuários
Observações	Permite rastreamento de localização dos usuários para funcionalidades baseadas em proximidade geográfica. Inclui

	dados de precisão e endereçamento.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação único da localização	Int	11	PK/Identity/Auto Increment
user_id	ID do usuário proprietário da localização	Int	11	FK para users(id)/Not Null
latitude	Coordenada de latitude	Decimal	10,8	Not Null
longitude	Coordenada de longitude	Decimal	11,8	Not Null
accuracy	Precisão da localização em metros	Decimal	8,2	Nullable
address	Endereço completo da localização	Text	-	Nullable
city	Cidade da localização	Varchar	100	Nullable
state	Estado/Província da localização	Varchar	100	Nullable
country	País da localização	Varchar	100	Nullable
created_at	Data e hora de criação do registro	Timestamp	-	Default CURRENT_TIMESTAMP

updated_at	Data e hora da última atualização	Timestamp	-	Default CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP

Tabela 11 - Dicionário de dados da tabela de denúncias

Tabela	Reports			
Descrição	Armazena denúncias/relatórios feitos por usuários contra outros usuários da plataforma			
Observações	Tabela para controle de moderação que permite aos usuários reportarem comportamentos inadequados. Possui relacionamento com a tabela 'users' tanto para o denunciante quanto para o denunciado e o administrador responsável pela análise.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**RESTRIÇÕES**
id	Código de identificação único do report	Int	11	PK/Identity/Auto Increment
reporter_id	ID do usuário que fez a denúncia	Int	11	FK para users(id)/Not Null
reported_id	ID do usuário que foi denunciado	Int	11	FK para users(id)/Not Null
reason	Motivo/descrição da denúncia	Text	-	Not Null
status	Status atual da denúncia	Enum	-	Values: 'pending', 'reviewed',

				'resolved', 'dismissed'/Default 'pending'
admin_notes	Observações do administrador sobre a análise	Text	-	Nullable
reviewed_by	ID do administrador que analisou a denúncia	Int	11	FK para users(id)/Nullable
created_at	Data e hora da criação da denúncia	Timestamp	-	Default CURRENT_TIMESTAMP
updated_at	Data e hora da última atualização	Timestamp	-	Default CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP

Tabela 12 - Dicionário de dados da tabela de visualização de sessões com chats

Visualização	v_chat_sessions			
Descrição	View que combina informações de sessões com dados relacionados de chats, usuários e membros			
Observações	Facilita consultas complexas que necessitam de dados agregados de múltiplas tabelas. Calcula automaticamente o status atual da sessão baseado em regras de negócio.			
CAMPOS				
NOME	**DESCRIÇÃO**	**TIPO DE DADO**	**TAMANHO**	**ORIGEM**

id	ID da sessão	Int		sessions.id
title	Título da sessão	Varchar		sessions.title
description	Descrição da sessão	Datetime		sessions.description
start_datetime	Data/hora de início	Datetime		sessions.start_datetime
end_datetime	Data/hora de fim	Decimal		sessions.end_datetime
max_players	Máximo de jogadores	Int		sessions.max_players
status	Status original da sessão	Enum		sessions.status
location	Local da sessão	Varchar		sessions.location
creator_id	ID do criador	Int		sessions.creator_id
chat_id	ID do chat associado	Int		sessions.chat_id
chat_name	Nome do chat	Varchar		chats.name
is_group	Indica se o chat é grupo	TinyInt		chats.is_group
creator_name	Nome do criador	Varchar		users.name
creator_username	Username do criador	Varchar		users.username
current_players	Número atual de jogadores	Int		COUNT(session_members)

	aceitos			
calculated_status	Status calculado baseado em regras	Varchar		Calculado

7 TECNOLOGIAS E FERRAMENTAS

Este capítulo tem como objetivo apresentar e detalhar as principais tecnologias empregadas durante o desenvolvimento do projeto, abordando ferramentas utilizadas desde a modelagem até a implementação, demonstrando como cada uma delas contribuiu para o projeto.

7.1 FIGMA

O Figma é uma ferramenta de *design* gráfico que se destaca por sua proposta centrada na colaboração em tempo real e na acessibilidade. Lançado em 2015, o Figma uniu as funcionalidades típicas de *softwares* nativos à flexibilidade do uso via navegador, permitindo que diferentes usuários trabalhem simultaneamente em um mesmo projeto, contribuindo para a agilidade dos fluxos de trabalho, a redução de problemas relacionados à versão de arquivos e a melhoria na comunicação entre membros da equipe de *design* (FIGMA, 2025).

Além disso, sua abordagem baseada em nuvem facilita o acompanhamento de projetos e o compartilhamento de informações, tornando o processo de criação mais fluido e integrado. Com uma interface orientada à prototipagem e ao *design* de interfaces digitais, o Figma se consolidou como uma das ferramentas mais utilizadas no planejamento visual de aplicações e sistemas interativos (FIGMA, 2025).

No contexto deste projeto, o Figma foi utilizado para a construção do protótipo de baixa fidelidade, servindo como base para o planejamento visual inicial da aplicação. Sua escolha se deu principalmente pela praticidade de uso diretamente no navegador, o que eliminou a necessidade de instalação de *softwares* adicionais e facilitou o acesso por todos os integrantes da equipe. A funcionalidade colaborativa da ferramenta foi essencial nesse processo, permitindo a construção conjunta das telas e facilitando a troca de ideias em tempo real, de modo a validar o primeiro design do sistema, possibilitando ajustes rápidos e eficientes antes da evolução para protótipos mais detalhados.

Figura 35 - Figma

Fonte: Figma, 2025.

7.2 DRAW.IO

O Draw.io é uma ferramenta *online* voltada à criação de diagramas de forma prática e acessível. Por operar diretamente no navegador, assim como o Figma, elimina a necessidade de instalações e oferece ao usuário suporte a uma grande variedade de modelos gráficos pré-prontos, como fluxogramas, diagramas de casos de uso, diagramas de classes, entre outros. A flexibilidade da plataforma permite que o usuário modele estruturas complexas com facilidade, utilizando uma biblioteca ampla de formas e conectores.

Outro diferencial do Draw.io está na sua capacidade de integrar-se com serviços de armazenamento em nuvem, como Google Drive e Dropbox, o que favorece o trabalho colaborativo e o versionamento de arquivos. A ferramenta também se destaca por seu compromisso com a acessibilidade e transparência dos dados, permitindo que os arquivos gerados possam ser exportados e reutilizados em outros ambientes.

A equipe optou pelo Draw.io devido à sua praticidade e fácil acesso via navegador, eliminando a necessidade de instalar programas adicionais. A ferramenta oferece uma interface intuitiva e uma ampla variedade de recursos que facilitaram a elaboração dos diagramas do projeto, tornando o processo mais ágil e eficiente.

Figura 36 - Draw.io

Fonte: Draw.io, 2025.

7.3 VISUAL STUDIO CODE

O Visual Studio Code (VSCode) é um ambiente de desenvolvimento integrado (IDE) gratuito e de código aberto, desenvolvido pela Microsoft, que se consolidou como uma das ferramentas mais utilizadas por programadores em todo o mundo. Reconhecido por sua leveza, rapidez e versatilidade, o VSCode oferece uma experiência fluida tanto para iniciantes quanto para desenvolvedores experientes. Seu principal diferencial está na capacidade de fornecer recursos avançados em uma interface simples e intuitiva, o que torna o processo de codificação mais produtivo e acessível. A compatibilidade com diversas linguagens de programação, como HTML, CSS, JavaScript, PHP, Python, entre outras, faz do VSCode uma IDE extremamente versátil para projetos de diferentes naturezas (VISUALSTUDIOCODE, 2025).

Durante o desenvolvimento do projeto CritMeet, o VSCode foi a principal ferramenta utilizada para escrever, organizar e editar o código-fonte do sistema, tanto no *front-end* quanto no *back-end*. Entre os principais recursos utilizados no VSCode durante o desenvolvimento do CritMeet, destacam-se o realce de sintaxe (*syntax highlighting*) e o autocompletar inteligente, que facilitaram a leitura e a escrita do código em linguagens como HTML, CSS, JavaScript e PHP, tornando o processo mais rápido e preciso. A equipe também fez uso de diversas extensões

específicas, voltadas para PHP, SQL, Bootstrap, além de ferramentas para validação de código, formatação automática e integração com o GitHub, o que contribuiu diretamente para a organização e qualidade do projeto. O terminal integrado da IDE possibilitou a execução de comandos no servidor local, testes de funcionalidades e acompanhamento de logs, tudo dentro do próprio ambiente de desenvolvimento. Além disso, o suporte ao controle de versão Git permitiu que os membros da equipe monitorassem alterações no código, realizassem commits e resolvessem conflitos de forma prática e eficiente, sem precisar sair da interface do editor.

Figura 37 - Visual Studio Code

Fonte: Visual Studio Code, 2025.

7.4 MYSQL WORKBENCH

O MySQL Workbench é uma ferramenta visual unificada voltada a arquitetos de banco de dados, desenvolvedores e administradores de banco de dados, cuja funcionalidade principal é a de oferecer uma interface gráfica que abrange desde o projeto visual de bancos de dados relacionais até a modelagem de dados, possibilitando a criação e manipulação de modelos entidade-relacionamento, bem como a engenharia direta e reversa de esquemas. A ferramenta inclui também recursos para desenvolvimento SQL, um painel visual para configuração de servidores, gerenciamento de usuários, entre outras funcionalidades. Ainda, suporta migração de dados e estruturas de outros sistemas gerenciadores de banco de dados relacionais, facilitando a transição para o ecossistema MySQL (MYSQL, 2025).

No desenvolvimento do CritMeet, o MySQL Workbench foi utilizado como

ferramenta central para a construção do modelo entidade-relacionamento (MER) do sistema, possibilitando a representação gráfica da estrutura de dados. A escolha dessa ferramenta se justifica por sua capacidade de facilitar a modelagem visual dos componentes do banco de dados, permitindo mapear entidades, atributos e relacionamentos de maneira clara e consistente, contribuindo para a documentação do banco de dados.

Figura 38 - MySQL Workbench

Fonte: Code4Pi, 2018.

7.5 MYSQL

O MySQL é um banco de dados relacional de código aberto utilizado para armazenamento e gerenciamento de banco de dados. Ele armazena dados em tabelas organizadas por esquemas e usa a linguagem SQL para consultar, atualizar e gerenciar esses dados. Graças ao suporte a transações ACID, o MySQL garante integridade mesmo em situações críticas, sendo usado tanto em projetos pessoais quanto em grandes aplicações como (ERICKSON, 2024).

Por ser *open source*, o MySQL conta com uma comunidade ativa que contribui com melhorias, correções e recursos de aprendizado. Seu suporte a JSON, alta compatibilidade com linguagens de programação e facilidade de uso o tornam uma escolha versátil e econômica para diferentes cenários. O sistema pode lidar com altos volumes de dados e conexões simultâneas, mantendo desempenho e

segurança mesmo em ambientes exigentes (ERICKSON, 2024).

No projeto CritMeet, utilizamos o MySQL como sistema de gerenciamento de banco de dados, hospedado em um servidor em nuvem fornecido pela VirtualCloud. A escolha do MySQL se deu por sua confiabilidade, desempenho e compatibilidade com outras tecnologias, especialmente com o PHP, que é utilizado nas páginas do nosso site para garantir a integração entre a aplicação e o banco de dados.

A infraestrutura em nuvem da VirtualCloud nos proporciona escalabilidade, estabilidade e recursos de segurança essenciais para o funcionamento contínuo do sistema. O banco de dados é acessado de forma segura pelas funcionalidades do CritMeet, permitindo o armazenamento e a recuperação eficiente das informações dos usuários e de suas interações. Essa combinação entre MySQL, PHP e nuvem oferece uma base sólida e flexível para o desenvolvimento e crescimento da nossa plataforma de *matchmaking* para jogadores de RPG.

Figura 39 - MySQL

Fonte: MySQL, 2025.

7.6 PHP

O PHP é uma linguagem de *script open source* amplamente utilizada, especialmente indicada para o desenvolvimento *web*. Diferentemente de linguagens que geram HTML através de múltiplos comandos, o PHP permite a inserção de código diretamente dentro de arquivos HTML, facilitando a criação de páginas

dinâmicas. O código PHP é executado no servidor, gerando o conteúdo que será enviado ao navegador, que recebe apenas o resultado final, sem acesso ao código fonte (PHP, 2025).

Além de ser simples para iniciantes, o PHP oferece recursos avançados para programadores experientes. Ele é capaz de realizar diversas tarefas, como coletar dados de formulários, gerar conteúdos dinâmicos, manipular arquivos e interagir com bancos de dados. O PHP funciona em vários sistemas operacionais e servidores *web*, suportando programação estruturada e orientada a objetos. Suas funcionalidades incluem também a geração de arquivos complexos, comunicação com outros serviços e suporte a uma ampla variedade de bancos de dados, o que o torna uma ferramenta poderosa e flexível para desenvolvimento *web* (PHP, 2025).

A escolha pela utilização da linguagem PHP no desenvolvimento do CritMeet fundamenta-se em sua facilidade de implementação e extensa compatibilidade com servidores *web*. O PHP possibilita a criação ágil de aplicações dinâmicas, sendo essencial para o funcionamento da plataforma. Além disso, sua consolidação no mercado e suporte robusto para integração com sistemas de gerenciamento de banco de dados e demais tecnologias conferem maior eficiência e flexibilidade ao processo de desenvolvimento.

Figura 40 - PHP

Fonte: PHP, 2025.

7.7 BOOTSTRAP

O Bootstrap é um *framework front-end* de código aberto que oferece uma série de componentes, estilos e funcionalidades baseados em HTML, CSS e JavaScript, com o objetivo de facilitar o desenvolvimento de interfaces web

modernas, funcionais e responsivas, sendo muito adotado pela comunidade por permitir que até mesmo iniciantes consigam aplicar estilos consistentes e profissionais com pouco esforço. Através de seu sistema de *grid*, classes utilitárias e componentes prontos, ele possibilita a construção de páginas adaptáveis a diferentes tamanhos de tela, atendendo tanto a *desktops* quanto a dispositivos móveis (LIMA, 2021).

Além da facilidade de uso, o Bootstrap também se destaca por sua compatibilidade com os principais navegadores e por permitir personalização por meio de variáveis e classes específicas. Sua adesão em projetos *web* contribui significativamente para a padronização do *layout* e a redução do tempo de desenvolvimento, tornando-se uma escolha estratégica para equipes que desejam otimizar o processo de criação sem abrir mão da qualidade visual e da responsividade da interface (LIMA, 2021).

Utilizamos o Bootstrap em nosso projeto por se tratar de um *framework* que agiliza significativamente o desenvolvimento da interface, permitindo a criação de páginas organizadas e visualmente agradáveis com menor esforço e tempo. Sua vasta biblioteca de componentes prontos, como botões, formulários, menus e sistemas de *grid*, nos proporcionou maior produtividade e padronização no *layout*. Além disso, sua fácil integração e ampla documentação facilitaram o aprendizado e a aplicação prática, tornando-o uma escolha eficiente e acessível para a equipe.

Figura 41 - Bootstrap

Fonte: Wikipedia, 2025

7.8 HTML5, CSS3, JAVASCRIPT

HTML (Linguagem de Marcação de Hipertexto), é uma tecnologia base utilizada na construção de páginas *web*. Ao contrário das linguagens de programação, que possibilitam o desenvolvimento de algoritmos e estruturas lógicas, o HTML é uma linguagem de marcação voltada à estruturação e organização de conteúdo em ambientes digitais. Ele permite definir títulos, parágrafos, *links*, imagens, listas, formulários e diversos outros elementos que compõem visualmente uma página acessada por navegadores (HILLMAN, 2023).

Com o avanço da internet, o HTML passou por diversas versões até chegar ao HTML5, amplamente utilizado atualmente. Essa versão introduziu elementos semânticos que melhoram a acessibilidade, o SEO (Otimização para Mecanismos de Busca) e a organização do código. Além disso, o HTML5 trouxe suporte nativo para mídias como vídeos e áudios, além de tipos de campos mais específicos para formulários, como datas e *e-mails*. Por ser interpretado diretamente pelo navegador, o HTML é indispensável para qualquer projeto web e está entre os primeiros conhecimentos de desenvolvimento *front-end* (HILLMAN, 2023).

Figura 42 - HTML5

Fonte: Wikipedia, 2025.

O CSS é uma linguagem usada para definir o estilo visual de páginas da web. O HTML organiza o conteúdo da página, como títulos, parágrafos, imagens ou listas.

Já o CSS define como esse conteúdo será apresentado ao usuário: cores, fontes, tamanhos, espaçamentos, posições e até animações. Ele funciona com regras que associam seletores, como nomes de elementos, classes ou IDs, a conjuntos de propriedades e valores. Por exemplo, é possível dizer que todos os parágrafos de uma página devem ter a cor azul e uma fonte de tamanho 18 *pixels*. Isso permite manter uma separação entre o conteúdo (no HTML) e a aparência (no CSS) (MDN, 2025).

Alguns conceitos fundamentais do CSS incluem a cascata, que define a prioridade entre regras conflitantes; a herança, que aplica estilos automaticamente a elementos filhos; e o modelo de caixas, que representa todo elemento visual como uma caixa composta por conteúdo, preenchimento interno (*padding*), borda e margem. O CSS3 é essencial para criar layouts organizados, responsivos e visualmente agradáveis. Ele facilita a manutenção e o reaproveitamento de estilos em diferentes partes de um *site* (MDN, 2025).

Figura 43 - CSS3

Fonte: TPSearchTool, 2022.

JavaScript é uma linguagem de programação leve e de uso geral, muito popular na *internet*. Ela é baseada em objetos e é usada principalmente para tornar as páginas *web* interativas, trabalhando junto com HTML e CSS na programação *front-end*. Com JavaScript, é possível criar ações dinâmicas, como responder a eventos do usuário, atualizar partes de uma página sem recarregá-la, fazer

animações e controlar elementos visuais. Além disso, apesar de ser tradicionalmente usada no navegador, JavaScript também pode ser executada no servidor (*back-end*), permitindo o processamento de dados e o funcionamento completo de aplicações *web* (SOUSA, 2022).

Interpretada diretamente pelos navegadores, essa linguagem tornou-se indispensável no desenvolvimento *web* moderno. Com a ajuda de *frameworks*, sua aplicação foi além da interface do usuário, alcançando também o *back-end*, aplicativos móveis e jogos. Essa versatilidade permite a criação de sistemas completos com uma única linguagem, facilitando o desenvolvimento de interfaces dinâmicas e aplicações integradas (SOUSA, 2022).

Figura 44 - JavaScript

Fonte: 1000logos, 2020.

A estruturação da interface da aplicação foi conduzida utilizando HTML5, CSS3 e JavaScript, tecnologias fundamentais no desenvolvimento *web*. O HTML5 foi responsável por organizar semanticamente o conteúdo das páginas, enquanto o CSS3 cuidou da estilização, definindo aspectos visuais como cores, tipografias e espaçamentos. O JavaScript complementou essa camada ao permitir a implementação de interações dinâmicas, viabilizando comportamentos reativos e responsivos nos elementos da interface. A integração harmoniosa dessas tecnologias proporcionou uma experiência fluida e funcional ao usuário.

7.9 AJAX

Segundo a página da Medium (2022), o AJAX consiste em um conjunto de técnicas utilizadas no desenvolvimento de aplicações web dinâmicas e interativas. Por meio de sua abordagem assíncrona, AJAX permite que dados sejam enviados e recebidos do servidor sem que seja necessário recarregar a página. Isso é possível graças à combinação de tecnologias como HTML, CSS, DOM, XML ou JSON, JavaScript e, principalmente, o objeto XMLHttpRequest, que realiza a comunicação com o servidor em segundo plano (QUEVEDO, 2022).

O processo de funcionamento do AJAX envolve a criação de uma requisição pelo navegador, seu envio ao servidor por meio do XMLHttpRequest, o recebimento da resposta com os dados solicitados e, por fim, a atualização dinâmica do conteúdo da página. Com isso, torna-se viável modificar partes específicas da interface conforme as interações do usuário, sem interromper a continuidade da navegação. Entre os principais benefícios desse modelo estão a interatividade com o usuário, o melhor desempenho da aplicação e a menor sobrecarga na comunicação com o servidor, visto que apenas os dados necessários são transmitidos (QUEVEDO, 2022).

O AJAX foi empregado na implementação das páginas *messages* e *group messages* com o objetivo de viabilizar a atualização em tempo real dos dados exibidos, sem a necessidade de recarregamento da página. Trata-se de um recurso da linguagem JavaScript que possibilita a comunicação assíncrona com o servidor, permitindo a recuperação e o envio de informações de forma contínua e eficiente. Essa abordagem contribuiu significativamente para que o CritMeet se tornasse uma aplicação mais dinâmica e interativa às ações do usuário.

Figura 45 - AJAX

Fonte: Wikipedia, 2025.

7.10 DIGITALOCEAN DROPLETS (VPS)

Os Droplets da DigitalOcean são máquinas virtuais de rápida configuração, projetadas para simplificar o uso da computação em nuvem por desenvolvedores, startups e pequenas empresas. Com uma interface intuitiva, suporte a APIs e ferramentas de automação de infraestrutura, oferecem alta disponibilidade e modelos de precificação previsíveis, tornando-se uma solução eficiente e acessível para a implantação de aplicações *web* e serviços online (DIGITALOCEAN, 2025).

Além disso, os Droplets contam com recursos como escalonamento automático, monitoramento integrado, *firewalls* e *backups*. Há diferentes tipos de instâncias otimizadas para uso geral, memória, CPU ou armazenamento, permitindo que os usuários escolham a configuração mais adequada às suas demandas. Trata-se, portanto, de uma alternativa escalável para projetos que exigem desempenho, flexibilidade e facilidade de gerenciamento na nuvem (DIGITALOCEAN, 2025).

Escolhemos utilizar o DigitalOcean Droplets em nosso projeto por se tratar de uma solução de VPS (Servidor Privado Virtual) simples e eficiente. A plataforma oferece facilidade na criação e gerenciamento de máquinas virtuais, além de contar com recursos essenciais como alta disponibilidade , escalabilidade sob demanda e ferramentas integradas de segurança e monitoramento, nos permitindo hospedar nossa aplicação com autonomia, flexibilidade e controle total sobre o ambiente, sem a complexidade excessiva presente em outras soluções de nuvem.

Figura 46 - DigitalOcean

Fonte: IconApe, 2021.

7.11 GIT BASH

Git Bash é uma ferramenta de linha de comando que emula a interface do Git CLI. Essa ferramenta proporciona um ambiente semelhante ao terminal Unix, permitindo que os usuários executem comandos do Git e interajam diretamente com repositórios de controle de versão. Além disso, o Git Bash oferece suporte a diversas funcionalidades típicas de sistemas baseados em Unix, facilitando o gerenciamento e o controle de projetos (GEEKSFORGEEKS, 2024).

Por meio do Git Bash, é possível realizar operações fundamentais no gerenciamento de versões, como clonar repositórios, realizar *commits*, criar e mesclar *branches*, entre outras ações essenciais ao fluxo de desenvolvimento colaborativo. Essa ferramenta é especialmente valiosa por oferecer uma experiência de linha de comando que muitos desenvolvedores já estão acostumados em sistemas Unix/Linux, promovendo maior flexibilidade e eficiência na automação de tarefas (GEEKSFORGEEKS, 2024).

O Git Bash foi utilizado no projeto CritMeet como ferramenta essencial para viabilizar a interface entre o ambiente de desenvolvimento local e o servidor remoto hospedado na plataforma DigitalOcean, onde está configurada uma máquina virtual com sistema operacional Debian. Por meio do Git Bash, foi possível estabelecer

uma comunicação via linha de comando (CLI), o que permitiu aos desenvolvedores realizarem tarefas administrativas diretamente no servidor, como o gerenciamento do banco de dados MySQL, a execução de comandos Unix e a manipulação do repositório versionado no GitHub. Dessa forma, o uso do Git Bash proporcionou maior controle, flexibilidade e eficiência nas etapas de implantação e manutenção do sistema.

Figura 47 - Git Bash

Fonte: SeekLogo, 2016.

7.12 GITHUB

O GitHub é uma plataforma online de hospedagem e gerenciamento de código-fonte que utiliza como base o sistema de controle de versão Git, amplamente adotado no desenvolvimento de *software* em nível mundial. Sua principal função é facilitar o trabalho colaborativo entre desenvolvedores, permitindo que múltiplos integrantes de uma equipe possam trabalhar simultaneamente em um mesmo projeto, de forma organizada, segura e com total controle sobre o histórico de alterações realizadas no código. Por meio de um sistema de versionamento eficiente, o GitHub registra cada modificação, possibilitando a visualização detalhada de quem alterou o quê, quando e por qual motivo, o que é fundamental para manter a integridade do projeto ao longo de todo o seu ciclo de vida (GITHUB, 2025).

Além das funcionalidades básicas de versionamento, o GitHub oferece uma série de recursos avançados que apoiam o desenvolvimento ágil e colaborativo de

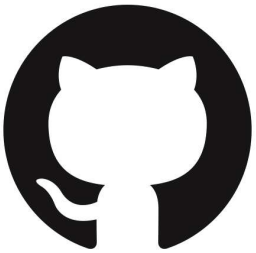
sistemas. Entre eles estão o gerenciamento de *branches*, que permite a criação de linhas de desenvolvimento paralelas, facilitando o trabalho em diferentes funcionalidades sem que haja conflitos entre os códigos; o controle de permissões de acesso, que garante segurança ao projeto ao limitar quem pode visualizar, editar ou aprovar alterações; e as ferramentas de integração contínua, que possibilitam a automatização de testes e *deploys* sempre que novas versões do código são enviadas para o repositório (GITHUB, 2025).

No desenvolvimento do CritMeet, o GitHub teve um papel essencial como ferramenta de versionamento e colaboração. Todo o código-fonte do projeto foi armazenado e gerenciado por meio de repositórios privados da equipe, garantindo que cada modificação pudesse ser documentada e revertida, se necessário. Isso trouxe segurança ao processo de desenvolvimento, permitindo recuperar versões anteriores do sistema em caso de falhas ou conflitos.

Durante as diferentes fases do projeto, como prototipação, implementação de funcionalidades e correção de *bugs*, os membros da equipe criaram *branches* específicas para trabalhar de forma isolada em determinadas tarefas. Posteriormente, as alterações eram integradas ao código principal por meio de *pull requests*, permitindo revisões e testes antes da fusão final. Essa prática evitou conflitos no código e promoveu uma estrutura de trabalho mais limpa e organizada.

O GitHub foi fundamental para a documentação técnica e o acompanhamento da evolução do projeto CritMeet. Essa documentação facilitou tanto a manutenção do sistema quanto a avaliação acadêmica, permitindo o rastreamento de versões e o entendimento das decisões técnicas. Além de servir como repositório central, o GitHub também contribuiu para o controle de qualidade e a colaboração eficiente entre os integrantes, sendo uma ferramenta indispensável para o sucesso do projeto.

Figura 48 - GitHub



Fonte: GitHub, 2025.

7.13 DEBIAN

O Debian é um sistema operacional de código aberto baseado no núcleo Linux, sendo reconhecido por sua estabilidade, segurança e compromisso com os princípios do *software* livre. Desenvolvido de forma colaborativa por uma comunidade global de voluntários, o projeto Debian tem como objetivo fornecer um sistema robusto e confiável, adequado tanto para usuários finais quanto para servidores. Ele se destaca por sua organização estruturada, com lançamentos regulares e bem documentados, além de um sistema de gerenciamento de pacotes eficiente (APT), que facilita a instalação, atualização e remoção de *softwares* (DEBIAN, 2025).

Além disso, o Debian serve como base para diversas outras distribuições amplamente utilizadas, como o Ubuntu, o que demonstra sua relevância no ecossistema de sistemas operacionais baseados em Linux. O suporte a uma vasta gama de pacotes e a ênfase na estabilidade o tornam ideal para projetos que demandam desempenho consistente e manutenção de longo prazo (DEBIAN, 2025).

A escolha do Debian como sistema operacional integrado ao Git Bash justifica-se pela sua estabilidade, robustez e ampla compatibilidade com ferramentas de desenvolvimento, características essenciais para ambientes de produção. Ao utilizar o Debian em um Droplet da DigitalOcean, garantimos um ambiente leve e seguro, otimizado para execução de comandos via terminal, facilitando a interação com o servidor remoto hospedado em máquina virtual. A integração do Git Bash permite acesso unificado ao *prompt* de comandos, ao MySQL e ao GitHub,

consolidando fluxos de trabalho como versionamento, administração de bancos de dados e *deploy* em um único ambiente, reduzindo complexidade operacional e aumentando a eficiência no gerenciamento de infraestrutura. Essa abordagem assegura coerência entre desenvolvimento local e remoto, além de aproveitar a vasta biblioteca de pacotes do Debian para dependências de *software*.

Figura 49 - Debian

Fonte: Wikipedia, 2025.

7.14 API FULLCALENDAR

A API FullCalendar[3] é uma biblioteca JavaScript amplamente utilizada no desenvolvimento de aplicações *web* que necessitam de funcionalidades de calendário interativo. Ela permite a exibição de eventos em formato de calendário, oferecendo recursos como visualizações por mês, semana e dia, navegação entre datas, criação de eventos dinâmicos, além de personalização de *layout* e estilo. Uma de suas principais vantagens é a flexibilidade na integração com sistemas *back-end*, permitindo que os eventos exibidos no calendário sejam carregados, atualizados e manipulados dinamicamente a partir de um banco de dados.

No projeto CritMeet, a API FullCalendar foi integrada ao sistema com o objetivo de facilitar o agendamento de sessões de RPG entre os usuários. Por meio dessa funcionalidade, os Mestres de Jogo podem criar e visualizar sessões

[3] Disponível em: https://fullcalendar.io/docs/view-api.

programadas, enquanto os jogadores têm acesso a um calendário interativo, onde podem consultar datas e horários disponíveis para participação nos encontros. A integração com o *back-end* em PHP e o banco de dados SQL permitiu que os eventos fossem armazenados de forma persistente, garantindo que os agendamentos permanecessem salvos e acessíveis a qualquer momento.

Além disso, a utilização do FullCalendar possibilitou a implementação de recursos de edição e exclusão de eventos, tornando a gestão das sessões de RPG mais dinâmica e intuitiva para os usuários da plataforma. A interface do calendário foi desenvolvida de maneira responsiva, garantindo o correto funcionamento tanto em navegadores de *desktop* quanto em dispositivos móveis, atendendo, assim, às necessidades de acessibilidade da comunidade de jogadores.

Essa integração com a API FullCalendar contribuiu significativamente para a experiência de usabilidade da aplicação, agregando uma ferramenta visual que torna o planejamento das sessões mais organizado e eficiente, promovendo maior engajamento entre os membros da comunidade CritMeet.

7.15 APIS DE GEOLOCALIZAÇÃO

A API Leaflet[4] é uma biblioteca de código aberto amplamente utilizada para a criação de mapas interativos em aplicações *web*. Desenvolvida com foco em desempenho e simplicidade, a Leaflet permite a visualização e manipulação de mapas diretamente no navegador, oferecendo recursos como marcação de pontos, personalização de camadas, *zoom*, interação com eventos de clique e integração com dados de geolocalização.

Uma das principais características da Leaflet é sua capacidade de trabalhar com diferentes fontes de mapas, sendo o OpenStreetMap[5] (OSM) uma das mais populares. O OpenStreetMap é um projeto colaborativo que oferece dados geográficos gratuitos e abertos, permitindo que desenvolvedores integrem mapas

[4] Disponível em: https://leafletjs.com/.

[5] Disponível em: https://www.openstreetmap.org/copyright.

detalhados em suas aplicações sem custos de licença. Essa integração facilita a exibição de localizações, rotas e áreas geográficas com alto nível de personalização.

No projeto CritMeet, a API Leaflet foi utilizada para exibir mapas interativos dentro da plataforma, permitindo que os usuários visualizem a localização de outros jogadores de RPG de mesa em suas proximidades. A visualização é alimentada por camadas de dados provenientes do OpenStreetMap, garantindo uma representação geográfica atualizada e de acesso livre. Por meio dessa implementação, foi possível apresentar aos usuários um mapa dinâmico, onde pontos de interesse (como perfis de outros jogadores) são plotados com base em coordenadas geográficas obtidas por meio de APIs de geolocalização.

Além disso, a utilização da Leaflet trouxe vantagens como baixo consumo de recursos, compatibilidade com diferentes dispositivos e suporte a diversas extensões e *plugins*, o que possibilita a futura adição de funcionalidades como cálculo de distâncias, rotas ou filtragem de usuários por área geográfica.

Essa integração entre a API Leaflet e o OpenStreetMap contribuiu de forma significativa para a usabilidade da plataforma, permitindo que a funcionalidade de localização de jogadores fosse apresentada de forma visual, acessível e intuitiva.

7.16 HTTPS

O HTTPS é um protocolo de internet amplamente utilizado por *sites* e aplicativos, sendo responsável por garantir a transferência segura de dados entre o navegador do usuário e o servidor *web*. Diferente do HTTP tradicional (que transmite informações em texto puro e é vulnerável à interceptação), o HTTPS adiciona uma camada de proteção por meio da criptografia. Essa camada adicional assegura que os dados, mesmo que capturados por terceiros, não possam ser compreendidos sem a chave criptográfica correta, resguardando assim informações sensíveis durante a navegação. Com isso, o HTTPS não apenas protege os dados, mas também promove maior confiança entre os usuários e os *sites* acessados (HOSTMÍDIA, 2025).

No contexto do CritMeet, a adoção do protocolo HTTPS foi uma medida essencial para garantir a segurança e a integridade dos dados trafegados entre os usuários e o servidor da aplicação. Como a plataforma envolve o cadastro de informações pessoais e possivelmente a troca de mensagens entre jogadores e mestres de RPG, é fundamental assegurar que essas informações estejam protegidas contra interceptações e acessos não autorizados. O uso do HTTPS permite a criptografia dessas comunicações, promovendo confidencialidade e autenticidade no acesso ao sistema. Além disso, contribui positivamente para a reputação da plataforma, evita alertas de segurança nos navegadores modernos e alinha o CritMeet às boas práticas exigidas por legislações de proteção de dados, como a LGPD.

7.17 .HTPASSWD E .HTACCESS

O *.htpasswd* é um recurso essencial de segurança utilizado em servidores *web*, com a finalidade de controlar o acesso a diretórios restritos por meio de autenticação básica. Ele pode armazenar pares de nomes de usuários e senhas criptografadas, sendo utilizado em conjunto com o arquivo *.htaccess*, onde se configuram as diretivas necessárias para definir o tipo de autenticação, o nome da área protegida e o caminho até o *.htpasswd*. A segurança deste mecanismo depende tanto da força do algoritmo de criptografia utilizado, quanto da correta definição de permissões de acesso ao arquivo, evitando que usuários não autorizados o leiam. Considerando também que as credenciais trafegam codificadas em Base64[6] durante a autenticação, é imprescindível o uso de HTTPS para garantir a confidencialidade das informações transmitidas (NGUYEN, 2024).

Em relação ao uso dessas tecnologias no desenvolvimento do CritMeet, a utilização dos arquivos *.htpasswd* e *.htaccess* foi adotada como uma medida de

[6] Base64 é um esquema de codificação que converte dados binários em uma representação textual, permitindo que informações originalmente não textuais possam ser armazenadas e transmitidas com segurança por meios projetados para lidar com texto.[7]

[7] Fonte: MDN. **Base64**. MDN. 2024. Disponível em: https://developer.mozilla.org/pt-BR/docs/Glossary/Base64. Acesso em: 15 jun. 2025

segurança fundamental para restringir o acesso a áreas administrativas e sensíveis da aplicação hospedada no servidor. O *.htpasswd* permite o controle de autenticação por meio do armazenamento seguro de credenciais criptografadas, enquanto o *.htaccess* define as diretivas de autenticação necessárias para que apenas usuários autorizados possam acessar determinados diretórios. Essa abordagem oferece uma camada adicional de proteção sem depender exclusivamente de mecanismos internos da aplicação, sendo especialmente relevante durante as fases de desenvolvimento, testes e manutenção, nas quais o controle rigoroso do acesso evita exposições indevidas de funcionalidades críticas ou dados sensíveis do sistema.

7.18 SSH, BCRYPT E SHA-256

O SSH é um protocolo utilizado no campo da administração de sistemas e na transferência segura de arquivos, especialmente em redes consideradas inseguras, estabelecendo uma conexão criptografada entre cliente e servidor, garantindo que todas as autenticações de usuário, comandos, saídas e transferências de dados estejam protegidas contra ataques e interceptações. O protocolo surgiu como resposta a uma falha de segurança em uma rede universitária finlandesa, levando à criação de uma solução confiável para *logins* remotos seguros. Desde então, o SSH evoluiu e tornou-se uma ferramenta essencial em *data centers* e grandes corporações, sendo empregado na administração de servidores, automação de processos por meio de chaves SSH, e substituindo protocolos antigos e vulneráveis (YLÖNEN, 2024).

Figura 50 - Secure Shell

Fonte: Secure Shell, 2025.

O Bcrypt é um algoritmo de *hashing* de senhas desenvolvido para oferecer um alto nível de segurança no armazenamento de credenciais em sistemas computacionais. Sua principal característica é o uso de técnicas adaptativas que tornam o processo de geração de *hashes* intencionalmente lento e, portanto, resistente a ataques de força bruta. Esse algoritmo funciona por meio da aplicação de múltiplas iterações de *hash* sobre a combinação de uma senha e um valor aleatório, gerado no momento do cadastro ou atualização da senha, de modo a evitar que senhas iguais resultem em *hashes* idênticos (VPNUNLIMITED, [20--]).

O SHA-256 é uma função de *hash* criptográfica pertencente à família SHA-2, desenvolvida pela Agência de Segurança Nacional dos Estados Unidos, com o objetivo de substituir o SHA-1, que apresentou vulnerabilidades ao longo do tempo. Sua principal função é transformar dados de qualquer tamanho em uma cadeia alfanumérica fixa de 256 bits, denominada valor *hash*. Esse valor é praticamente único para cada entrada distinta, garantindo que pequenas alterações no dado de origem resultem em grandes mudanças no *hash* gerado. O algoritmo é determinístico, ou seja, produzirá sempre o mesmo valor de saída para uma entrada específica, característica essencial para a verificação da integridade de dados entre sistemas distintos (LEPCHA, 2023).

No desenvolvimento do CritMeet, a utilização do protocolo SSH foi essencial para garantir um ambiente seguro na administração remota do servidor de produção, especialmente em operações como manutenção e *deploy*. A escolha pelo SSH está fundamentada na sua capacidade de estabelecer canais de comunicação criptografados, evitando a exposição de credenciais em texto plano e restringindo o acesso apenas a desenvolvedores previamente autorizados.

Os algoritmos Bcrypt e SHA-256 foram empregados em dois contextos distintos, ambos voltados à segurança da aplicação. O uso do SHA-256 está voltado para a geração de chaves criptográficas para acesso ao servidor, garantindo que apenas dispositivos autorizados pudessem realizar operações sensíveis, reforçando a proteção do ambiente de desenvolvimento, impedindo acessos não autorizados e promovendo a integridade dos dados do sistema.

Ademais, o Bcrypt foi incorporado ao mecanismo de *login* e registro de usuários, atuando na criptografia das senhas armazenadas no banco de dados. Em vez de manter as credenciais em texto simples, o sistema as transforma em *hashes* criptográficos, impossibilitando sua leitura direta mesmo em caso de vazamento. Essa prática é essencial para a proteção de informações pessoais, alinhando o CritMeet às boas práticas de segurança digital e mitigando riscos relacionados à exposição de dados sensíveis.

8 DESENVOLVIMENTO

Esta seção apresenta o desenvolvimento da aplicação CritMeet, detalhando as etapas realizadas e destacando a importância de cada fase para a construção e conclusão do sistema. O processo foi estruturado para garantir que todas as partes da aplicação se integrassem de forma harmoniosa, atendendo às necessidades dos usuários que buscam conectar-se com outros jogadores de RPG de mesa.

8.1 BANCO DE DADOS

Um banco de dados é uma estrutura organizada de dados, normalmente armazenados em sistemas computacionais, cuja função central é permitir que essas informações possam ser acessadas, manipuladas, atualizadas e gerenciadas de forma eficiente e segura. Ele é gerido por um Sistema de Gerenciamento de Banco de Dados (DBMS), que atua como intermediário entre os dados e os usuários ou aplicações, possibilitando desde operações simples de inserção até consultas complexas. A maioria dos bancos de dados atuais utiliza a linguagem SQL (Linguagem de Consulta Estruturada) para comunicação, permitindo a criação e consulta de dados com flexibilidade (ORACLE, 2020).

Ao escolher o sistema de gerenciamento de banco de dados MySQL para o desenvolvimento da aplicação, essa decisão foi motivada pela familiaridade da equipe com a tecnologia, especialmente em ambientes *web*. Por ser um sistema de banco de dados relacional de código aberto, o MySQL é diversas vezes adotado por grandes plataformas online devido à sua capacidade de lidar com grandes volumes de dados e múltiplas transações simultâneas com eficiência.

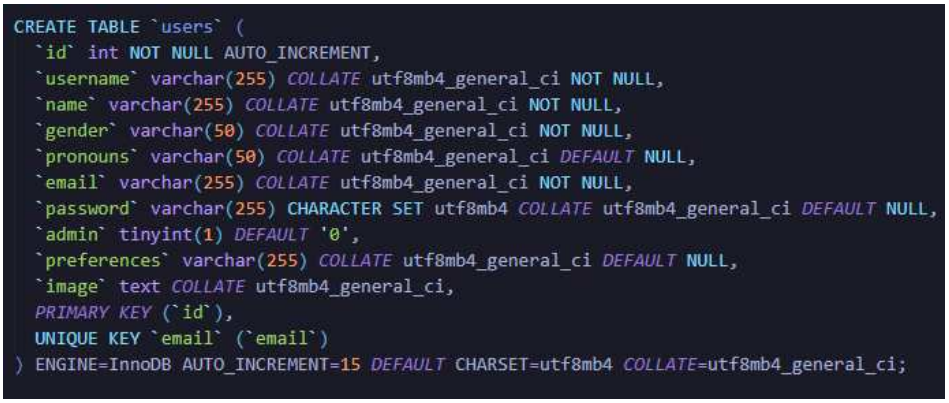
Com a evolução do projeto e sua migração para um ambiente de produção, o banco passou a ser hospedado em um servidor Apache2 executando em uma máquina virtual Debian 12 na plataforma Digital Ocean. A manipulação do banco em ambiente de produção é realizada principalmente através de linha de comando (CLI) utilizando o Git Bash, proporcionando maior controle e possibilitando a automação

de operações através de *scripts*.

Para apoiar a documentação e visualização da estrutura do banco, foi utilizado o MySQL Workbench para a geração do Modelo de Entidade e Relacionamento (MER), facilitando a compreensão da arquitetura de dados implementada.

O banco de dados critmeet foi projetado com foco na modularidade e integridade referencial, utilizando a *engine* InnoDB para suporte completo a transações ACID e chaves estrangeiras. A estrutura implementa o conjunto de caracteres UTF8MB4, garantindo suporte completo a *emojis* e caracteres especiais, aspecto importante para uma aplicação social contemporânea.

Figura 51 - Tabela de usuários



```
CREATE TABLE `users` (
  `id` int NOT NULL AUTO_INCREMENT,
  `username` varchar(255) COLLATE utf8mb4_general_ci NOT NULL,
  `name` varchar(255) COLLATE utf8mb4_general_ci NOT NULL,
  `gender` varchar(50) COLLATE utf8mb4_general_ci NOT NULL,
  `pronouns` varchar(50) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `email` varchar(255) COLLATE utf8mb4_general_ci NOT NULL,
  `password` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci DEFAULT NULL,
  `admin` tinyint(1) DEFAULT '0',
  `preferences` varchar(255) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `image` text COLLATE utf8mb4_general_ci,
  PRIMARY KEY (`id`),
  UNIQUE KEY `email` (`email`)
) ENGINE=InnoDB AUTO_INCREMENT=15 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Fonte: elaborado pelos autores.

A tabela users constitui o núcleo do sistema de autenticação e gerenciamento de perfis de usuário da plataforma. Esta tabela armazena todas as informações essenciais dos usuários cadastrados, incluindo dados de identificação (*username*, *name*, *email*), informações pessoais com foco na inclusividade social (*gender*, *pronouns*), credenciais de acesso (*password*), configurações de perfil (*preferences*, *image*) e controle de privilégios administrativos (*admin*). A constraint UNIQUE no campo email garante a unicidade de cada usuário no sistema.

Figura 52 - Tabela de sessões

```sql
CREATE TABLE `sessions` (
  `id` int NOT NULL AUTO_INCREMENT,
  `title` varchar(255) COLLATE utf8mb4_general_ci NOT NULL,
  `description` text COLLATE utf8mb4_general_ci,
  `start_datetime` datetime NOT NULL,
  `end_datetime` datetime NOT NULL,
  `max_players` int NOT NULL DEFAULT '4',
  `current_players` int NOT NULL DEFAULT '1',
  `status` enum('active','full','cancelled','completed') COLLATE utf8mb4_general_ci DEFAULT 'active',
  `location` varchar(255) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `creator_id` int NOT NULL,
  `chat_id` int DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `creator_id` (`creator_id`),
  KEY `idx_sessions_chat_id` (`chat_id`),
  KEY `idx_sessions_chat_start` (`chat_id`,`start_datetime`),
  CONSTRAINT `sessions_ibfk_1` FOREIGN KEY (`creator_id`) REFERENCES `users` (`id`) ON DELETE CASCADE,
  CONSTRAINT `sessions_ibfk_2` FOREIGN KEY (`chat_id`) REFERENCES `chats` (`id`) ON DELETE SET NULL
) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Fonte: elaborado pelos autores.

A tabela sessions representa o core funcional do sistema, responsável pelo gerenciamento das sessões de RPG criadas pelos usuários. Cada registro armazena informações completas sobre uma sessão, incluindo metadados (*title*, *description*), agendamento temporal (*start_datetime*, *end_datetime*), controle de capacidade (max_players, current_players), estados bem definidos através de ENUM (*status*: *active*, *full*, *cancelled*, *completed*), localização física para sessões presenciais (*location*), identificação do criador (*creator_id*) e integração com o sistema de comunicação (*chat_id*). A tabela implementa chaves estrangeiras para garantir integridade referencial com as tabelas users e chats.

Figura 53 - Tabela de localização de usuário

```sql
CREATE TABLE `user_locations` (
  `id` int NOT NULL AUTO_INCREMENT,
  `user_id` int NOT NULL,
  `latitude` decimal(10,8) NOT NULL,
  `longitude` decimal(11,8) NOT NULL,
  `accuracy` decimal(8,2) DEFAULT NULL,
  `address` text,
  `city` varchar(100) DEFAULT NULL,
  `state` varchar(100) DEFAULT NULL,
  `country` varchar(100) DEFAULT 'Brasil',
  `created_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idx_user_id` (`user_id`),
  KEY `idx_updated_at` (`updated_at`),
  KEY `idx_location` (`latitude`,`longitude`),
  CONSTRAINT `user_locations_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Fonte: elaborado pelos autores.

O sistema de geolocalização é implementado através da tabela

user_locations, que armazena informações precisas de localização dos usuários para facilitar a busca por sessões próximas. A tabela utiliza coordenadas geográficas com precisão de 8 casas decimais (latitude, longitude) para precisão métrica, complementadas por informações de endereçamento (*address*, *city*, *state*, *country*) com valor padrão 'Brasil' considerando o público-alvo. Inclui campos de auditoria temporal (*created_at*, *updated_at*) e um índice composto *idx_location* otimizado para consultas geoespaciais eficientes.

Figura 54 - Tabela de amigos

```
CREATE TABLE `friends` (
  `id` int NOT NULL AUTO_INCREMENT,
  `user_id` int NOT NULL,
  `friend_id` int NOT NULL,
  `status` enum('pending','accepted') COLLATE utf8mb4_general_ci DEFAULT 'pending',
  `created_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `fk_user` (`user_id`),
  KEY `fk_friend` (`friend_id`),
  CONSTRAINT `fk_friend` FOREIGN KEY (`friend_id`) REFERENCES `users` (`id`) ON DELETE CASCADE,
  CONSTRAINT `fk_user` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=23 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Fonte: elaborado pelos autores.

A implementação do sistema de relacionamentos sociais é gerenciada pela tabela friends, que segue o padrão bidirecional onde cada relacionamento de amizade requer aceitação mútua. A tabela utiliza um sistema de estados através do campo status (*pending*, *accepted*) para controlar o fluxo de solicitações de amizade. Os campos *user_id* e *friend_id* estabelecem a relação entre usuários, com constraints de chave estrangeira configuradas para CASCADE, garantindo a integridade dos dados quando usuários são removidos do sistema.

Figura 55 - Tabela de membros de sessão

```
CREATE TABLE `session_members` (
  `id` int NOT NULL AUTO_INCREMENT,
  `session_id` int NOT NULL,
  `user_id` int NOT NULL,
  `status` enum('pending','accepted','declined') CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci DEFAULT 'pending',
  `joined_at` timestamp NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `session_id` (`session_id`),
  KEY `user_id` (`user_id`),
  KEY `idx_session_members_status` (`session_id`,`status`),
  CONSTRAINT `session_members_ibfk_1` FOREIGN KEY (`session_id`) REFERENCES `sessions` (`id`) ON DELETE CASCADE,
  CONSTRAINT `session_members_ibfk_2` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Fonte: elaborado pelos autores.

Esta tabela gerencia a participação dos usuários nas sessões de RPG através de um sistema de convites com controle de estados. Implementa três possíveis status de participação (*pending*, *accepted*, *declined*) através do campo status, permitindo um fluxo controlado de entrada em sessões. A tabela estabelece a relação muitos para muitos entre usuários e sessões, incluindo *timestamp* de adesão (*joined_at*) e um índice composto *idx_session_members_status* para otimização de consultas de participação.

Figura 56 - Tabela de chats

```
CREATE TABLE `chats` (
  `id` int NOT NULL AUTO_INCREMENT,
  `is_group` tinyint(1) NOT NULL DEFAULT '0',
  `created_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `name` varchar(255) COLLATE utf8mb4_general_ci DEFAULT NULL,
  `creator_id` int NOT NULL,
  `image` varchar(255) COLLATE utf8mb4_general_ci DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=27 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Fonte: elaborado pelos autores.

A tabela *chats* é o centro do sistema de comunicação da plataforma, gerenciando tanto conversas individuais quanto em grupo. Implementa diferenciação entre tipos de chat através do campo booleano *is_group*, inclui metadados como nome do chat (*name*), identificação do criador (*creator_id*), imagem personalizada (*image*) e timestamp de criação (*created_at*). Esta tabela serve como base para todo o sistema de mensageria integrado às sessões de RPG.

Figura 57 - Tabela de membros de chat

```
CREATE TABLE `chat_members` (
  `id` int NOT NULL AUTO_INCREMENT,
  `chat_id` int NOT NULL,
  `user_id` int NOT NULL,
  `role` enum('member','admin') COLLATE utf8mb4_general_ci DEFAULT 'member',
  `last_seen` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `chat_id` (`chat_id`),
  KEY `user_id` (`user_id`),
  CONSTRAINT `chat_members_ibfk_1` FOREIGN KEY (`chat_id`) REFERENCES `chats` (`id`) ON DELETE CASCADE,
  CONSTRAINT `chat_members_ibfk_2` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=55 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Fonte: elaborado pelos autores.

Gerencia a participação dos usuários nos diferentes *chats* da plataforma, implementando um sistema de cargos (*member*, *admin*) para controle de permissões dentro de cada conversa. A tabela estabelece a relação muitos para muitos entre usuários e *chats*, incluindo funcionalidade de controle de visualização através do campo *last_seen* para implementação de recursos como "mensagens não lidas" e status de atividade dos participantes.

Figura 58 - Tabela de mensagens

```
CREATE TABLE `messages` (
  `id` int NOT NULL AUTO_INCREMENT,
  `chat_id` int NOT NULL,
  `sender_id` int NOT NULL,
  `content` text COLLATE utf8mb4_general_ci NOT NULL,
  `timestamp` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `chat_id` (`chat_id`),
  KEY `sender_id` (`sender_id`),
  CONSTRAINT `messages_ibfk_1` FOREIGN KEY (`chat_id`) REFERENCES `chats` (`id`) ON DELETE CASCADE,
  CONSTRAINT `messages_ibfk_2` FOREIGN KEY (`sender_id`) REFERENCES `users` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=41 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Fonte: elaborado pelos autores.

Armazena todas as mensagens trocadas nos chats da plataforma, formando o histórico completo de comunicação entre usuários. Cada mensagem é identificada pelo chat de origem (*chat_id*), remetente (*sender_id*), conteúdo textual (*content*) e timestamp preciso de envio (*timestamp*). A tabela utiliza o tipo TEXT para suportar mensagens de tamanho variável e implementa cascata de exclusão para manter a integridade quando chats ou usuários são removidos.

Figura 59 - Tabela de denúncias

```sql
CREATE TABLE `reports` (
  `id` int NOT NULL AUTO_INCREMENT,
  `reporter_id` int NOT NULL,
  `reported_id` int NOT NULL,
  `reason` text CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci NOT NULL,
  `status` enum('pending','reviewed','resolved','dismissed') CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci DEFAULT 'pending',
  `admin_notes` text CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci,
  `reviewed_by` int DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `reporter_id` (`reporter_id`),
  KEY `reported_id` (`reported_id`),
  KEY `reviewed_by` (`reviewed_by`),
  KEY `idx_status` (`status`),
  KEY `idx_created_at` (`created_at`),
  CONSTRAINT `reports_ibfk_1` FOREIGN KEY (`reporter_id`) REFERENCES `users` (`id`) ON DELETE CASCADE,
  CONSTRAINT `reports_ibfk_2` FOREIGN KEY (`reported_id`) REFERENCES `users` (`id`) ON DELETE CASCADE,
  CONSTRAINT `reports_ibfk_3` FOREIGN KEY (`reviewed_by`) REFERENCES `users` (`id`) ON DELETE SET NULL
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

Fonte: elaborado pelos autores.

Implementa o sistema de moderação da plataforma, permitindo que usuários reportem comportamentos inadequados ou violações das regras de conduta. A tabela gerencia um fluxo completo de análise administrativa através dos campos de identificação (*reporter_id*, *reported_id*), justificativa (*reason*), controle de status (*pending*, *reviewed*, *resolved*, *dismissed*), anotações administrativas (*admin_notes*), identificação do moderador responsável (*reviewed_by*) e timestamps de auditoria (*created_at*, *updated_at*). Inclui índices otimizados para consultas de moderação frequentes.

Figura 60 - Camada de view

View: v_chat_sessions

Columns:

id	int
title	varchar(255)
description	text
start_datetime	datetime
end_datetime	datetime
max_players	int
status	enum('active','full','cancelled','completed')
location	varchar(255)
creator_id	int
chat_id	int
chat_name	varchar(255)
is_group	tinyint(1)
creator_name	varchar(255)
creator_username	varchar(255)
current_players	bigint
calculated_status	varchar(9)

Fonte: elaborado pelos autores.

Esta *view* materializada agrega informações de múltiplas tabelas para otimizar

consultas frequentes relacionadas às sessões e seus chats associados. Implementa lógica de negócio automática para cálculo do status real das sessões através do campo *calculated_status*, considerando fatores como capacidade máxima atingida e data de término. A *view* centraliza informações de sessões, *chats*, criadores e participantes, fornecendo uma interface simplificada para consultas complexas que envolvem múltiplas entidades do sistema.

8.2 BACK-END

De acordo com o artigo publicado por Santana (2024) na plataforma da Alura, o *back-end* é a parte da aplicação que tem como principais responsabilidades executar lógicas complexas e armazenar os dados da aplicação. Ao contrário do *front-end*, o *back-end* é executado em servidores conectados à internet, muitas vezes hospedados em plataformas de computação em nuvem (SANTANA, 2024). Esta camada confere ao *software* propriedades fundamentais como segurança, desempenho e confiabilidade, sendo indispensável ao longo de todo seu desenvolvimento.

No contexto do projeto em questão, o *back-end* desempenha um papel essencial para o funcionamento estruturado e seguro da aplicação. Como a plataforma foi idealizada para promover o encontro entre jogadores de RPG, é necessário lidar com funcionalidades como registro e autenticação de usuários, gerenciamento de perfis, controle de permissões e a mediação das interações entre os participantes. Todas essas operações exigem uma lógica de negócio robusta, que é executada exclusivamente no *back-end*. É nessa camada da aplicação que são realizadas operações mais complexas, como o armazenamento e recuperação de dados em bancos de dados, validações de segurança e a comunicação com servidores, garantindo que todas as ações realizadas no *front-end* estejam devidamente processadas e registradas e intermediando a comunicação do *front-end* com o banco de dados.

O *back-end* do CritMeet foi desenvolvido utilizando PHP 8.2.28 como linguagem principal, aproveitando as funcionalidades modernas de orientação a

objetos, sintaxe aprimorada e otimizações de performance da versão mais recente. A arquitetura incorpora JavaScript para processamento assíncrono, requisições Ajax para comunicação dinâmica com o *front-end*, e manipulação eficiente de dados JSON para intercâmbio de informações estruturadas. A escolha estratégica do PHP fundamentou-se em três pilares: estabilidade comprovada em aplicações *web*, extensa documentação comunitária e facilidade de integração com sistemas de terceiros. O desenvolvimento foi realizado integralmente no Visual Studio Code, proporcionando um ambiente robusto com ferramentas avançadas de depuração e extensões especializadas para desenvolvimento PHP.

8.2.1 ROTAS DE ACESSO

O sistema de rotas do CritMeet é o mecanismo central que direciona todas as requisições HTTP para os recursos adequados da aplicação. Implementado no arquivo *router.php*, ele funciona como um GPS “inteligente” que interpreta URLs amigáveis, transformando URLs do tipo *critmeet.com.br/login* em caminhos físicos no servidor (*/pages/login/index.php*), segue princípios RESTful, mantendo uma estrutura lógica e humanamente legível, eliminando extensões de arquivo (*.php*) e parâmetros complexos e garante a segurança bloqueando acesso direto a arquivos sensíveis (*.env*, *.htpasswd*, etc.) via regras no *.htaccess*.

Figura 61 - Rotas de acesso

```php
// router.php
$requested_path = trim($_SERVER['REQUEST_URI'], '/');

// Remove base do projeto (ex: /critmeet)
$base_path = 'critmeet';
$clean_path = str_replace($base_path, '', $requested_path);

// Mapeamento para páginas físicas
if ($clean_path === '') {
    require __DIR__ . '/pages/home/index.php'; // Homepage
} elseif (file_exists("/pages/{$clean_path}/index.php")) {
    require "/pages/{$clean_path}/index.php"; // Páginas regulares
} else {
    http_response_code(404);
    require '/pages/404/index.php'; // Página de erro
}
```

Fonte: elaborado pelos autores.

8.2.2 PROTEÇÃO DE DADOS

A segurança de dados da aplicação CritMeet foi concebida de forma multicamadas, contemplando desde o armazenamento de credenciais até o controle de acesso a áreas sensíveis do sistema. A seguir, detalham-se os principais recursos empregados.

- **Criptografia e Autenticação Segura:** as senhas dos usuários são protegidas utilizando o algoritmo Bcrypt, que incorpora *salting* e múltiplas iterações para mitigar ataques de força bruta. O sistema adota uma abordagem híbrida para suportar senhas legadas, realizando migração automática para *hashes* seguros, garantindo a confidencialidade mesmo em casos de vazamento de banco de dados, uma vez que os *hashes* são irrecuperáveis sem a senha original:

Figura 62 - Autenticação segura

```php
// Sistema híbrido de validação de senhas
$password_valid = false;
if (password_verify($current_password, $user['password'])) {
    $password_valid = true; // Hash moderno (padrão atual)
} elseif ($user['password'] === $current_password) {
    $password_valid = true; // Texto plano (dados legados)
    // Migração automática para hash
    $hashed_password = password_hash($current_password, PASSWORD_DEFAULT);
    $update_query = "UPDATE users SET password = ? WHERE id = ?";
    $update_stmt = $mysqli->prepare($update_query);
    $update_stmt->bind_param("si", $hashed_password, $user_id);
    $update_stmt->execute();
}
```

Fonte: elaborado pelos autores.

- **Sessões PHP e Controle de Acesso:** a autenticação é controlada por sessões PHP com verificação obrigatória nas páginas protegidas, impedindo o acesso não autorizado:

Figura 63 - Controle de acesso

```php
session_start();
$user_id = $_SESSION['user_id'] ?? null;

// Verificação de autenticação obrigatória
if (!$user_id) {
    echo "<script>
        alert('Usuário não autenticado.');
        window.location.href='../../pages/login/';
    </script>";
    exit();
}
```

Fonte: elaborado pelos autores.

Adicionalmente, usuários com privilégios administrativos são verificados com base em seu papel na base de dados:

Figura 64 - Verificação de permissões administrativas

```php
$is_admin = false;
if ($user_id) {
    $query = "SELECT admin FROM users WHERE id = ?";
    $stmt = $mysqli->prepare($query);
    $stmt->bind_param("i", $user_id);
    $stmt->execute();
    $result = $stmt->get_result();
    if ($result && $row = $result->fetch_assoc()) {
        $is_admin = $row['admin'] == 1;
    }
    $stmt->close();
}
```

Fonte: elaborado pelos autores.

- **HTTPS e SSH:** toda comunicação entre cliente e servidor é criptografada via HTTPS, assegurando a confidencialidade dos dados durante a navegação, especialmente em operações sensíveis como login e troca de mensagens. Para administração remota, o protocolo SSH é utilizado, evitando a exposição de credenciais e permitindo o uso de chaves criptográficas;
- **Hash SHA-256 e Acesso Restrito:** a aplicação utiliza SHA-256 para geração de chaves de autenticação e acesso ao ambiente de produção. Esse

algoritmo garante integridade e autenticidade dos dados transmitidos, sendo amplamente adotado em ambientes que exigem alta segurança;

- **.htaccess e .htpasswd:** para proteção de diretórios administrativos e sensíveis, foram utilizados os arquivos *.htaccess* e *.htpasswd*, permitindo autenticação básica em nível de servidor:

Figura 65 - Configuração de autenticação básica

```
AuthType Basic
AuthName "Área Restrita - CritMeet"
AuthUserFile /etc/apache2/.htpasswd
Require valid-user
```

Fonte: elaborado pelos autores.

As credenciais são armazenadas de forma criptografada no arquivo *.htpasswd*, e o tráfego dessas informações é protegido por HTTPS, evitando interceptações.

- **Validação e Sanitização de Dados:** A CritMeet implementa rigorosa validação de entradas tanto no *frontend* quanto no *backend*, utilizando *filter_input* e *htmlspecialchars()* para mitigar ataques como XSS e *SQL Injection*:

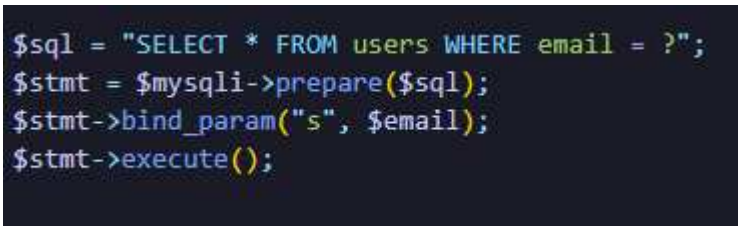
Figura 66 - Validação de dados

```
// Validação de email
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
    throw new Exception('Email inválido.');
}

// Validação de senha
if (strlen($password) < 6) {
    throw new Exception('A senha deve ter pelo menos 6 caracteres.');
}

// Validação de entrada com filtros
$group_id = filter_input(INPUT_GET, 'group_id', FILTER_VALIDATE_INT);
$content = trim($_POST['content'] ?? '');
$new_group_name = trim($_POST['group_name']);

if (empty($content)) {
    throw new Exception('Conteúdo da mensagem não pode estar vazio.');
}
```

Fonte: elaborado pelos autores.

Além disso, todas as consultas SQL utilizam *prepared statements* com *bind_param*, impedindo injeções maliciosas e reforçando a robustez da aplicação contra ataques externos.

Figura 67 - Uso de prepared statement para prevenção de SQL Injection



Fonte: elaborado pelos autores

8.3 FRONT-END

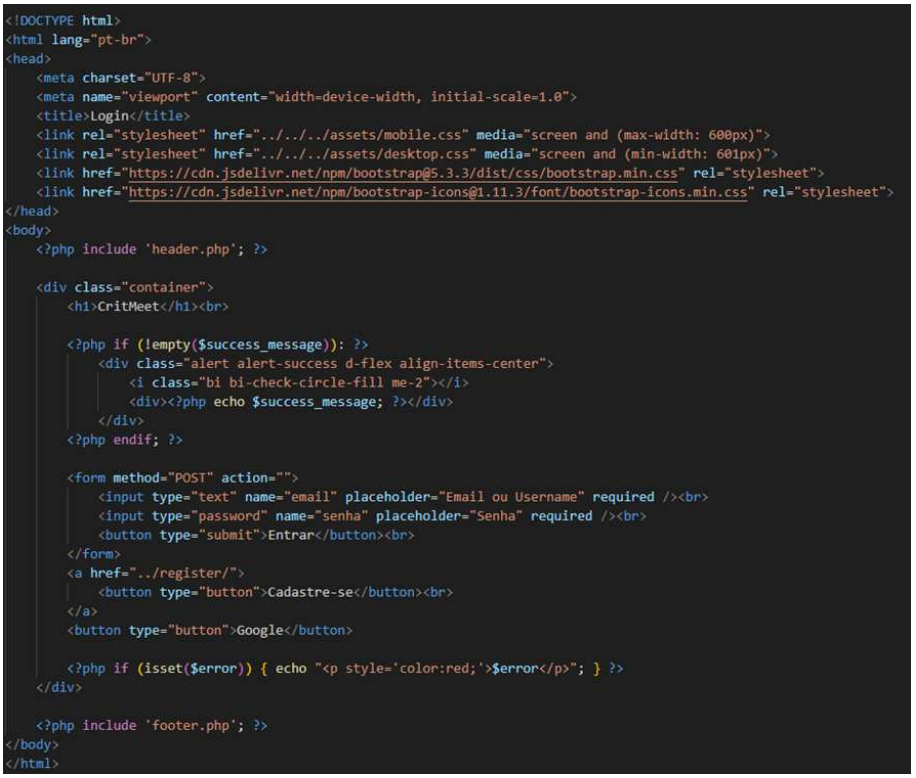
No contexto do desenvolvimento de tecnologia, o *front-end* representa a camada visual de um *site* ou aplicativo, sendo responsável por criar a interface com a qual os usuários interagem diretamente. Essa parte do sistema é composta por elementos gráficos como botões, menus e demais componentes visuais, cuja implementação se dá por meio de linguagens específicas, como HTML, CSS e JavaScript. O HTML é utilizado para estruturar o conteúdo da página, o CSS define a estética visual dos elementos, e o JavaScript adiciona interatividade e comportamentos dinâmicos. Dessa forma, o front-end desempenha um papel crucial na construção de uma experiência do usuário agradável, intuitiva e responsiva, sendo essencial para a navegação, a execução de ações e a interação geral dentro de plataformas digitais (AWARI, 2023).

O desenvolvimento do front-end do CritMeet foi guiado por princípios de responsividade, acessibilidade e usabilidade, com foco na construção de uma interface clara, funcional e compatível com diferentes dispositivos. Adotou-se uma abordagem *mobile-first*, garantindo que a aplicação fosse plenamente utilizável em telas pequenas, como smartphones, sem comprometer a experiência em

dispositivos maiores.

A base estrutural das páginas foi desenvolvida com HTML5, seguindo práticas de marcação semântica para tornar o código mais compreensível e otimizado para motores de busca. A estilização visual ficou a cargo do CSS3, utilizando media queries, *flexbox* e Bootstrap 5.3.3 para assegurar um layout responsivo e adaptável a diferentes resoluções de tela. O uso do Bootstrap também permitiu acelerar o desenvolvimento por meio de componentes reutilizáveis e sistemas de grid pré-configurados.

Figura 66 - Front-end: Tela de login



```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Login</title>
    <link rel="stylesheet" href="../../../assets/mobile.css" media="screen and (max-width: 600px)">
    <link rel="stylesheet" href="../../../assets/desktop.css" media="screen and (min-width: 601px)">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css" rel="stylesheet">
</head>
<body>
    <?php include 'header.php'; ?>

    <div class="container">
        <h1>CritMeet</h1><br>

        <?php if (!empty($success_message)): ?>
            <div class="alert alert-success d-flex align-items-center">
                <i class="bi bi-check-circle-fill me-2"></i>
                <div><?php echo $success_message; ?></div>
            </div>
        <?php endif; ?>

        <form method="POST" action="">
            <input type="text" name="email" placeholder="Email ou Username" required /><br>
            <input type="password" name="senha" placeholder="Senha" required /><br>
            <button type="submit">Entrar</button><br>
        </form>
        <a href="../register/">
            <button type="button">Cadastre-se</button><br>
        </a>
        <button type="button">Google</button>

        <?php if (isset($error)) { echo "<p style='color:red;'>$error</p>"; } ?>
    </div>

    <?php include 'footer.php'; ?>
</body>
</html>
```

Fonte: elaborado pelos autores.

Para a camada de interatividade, foi utilizada a linguagem JavaScript (ES6+), permitindo a manipulação dinâmica de elementos da interface e o controle de ações do usuário. Funções como atualizações em tempo real, troca de abas, validações de formulário e notificações visuais foram implementadas para aumentar a fluidez da navegação.

Além disso, foi adotada a biblioteca jQuery 3.6.0, especialmente em funcionalidades que envolvem requisições AJAX, como a troca de mensagens no sistema de chat, curtidas em perfis e busca de jogadores próximos. Essa integração com o back-end em PHP permitiu uma comunicação assíncrona entre cliente e servidor, melhorando o desempenho e a experiência de uso.

Figura 67 - Front-end: Funcionalidades com atualização em tempo real

Fonte: elaborado pelos autores.

O design visual do CritMeet também incorporou a biblioteca Bootstrap Icons 1.11.3, responsável pela uniformização de ícones utilizados em botões, menus e indicadores de status. Essa padronização contribuiu para uma interface mais limpa, moderna e intuitiva.

Outro ponto de destaque foi a implementação de validações em tempo real no front-end, que forneceram feedback imediato ao usuário durante o preenchimento de formulários, reduzindo erros e aumentando a eficiência da navegação.

Figura 68 - Front-end: Formulário com validação visual na edição de perfil

```php
if ($stmt->execute()) {
    $success_message = "Informações do usuário atualizadas com sucesso!";
    // Recarregar dados do usuário
    $query = "SELECT * FROM users WHERE id = ?";
    $stmt2 = $mysqli->prepare($query);
    $stmt2->bind_param("i", $user_id);
    $stmt2->execute();
    $result = $stmt2->get_result();
    if ($result && $row = $result->fetch_assoc()) {
        $user = $row;
    }
} else {
    $error_message = "Erro ao atualizar as informações do usuário.";
}
```

Fonte: elaborado pelos autores.

8.3.1 PÁGINAS

A organização da estrutura de páginas no projeto CritMeet foi fundamental para garantir a clareza, manutenção e escalabilidade do código-fonte ao longo do desenvolvimento. Para isso, cada página do sistema foi alocada dentro de diretórios específicos, seguindo uma divisão lógica por funcionalidades. Essa abordagem permitiu que os arquivos relacionados a uma mesma funcionalidade, como o código da página, os estilos CSS e os scripts JavaScript fossem mantidos juntos, facilitando tanto a leitura quanto a atualização futura do sistema.

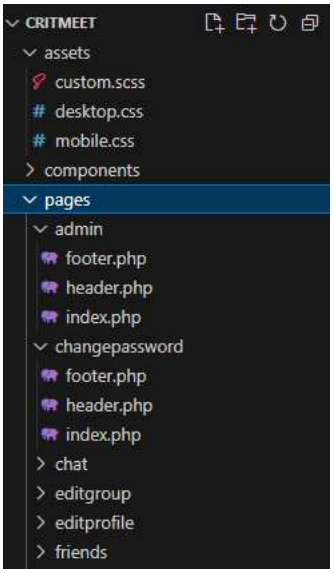
Em cada diretório, o arquivo principal é o index.php, responsável por estruturar o conteúdo da página. Dentro desse arquivo, são feitas as chamadas para os arquivos header.php e footer.php, que contém os elementos de cabeçalho e rodapé padrão da aplicação. Essa separação facilita a manutenção de elementos comuns, pois qualquer alteração no layout ou nos links de navegação, por exemplo, pode ser feita de forma centralizada apenas no header.php e footer.php, refletindo automaticamente em todas as páginas do sistema.

Além disso, dentro do corpo do index.php, são chamadas as funções universais da aplicação, que incluem recursos como autenticação de usuários, carregamento de componentes reutilizáveis (menus, perfis, sistema de mensagens)

e validações de sessão. Esse formato permite que cada página carregue de forma dinâmica apenas os módulos necessários, evitando sobrecarga e mantendo a eficiência no carregamento.

A seguir, um exemplo da estrutura adotada:

Figura 69 - Páginas do CritMeet



Fonte: elaborado pelos autores.

A estilização visual (CSS) de cada página também está localizada dentro de seu respectivo diretório, garantindo que cada seção da aplicação possua um estilo próprio, sem interferir no layout das demais páginas. Essa divisão favorece a personalização de cada área, ao mesmo tempo que mantém o projeto organizado e de fácil leitura para futuros ajustes.

Essa estratégia de estruturação, baseada em modularidade e reaproveitamento de código, não apenas facilitou o desenvolvimento da aplicação, como também tornou o sistema mais escalável, permitindo a adição de novas páginas e funcionalidades de forma simples e eficiente.

Figura 70 - Exemplo de estrutura de página

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Home</title>
    <link rel="stylesheet" href="../../../assets/mobile.css" media="screen and (max-width: 600px)">
    <link rel="stylesheet" href="../../../assets/desktop.css" media="screen and (min-width: 601px)">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.11.3/font/bootstrap-icons.min.css" rel="stylesheet">
</head>
<body>
    <?php include 'header.php';
    require_once __DIR__ . '/../../config.php';
    ?>

    <div class="container">
        <h1>CritMeet</h1>
        <h2>Saudações!</h2>
        <h2>Vamos começar a sua jornada!</h2>
        <a href="../../login/">
            <button type="button">INICIAR</button>
        </a>
    </div>

    <?php include 'footer.php'; ?>

</body>
</html>
```

Fonte: elaborado pelos autores.

8.3.2 CONSUMO DA API

Todas as páginas da aplicação foram pensadas para manter consistência visual e funcional, respeitando a hierarquia de informações e facilitando o uso por jogadores iniciantes e experientes. O front-end também foi projetado para se integrar com APIs externas, como o Leaflet.js e o FullCalendar, sem comprometer a performance.

Figura 71 - Script do calendário integrado

```
<script>
    let calendar;

    document.addEventListener('DOMContentLoaded', function() {
        document.getElementById('scheduledSessions').addEventListener('shown.bs.collapse', function () {
            if (!calendar) {
                <?php echo $calendar->getCalendarScript(); ?>
                initializeCalendar();
            }
        });
    });
</script>
```

Fonte: elaborado pelos autores.

9 CONSIDERAÇÕES FINAIS

O projeto CritMeet surgiu com a proposta de ser uma solução no cenário dos jogos de RPG de mesa, visando conectar mestres e jogadores por meio de uma plataforma digital acessível e funcional. Inspirado em mecanismos de aplicativos de relacionamento, o sistema foi idealizado para facilitar a formação de grupos de jogo, promovendo a inclusão, a praticidade e a descoberta de novas mesas de jogo com base em interesses, localização e disponibilidade. Ao longo do processo de concepção, ficou evidente a necessidade de mais recursos de modo que pudéssemos entregar não somente uma aplicação funcional, mas também que atendesse às necessidades dos jogadores de RPG de mesa.

Durante o desenvolvimento do CritMeet, foram aplicados conceitos técnicos fundamentais que abrangeram desde a estruturação do banco de dados até a definição detalhada dos fluxos de interação do sistema. Através do uso do MySQL Workbench, a equipe elaborou um modelo entidade-relacionamento (MER), garantindo a integridade, consistência e escalabilidade dos dados, que serviram como alicerce para a implementação de funcionalidades mais avançadas. Simultaneamente, a elaboração de diagramas de sequência e diagramas de caso de uso contribuiu para o mapeamento e validação dos comportamentos previstos para os usuários, permitindo identificar possíveis falhas e promover melhorias na usabilidade desde as etapas iniciais do projeto.

Outro aspecto relevante foi a definição da arquitetura do sistema, que foi planejada com foco na escalabilidade e na flexibilidade para futuras expansões. Embora o desenvolvimento inicial tenha sido direcionado para o ambiente web, desde as fases iniciais considerou-se a possibilidade de adaptação para dispositivos móveis, reconhecendo que grande parte do público-alvo utiliza smartphones como principal meio de acesso a esse tipo de aplicação. Essa escolha estratégica garante que, no futuro, o CritMeet possa receber novas funcionalidades e ser disponibilizado em diferentes plataformas, ampliando seu alcance e facilitando o acesso por um número ainda maior de usuários.

A distribuição das tarefas entre os integrantes da equipe foi realizada de

acordo com as habilidades técnicas individuais, o que contribuiu para um fluxo de trabalho mais ágil e colaborativo. Essa divisão estratégica permitiu o desenvolvimento paralelo de diferentes camadas do sistema, incluindo a interface do usuário, a lógica de negócios e o gerenciamento de dados. Além disso, a troca contínua de ideias e o compartilhamento de feedbacks durante todas as fases do projeto criaram um ambiente de trabalho dinâmico e criativo, onde as decisões foram baseadas em testes práticos e observações de desempenho.

Durante o desenvolvimento, a equipe enfrentou diversos desafios, como questões de compatibilidade entre tecnologias, dificuldades na integração dos diferentes componentes do sistema e a necessidade de revisões constantes nas funcionalidades para alinhá-las melhor às expectativas dos usuários. Apesar desses obstáculos, o resultado final foi um sistema coeso e alinhado à proposta inicial, que não apenas contempla as funcionalidades planejadas, mas também se apresenta como uma plataforma sólida e com potencial para expansão, visando a real integração nos espaços da comunidade de jogadores de RPG de mesa.

Em conclusão, o projeto evidenciou como a combinação entre uma ideia inovadora, o uso de ferramentas adequadas e o comprometimento de uma equipe engajada pode resultar em um produto funcional e com aplicabilidade real. O CritMeet vai além de um simples trabalho acadêmico, configurando-se como uma solução concreta para atender as necessidades de uma comunidade em constante crescimento, que busca espaços mais eficientes para encontro e organização de grupos de RPG de mesa. As fases de concepção, modelagem, desenvolvimento e testes proporcionaram à equipe um aprendizado técnico e interpessoal significativo, reforçando a importância da interdisciplinaridade, do planejamento estruturado e da escuta ativa durante o processo de desenvolvimento de software.

REFERÊNCIAS

ALFF, Francilvio Roberto. **Especificação de casos de uso: 5 passos**. Análise de Requisitos. 2024. Disponível em: https://analisederequisitos.com.br/como-escrever-casos-de-uso/. Acesso em: 7 jun. 2025.

ALMENARA, Igor. **Como funciona a seção "pessoas que você talvez conheça" no Facebook**. Canal Tech. 2021. Disponível em: https://canaltech.com.br/redes-sociais/como-funciona-a-secao-pessoas-que-talvez-voce-conheca-no-facebook-202979/. Acesso em: 11 nov. 2024.

ASPROFISSÕES. **O que é entidade.** As Profissões. 2024. Disponível em: https://asprofissoes.com/engenharia-de-software/glossario/o-que-e-entidade-definicao-e-importancia/. Acesso em: 14 jun. 2025.

AWARI. **O que é front-end:** guia completo para iniciantes em tecnologia. Awari. 2023. Disponível em: https://awari.com.br/o-que-e-front-end-guia-completo-para-iniciantes-em-tecnologia/. Acesso em: 16 jun. 2025.

AWS. **O que é o Scrum?**. AWS. 2024. Disponível em: https://aws.amazon.com/pt/what-is/scrum/. Acesso em: 10 jun. 2025.

AWS. **O que é uma aplicação web?**. AWS. 2024. Disponível em: https://aws.amazon.com/pt/what-is/web-application/. Acesso em: 2 jun. 2025.

CHAI, Wesley. ***Data dictionary*.** TechTarget. 2022. Disponível em: https://www.techtarget.com/searchapparchitecture/definition/data-dictionary. Acesso em: 12 jun. 2025.

DEBIAN. **Sobre o Debian.** Debian. 2025. Disponível em: https://www.debian.org/intro/about.pt.html. Acesso em: 12 jun. 2025.

DEVMEDIA. **Modelagem de dados 1:** Entidades. DevMedia. 2007. Disponível em: https://www.devmedia.com.br/modelagem-de-dados-1-entidades/4140. Acesso em: 14. jun. 2025.

DIGITALOCEAN. ***About*.** DigitalOcean. 2025. Disponível em: https://www.digitalocean.com/about. Acesso em: 12 jun. 2025.

DIGITALOCEAN. ***Droplets*.** DigitalOcean. 2025. Disponível em: https://www.digitalocean.com/products/droplets. Acesso em: 12 jun. 2025.

ERICKSON, Jeffrey. **MySQL:** Entendendo o que é e como é usado. Oracle. 2024. Disponível em: https://www.oracle.com/br/mysql/what-is-mysql/. Acesso em: 13 jun. 2025.

ESCOLADNC. **Entendendo os tipos de cardinalidade em modelagem de banco de dados.** Escola DNC. 2025. Disponível em: https://www.escoladnc.com.br/blog/entendendo-os-tipos-de-cardinalidade-em-modelagem-de-bancos-de-dados. Acesso em: 11 jun. 2025.

FIGMA. **Sobre nós.** Figma. 2025. Disponível em: https://www.figma.com/pt-br/about/. Acesso em: 12 jun. 2025.

FLORIANO, Jeronima. **O que são requisitos de software e porquê é importante entendê-los.** DevCommunity. 2023. Disponível em: https://dev.to/jeronimafloriano/o-que-sao-requisitos-de-software-e-porque-e-importante-entende-los-bjn. Acesso em: 30 mai. 2025

GEEKSFORGEEKS. **Git Bash.** Geeks for Geeks. 2024. Disponível em: https://www.geeksforgeeks.org/git/working-on-git-bash/. Acesso em: 14 jun. 2025.

GITHUB. ***Why choose GitHub?***. GitHub. 2025. Disponível em: https://github.com/why-github. Acesso em: 16 jun. 2025.
GOUML. ***Class Diagram Tutorial:*** *A Comprehensive Guide*. GoUML. 2025. Disponível em: https://www.go-uml.com/class-diagram-tutorial-a-comprehensive-guide/. Acesso em: 11 jun. 2025.

HAWKES-ROBINSON, William. ***Improving Access to Tabletop Role-Playing Games by Overcoming High Barriers to Introductory Play***. [*S. l.*], 2020. Preprint. Disponível em: https://www.researchgate.net/publication/352555006. Acesso em: 10 jun. 2025.

HILLMAN, Mônica. **HTML:** o que é, importância para a web, como aprender e um guia para iniciantes. Alura. 2023. Disponível em: https://www.alura.com.br/artigos/html. Acesso em: 12 jun. 2025.

HOSTMÍDIA. **HTTPS:** o que é, como funciona e por que é importante?. HostMídia. 2025. Disponível em: https://www.hostmidia.com.br/blog/o-que-e-https/. Acesso em: 14 jun. 2025.

IBM. **Diagramas de Caso de Uso**. IBM. 2021. Disponível em: https://www.ibm.com/docs/pt-br/rsm/7.5.0?topic=diagrams-use-case. Acesso em: 4 jun. 2025.

LEPCHA, Mensholong. ***What is SHA-256?***. Techopedia. 2023. Disponível em: https://www.techopedia.com/definition/sha-256. Acesso em: 15 jun. 2025.

LIMA, Guilherme. **Bootstrap:** o que é, documentação, como e quando usar. Alura. 2021. Disponível em: https://www.alura.com.br/artigos/bootstrap. Acesso em: 12 jun. 2025.

LUCIDCHART. **O que é um diagrama de sequência UML?**. Lucidchart. 2025. Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-diagrama-de-sequencia-uml. Acesso em: 10 jun. 2025.

LUCIDCHART. **O que é um diagrama UML?**. Lucidchart. 2025. Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-uml. Acesso em: 30 mai. 2025.

MARTINS, Renata. **Qual a importância da tecnologia e seus benefícios?**. Grupo ABL. 2023. Disponível em: https://www.grupoabl.com.br/post/qual-a-import%C3%A2ncia-da-tecnologia-e-seus-benef%C3%ADcios. Acesso em: 28 mai. 2025.

MATHIAS, Lucas. **Qual a diferença entre pesquisa qualitativa e pesquisa quantitativa:** guia completo. MindMiners. 2022. Disponível em: https://mindminers.com/blog/pesquisa-qualitativa-quantitativa/. Acesso em: 13 jun. 2025.

MDN. **CSS.** MDN. 2025. Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/CSS. Acesso em: 12 jun. 2025.

MIRO. **Modelo de protótipo de baixa fidelidade.** Miro. 2025. Disponível em: https://miro.com/pt/modelos/prototipo-baixa-fidelidade/. Acesso em: 29 mai. 2025.

MYSQL. **Workbench.** MySQL. 2025. Disponível em: https://www.mysql.com/products/workbench/. Acesso em: 15 jun. 2025.

NGUYEN, Thien. ***Understanding .htpasswd:*** *A Comprehensive Guide*. DevCommunity. 2024. Disponível em: https://dev.to/quangthien27/understanding-htpasswd-a-comprehensive-guide-4c31. Acesso em: 14 jun. 2025.

ORACLE. **O que é um banco de dados?**. Oracle. 2020. Disponível em: https://www.oracle.com/br/database/what-is-database/. Acesso em 15 jun. 2025.

PHP. **O que é o PHP e o que ele pode fazer?**. PHP. 2025. Disponível em: https://www.php.net/manual/pt_BR/introduction.php. Acesso em: 12 jun. 2025.

QUEVEDO, Gabriele. **Explorando o AJAX:** como criar aplicações web dinâmicas e interativas. Medium. 2022. Disponível em: https://medium.com/@gabrielequevedo/o-que-%C3%A9-ajax-27d44dde12. Acesso em: 14 jun. 2025.

ROSA, Ângela. **Protótipo:** entenda o que é, tipos exemplos e como fazer na prática!.SoftDesign. 2024. Disponível em: https://softdesign.com.br/blog/prototipo-baixa-e-alta-fidelidade/. Acesso em: 29 mai. 2025.

RPGMAISBARATO. **A origem do RPG de mesa**. RPG Mais Barato. 2018. Disponível em: https://rpgmaisbarato.com/blog/rpg-de-mesa-origem-como-jogar/. Acesso em: 9 nov. 2024.

SANTANA, Eduardo Felipe. **Back-end:** o que é e um guia para iniciar a área. Alura. 2024. Disponível em: https://www.alura.com.br/artigos/backend. Acesso em: 15 jun. 2025.

SCIENCEDIRECT. ***Entity Relationship Model.*** ScienceDirect. 2025. Disponível em: https://www.sciencedirect.com/topics/computer-science/entity-relationship-model. Acesso em: 11 jun. 2025.

SOUSA, Priscila. **JavaScript - O que é, funcionamento, conceito e definição**. Conceito.de. 2022. Disponível em: https://conceito.de/javascript. Acesso em: 12 jun. 2025.

VISUALSTUDIOCODE. ***Why did we build Visual Studio Code?***. Visual Studio Code. 2025. Disponível em: https://code.visualstudio.com/Docs/editor/whyvscode. Acesso em: 16 jun. 2025.

VPNUNLIMITED. **Bcrypt.** VPN Unlimited. [20--]. Disponível em: https://www.vpnunlimited.com/pt/help/cybersecurity/bcrypt. Acesso em: 15 jun. 2025.

WALSH, O.; LINEHAN, C. ***Roll for Insight:*** *Understanding How the Experience of Playing Dungeons & Dragons Impacts the Mental Health of an Average Player*. International Journal of Role-Playing. 2024. Disponível em: https://journals.uu.se/IJRP/article/view/321. Acesso em: 13 jun. 2025.

YLÖNEN, Tatu. ***What is SSH (Secure Shell)?***. Secure Shell. 2024. Disponível em: https://www.ssh.com/academy/ssh. Acesso em: 16 jun. 2025.