Universidade Federal do Pará
Campus Universitário de Castanhal - CCAST
Faculdade de Computação - FACOMP
Bacharelado em Engenharia de Computação

Modelagem e implementação de um ambiente simulado para atendimento ao cliente de uma plataforma de gestão de imóveis

Karol Wojtyla Sousa Nascimento

UFPA / CCAST / FACOMP
Campus Universitário de Castanhal
Castanhal - Pará - Brasil
2025

Universidade Federal do Pará

Campus Universitário de Castanhal - CCAST

Faculdade de Computação - FACOMP

Bacharelado em Engenharia de Computação

Karol Wojtyla Sousa Nascimento

Modelagem e implementação de um ambiente simulado para atendimento ao cliente de uma plataforma de gestão de imóveis

Monografia submetida à avaliação da Banca Examinadora aprovada pelo colegiado da Faculdade de computação da Universidade Federal do Pará e julgada adequada para a obtenção do Grau de Bacharelado em Engenharia de Computação.

Orientador: Profa. Dra. Yomara Pinheiro Pires

UFPA / CCAST / FACOMP

Campus Universitário de Castanhal

Castanhal - Pará - Brasil

2025

UNIVERSIDADE FEDERAL DO PARÁ
CAMPUS UNIVERSITÁRIO DE CASTANHAL - CCAST
FACULDADE DE COMPUTAÇÃO - FACOMP
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

Modelagem e implementação de um ambiente simulado para atendimento ao cliente de uma plataforma de gestão de imóveis

Autor: Karol Wojtyla Sousa Nascimento

Monografia apresentada à Faculdade de Computação e aprovada para a obtenção do título de Bacharel em Engenharia de Computação.

APROVADO EM: 16 de setembro de 2025.

Banca Examinadora:

Profa. Dra. Yomara Pinheiro Pires
Orientador FACOMP/CCAST/UFPA

Prof.ª Drº. Igor Ruiz Gomes
(Avaliador Interno – FACOMP/CCAST/UFPA)

Prof.ª Drº. José Jailton Henrique Ferreira Júnior
(Avaliador Interno – FACOMP/CCAST/UFPA)

Visto:

Profª. Drª. Yomara Pinheiro Pires
(Diretora da Faculdade de Computação/CCAST/UFPA)

Castanhal - Pará - Brasil

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISBD
Sistema de Bibliotecas da Universidade Federal do Pará
Gerada automaticamente pelo módulo Ficat, mediante os dados fornecidos pelo(a) autor(a)

N244m Nascimento, Karol Wojtyla Sousa.
Modelagem e implementação de um ambiente simulado para atendimento ao cliente de uma plataforma de gestão de imóveis / Karol Wojtyla Sousa Nascimento. — 2025.
63 f. : il. color.

Orientador(a): Profª. Dra. Yomara Pinheiro Pires
Trabalho de Conclusão (Graduação) - Universidade Federal do Pará, Campus Universitário de Castanhal, Faculdade de Engenharia da Computação, Castanhal, 2025.

1. Engenharia de Software. 2. Atendimento ao cliente. 3. Ambientes simulados. 4. Suporte técnico. 5. LGPD. I. Título.

CDD 005.12

Dedico este trabalho a todos que foram meu lar, desde o início de tudo isso. A Deus, pelo dom da vida e por tudo que operou nela nestes cinco anos. A meus pais, Sônia e Eduardo, e meus irmãos, Kauã e Karoline, que são a minha maior motivação para continuar, todos os dias, a ser o melhor de mim. A Camile, por toda a parceria e companheirismo mútuo dado nesses três anos 'e pouquinho'. Aos amigos, família e colegas de jornada. A vocês, que me mostraram que a minha casa não tem parede, não tem janela, nem tem porta pra trancar. Levo comigo o privilégio de ter encontrado, em cada um de vocês, o meu verdadeiro lar.

Agradecimentos

Tenho uma memória muito marcante de quando era criança: estava deitado no colo do meu pai, no intervalo de alguma novela das 21h da Globo, naquelas TVs de tubo grandonas, quando começou a passar um comercial de universidade. É estranho, mas naquele momento senti um aperto enorme no peito e comecei a chorar. Minha mãe, assustada, perguntou: "Que foi, menino?!" e tudo que eu consegui responder foi: "Mãe, eu não sei se vou conseguir fazer uma faculdade".

A cabeça de uma criança de nove anos é mesmo peculiar. Mas, de certo modo, talvez eu não estivesse tão errado assim. Naquele momento, eu realmente não conseguia fazer uma faculdade. Não conseguia apresentar um seminário. Não conseguia escrever um TCC. Mas hoje, quase treze anos depois, estamos aqui, né? Mas esse trabalho só é possível não porque eu fiquei mais inteligente ou porque algo mágico aconteceu comigo. É possível porque, nesse caminho, eu encontrei a minha maior força – e o mais bonito é que ela não vem de dentro de mim, mas de fora.

Por isso, agradeço a Deus, meu alicerce em tudo isso. Agradeço por não ter me deixado desistir em nenhum momento nesses cinco anos. Por ter me guardado e me guiado, em Sua infinita graça, no Seu caminho, que apesar de estreito, me levou à vida. E eu sou eternamente grato por tê-lo trilhado.

Agradeço à minha família: meu pai, Eduardo; minha mãe, Sônia; e meus irmãos, Kauã e Karoline – mais conhecida pelo nome artístico "Fofinha". Vocês foram os pilares que me sustentaram e me mantiveram motivado. Quando a saudade apertava e tudo parecia incerto, eram vocês que, mesmo sem perceber, me lembravam da razão de eu viver tudo isso. É por vocês. Sempre foi por vocês.

Agradeço à Camile, por toda a parceria e companheirismo compartilhados desde que eu coloquei os pés em Castanhal. Obrigado por me fazer enxergar uma versão de mim que talvez eu jamais conhecera. Obrigado por ter sido luz nos meus dias mais sombrios; calor nos dias mais frios; e nos dias de luz e brilho, obrigado por ter sido você. Obrigado por ter sido lar.

Agradeço aos amigos e colegas que fizeram parte dessa caminhada e tornaram essa jornada mais leve. Aridan, Beatriz, Leandro, Lays, Ítalo e Geovane, obrigado por terem me aturado e me ouvido nesses anos. Carrego vocês com muito carinho aqui dentro. Sem vocês eu teria um pouco mais de QI – mas definitivamente seria uma pessoa mais triste.

Agradeço à Yomara, minha orientadora, que não apenas guiou este trabalho, mas foi um dos meus principais pontos de apoio quando eu tinha uma ideia mirabolante de

trabalho, projeto ou simplesmente um devaneio. Obrigado por toda a paciência, pela parceria e pela confiança até aqui. Eu me orgulharia imensamente se um dia conseguir ser, ao menos, metade do profissional – e da pessoa – que você é.

Agradeço também à Seazone, em especial aos colegas Roberto Campos, Natália Bessa, a todo o time de hosting e a cada pessoa que colaborou para que este trabalho fosse possível. Este trabalho se torna mais especial por todo o apoio e confiança depositados em mim e no meu trabalho. Sou profundamente grato por ter encontrado, no trabalho, esse espaço de acolhida.

Por último, mas não menos importante, agradeço à UFPA. Afinal, se não fosse por ela, este trabalho simplesmente não existiria. Minha gratidão a todos os professores, técnicos, auxiliares, equipes de apoio, bolsistas e a cada pessoa que, dia após dia, faz essa engrenagem girar. Obrigado por manterem viva a missão de transformar vidas por meio da educação.

Obrigado, pessoal, por mostrarem ao Karol de 9 anos que ele não precisava ter medo, porque, um dia, estaria cercado de pessoas incríveis que o ajudariam a chegar até aqui. Hoje, posso dizer com orgulho: nós conseguimos. De coração, obrigado.

"Incendeie o seu coração."

Kyojuro Rengoku

Resumo

O atendimento ao cliente é um fator estratégico para empresas de base tecnológica, sobretudo em plataformas digitais de gestão de imóveis por temporada, nas quais a eficiência do suporte impacta diretamente a experiência do usuário. Muitas vezes, a equipe de atendimento não possui aparatos técnicos suficientes para reproduzir problemas relatados, o que gera atrasos, sobrecarga nos times de desenvolvimento e insatisfação dos clientes. Diante desse cenário, este trabalho apresenta a modelagem e implementação de um ambiente simulado capaz de reproduzir a perspectiva do cliente de forma segura, eliminando a necessidade de acesso direto às suas credenciais e garantindo conformidade com a Lei Geral de Proteção de Dados (LGPD).

A pesquisa é classificada como aplicada, de natureza exploratória, experimental e descritiva. O desenvolvimento da solução foi guiado por um levantamento de requisitos junto ao suporte técnico, culminando na criação da ferramenta "Perspectiva do Anfitrião". Essa ferramenta foi implementada utilizando React e TypeScript no front-end, Django e PostgreSQL no back-end, além de Docker e Amazon Web Services (AWS) para provisionamento. O sistema opera em modo somente visualização, permitindo ao suporte acessar a interface da aplicação exatamente como o cliente, sem riscos de manipulação indevida.

Os resultados mostraram que a ferramenta proporciona ganhos em eficiência e objetividade, permitindo diagnósticos mais ágeis, redução no tempo de resposta e encaminhamentos mais claros ao time técnico. Recursos como autenticação via credenciais institucionais, pesquisa de anfitriões, lista de favoritos e visualização mobile ampliaram a usabilidade. Constatou-se ainda a mitigação de riscos relacionados à privacidade, uma vez que não há uso de credenciais reais.

Conclui-se que a solução desenvolvida contribui para melhorar a qualidade do suporte em empresas de gestão de imóveis por temporada, podendo servir como modelo para outras organizações que demandem ambientes simulados seguros e eficazes.

Palavras-chaves: Engenharia de Software. Atendimento ao cliente. Ambientes simulados. Suporte técnico. LGPD.

Abstract

Customer service is a strategic factor for technology-based companies, especially on digital vacation rental management platforms, where efficient support directly impacts the user experience. Often, the support team lacks the necessary technical tools to reproduce reported issues, which leads to delays, an overload on development teams, and customer dissatisfaction. Given this scenario, this paper presents the modeling and implementation of a simulated environment capable of safely reproducing the customer's perspective, eliminating the need for direct access to their credentials and ensuring compliance with the General Data Protection Law (LGPD).

The research is classified as applied, with an exploratory, experimental, and descriptive nature. The development of the solution was guided by a requirements survey with the technical support team, culminating in the creation of the "Host Perspective" tool. This tool was implemented using React and TypeScript for the front-end, Django and PostgreSQL for the back-end, and Docker and Amazon Web Services (AWS) for provisioning. The system operates in view-only mode, allowing support to access the application's interface exactly as the customer would, without the risk of improper manipulation.

The results showed that the tool provides gains in efficiency and objectivity, allowing for faster diagnoses, reduced response times, and clearer referrals to the technical team. Features such as authentication via institutional credentials, host search, a favorites list, and mobile viewing enhanced usability. It was also found that it mitigates privacy-related risks, since real credentials are not used.

It is concluded that the developed solution contributes to improving the quality of support in vacation rental management companies and can serve as a model for other organizations that require secure and effective simulated environments.

Keywords: Software Engineering. Customer service. Simulated environments. Technical support. LGPD.

Lista de Figuras

Figura 1 – Componentes do *TypeScript* 18
Figura 2 – Relação conjuntiva entre JavaScript e *TypeScript* 19
Figura 3 – Exemplo de estilização dinâmica com Styled Components 21
Figura 4 – Exemplo de fluxo de informações em um sistema web 21
Figura 5 – Fluxo de uma REST API 22
Figura 6 – Exemplo de utilização do Redis 25
Figura 7 – Funcionamento do acesso a uma aplicação web 26
Figura 8 – Processo de construção de um conteiner com Docker 27
Figura 9 – Arquitetura *as is* da solução 29
Figura 10 – Fluxo básico de login com JWT 30
Figura 11 – Fluxo de autenticação como *view-only* 31
Figura 12 – Fluxo esperado de jornada do usuário 34
Figura 13 – Tela de *login* da aplicação completa 35
Figura 14 – Página inicial da aplicação 35
Figura 15 – Visão geral da Perspectiva do Anfitrião 36
Figura 16 – Exemplo de pesquisa por anfitrião 37
Figura 17 – Tela de lista de anfitriões favoritos 38
Figura 18 – Tela de carregamento da perspectiva do anfitrião 39
Figura 19 – Tela de perspectiva do anfitrião autenticada 39
Figura 20 – Tela de perspectiva do anfitrião na versão *mobile* 40

Lista de Tabelas

Tabela 1 – Requisitos de desenvolvimento da solução 16
Tabela 2 – Funcionalidades acessíveis na Perspectiva do Anfitrião 41

Lista de abreviaturas e siglas

API	Application Programming Interfaces
AWS	Amazon Web Services
B2B	Business to Business
B2C	Business to Customer
CS	Customer Success
CSS	Cascading Style Sheets
DOM	Document Object Model
EC2	Amazon Elastic Compute Cloud
FCR	First Call Resolution
GDPR	Regulamento Geral de Proteção de Dados (*General Data Protection Regulation*)
HTTP	Hypertext Transfer Protocol
JS	JavaScript
JSON	JavaScript Object Notation
JWT	JSON Web Token
LGPD	Lei Geral de Proteção de Dados
ORM	Object Relational Mapper
OTA	Online Travel Agencies
QA	Quality Assurance
REST	Representational State Transfer
SaaS	Software as a Service
SGBD	Sistema de Gerenciamento de Banco de Dados
SQL	Structured Query Language
TI	Tecnologia da Informação
URL	Uniform Resource Locator

Sumário

1 INTRODUÇÃO 1
1.1 Objetivos 2
1.1.1 Objetivos específicos 2
1.2 Organização do trabalho 3

2 FUNDAMENTAÇÃO TEÓRICA 4
2.1 Considerações iniciais sobre o capítulo 4
2.2 Caracterização da empresa de short-stay 4
2.3 Atendimento ao cliente 6
2.4 Ambientes de teste para simulação 8
2.5 Segurança da informação e conformidade com LGPD 10
2.6 Considerações finais sobre o capítulo 12

3 MATERIAIS E MÉTODOS 14
3.1 Considerações iniciais sobre o capítulo 14
3.2 Metodologia 14
3.3 Levantamento de requisitos 15
3.4 Projeto e desenvolvimento da solução 17
3.4.1 Front-End 17
3.4.1.1 TypeScript 17
3.4.1.2 ReactJS 19
3.4.1.3 Vite 20
3.4.1.4 Styled Components 20
3.4.2 Back-End 21
3.4.2.1 REST APIs 22
3.4.2.2 Django 23
3.4.2.3 PostgreSQL 24
3.4.2.4 Redis 25
3.4.2.5 Celery 25
3.4.3 Infraestrutura e provisionamento da solução 26
3.4.3.1 Docker 27
3.4.3.2 Amazon Web Services (AWS) 27
3.5 Arquitetura da solução 28
3.6 Fluxo de autenticação 30
3.7 Considerações finais sobre o capítulo 32

4 RESULTADOS E DISCUSSÃO 33
4.1 Considerações iniciais sobre o capítulo 33
4.2 Validação funcional da ferramenta 33
4.2.1 Autenticação em ambiente produtivo 34
4.2.2 Acesso à ferramenta de simulação 36
4.2.3 Visualização no ambiente simulado 38
4.3 Síntese dos resultados 41
4.3.1 Contribuições para a eficiência do suporte técnico 42
4.3.2 Segurança e conformidade 42
4.4 Considerações finais sobre o capítulo 43

5 CONCLUSÕES 44

REFERÊNCIAS 46

1 Introdução

O atendimento ao cliente sempre foi um setor essencial no meio corporativo, mesmo antes das transformações industriais mais recentes, que intensificaram o volume de interações e aceleraram o fluxo de informações. Alves (2017) reflete que a globalização, o avanço da tecnologia e a informação em abundância tornaram o cliente mais exigente e consciente em seu consumo, forçando as empresas a adotarem novas estratégias para manterem-se ativas no mercado.

Em concordância a essa afirmação, Schon (2022) ressalta que a gestão de atendimento ao cliente com qualidade e assertividade está intimamente relacionada ao sucesso de uma organização; e sugere que processos de relacionamento com o cliente sejam sempre reinventados, garantindo um atendimento mais alinhado tanto às suas necessidades declaradas quanto às reais.

No universo da tecnologia da informação, o cenário não é diferente: os recursos tecnológicos presentes no ecossistema das empresas de TI são constantemente adaptados e integrados para aprimorar e tornar mais eficientes as interações entre marcas e consumidores. O objetivo desse movimento organizacional é reduzir gargalos que comprometem o tempo de espera, tornando o atendimento mais eficiente, assertivo e personalizado. Além disso, busca-se otimizar a *First Call Resolution* (FCR), uma métrica essencial que avalia a capacidade de resolver solicitações no primeiro contato, melhorando a experiência do cliente e a eficácia do atendimento (ZENDESK, 2023).

Essa necessidade por assertividade e qualidade no atendimento se agrava quando o segmento de negócio da empresa em questão é um SaaS[1] ou possui uma ferramenta digital de apoio ao negócio. Isso porque as equipes de atendimento nem sempre tem o conhecimento técnico ou são treinadas para lidar com situações mais complexas ou pontuais, que envolvam uma análise mais próxima do usuário. Esse entrave, por conseguinte, gera um atraso no atendimento, aumenta a frustração do cliente e acarreta na geração de maiores cargas de trabalho aos times técnicos (SOUZA, 2024).

Como solução para esse viés, empresas de tecnologia têm investido no desenvolvimento de ambientes simulados que reproduzem a experiência do cliente, proporcionando um atendimento mais assertivo. Nesse sentido, Souza (2024) apresenta o conceito de "simulação espelho/modelo", definindo-o como "uma técnica utilizada em ambientes de desenvolvimento para validar o funcionamento de tecnologias ou recursos em cenários similares ou idênticos aos do cliente".

Esses ambientes simulados permitem que as equipes de suporte testem fluxos de

[1] Sigla para "*Software as a Service*" ou *Software* como Serviço

usabilidade das ferramentas, reproduzindo com precisão os passos do cliente para identificar possíveis lacunas de forma segura, sem impactar a experiência do usuário final (SOUZA, 2024). Ao recriar cenários autênticos, a equipe pode oferecer respostas mais ágeis, seja solucionando diretamente a demanda do cliente ou fornecendo ao time técnico insumos bem contextualizados sob a ótica da engenharia, reduzindo o tempo investido em *discoverys* técnicos.

Diante deste contexto, este trabalho visa desenvolver uma solução que possibilite a simulação do ambiente do cliente para equipes de suporte técnico de uma empresa de aluguel de imóveis por temporada, sediada em Florianópolis/SC e com atuação em mais de 11 estados brasileiros.

A proposta visa aprimorar a eficiência e a objetividade no atendimento ao cliente, eliminando a necessidade de acesso direto às credenciais dos usuários e garantindo conformidade com as normas de proteção de dados, preservando sua privacidade e segurança.

Além disso, este estudo explora os impactos dessa abordagem na otimização de processos internos, redução do tempo de resposta e aprimoramento da experiência do usuário. Ao enfrentar essa problemática, a pesquisa contribui para o desenvolvimento de práticas mais seguras e eficazes no suporte a plataformas de gestão de imóveis por temporada, fortalecendo a confiabilidade e a competitividade do serviço.

1.1 Objetivos

Este trabalho tem por objetivo desenvolver um solução que simule o ambiente do cliente para equipes de suporte técnico de uma empresa de aluguel de imóveis por temporada.

1.1.1 Objetivos específicos

Como objetivos específicos da pesquisa, pretende-se:

(a) Analisar as dificuldades enfrentadas pelo suporte técnico na resolução de chamados relacionados a problemas exclusivos do ambiente do cliente;

(b) Projetar e desenvolver um ambiente simulado que permita à equipe de suporte visualizar o sistema como o cliente franqueado, sem comprometer a privacidade e segurança dos dados;

(c) Realizar testes funcionais da plataforma em ambiente produtivo, avaliando sua eficácia esperada na otimização do suporte técnico.

1.2 Organização do trabalho

Este estudo está organizado da seguinte forma:

Capítulo 1: Apresenta uma introdução à temática do trabalho, contextualizando o problema de pesquisa e justificando sua relevância no cenário especificado. Além disso, define os objetivos do estudo e sua contribuição esperada para o campo de suporte técnico e gestão de imóveis por temporada.

Capítulo 2: Explora conceitos essenciais para a compreensão da relevância do estudo e seus impactos no contexto aplicado.

Capítulo 3: Descreve a metodologia adotada, destacando as premissas metodológicas do estudo, os recursos tecnológicos utilizados no desenvolvimento e os aspectos relacionados ao funcionamento da ferramenta, como autenticação e jornada de uso.

Capítulo 4: Expõe os resultados alcançados, com ênfase na validação funcional da ferramenta. São apresentados os principais fluxos de uso, a interface desenvolvida e os recursos implementados, evidenciando a aplicabilidade prática da solução no contexto do suporte técnico. Além disso, analisa criticamente as contribuições da ferramenta em termos de eficiência do suporte técnico, segurança e conformidade, bem como usabilidade e experiência do usuário.

Capítulo 5: Apresenta as conclusões do trabalho, retomando o tema e os objetivos da pesquisa, destacando os resultados obtidos e apontando possíveis desdobramentos para trabalhos futuros.

2 Fundamentação Teórica

2.1 Considerações iniciais sobre o capítulo

O presente capítulo tem como objetivo oferecer o embasamento teórico necessário para sustentar o desenvolvimento da pesquisa, abordando conceitos, práticas e referenciais relacionados ao mercado de *short-stay* [1], ao atendimento ao cliente e ao uso de ambientes simulados no contexto de suporte técnico. Além disso, são discutidos aspectos ligados à segurança da informação e à conformidade com a Lei Geral de Proteção de Dados (LGPD), que constituem fundamentos indispensáveis para a proposição de soluções tecnológicas que envolvem o tratamento de dados pessoais.

A intenção é estabelecer uma base sólida de conhecimento que permita compreender as motivações do estudo e justifique as escolhas metodológicas e técnicas descritas nos capítulos seguintes.

2.2 Caracterização da empresa de short-stay

O mercado de *short-stay*, vem apresentando uma demanda crescente por soluções tecnológicas voltadas à gestão de propriedades e reservas. Diante desse cenário, torna-se essencial garantir um suporte técnico eficiente, capaz de otimizar a experiência dos usuários e a operação dos gestores. Essa necessidade reforça a importância de inovações no atendimento ao cliente, permitindo maior agilidade e precisão nas interações (CAVALCANTI, 2023).

O presente estudo foi desenvolvido em uma empresa do setor de *short-stay*, especializada na oferta de imóveis para locação de curta duração. Fundada em 2018 e sediada em Florianópolis/SC, a organização consolidou-se como um dos principais agentes desse mercado, reunindo atualmente uma base de mais de mil clientes no segmento *Business to Business* (B2B)[2] e administrando uma carteira com mais de dois mil imóveis distribuídos em 50 destinos localizados em 11 estados brasileiros. No segmento *Business to Consumer* (B2C)[3], a empresa mantém um fluxo operacional expressivo, registrando diariamente mais de mil reservas de hóspedes que buscam soluções de hospedagem por temporada.

Embora semelhante ao aluguel tradicional, essa modalidade apresenta diferenças significativas, especialmente no aspecto contratual: enquanto locações convencionais duram meses ou anos, o *short-stay* varia de alguns dias a semanas. Esse modelo é particularmente

[1] Modelo de aluguel de imóveis para estadias de curto prazo.
[2] Modelo de negócio no qual uma empresa fornece produtos ou serviços diretamente a outras empresas.
[3] Modelo de negócio no qual a empresa fornece produtos ou serviços diretamente ao consumidor final.

popular em destinos turísticos, onde viajantes buscam acomodações confortáveis e bem localizadas, garantindo mais flexibilidade e praticidade em comparação a hospedagens tradicionais (CAVALCANTI, 2023).

De modo geral, para os consumidores, o aluguel por temporada oferece diversas vantagens. Além de proporcionar uma experiência mais personalizada e acolhedora em comparação aos hotéis tradicionais, essa modalidade permite acesso a imóveis mobiliados e equipados, oferecendo maior conforto e conveniência durante a estadia (SEAZONE, 2023).

Na perspectiva do investidor, o mercado de aluguel por temporada apresenta oportunidades promissoras. Investir em imóveis destinados a essa modalidade pode resultar em retornos financeiros atrativos, especialmente em regiões com alta demanda turística. A flexibilidade dos contratos permite ajustar os períodos de locação conforme a demanda sazonal, maximizando a ocupação e os rendimentos (MORCELLI, 2024).

No contexto da empresa caracterizada como objeto de estudo do presente trabalho, existe ainda um terceiro papel, tomado como o grande diferencial competitivo do seu modelo de negócio: o anfitrião. Segundo Crescenzo (2021), "muitos proprietários preferem deixar seus imóveis vazios do que alugar a um preço baixo, em parte devido ao trabalho de se realizar o *check-in* e o *check-out* do hóspede, além da limpeza do imóvel". O papel do anfitrião emerge como uma solução para preencher esta lacuna e estender o público de investidores da empresa.

Para o hóspede, o anfitrião atua como seu próprio nome já induz. Ele é responsável por receber, apresentar o espaço, efetuar o *check-out* na saída do hóspede, além de garantir a limpeza do local para próximas reservas e ser o ponto focal para qualquer intercorrência na estadia. Além disso, o anfitrião, por ter uma relação de franqueado à empresa, recebe comissões por reservas e taxas de limpeza realizadas no imóvel alugado.

Com a inserção desse elemento no fluxo do negócio, foi observado um crescimento de 30% no faturamento dos imóveis que começavam a ser administrados pela empresa em questão e um crescimento no portfólio de imóveis de 10% ao mês (CRESCENZO, 2021).

Dado o papel estratégico do anfitrião como diferencial competitivo da empresa em relação às demais *Online Travel Agencies* (OTAs), desde os primeiros anos de operação da empresa, foi concebida uma ferramenta voltada especificamente para anfitriões, inserida como produto de suporte ao negócio. Essa aplicação centraliza informações essenciais para a gestão da franquia e amplia a autonomia do anfitrião no acompanhamento de suas atividades, sendo um sistema web capaz de:

- Sumarizar de forma automática para o anfitrião os apartamentos e casas presentes em sua carteira de imóveis;
- Especificar as movimentações de *check-in* e *check-out* em formato de agenda, para

que a gestão do tempo seja otimizada e a entrega de valor ao produto seja garantida.

- Registrar despesas relacionadas à mudanças no imóvel – seja por manutenção, danos ou por melhoría da experiência do hóspede.
- Acompanhar o rendimento e movimentações financeiras da franquia.
- Gerenciar dados pessoais, dados bancários e cadastro de co-anfitriões para o gerenciamento de pessoal.

Contudo, frente à falhas pontuais do sistema ou dúvidas em relação ao seu uso, a ação imediata dos franqueados é o encaminhamento à equipe de suporte da plataforma, também chamado de *Customer Success* (CS). A equipe de CS, por sua vez, não possui uma visualização imediata dos erros, por não conseguir acessar o sistema do anfitrião. Isso acarreta numa carga de trabalho para as equipes de desenvolvimento, que, por conseguinte, são incumbidos de realizar *discoverys* técnicos e elaborar planos de ação para a possível correção.

Conforme destacado na seção anterior, o projeto desenvolvido neste estudo tem como objetivo mitigar esses entraves, oferecendo uma visão segura e fiel da conta do anfitrião, ao reproduzir informações da reserva, dados pessoais e demais elementos relevantes.

2.3 Atendimento ao cliente

O atendimento ao cliente é um dos pilares fundamentais para o sucesso de qualquer empresa, independentemente do seu segmento. Ele compreende o conjunto de ações e estratégias voltadas para a interação entre a empresa e seus consumidores, visando não apenas solucionar dúvidas e problemas, mas também criar uma experiência positiva e fortalecer o relacionamento com o público. Um bom atendimento não se limita ao suporte técnico ou às vendas; ele deve abranger toda a jornada do cliente, desde o primeiro contato até o pós-venda, garantindo satisfação, fidelização e, consequentemente, a valorização da marca (ANDRADE, 2022).

Entender a figura do cliente, nesse escopo, é substancial para moldar os processos de atendimento, com ênfase em qualidade e construção de relacionamentos. Costa (2015) define o cliente como a razão da existência de uma organização, todas as decisões tomadas por esse ator determinam se a empresa terá prosperidade ou não. No entanto, clientes configuram um grupo social complexo, contendo inúmeras camadas que definem seu posicionamento frente ao uso de um serviço. Nesse contexto, é necessário que as empresas mapeiem, conheçam e sirvam todos os seus componentes operacionais em prol da satisfação do cliente (ANDRADE, 2022).

Com um mercado atualmente aquecido em boa parcela dos serviços e produtos, a satisfação do cliente coloca-se como um fator elementar no processo de fidelização dos clientes. É importante pensar, enquanto organização, em todos os aspectos que fortalecem esses laços e aproximam os clientes da figura de um parceiro comercial. Ademais, é importante lembrar que em ocorrências de insatisfação com determinado produto ou serviço, um consumidor pode assumir o papel de vetor da opinião negativa, acarretando na perda de potenciais oportunidades de negócio (ANDRADE, 2022).

A satisfação do cliente pode ser trabalhada sob diversas formas: desde a primeira abordagem, triagem, venda, até a abordagem no pós-venda e suporte. Quanto a isso, Schon (2022) percebe o atendimento pós-venda como um fator de destaque no processo de relacionamento com o cliente. Segundo o autor, um atendimento pós-venda bem estruturado, que coleta *feedbacks* e consulta os clientes para melhoria contínua do produto, promovem uma sensação de conforto, importância e confiança nos clientes, além de denotar um diferencial de gestão em relação aos concorrentes.

Contudo, não é necessário somente entregar um fluxo de atendimento, seja ele de pré-venda, venda ou pós-venda, mas entregá-lo com qualidade. De acordo com Souza *et al.* (2019), a qualidade, na perspectiva do consumidor, está relacionada ao valor e à utilidade intrínseca do produto ofertado, podendo ou não estar ligada ao preço. Essa abordagem sugere que, qualidade, na verdade, pode ser entendida como um grupo de características, atributos, ou elementos que conjuntamente formam bens e serviços.

Dessa forma, no contexto do atendimento ao cliente, a qualidade não se resume apenas a fornecer uma resposta ou solucionar um problema, mas também envolve a maneira como isso é feito. Aspectos como o tempo de espera, a clareza e a empatia na comunicação, a eficiência da solução apresentada e a experiência geral do cliente durante o processo são fatores determinantes para um atendimento realmente eficaz e satisfatório.

Para garantir a qualidade no atendimento, são adotadas inúmeras ferramentas, *frameworks* e abordagens, afim de prover a melhor experiência ao consumidor. Nesse sentido, a tecnologia surge como um grande auxiliador, uma vez que, "com o alastramento da tecnologia no mundo dos negócios, uma considerável parcela da população deixa de lado relações presenciais e optam pela rapidez e agilidade" em meios digitais (SCHON; LIRA, 2022).

Dessa forma, é de extrema importância prover de uma infraestrutura preparada para receber solicitações dos clientes e respondê-las o mais rápido possível. Andrade (2022) discorre que, para mitigar – ou mesmo atenuar – essas demandas ágeis, é necessário investir em um plano de ação focado na capacitação de equipes, humanização de automações e estímulo do *feedback*.

O desenvolvimento de sistemas e produtos voltados para o atendimento digital

também conota uma estratégia essencial para garantir eficiência e qualidade na comunicação com os clientes. Soluções como *chatbots* com inteligência artificial, centrais de atendimento *omnichannel*[4], simulação de ambiente do cliente, bases de conhecimento automatizadas e plataformas de autoatendimento possibilitam respostas rápidas e personalizadas, reduzindo o tempo de espera e melhorando a experiência do usuário (ZENDESK, 2023).

Ao integrar essas soluções com uma abordagem centrada no usuário, as empresas conseguem não apenas otimizar seus processos internos, mas também fortalecer o relacionamento com os clientes. A tecnologia, quando bem aplicada, não substitui o atendimento humano, mas complementa e potencializa sua eficiência, garantindo que as interações sejam mais ágeis, acessíveis e personalizadas. Dessa forma, o investimento em inovação no atendimento digital não só melhora a experiência do consumidor, mas também contribui para a fidelização e o crescimento sustentável da empresa.

2.4 Ambientes de teste para simulação

A implementação de ambientes simulados para testes representa uma estratégia fundamental para aprimorar o processo de atendimento, beneficiando tanto os clientes quanto as equipes internas de desenvolvimento e produto (OLIVEIRA, 2023). Ao possibilitar a reprodução fiel do ambiente do usuário, essa abordagem contribui para a identificação e resolução ágil de problemas, reduzindo o tempo de resposta e melhorando a experiência do cliente (ZENDESK, 2023).

Os ambientes de teste para simulação constituem uma abordagem essencial no desenvolvimento e manutenção de sistemas computacionais modernos, permitindo a reprodução controlada de cenários reais para fins de avaliação, diagnóstico e resolução de problemas. Segundo Sommerville (2011), esses ambientes são réplicas isoladas de sistemas em produção, projetados para imitar com precisão o comportamento e as características do ambiente real, sem os riscos associados à manipulação direta de dados e processos críticos.

De acordo com Myers (2012), os ambientes simulados apresentam-se como ferramentas fundamentais para o teste de *software*, pois permitem executar, analisar e validar o comportamento do sistema sob condições controladas. Essa característica possibilita que desenvolvedores e equipes de suporte identifiquem falhas, avaliem o desempenho e verifiquem a conformidade com requisitos estabelecidos, antes da implementação em produção ou durante a análise de problemas reportados.

A fidelidade da simulação é um aspecto crucial para a eficácia dos ambientes de teste. Myers (2012) argumenta que quanto mais o ambiente simulado se aproxima do ambiente de produção em termos de configuração, dados e comportamento, mais confiáveis

[4] Estratégia de vendas que integra todos os canais de uma empresa, tanto físicos quanto digitais, para oferecer uma experiência de compra fluida ao cliente

serão os resultados obtidos nos testes. Essa similitude permite identificar problemas que poderiam passar despercebidos em ambientes simplificados demais.

No entanto, a criação e manutenção de ambientes de teste fidedignos apresentam desafios significativos. Jorgensen (2013) aponta que a complexidade dos sistemas modernos, com suas múltiplas integrações e dependências, torna a replicação exata do ambiente de produção uma tarefa árdua. Além disso, há considerações importantes relacionadas ao volume de dados, à performance e aos custos de infraestrutura.

Uma estratégia cada vez mais adotada para contornar esses desafios é a utilização de técnicas de virtualização e containerização. Tecnologias como Docker e Kubernetes têm revolucionado a forma como os ambientes de teste são implementados, permitindo criar réplicas consistentes e isoladas de sistemas complexos com maior facilidade e menor custo (MANOR et al., 2025).

Outra tendência relevante é a utilização de dados sintéticos ou mascarados nos ambientes de teste. Essa abordagem permite simular volumes e variedades de dados semelhantes aos encontrados em produção, sem expor informações sensíveis dos usuários, o que é particularmente importante em setores regulados ou que lidam com dados pessoais (MOURA; COUTINHO, 2024).

No contexto específico das equipes de suporte técnico e atendimento ao cliente, os ambientes simulados desempenham um papel ainda mais crítico. Segundo Hatter (2014), esses ambientes permitem que os analistas de suporte reproduzam problemas reportados pelos usuários, analisem seu comportamento e testem soluções sem afetar os sistemas em produção ou comprometer dados reais dos clientes.

Souza (2024) destaca as vantagens em se adotar sistemas de simulação no processo de atendimento ao cliente. Dentre estas, incluem:

- Reduz o estresse e a ansiedade da equipe de suporte ao lidar com situações complexas em cenários reais;
- Permite que a equipe de suporte pratique diferentes situações, desenvolvendo suas habilidades de comunicação e resolução de problemas em um ambiente controlado;
- Aumenta a eficiência e a rapidez na resolução de problemas, diminuindo o tempo de espera e a frustração dos clientes;
- Treinamento de novos membros da equipe, familiarizando-os com ferramentas, processos e a base de conhecimento do *software*;
- Preparação para lançamentos e eventos, ajudando a equipe a lidar com picos de demanda ou novos produtos e funcionalidades.

Além disso, a relevância dessa prática se intensifica em empresas cujo software desempenha um papel essencial na entrega de valor ao produto ou serviço final. Isso se aplica a diversos setores da indústria, incluindo o mercado de aluguel de imóveis por temporada (*short stay*), onde plataformas digitais são fundamentais para a operação eficiente do negócio. A adoção de ambientes simulados, nesse contexto, não apenas aprimora a qualidade do suporte técnico, mas também fortalece a confiabilidade e a competitividade da empresa no mercado.

No setor de hospedagem e *short-stay*, que envolve o processamento de dados sensíveis como informações pessoais e financeiras de hóspedes, a utilização de ambientes simulados para suporte técnico torna-se ainda mais relevante. Eles possibilitam que a equipe de CS visualize e reproduza os cenários reportados pelos anfitriões, sem necessidade de acesso direto às suas contas reais.

Ao implementar um ambiente que replica com fidelidade a experiência do usuário, mas utiliza dados seguros e controlados, a solução proposta não apenas agiliza o processo de diagnóstico e resolução de problemas, mas também reforça o compromisso da empresa com a proteção de dados e a conformidade regulatória, aspecto cada vez mais relevante no cenário atual, especialmente à luz da LGPD e outras regulamentações semelhantes.

2.5 Segurança da informação e conformidade com LGPD

A segurança da informação constitui um conjunto de práticas, políticas e procedimentos destinados a proteger dados e sistemas contra acessos não autorizados, uso indevido, modificação, divulgação ou destruição. Segundo (NEVES et al., 2021):

> O surgimento da globalização bem como a expansão da internet, trouxe grande quantidade de dados gerados a cada minuto em inúmeras transações comerciais e/ou sociais, demonstrando a necessidade de definição de parâmetros em forma de lei, para que haja total responsabilidade na conduta desta relação, necessitando de normatizações para a segurança da informação.

Dessa forma, a segurança da informação pode ser definida como uma área do conhecimento dedicada à proteção de ativos de informação contra acessos não autorizados, alterações indevidas ou indisponibilidade, buscando preservar três pilares fundamentais: confidencialidade, integridade e disponibilidade das informações (SÊMOLA, 2014).

Segundo Semola (2014), o pilar de confidencialidade refere-se à garantia de que a informação seja acessível apenas por pessoas autorizadas; a integridade assegura que os dados não sejam alterados de forma indevida ou corrompidos; e a disponibilidade garante que a informação esteja acessível quando necessária. Em uma visão mais contemporânea, Brasil (2024) acrescenta um quarto pilar: a autenticidade. O papel da autenticidade é

assegurar que a informação foi produzida, expedida, modificada ou destruída por uma determinada pessoa física, equipamento, sistema, órgão ou entidade. Os quatro conceitos, conjuntamente, formam um arcabouço mais completo para a proteção das informações.

No contexto atual, marcado pela transformação digital e pelo aumento exponencial na geração e no processamento de dados, a segurança da informação tornou-se uma preocupação central para organizações de todos os portes e segmentos. Conforme aponta (SANTOS; SILVA, 2021), o aumento da capacidade de comunicação, a interação entre sistemas, a evolução das redes convergentes e o surgimento das redes móveis vêm proporcionando comunicação contínua e em diversos locais, possibilitando várias maneiras de acesso à informação e, muitas vezes, esse acesso é indevido.

Em resposta a esse cenário e visando estabelecer diretrizes para a proteção de dados pessoais, diversas regulamentações foram criadas em todo o mundo. Entre elas, destaca-se o Regulamento Geral de Proteção de Dados (GDPR) na Europa, que serviu como base para a elaboração da Lei Geral de Proteção de Dados Pessoais (LGPD) no Brasil (GOMES et al., 2025).

A LGPD, Lei nº 13.709/2018, entrou em vigor em setembro de 2020 e representa um marco na regulamentação da proteção de dados no Brasil. Ela estabelece regras claras sobre coleta, tratamento, armazenamento e compartilhamento de dados pessoais, sejam eles digitais ou físicos, impondo às organizações a responsabilidade pela adequação de seus processos e sistemas (BRASIL, 2018).

De acordo com (GOMES et al., 2025), a LGPD fundamenta-se em princípios como publicidade, exatidão dos dados, finalidade, livre acesso e segurança lógica. Esses princípios visam assegurar que o tratamento de dados pessoais ocorra de maneira ética, transparente e respeitosa aos direitos dos titulares.

Um aspecto crucial da LGPD é a exigência de que as organizações implementem medidas técnicas e administrativas de segurança capazes de proteger os dados pessoais contra acessos não autorizados e situações acidentais ou ilícitas de destruição, perda, alteração, comunicação ou difusão (BRASIL, 2018). Isso inclui a adoção de controles de acesso rigorosos, políticas de segurança abrangentes, procedimentos para resposta a incidentes, entre outras práticas recomendadas de segurança da informação.

Além disso, a lei estabelece sanções significativas para casos de não conformidade, que variam de vão desde advertências e multas, até a proibição total ou parcial do exercício da atividade da empresa (GOMES et al., 2025). Isso ressalta a importância de uma abordagem proativa e estruturada para a proteção de dados, não apenas como uma obrigação legal, mas também como um diferencial competitivo e uma demonstração de compromisso com a privacidade dos usuários.

Enviesando esta discussão para o contexto do desenvolvimento de *software*, Moura

(2024) defende que:

> A LGPD se torna um aspecto a ser considerado dentro do processo de desenvolvimento de aplicacões e sistemas, pois envolve a necessidade de funcionalidades estarem aderentes a seus princípios. Isso implica em uma forte relação com requisitos de *software*. Desse modo, por exemplo, os profissionais de desenvolvimento de *software* devem aprimorar seu aprendizado em relação a técnicas que garantam a privacidade e o cumprimento dos princípios da LGPD.

Sendo assim, é imprescindível considerar o alinhamento à LGPD no desenvolvimento não somente deste trabalho, mas também em qualquer solução. Aplicada à qualquer setor da economia que lide com dados sensíveis. Adotar essa prática não somente induz a um sistema mais seguro e confiável, mas inibe as chances de sanções jurídicas para com as empresas quanto à LGPD.

No contexto específico do mercado de *short-stay* e do sistema de gestão para anfitriões abordado neste estudo, a segurança da informação e a conformidade com a LGPD assumem dimensões ainda mais críticas. Isso porque esses sistemas processam diversos tipos de dados pessoais, incluindo informações de identificação (como nome, CPF, RG), dados de contato (e-mail, telefone) e informações financeiras (dados de pagamento, histórico de transações). A utilização desses dados sensíveis requer uma abordagem rigorosa em relação à privacidade e segurança.

Nesse sentido, a implementação de ambientes simulados para suporte técnico, como proposto neste estudo, apresenta-se como uma estratégia alinhada aos princípios da LGPD, especialmente ao princípio da minimização de dados. De acordo com (FRAZÃO; OLIVA; TEPEDINO, 2019), esse princípio orienta que apenas os dados estritamente necessários para uma finalidade específica devem ser coletados e processados, reduzindo assim os riscos associados ao tratamento excessivo de informações pessoais.

Ao utilizar ambientes simulados, as equipes de suporte técnico podem diagnosticar e resolver problemas sem acessar diretamente os dados reais dos clientes, o que diminui significativamente a exposição dessas informações e o risco de incidentes de segurança. Ao permitir que a equipe de suporte técnico visualize e reproduza cenários de erro sem manipular diretamente os dados reais, a solução proposta neste estudo reforça o compromisso da empresa com a proteção de dados e a conformidade regulatória.

2.6 Considerações finais sobre o capítulo

Em síntese, o capítulo apresentou os principais elementos teóricos que sustentam o desenvolvimento da solução proposta. Foram exploradas as particularidades do mercado de *short-stay*, destacando o papel estratégico do anfitrião, além da centralidade do atendimento ao cliente como diferencial competitivo. Também se discutiu a relevância dos ambientes de

simulação como prática de apoio às equipes de suporte, ressaltando seus benefícios em termos de eficiência, aprendizado e mitigação de riscos.

Por fim, evidenciou-se a importância da segurança da informação e da conformidade com a LGPD como diretrizes fundamentais para qualquer sistema que manipule dados sensíveis. Esses referenciais, articulados, fornecem a sustentação conceitual para a proposta da pesquisa, ao mesmo tempo em que reforçam sua pertinência prática e acadêmica.

3 Materiais e Métodos

3.1 Considerações iniciais sobre o capítulo

O presente capítulo tem como objetivo apresentar os materiais e métodos empregados na realização desta pesquisa, descrevendo os fundamentos metodológicos que orientam o estudo, os recursos tecnológicos que possibilitaram a implementação da solução proposta e o fluxo de funcionamento da ferramenta resultante do desenvolvimento.

3.2 Metodologia

Este trabalho caracteriza-se como uma pesquisa aplicada, voltada para o desenvolvimento de um ambiente simulado que reproduza a experiência dos anfitriões franqueados de uma empresa de aluguel por temporada para a equipe de suporte. O objetivo central é criar uma solução prática e funcional que responda a um problema real, aprimorando a eficiência do atendimento ao cliente sem comprometer a segurança dos dados dos clientes.

Ademais, o estudo pode ser classificado como de natureza exploratória e experimental. O campo exploratório tem por objetivo investigar soluções existentes, tecnologias aplicáveis e melhores práticas para melhorar o suporte técnico e segurança da informação em empresas de tecnologia, em especial aquelas voltadas à temática de simulação da perspectiva do cliente.

No âmbito experimental, os esforços se concentrarão na realização de testes práticos da solução implementada, por meio da aplicação de um formulário (*survey*) com perguntas abertas e fechadas, abrangendo aspectos quantitativos e qualitativos relacionados ao uso da ferramenta, sua usabilidade, experiência de interface (UI/UX) e impacto operacional percebido pelas equipes de suporte.

Por fim, pode-se atribuir a este trabalho também a característica descritiva, tendo em vista a necessidade de perpassar pelos desafios que permeiam o cotidiano dos profissionais de suporte técnico da empresa. Além disso, será necessário detalhar a implementação e arquitetura do projeto, bem como seus efeitos na experiência do usuário e na eficiência operacional, com foco nos benefícios proporcionados pela solução desenvolvida.

Portanto, as etapas metodológicas desta pesquisa são:

- Levantamento de requisitos do *software* por meio da análise das dificuldades enfrentadas pelo suporte técnico na resolução de chamados relacionados a problemas exclusivos do ambiente do cliente;

- Projeto e desenvolvimento de um ambiente simulado para apoio à esse processo, destacando as tecnologias para construção de *front-end* e *back-end*, e destacando o método de provisionamento da solução para acesso contínuo.

3.3 Levantamento de requisitos

O levantamento de requisitos é uma etapa fundamental no processo de desenvolvimento de *software*, consistindo na coleta de informações sobre as necessidades e expectativas dos *stakeholders* [1] para a criação de um sistema que atenda às suas demandas. Trata-se de um processo que busca identificar, documentar e validar as necessidades dos usuários e as restrições que guiarão o desenvolvimento (SILVA, 2023).

Conforme aponta Pressman (2016), essa é a base para a definição do escopo do projeto, estabelecendo um entendimento mútuo entre todas as partes interessadas sobre o que deve ser construído. A negligência desta fase está diretamente relacionada ao fracasso de muitos projetos de *software*, uma vez que a construção de uma solução elegante para o problema errado não atende às necessidades de ninguém.

Os requisitos de um *software* são comumente classificados em duas categorias principais: requisitos funcionais e não-funcionais (PIHLAJANIEMI, 2023). Os requisitos funcionais descrevem o comportamento do sistema, ou seja, as funções e serviços que ele deve executar para atender às necessidades dos usuários. Por outro lado, os requisitos não-funcionais especificam os critérios de qualidade e as restrições sob as quais o sistema deve operar, abrangendo aspectos como desempenho, segurança, usabilidade e confiabilidade (SILVA, 2023).

Neste trabalho, o levantamento de requisitos foi conduzido por meio da análise das dificuldades enfrentadas pela equipe de suporte técnico da empresa, conforme detalhado previamente. A partir dessa análise, foram especificadas as funcionalidades essenciais para o ambiente de simulação proposto, de forma que ele atenda às necessidades do usuário (suporte) e agreguem valor aos seus processos operacionais. A tabela 1 faz uma sumarização dos 20 principais requisitos da aplicação, divididos entre requisitos funcionais e não-funcionais, levantados para o desenvolvimento da solução.

[1] Todas as partes (pessoas) envolvidas e interessadas no *software*

Tabela 1 – Requisitos de desenvolvimento da solução

Referência	Tipo	Descrição
RF01	Funcional	A ferramenta deve requerer login prévio na aplicação matriz com credenciais institucionais.
RF02	Funcional	A ferramenta deve estar disponível apenas usuários com perfil de Suporte e Admin
RF03	Funcional	A ferramenta deve estar disponível nos menus de navegação da aplicação matriz
RF04	Funcional	A ferramenta deve permitir pesquisar anfitriões na base de dados por ID ou nome
RF05	Funcional	A pesquisa na ferramenta deve ter um limite mínimo de 3 caracteres para retorno
RF06	Funcional	A pesquisa deve retornar uma listagem com os possíveis anfitriões, com base nos seus parâmetros
RF07	Funcional	A ferramenta deve permitir definir anfitriões como favoritos e apresentá-los em uma listagem exclusiva
RF08	Funcional	Todas as listagens de anfitriões devem ser paginadas
RF09	Funcional	A ferramenta deve exibir um feedback de carregamento ao selecionar um anfitrião
RF10	Funcional	A ferramenta deve exibir a interface sob a mesma visão que um anfitrião possui
RF11	Funcional	A ferramenta deve bloquear qualquer operação de escrita (edição/exclusão/criação)
RF12	Funcional	Os dados exibidos na simulação devem corresponder ao que o anfitrião veria no ambiente produtivo
RF13	Funcional	A ferramenta deve permitir que a interface da perspectiva do anfitrião seja renderizada em modo mobile e desktop
RF14	Funcional	A ferramenta deve permitir o redimensionamento (zoom) da tela de perspectiva
RF15	Funcional	A ferramenta deve permitir logout voluntário
RNF01	Não-funcional	A ferramenta deve persistir a lista de anfitriões favoritos localmente no navegador
RNF02	Não-funcional	A ferramenta deve utilizar tokens JWT para autenticação, de ponta a ponta
RNF03	Não-funcional	A ferramenta deverá ser desenvolvida com React (TypeScript) como tecnologia principal de front-end
RNF04	Não-funcional	A ferramenta deverá ser desenvolvida com Django (Python) como tecnologia principal de back-end
RNF05	Não-funcional	A ferramenta deverá utilizar o mesmo guia de estilos da aplicação matriz

Fonte: AUTOR, 2025

Uma vez definidos e validados os requisitos, a próxima etapa do processo de desenvolvimento reside no projeto detalhado e na implementação da solução. A seção seguinte detalha o projeto e o desenvolvimento do *software*, pairando sobre tópicos como

as tecnologias, padrões e arquitetura utilizados para a construção da ferramenta.

3.4 Projeto e desenvolvimento da solução

3.4.1 Front-End

A camada de *Front-End* de um projeto refere-se ao conjunto de ferramentas combinadas para prover aos usuários uma interface de interação, a qual é a porta de entrada para informações precisas para processamentos futuros de uma aplicação web (KREMER, 2023). Segundo Gorai (2025), conota uma camada estratégica na entrega de serviços digitais, sendo o meio pelo qual desenvolvedores podem proporcionar experiências intuitivas e atrativas para os usuários no uso de plataformas web.

O autor também destaca a relevância de aplicações web bem diagramadas, não apenas como um meio de atrair novos clientes, mas também como uma estratégia eficaz para fortalecer o engajamento entre a empresa e seu público, evitar frustrações e más experiências dos clientes e, por conseguinte, aumentar oportunidades de negócio (GORAI et al., 2025).

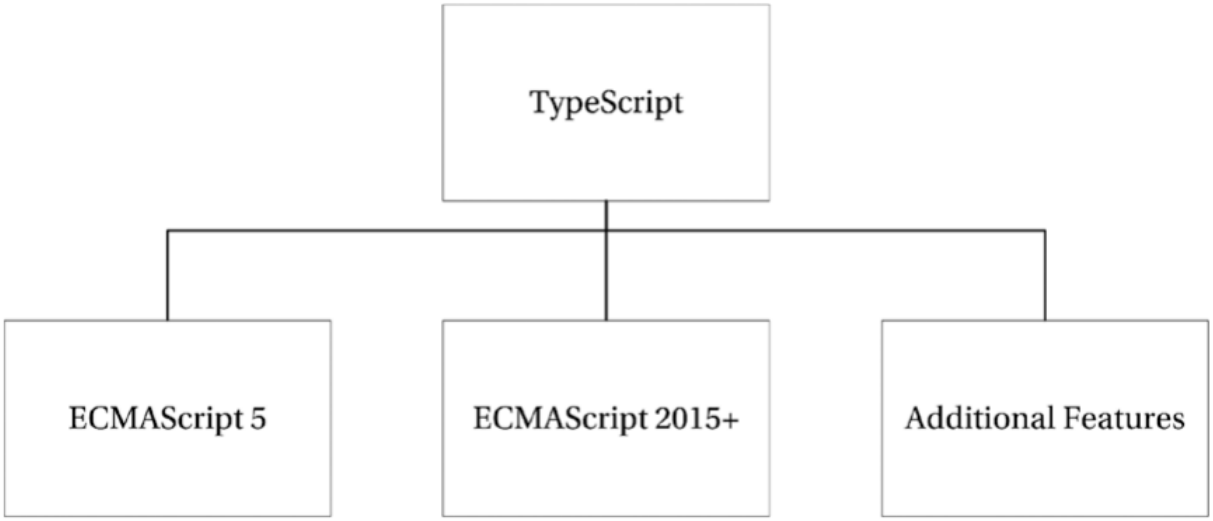
Nos últimos anos, além do design, a performance das aplicações web tornou-se um fator crucial na experiência do usuário. De acordo com Ekpobimi (2024), um front-end bem otimizado — capaz de reduzir tempos de carregamento, evitar mudanças inesperadas de layout e minimizar atrasos — é essencial para aprimorar a usabilidade. Essa otimização não apenas melhora a satisfação do usuário, mas também reduz a taxa de abandono por frustração, contribuindo para um maior engajamento e retenção.

Diante disso, as tecnologias selecionadas para o desenvolvimento da camada de Front-End foram escolhidas com base em sua capacidade de proporcionar um equilíbrio entre design, performance e usabilidade. A combinação dessas soluções visa não apenas proporcionar uma ferramenta útil de simulação para os times de atendimento com a plataforma, mas fazer com que essa experiência seja proveitosa e, consequentemente, produtiva, alinhando-se às melhores práticas no desenvolvimento web.

3.4.1.1 TypeScript

O *TypeScript* é uma linguagem criada e mantida pela Microsoft desde 2012, com o objetivo de tornar o desenvolvimento de aplicações em JavaScript mais escaláveis, podendo suportar milhares de linhas de código. A linguagem é considerada um *superset* do JavaScript, por trazer uma tecnologia que combina toda a linguagem JavaScript (ECMAScript), adicionando funções adicionais para melhorar o desenvolvimento de aplicações web (FENTON, 2018).

Figura 1 – Componentes do *TypeScript*



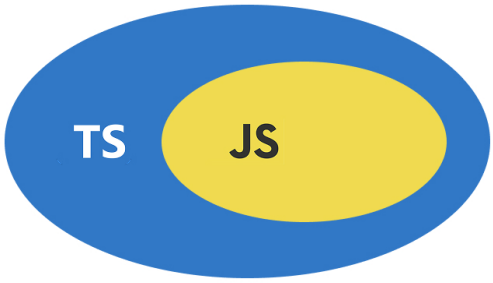
Fonte: FENTON, 2018

O grande diferencial do *TypeScript* em relação às linguagens anteriormente citadas é a introdução de um sistema de tipagem estática opcional, permitindo a definição explícita de tipos para variáveis, funções e estruturas de dados. Essa abordagem mitiga erros comuns do JavaScript, como operações entre tipos incompatíveis, ao antecipar a detecção de inconsistências para a fase de compilação. Além disso, a tipagem estática melhora a segurança do código, facilita a manutenção e aprimora a escalabilidade de projetos, combinando a flexibilidade do JavaScript com os benefícios de linguagens tipadas (MICROSOFT, 2024).

A introdução de estruturas de dados mais complexas no *TypeScript* também representa um diferencial significativo para a linguagem, tornando-a uma escolha robusta para aplicações de grande escala. Com recursos avançados, como tipos genéricos, *interfaces*, *unions* e *intersections*, o *TypeScript* facilita a construção de sistemas modulares e altamente reutilizáveis – como o que será desenvolvido neste trabalho –, reduz erros e melhora a legibilidade do código (FENTON, 2018).

Por fim, é importante destacar o conceito de superset mencionado no início desta seção. Na literatura, o *TypeScript* é frequentemente descrito como um "superconjunto" do JavaScript, conforme ilustrado na Figura 2. Isso significa que a linguagem expande as funcionalidades do JavaScript sem comprometer sua compatibilidade, permitindo que qualquer código *TypeScript* válido seja convertido para JavaScript puro (FENTON, 2018).

Figura 2 – Relação conjuntiva entre JavaScript e *TypeScript*



Fonte: AUTOR, 2025

Essa característica garante a interoperabilidade com navegadores e outros ambientes que executam JavaScript, facilitando a adoção gradual da linguagem. Além disso, a sintaxe, integrações e os conceitos fundamentais do JavaScript são preservados no *TypeScript*, o que reduz a curva de aprendizado para desenvolvedores familiarizados com a linguagem original, ao mesmo tempo em que introduz recursos mais avançados para um desenvolvimento mais seguro e escalável (MICROSOFT, 2024).

No contexto deste trabalho, o *TypeScript* desempenha um papel fundamental como linguagem de programação base para a construção do *Front-End* da aplicação. A combinação da linguagem com as tecnologias posteriormente listadas, são substanciais para a criação modular de componentes seguros, reutilizáveis e robustos, reduzindo o risco de erros em tempo de execução e aumentando a eficiência na etapa de desenvolvimento.

3.4.1.2 ReactJS

O ReactJS é uma biblioteca baseada em JavaScript para construção de interfaces de usuário e será a principal tecnologia da camada de *Front-End* da aplicação. Ele é uma ferramente código aberto multi-plataforma, desenvolvida pelo Meta em meados de 2013, e se propõe a instaurar um novo paradigma na construção de interfaces de usuário ao introduzir o conceito de componentização – ideal que propõe a criação de interfaces modulares, reutilizáveis e mais fáceis de manter (MDN, 2024).

A escolha do React para este projeto se dá pela necessidade de uma interface responsiva e altamente performática, capaz de reproduzir fielmente o ambiente do cliente na simulação para o suporte técnico. A arquitetura baseada em componentes reutilizáveis também facilita a manutenção e evolução da aplicação, enquanto a eficiência do Virtual DOM melhora o tempo de resposta da interface (MDN, 2024). Além disso, sua ampla adoção no mercado e compatibilidade com diversas bibliotecas garantem maior flexibilidade no desenvolvimento da solução.

3.4.1.3 Vite

O Vite é uma ferramenta de *build* (compilação) moderna que proporciona um ambiente de desenvolvimento rápido e eficiente para aplicações web. Criado para otimizar a experiência do desenvolvedor, o Vite foi projetado para superar limitações de ferramentas tradicionais, oferecendo tempos de inicialização instantâneos e uma experiência de recarga a quente (*hot module replacement*) extremamente ágil (VITE, 2024).

Uma das principais vantagens do Vite é seu mecanismo de compilação baseado em *ES Modules*, que permite carregar e compilar arquivos sob demanda, em vez de realizar uma transpilação completa antes da execução. Isso resulta em um ambiente de desenvolvimento mais responsivo, eliminando a necessidade de processos demorados de *bundling*[2] durante a fase de desenvolvimento (VITE, 2024).

Além da performance, o Vite oferece compatibilidade com diversas tecnologias modernas, como React, Vue e Svelte, permitindo que os desenvolvedores trabalhem de forma eficiente com diferentes frameworks e bibliotecas. A escolha do Vite para este projeto se justifica pela necessidade de um fluxo de desenvolvimento ágil, com atualizações instantâneas na interface e uma configuração simplificada, favorecendo um código mais modular e otimizado.

3.4.1.4 Styled Components

O Styled Components é uma biblioteca de estilização para aplicações React baseada no conceito de CSS-in-JS. Essa abordagem de estilização permite que os estilos sejam definidos diretamente dentro dos componentes React, utilizando a sintaxe de *template literals* do JavaScript. Dessa forma, a estilização fica encapsulada dentro de cada componente, garantindo maior modularidade e evitando conflitos de nomenclatura entre classes CSS (STYLED-COMPONENTS, 2025).

Uma das principais vantagens do Styled Components é a capacidade de definir estilos dinâmicos com base em propriedades especificadas do componente. A figura 3 ilustra um exemplo dessa funcionalidade em um botão que possui uma propriedade “tema”. Na ilustração é possível ver que quando a propriedade é igual a “light” ou “dark”, o componente assume uma estilização diferente, que pode ser configurada de forma lógica com JavaScript.

[2] processo de agrupar múltiplos arquivos de código para serem carregados pelo navegador

Figura 3 – Exemplo de estilização dinâmica com Styled Components

Fonte: AUTOR, 2025

Essa *feature* possibilita a criação de interfaces mais flexíveis e adaptáveis sem a necessidade de manipular classes ou estilos inline. Além disso, a biblioteca oferece suporte a temas globais, permitindo a aplicação de estilos consistentes em toda a aplicação por meio do ThemeProvider (STYLED-COMPONENTS, 2025).

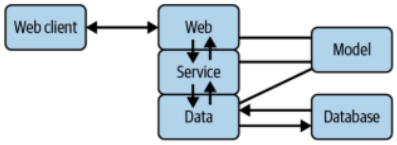
Além disso, a performance é otimizada por meio da remoção automática de estilos não utilizados, contribuindo para um carregamento mais eficiente da aplicação. Dessa forma, o uso do Styled Components se destaca como uma solução moderna e escalável para estilização em React, oferecendo maior organização, reaproveitamento de código e flexibilidade no desenvolvimento de interfaces visuais (STYLED-COMPONENTS, 2025).

3.4.2 Back-End

As interações de usuário na camada de *front-End*, conforme a complexidade do projeto, acabam gerando um volume de dados para o sistema. Estes dados são tratados e processados pela camada de *back-end* da aplicação, que compreende toda a lógica que alimenta o sistema, no lado do servidor (GORAI et al., 2025). Segundo Kremer (2022):

> Criar, editar/atualizar e coletar dados são alguns dos processos mais frequentemente associados ao desenvolvimento de *back-end*. Alguns exemplos de linguagens de script comuns usadas são JavaScript, PHP, Ruby e Python. Com essas linguagens, um desenvolvedor de *back-end* pode criar algoritmos e lógica de negócios para manipular os dados que foram recebidos no desenvolvimento de *front-end*.

Figura 4 – Exemplo de fluxo de informações em um sistema web



Fonte: MASSÉ, 2011

Além disso, assim como abordado na seção sobre *Front-End*, o desempenho e a performance são fatores fundamentais no planejamento da arquitetura de sistemas web. A

camada de servidor, hoje, não se limita apenas a fornecer serviços, mas busca entregá-los de forma eficiente, com alta velocidade e o menor custo computacional possível. Essa otimização é essencial para garantir escalabilidade, reduzir latências e otimizar a alocação de recursos, tornando a aplicação mais responsiva e acessível (CARMO; FERREIRA; FIGUEIREDO, 2024).

Sendo assim, as tecnologias elencadas para o desenvolvimento da camada de *back-end* da aplicação foram definidas pensando no melhor aproveitamento geral de aspectos como robustez, escalabilidade e segurança, aspectos amplamente discutidos na literatura. A adoção dessas ferramentas visa não apenas estabelecer, de maneira eficiente, a interface entre a base de anfitriões da empresa e o *Front-End*, mas também garantir um ambiente confiável para a simulação de suas interações, proporcionando uma experiência segura e consistente para o time de atendimento.

3.4.2.1 REST APIs

Antes de efetivamente entrar no contexto das tecnologias de *back-end*, é importante conhecer um conceito fundamental para a plena comunicação entre as camadas arquiteturais da aplicação: as APIs REST. As *Representational State Transfer* (REST) *Application Programming Interfaces* (API's) são um dos padrões mais utilizados para a comunicação entre sistemas modernos na web. Baseiam-se na arquitetura REST, que segue um conjunto de princípios para a construção de serviços escaláveis, eficientes e independentes de plataforma (MASSé, 2011).

Utilizar uma arquitetura REST para a transferência de dados entre as camadas da aplicação, significa utilizar métodos *Hypertext Transfer Protocol* (HTTP) baseados em requisições e respostas tanto para o processo de transferência, quanto para normalização do formato. Dessa forma, o *back-end* consegue fornecer diferentes *endpoints* – caminho específico dentro de uma API, que retorna um dado específico – para diferentes funcionalidades no sistema, para que o *front-end*, através de chamadas, possa acessá-los e garantir os dados necessários para o usuário (GORAI et al., 2025).

Figura 5 – Fluxo de uma REST API

Fonte: MASSÉ, 2011

Gorai (2025) especifica o fluxo apresentado na figura acima em três etapas:

1. **Cliente (*Frontend*):**

- Faz uma requisição HTTP para um endpoint específico da API (URL) no servidor.
- Especifica o método da requisição (GET, POST, PUT, DELETE) e a ação desejada.
- Pode incluir um corpo de requisição com dados para ações específicas, como criação ou atualização.

2. **Servidor (*Backend*):**

- Recebe a requisição e identifica o endpoint alvo com base na URL e no método.
- Processa a requisição, acessando bancos de dados, realizando cálculos ou interagindo com outros serviços.
- Prepara uma resposta contendo os dados solicitados, o código de status (por exemplo, 200 para sucesso) e quaisquer informações adicionais.

3. **Cliente:**

- Recebe a resposta e interpreta o código de status e o conteúdo dos dados.
- Atualiza a interface do usuário ou executa outras ações com base nas informações retornadas.

Dada sua simplicidade e ampla aplicabilidade, as REST APIs tornaram-se um recurso essencial para aplicações web modernas. Quando bem projetadas, permitem a construção de sistemas distribuídos robustos, facilitam integrações entre serviços e viabilizam aplicações altamente escaláveis (MASSé, 2011). No contexto deste trabalho, as REST APIs são amplamente utilizadas para a aquisição e reprodução fiel dos dados dos anfitriões dentro do sistema simulado, garantindo uma experiência realista e precisa.

3.4.2.2 Django

O desenvolvimento de aplicações web modernas frequentemente envolve a utilização de *frameworks*, que consistem em um conjunto de bibliotecas que fornecem funcionalidades genéricas e abstraem tarefas de baixo nível, permitindo que o desenvolvedor se concentre na lógica de negócio do *software* (LATHKAR, 2025). Nesse contexto, o Django se destaca como um *framework* de alto nível para a linguagem Python, projetado para incentivar o desenvolvimento rápido e um design pragmático e limpo (DJANGO, 2025).

Sua função primordial é facilitar a criação de aplicações web complexas e orientadas a dados, estruturando o *back-end* do sistema a partir de uma variação da arquitetura MVC (*Model-View-Controller*), conhecida como MVT (*Model-View-Template*) (LATHKAR, 2025). O *framework* é responsável por gerenciar todo o ciclo de requisição e resposta do

protocolo HTTP, processar a interação com o banco de dados por meio de um Mapeador Objeto-Relacional (ORM) e renderizar o conteúdo a ser apresentado ao cliente (LATHKAR, 2025).

Uma das vantagens mais significativas do Django reside em sua filosofia "baterias incluídas" (*batteries-included*), que se traduz na incorporação de um ecossistema robusto de componentes prontos para uso em tarefas comuns do desenvolvimento web. Entre esses componentes, destacam-se um sistema completo de autenticação de usuários, uma interface de administração gerada automaticamente, e proteções nativas contra vulnerabilidades de segurança prevalentes, como injeção de SQL, cross-site scripting (XSS) e cross-site request forgery (CSRF) (DJANGO, 2025).

Adicionalmente, o *framework* é reconhecido por sua alta escalabilidade, sendo utilizado por aplicações de grande tráfego, e por ser mantido por uma comunidade ativa que produz uma documentação abrangente, o que o torna uma ferramenta versátil e confiável (DJANGO, 2025).

No escopo deste trabalho, o Django figura como o componente central da arquitetura de *back-end*. A ferramenta é responsável por estruturar e gerenciar as requisições HTTP advindas da aplicação cliente (*front-end*), orquestrando a lógica de negócio e a integração de serviços auxiliares (e.g. Celery, Redis e outros). Por conseguinte, o *framework* garante que as solicitações sejam processadas de forma eficiente, segura e assíncrona, provendo respostas adequadas ao usuário e assegurando a consistência e a integridade das alterações no banco de dados da aplicação.

3.4.2.3 PostgreSQL

O PostgreSQL é um Sistema de Gerenciamento de Banco de Dados (SGBD) baseado na versão 4.2 do POSTEGRES, tendo sua origem na Universidade da California, em 1995. Essa ferramenta propõe um suporte a boa parte das funcionalidades do MySQL convencional, porém, combinando tecnologias modernas, como consultas (*queries*) complexas, chaves estrangeiras, *triggers*, visões atualizáveis e integridade transacional (POSTGRESQL, 2025).

Tendo em vista que os bancos de dados são um dos principais elementos de aplicações web atuais, o PostgresSQL emerge como uma solução precisa, de código aberto e uso gratuito, capaz de lidar com grandes volumes de dados de forma eficiente, o tornando ideal para médias e grandes empresas. Além disso, suas *features* adicionais são fatores que o coloca a frente de outros SGBDs, uma vez que sua integração pode ser feita através de inúmeras linguagens de programação e seu padrão de escrita de *queries* é mais compreensivo que os demais (RAVSHANOVICH, 2024).

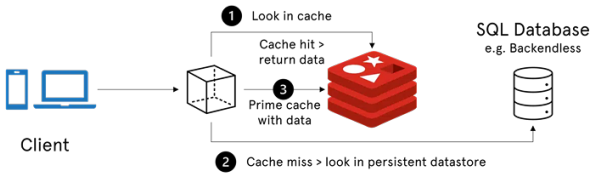
Neste trabalho, o PostgreSQL é utilizado como o banco de dados principal da

aplicação. Sua robustez e suporte transacional garantem a consistência dos dados, enquanto sua compatibilidade com o Django facilita a integração e manipulação eficiente das informações necessárias para a simulação do ambiente do cliente.

3.4.2.4 Redis

O Redis é um sistema de armazenamento em memória altamente eficiente, projetado para oferecer acesso ultrarrápido a dados estruturados. Com suporte a múltiplos tipos de estrutura, como listas, conjuntos e *hashes*, ele vai além de um simples banco de dados chave-valor, permitindo aplicações complexas, como *caching* dinâmico, filas de mensagens e controle de sessões. Sua arquitetura baseada em RAM possibilita tempos de resposta na ordem de microssegundos, tornando-o uma escolha ideal para aplicações que exigem alto desempenho e baixa latência (REDIS, 2025).

Figura 6 – Exemplo de utilização do Redis



Fonte: SARFRAZ, M. 2023. Disponível em: https://venturenox.com/blog/the-power-of-redis-in-transforming-real-time-applications. Acesso em 21 jul. 2025.

A Figura 6 apresenta um exemplo de fluxo do Redis dentro de uma aplicação. Quando um cliente realiza uma requisição, a aplicação primeiro verifica se os dados solicitados estão armazenados em *cache* no Redis. Se os dados forem encontrados, eles são imediatamente disponibilizados ao *front-end*, garantindo uma resposta rápida. Caso contrário, a aplicação consulta o banco de dados por meio de requisições HTTP para recuperar as informações necessárias. Após obter os dados, além de retorná-los ao cliente, a aplicação os armazena no Redis, permitindo que futuras requisições sejam atendidas de forma mais eficiente, reduzindo a carga sobre o banco de dados e melhorando o desempenho do sistema.

No contexto deste projeto, o Redis é utilizado para otimizar o desempenho da aplicação, reduzindo o tempo de resposta em requisições recorrentes relacionadas aos dados dos anfitriões franqueados, afim de minimizar a sobrecarga no banco de dados principal. Dessa forma, a experiência de uso na aplicação será melhora, proporcionando mais fluidez e responsividade, de forma alinhada às necessidades do sistema.

3.4.2.5 Celery

O Celery é um sistema de fila de tarefas assíncronas amplamente utilizado para processar operações em segundo plano de forma eficiente. Ele permite a execução distribuída

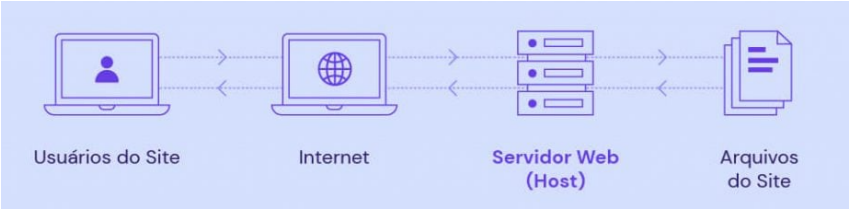
e paralela de tarefas, sendo ideal para aplicações que exigem processamento intensivo ou a execução de tarefas demoradas sem bloquear a resposta da aplicação (CELERY, 2025).

Neste trabalho, o Celery é utilizado para lidar com operações assíncronas, como envio de notificações, processamento de relatórios e outras tarefas que exigem execução em segundo plano. Integrado ao Redis como *broker*, ele garante eficiência na distribuição e execução das tarefas, reduzindo a carga sobre a aplicação principal e melhorando a responsividade do sistema.

3.4.3 Infraestrutura e provisionamento da solução

Para garantir que todas as camadas projetadas com as tecnologias mencionadas estejam acessíveis à equipe de atendimento por meio de um navegador web, é essencial realizar o provisionamento dessas soluções em uma infraestrutura de hospedagem adequada. De acordo com Santos (2023), a hospedagem pode ser definida como o "local onde os dados serão armazenados para garantir o funcionamento correto da ferramenta". De maneira mais ampla, a figura 7 ilustra uma sistematização do acesso às aplicações, destacando o fluxo de interações entre os diferentes atores do fluxo.

Figura 7 – Funcionamento do acesso a uma aplicação web



Fonte: HOSTINGER. 2025. Disponível em: https://www.hostinger.com/ph/tutorials/what-is-a-web-server. Acesso em 21 jul. 2025.

Nessa imagem, é possível observar que, mais que um lugar onde os dados da aplicação são armazenados, uma hospedagem é um servidor que exerce um papel fundamental na liberação do acesso à aplicação em si, trazendo um aspecto de segurança para suas atribuições. Os serviços de hospedagem de sites e aplicações Web também fornecem suporte adicional, como de backup e verificação de performance, permitindo que os desenvolvedores voltem-se à tarefas mais estratégicas, ao invés de despender energia para configurar ambientes e mitigar falhas nesse sentido (AWS, 2025b).

Além do provisionamento do servidor, é crucial preparar o ambiente para garantir o funcionamento adequado da aplicação, tanto em termos de desenvolvimento quanto de lançamento de novas funcionalidades. Dessa forma, as tecnologias e plataformas a serem adotadas foram escolhidas com o objetivo de assegurar uma solução que seja não apenas altamente disponível e escalável, mas também segura, garantindo uma experiência eficiente e confiável para os usuários finais.

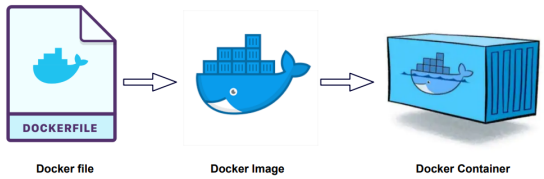
3.4.3.1 Docker

O Docker é uma plataforma amplamente utilizada para desenvolver, enviar e executar aplicativos. Ele é capaz de "empacotar" e executar um aplicativo em um ambiente isolado, chamado de contêiner (DOCKER, 2025). Os contêineres, por sua vez, encapsula uma aplicação junto com suas dependências, bibliotecas, binários e arquivos de configuração em um único pacote que pode ser executado consistentemente em diferentes sistemas ou máquinas (MANOR et al., 2025).

A utilização do Docker se justifica pela facilidade em lidar estes conteineres, tanto no sentido de portabilidade da aplicação, quanto para provisionar a mesma aplicação em diferentes ambientes, como desenvolvimento, teste, staging e produção. Nesse sentido, é possível garantir que todos esses ambientes sejam configurados de maneira idêntica, sem surpresas ao mover a aplicação entre eles. Isso reduz significativamente os erros de configuração e aumenta a eficiência da equipe de desenvolvimento (DOCKER, 2025).

A portabilidade do Docker é um dos seus maiores destaques, permitindo que os desenvolvedores criem contêineres desde a etapa inicial com a criação do arquivo Dockerfile. Esse arquivo contém instruções que definem o ambiente e as dependências da aplicação. Com as configurações especificadas, o próximo passo é construir a imagem Docker (com o comando build), a qual serve como base para a criação do contêiner (DOCKER, 2025). Esse fluxo pode ser observado na figura 8.

Figura 8 – Processo de construção de um conteiner com Docker



Fonte: CAMPOS, Paulo. 2023. Disponível em: https://medium.com/@paulo.campos.dev/introdução-a-docker-tudo-o-que-você-precisa-saber-para-criar-seus-primeiros-contêineres. Acesso em 21 jul. 2025.

No contexto deste trabalho, o Docker é utilizado para garantir que todas as partes do sistema, como o *back-end*, o banco de dados e o *front-end*, sejam executadas de forma consistente e isolada, em containers que podem ser facilmente provisionados e escalados em diferentes ambientes, desde o desenvolvimento até a produção.

3.4.3.2 Amazon Web Services (AWS)

A Amazon Web Services (AWS) é uma plataforma de computação em nuvem abrangente, oferecendo mais de 200 serviços completos em data centers distribuidos globalmente. Lançada em 2006, a AWS tem se consolidado como líder no mercado de

infraestrutura cloud, proporcionando soluções que vão desde serviços computacionais e armazenamento até inteligência artificial e Internet das Coisas (IoT) (AWS, 2025c).

Segundo Wittig (2023), a principal proposta de valor da AWS reside na eliminação da necessidade de investimentos iniciais em infraestrutura física, permitindo que empresas de todos os portes adotem um modelo de pagamento conforme o uso. Essa abordagem possibilita escalabilidade sob demanda, alta disponibilidade e redundância geográfica, fatores cruciais para aplicações modernas como a desenvolvida neste trabalho.

No contexto deste projeto, a AWS desempenhará um papel substancial no provisionamento da infraestrutura necessária para hospedar a aplicação de simulação. A plataforma foi escolhida por sua confiabilidade, segurança e capacidade de escalar recursos conforme a demanda, além de oferecer ferramentas robustas para monitoramento e gerenciamento da infraestrutura. Através da AWS, será possível disponibilizar a aplicação para toda a equipe de suporte técnico, garantindo alta disponibilidade e desempenho consistente.

Para a finalidade de provisionamento da solução na web utilizados no caso de uso deste trabalho, foram utilizados o **Amazon Elastic Compute Cloud (EC2),** este é um serviço web que disponibiliza capacidade computacional segura e redimensionável na nuvem. Foi projetado para tornar a computação em escala na web mais acessível para desenvolvedores, permitindo o controle completo sobre os recursos computacionais e a possibilidade de executar aplicações em um ambiente confiável (AWS, 2025a).

De acordo com Wittig (2023), o EC2 se destaca pela flexibilidade que oferece aos usuários, permitindo a seleção entre diversas instâncias com diferentes capacidades de CPU, memória, armazenamento e rede; a capacidade de aumentar ou diminuir recursos conforme a demanda; e a facilidade de acoplamento com outros serviços do ecossitema AWS, como sistemas de armazenamento, banco de dados e redes.

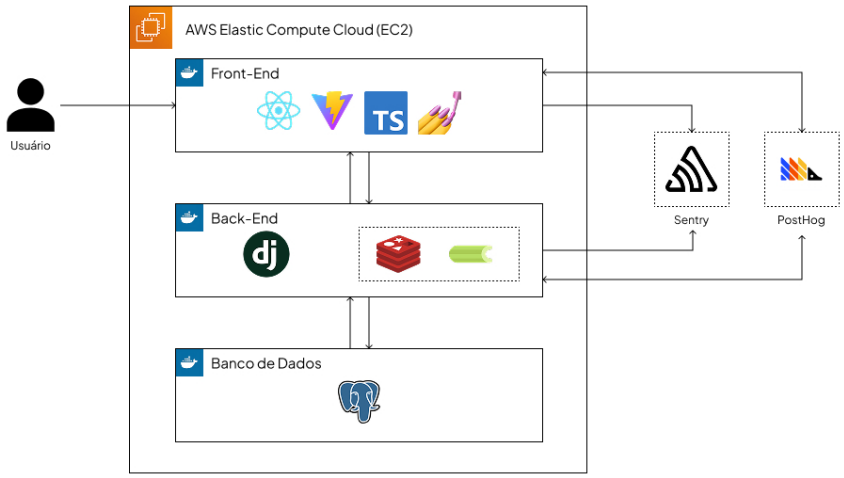
O EC2, neste trabalho, é utilizado como componente central para hospedagem da aplicação. As instâncias EC2 executarão os contêineres Docker que encapsulam tanto o *front-end* quanto o *back-end* da solução, aproveitando a configuração previamente estabelecida nos ambientes de desenvolvimento. Para garantir alta disponibilidade, serão implementadas múltiplas instâncias distribuídas em diferentes zonas de disponibilidade, com balanceamento de carga para distribuir eficientemente as requisições.

3.5 Arquitetura da solução

A análise dos fluxos de operação e do stack tecnológico empregado no desenvolvimento permite a elaboração de um modelo arquitetural *as-is* da solução proposta, detalhando suas integrações sistêmicas e processos técnicos. Conforme ilustrado na Figura 9, o diagrama apresenta a disposição dos componentes tecnológicos e a dinâmica de suas

interações, especificando os tempos e movimentos das interações entre os módulos da aplicação.

Figura 9 – Arquitetura *as is* da solução



Fonte: AUTOR, 2025

No diagrama, todas as camadas estão conteinerizadas, afim de garantir portabilidade tanto para o desenvolvimento, quanto para o *deploy* da aplicação. A camada de *front-end* é a grande responsável por captar interações e receber entradas de dados pelos usuários do time de atendimento. Esses dados são processados pela camada de *back-end*, que busca registros correlatos em *cache*. Caso existam, o *back-end* retorna imediatamente os dados solicitados através de *endpoints*; caso contrário, ele busca o registro solicitado na requisição na camada de banco de dados e também os retorna, caso existam nas tabelas.

Todas essas interações acontecem e são mantidas operantes dentro de uma instância EC2 da AWS, cuja finalidade central é provisionar a aplicação para acesso pelos navegadores dos colaboradores da empresa de aluguel por temporada.

Quanto ao Sentry e PostHog, que podem ser observados fora do ecossistema da AWS, estarão diretamente conectados tanto ao front, quanto ao *back-end*. O Sentry apresenta uma relação unidirecional para com essas camadas, uma vez que quando a captura de erros ocorre, estes são sumarizados e mostrados na própria plataforma do Sentry. Já o PostHog apresenta uma relação bidirecional, afinal, tanto ele captura eventos da aplicação – e gera uma visualização em sua plataforma – quanto ele manda dados booleanos ou de contexto para as aplicações, através de *feature flags*.

Dessarte, a arquitetura apresentada demonstra uma solução robusta e bem estruturada para atender às regras de negócio, com separação clara de responsabilidades entre as camadas de apresentação, lógica de negócio e persistência de dados. A combinação de conteinerização e o próprio provisionamento da plataforma garantem sua escalabilidade horizontal e alta disponibilidade, visando suportar adequadamente este processo crítico

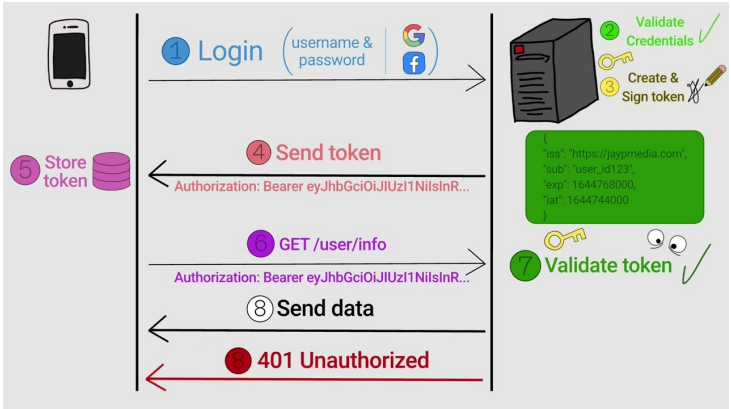
dentro do time de atendimento. Além disso, a arquitetura modular facilita futuras manutenções e expansões funcionais, permitindo que novos recursos sejam incorporados sem comprometer a estabilidade do sistema existente.

3.6 Fluxo de autenticação

Um dos principais desafios de engenharia deste trabalho consistiu em viabilizar a simulação da autenticação como anfitrião sem a necessidade de utilização de credenciais reais. Esse mecanismo não apenas se configura como um módulo central da solução, mas também responde diretamente à exigência de segurança do negócio, já que seria inviável fornecer acesso direto aos dados de clientes para fins de suporte. Em termos técnicos, o problema pode ser resumido na necessidade de permitir que um colaborador autenticado na plataforma pudesse "assumir" a perspectiva de um anfitrião, preservando ao mesmo tempo a integridade dos dados.

Nesse contexto, o fluxo inicia-se no momento em que o colaborador de suporte insere suas credenciais institucionais na tela de login da aplicação. Nesse instante, é realizada uma requisição HTTP do tipo *POST* para o endpoint `/login`, localizado no *back-end*. Esse *endpoint* é responsável por validar as credenciais: a senha informada é criptografada em trânsito, e em seguida o sistema realiza a busca no banco de dados pelo *e-mail* e senha fornecidos. Caso não haja correspondência, a resposta retornada é o código de status HTTP 401 (*Unauthorized*). Quando as credenciais estão corretas, o sistema retorna um status 200 (*OK*) e emite um *token* JSON Web Token (JWT).

Figura 10 – Fluxo básico de login com JWT



Fonte: JAYPMEDIA. 2024. Disponível em: https://dev.to/jaypmedia/jwt-explained-in-4-minutes-with-visuals-g3n. Acesso em 21 jul. 2025.

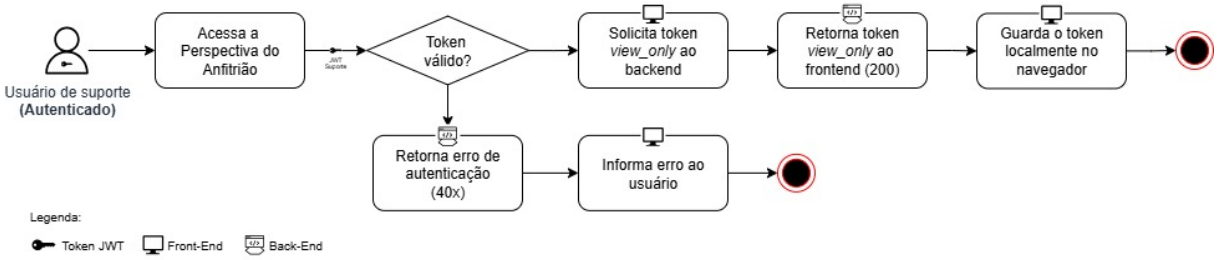
O JWT, conforme definido no padrão aberto RFC 7519, representa um formato compacto para a transmissão segura de informações entre cliente e servidor por meio de objetos *JSON* (MONTANHEIRO; CARVALHO; RODRIGUES, 2017). No contexto desta

aplicação, o *token* gerado é armazenado no navegador do usuário de suporte, tanto em *cookies* quanto no *localStorage*. Esse *token* acompanha todas as requisições subsequentes à API, servindo como credencial de autorização para operações compatíveis com o perfil do colaborador.

Uma vez autenticado, o usuário pode pesquisar anfitriões na base de dados. Quando seleciona um deles para visualizar sua perspectiva, inicia-se um novo fluxo de autenticação específico para esse caso. A aplicação abre uma nova a sessão da aplicação na tela de perspectiva, acrescentando parâmetros na URL que indicam o modo de visualização e o identificador do anfitrião escolhido (por exemplo: `/?mode=view&user_id=ID_DO_USUARIO`). A partir disso, um *middleware* implementado no *front-end* intercepta a solicitação, reconhece o pedido de acesso em modo somente leitura (*view-only*) e envia uma requisição ao *endpoint* `/login/view-only-token`, no *back-end*.

Esse *endpoint* executa duas tarefas fundamentais: em primeiro lugar, valida o *token* JWT original do suporte, garantindo que a solicitação parta de um usuário devidamente autenticado; em seguida, emite um novo JWT configurado exclusivamente para o modo *view-only*m conforme ilustrado na figura 11. Esse *token* diferenciado é gravado no navegador e possui permissões restritas, permitindo apenas operações de leitura, sem qualquer possibilidade de inserção, atualização ou exclusão de dados.

Figura 11 – Fluxo de autenticação como *view-only*



Fonte: AUTOR, 2025

Nesse estágio, coexistem dois *tokens* no navegador: (i) o *token* institucional do colaborador de suporte, que garante acesso às funcionalidades próprias de seu papel; e (ii) o *token* restrito da perspectiva do anfitrião, utilizado quando o usuário navega em URLs que contêm os parâmetros de visualização. A alternância entre os dois *tokens* é orquestrada pelo *middleware* implementado na aplicação, que identifica o contexto de cada requisição e aplica o *token* apropriado para autorização junto à API.

Esse arranjo técnico garante que a equipe de suporte consiga simular a experiência do anfitrião com elevado grau de fidelidade, mas em um ambiente controlado e seguro. A separação entre *tokens* de suporte e *tokens* de visualização impede ações indevidas e

assegura que os dados sensíveis da base operacional permaneçam íntegros, ainda que o colaborador esteja reproduzindo a perspectiva de um cliente real.

3.7 Considerações finais sobre o capítulo

O presente capítulo apresentou os materiais e métodos empregados no desenvolvimento da ferramenta proposta. Inicialmente, foram discutidas as premissas metodológicas que fundamentaram a pesquisa, seguidas de um detalhamento das tecnologias que possibilitaram a construção da solução. Em sequência, descreveu-se o funcionamento da ferramenta, contemplando aspectos de arquitetura e autenticação da solução.

De modo geral, buscou-se alinhar o percurso metodológico aos objetivos do estudo, assegurando a robustez técnica da implementação e a clareza necessária para sua compreensão. Esse conjunto de descrições estabelece a base para a análise crítica da solução, cujos resultados serão explorados nos capítulos seguintes.

4 Resultados e Discussão

4.1 Considerações iniciais sobre o capítulo

Este capítulo apresenta os resultados obtidos com a implementação da ferramenta de suporte, denominada "Perspectiva do Anfitrião". A abordagem adotada para a exposição dos resultados consiste na validação funcional da solução, demonstrando que os fluxos de utilização e os mecanismos de segurança operam conforme os requisitos definidos no escopo do projeto. A jornada do usuário, detalhada a seguir, serve como evidência prática de que a ferramenta atende aos seus objetivos centrais, fornecendo uma réplica fiel e segura do ambiente do cliente para a equipe de suporte.

4.2 Validação funcional da ferramenta

A validação funcional constitui uma etapa metodológica essencial no desenvolvimento de sistemas, tendo por finalidade verificar se a solução implementada atende aos requisitos especificados e opera de acordo com as expectativas dos usuários finais (IEEE, 2017). No contexto deste trabalho, a validação funcional da ferramenta "Perspectiva do Anfitrião" foi conduzida para assegurar que o sistema desenvolvido cumprisse os objetivos propostos de fornecer um ambiente simulado seguro e eficaz para a equipe de suporte técnico. Esta validação envolveu a análise sistemática dos fluxos operacionais, mecanismos de segurança e funcionalidades implementadas, com foco na verificação da conformidade entre os requisitos funcionais e não funcionais estabelecidos na metodologia e a solução entregue.

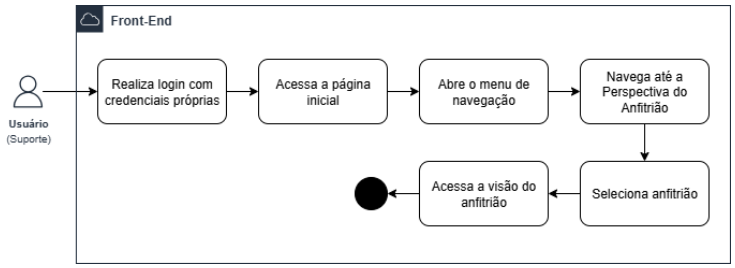
O processo de validação foi realizado por uma equipe multidisciplinar composta por um profissional da área de produto e um especialista em *Quality Assurance* (QA) da empresa, totalizando dois avaliadores com expertise complementar. A escolha destes perfis profissionais justifica-se pela necessidade de uma análise abrangente que contemplasse tanto os aspectos de usabilidade e adequação ao negócio quanto os critérios técnicos de qualidade e conformidade. Os avaliadores conduziram testes estruturados baseados em cenários de uso real, verificando a operacionalidade das funcionalidades desenvolvidas e a aderência aos requisitos de segurança estabelecidos, particularmente no que se refere às restrições de acesso em modo view-only e à proteção de dados sensíveis.

A metodologia de validação adotada baseou-se em uma análise comparativa entre os requisitos especificados e as funcionalidades implementadas, permitindo a identificação de lacunas, oportunidades de melhoria e necessidades de correção antes da disponibilização

da ferramenta ao usuário final. Este processo de validação sistemática foi essencial para o refinamento da ferramenta, assegurando que ela atendesse efetivamente às necessidades operacionais da equipe de suporte e aos padrões de segurança exigidos pela organização.

Sendo assim, a partir das bases arquiteturais, tecnologias de desenvolvimento e da condução de refinamentos apresentados anteriormente, foi construída a ferramenta de simulação, concebida para apoiar a atuação da equipe de suporte no segmento de franquias da empresa de *short-stay*. O diagrama de sequência do fluxo de utilização do sistema pode ser observado na figura 12:

Figura 12 – Fluxo esperado de jornada do usuário

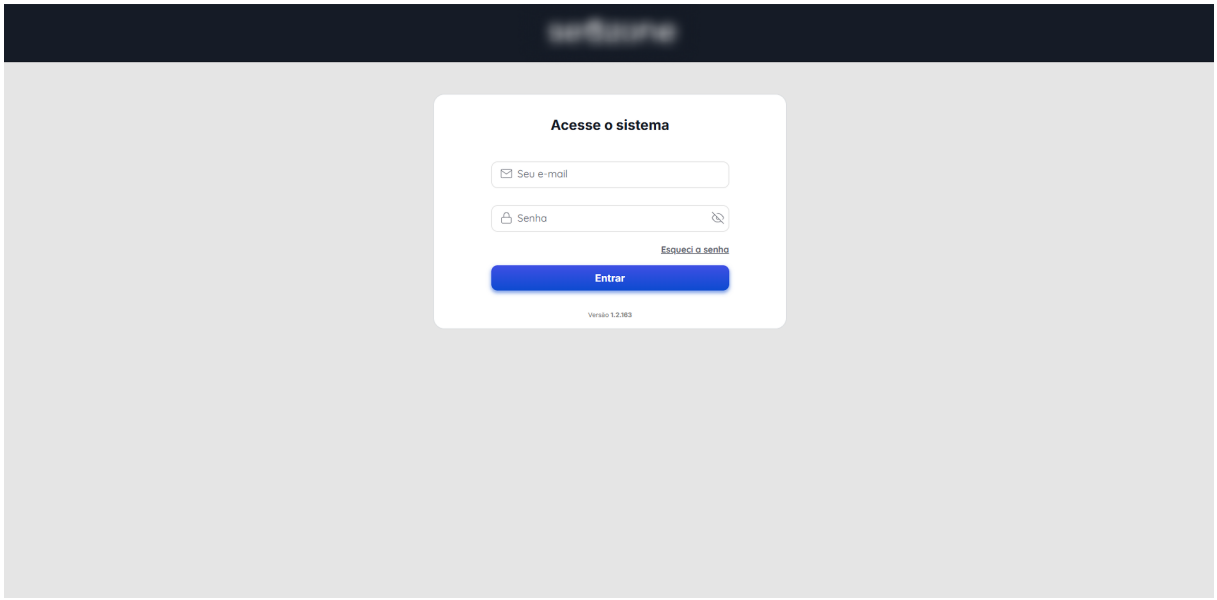


Fonte: AUTOR, 2025

4.2.1 Autenticação em ambiente produtivo

A primeira etapa validada corresponde ao processo de autenticação inicial. A Figura 13 confirma a implementação da tela de login, onde o colaborador do suporte acessa a aplicação utilizando credenciais institucionais previamente configuradas. Esse fluxo garante que apenas usuários autorizados tenham acesso ao ambiente de simulação, em conformidade com os requisitos de segurança do projeto.

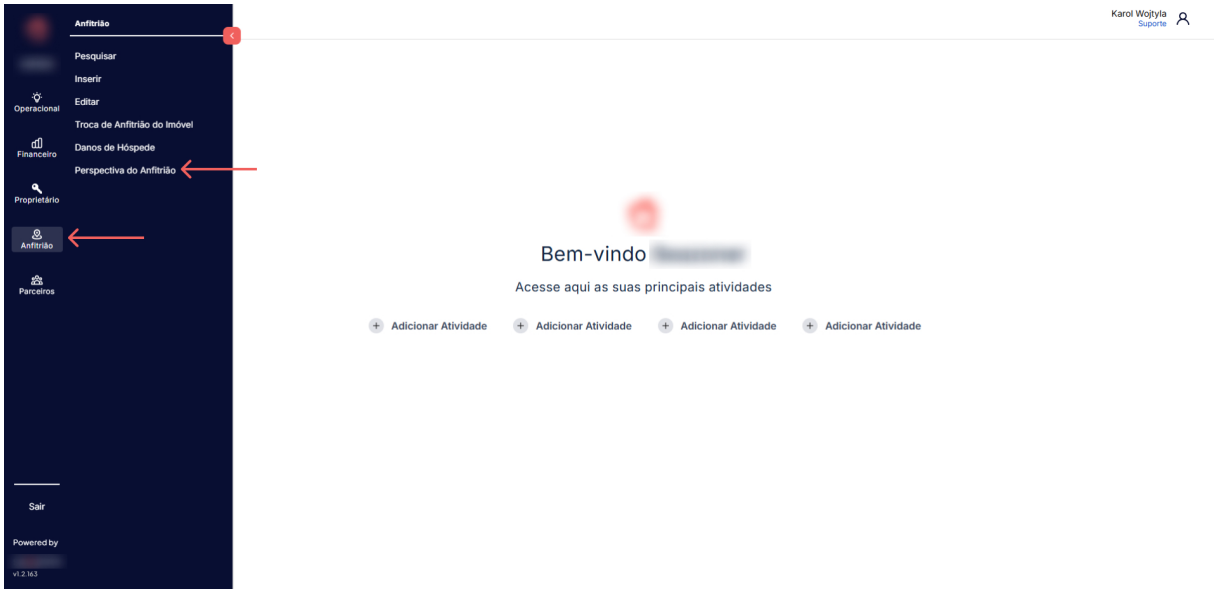
Figura 13 – Tela de *login* da aplicação completa



Fonte: AUTOR, 2025

Uma vez autenticado, o usuário é direcionado à página inicial da aplicação (Figura 14), que reúne o painel principal e as funcionalidades disponíveis. Essa tela valida a correta implementação da navegação, permitindo o acesso intuitivo ao módulo desenvolvido.

Figura 14 – Página inicial da aplicação

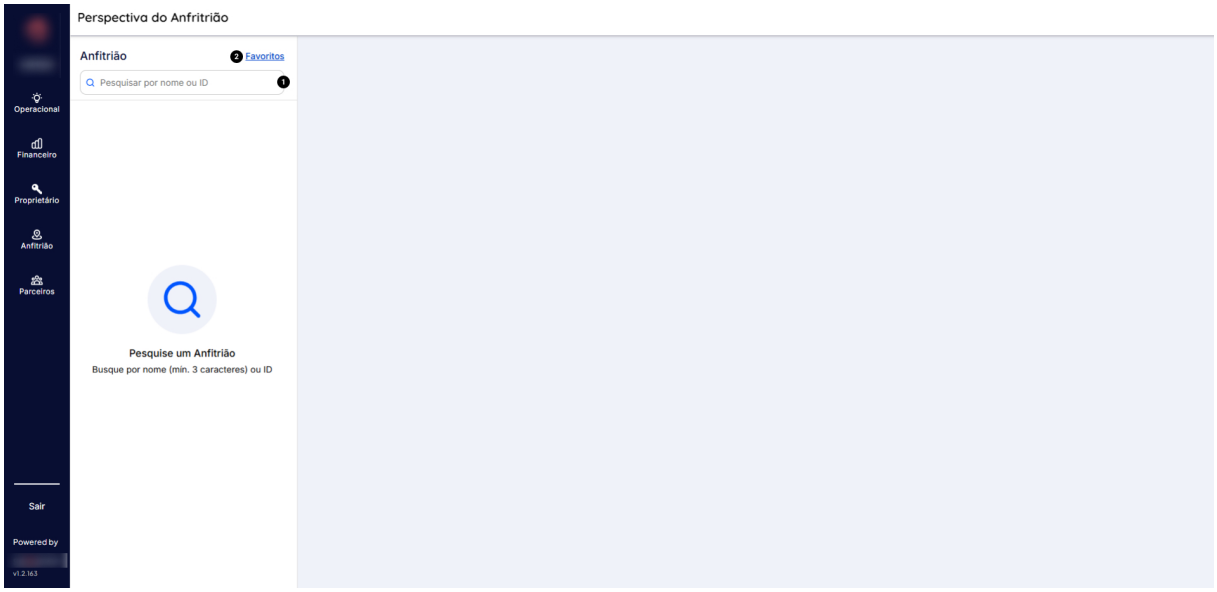


Fonte: AUTOR, 2025

4.2.2 Acesso à ferramenta de simulação

Ao selecionar o menu “Anfitrião” e a opção “Perspectiva do Anfitrião”, o usuário é redirecionado à *dashboard* da ferramenta de simulação. A Figura 15 demonstra a correta implementação desse módulo, que se consolida como o núcleo da solução, reunindo os recursos necessários para a simulação da experiência do anfitrião.

Figura 15 – Visão geral da Perspectiva do Anfitrião

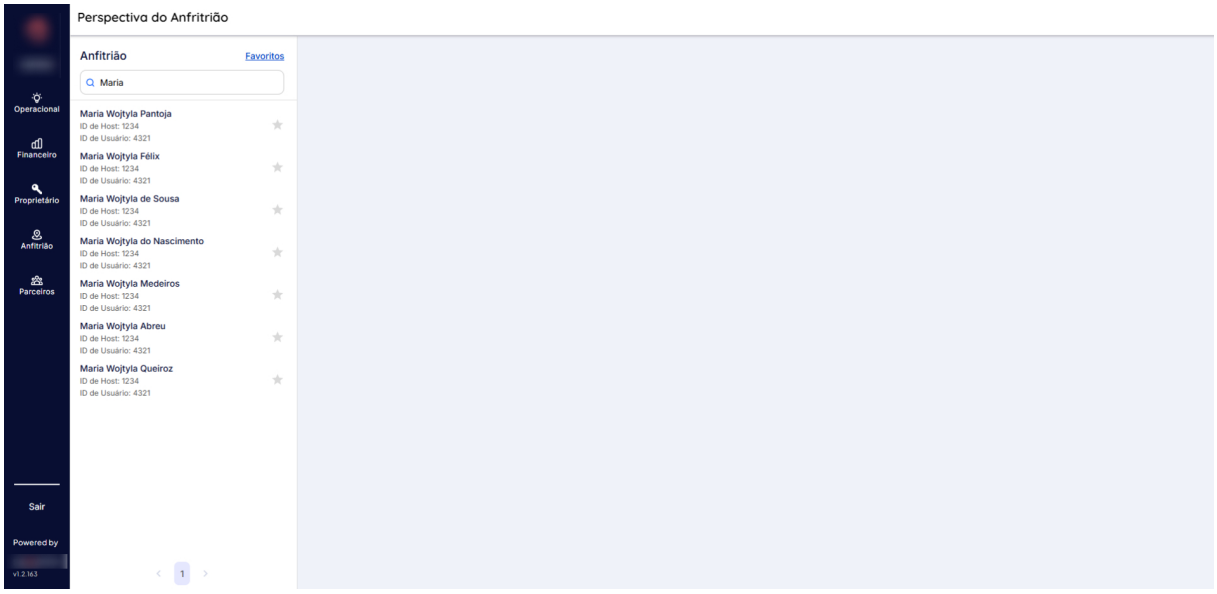


Fonte: AUTOR, 2025

A partir desse painel, o colaborador pode interagir com diferentes funcionalidades desenvolvidas para otimizar sua rotina de atendimento. Entre as principais, destacam-se:

1. **Pesquisa de anfitriões:** esta funcionalidade foi implementada para permitir encontrar mais rapidamente o anfitrião solicitante de suporte na base de dados da empresa. Este fluxo foi implementado para combinar tanto comodidade na navegação – ao filtrar os anfitriões mais facilmente por nome ou ID de Anfitrião – quanto para economizar processamento e recursos do servidor – ao implementar uma filtragem a partir de 3 caracteres ou mais. Quando digitados os 3 caracteres mínimos, são mostrados os resultados possíveis imediatamente, para que o usuário possa escolher o usuário o qual deseja visualizar, conforme exemplifica a figura 16.

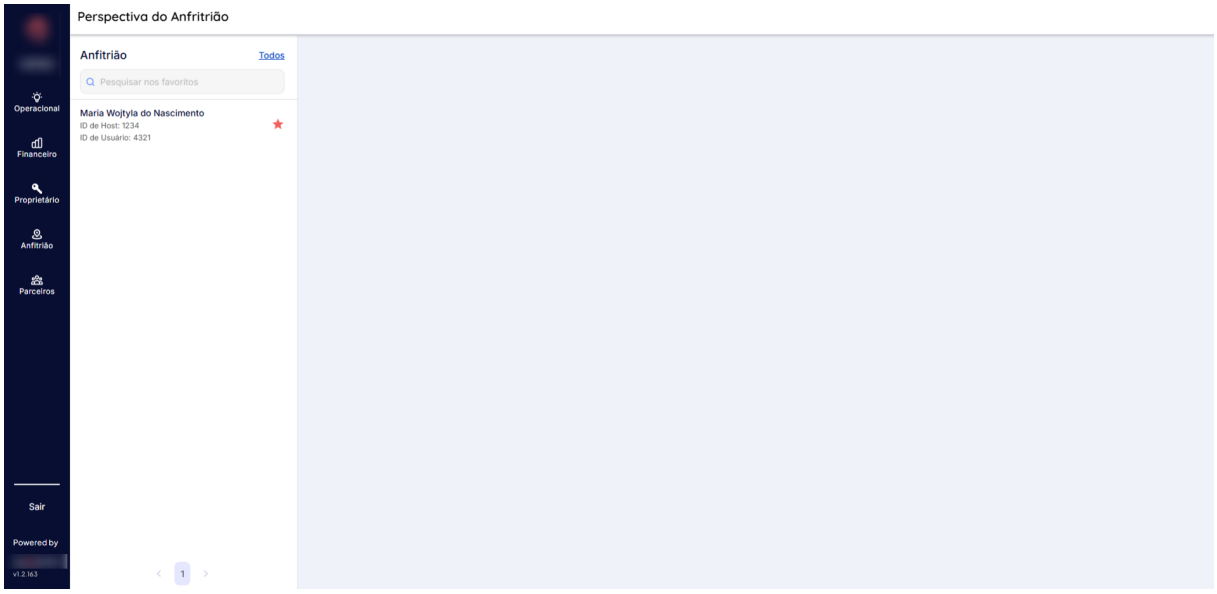
Figura 16 – Exemplo de pesquisa por anfitrião



Fonte: AUTOR, 2025

2. **Anfitriões favoritos:** possibilita que o colaborador de CS mantenha uma relação personalizada de anfitriões frequentemente atendidos. Essa funcionalidade foi concebida para agilizar o acesso em casos de alta recorrência de chamados ou no acompanhamento de *tickets* que demandam suporte prolongado. A lista é armazenada localmente no navegador, de modo que cada usuário pode gerenciar suas próprias preferências, de forma independente e escalável. A Figura 17 apresenta um exemplo de utilização dessa funcionalidade.

Figura 17 – Tela de lista de anfitriões favoritos

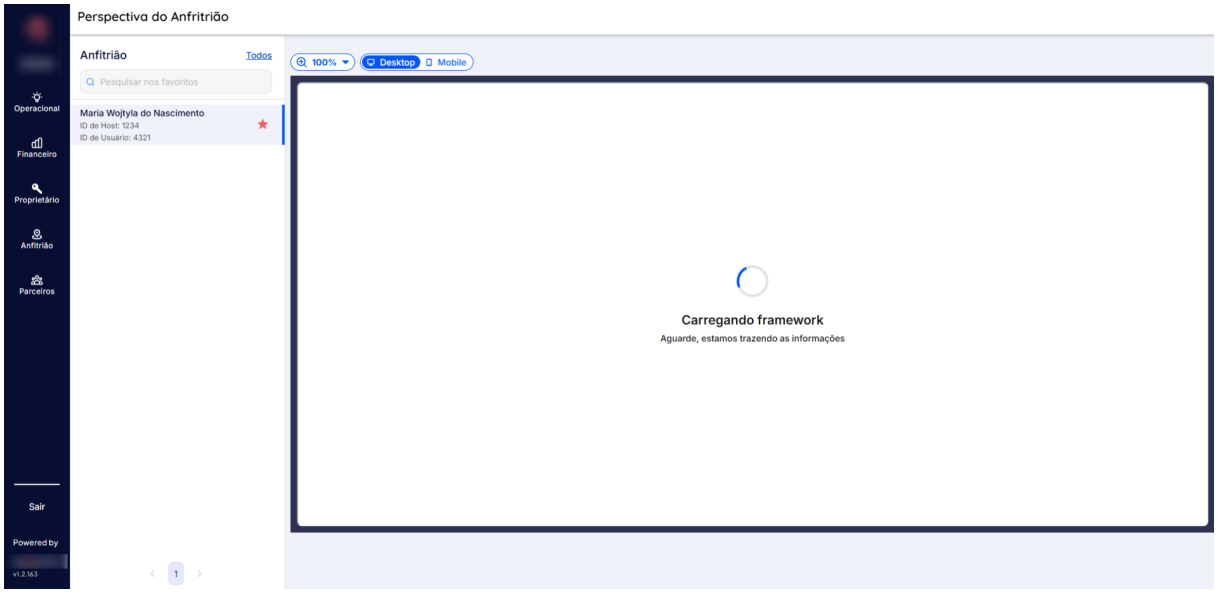


Fonte: AUTOR, 2025

4.2.3 Visualização no ambiente simulado

Independentemente da forma de acesso escolhida – pesquisa ou lista de favoritos – a interação converge para uma mesma ação: a abertura da perspectiva do anfitrião. Esse processo foi validado por meio de um fluxo que redireciona o usuário para uma nova sessão em modo *view-only*, impedindo qualquer alteração nos dados. A Figura 18 apresenta a tela de carregamento, implementada como mecanismo de *feedback* ao usuário durante o processamento da requisição.

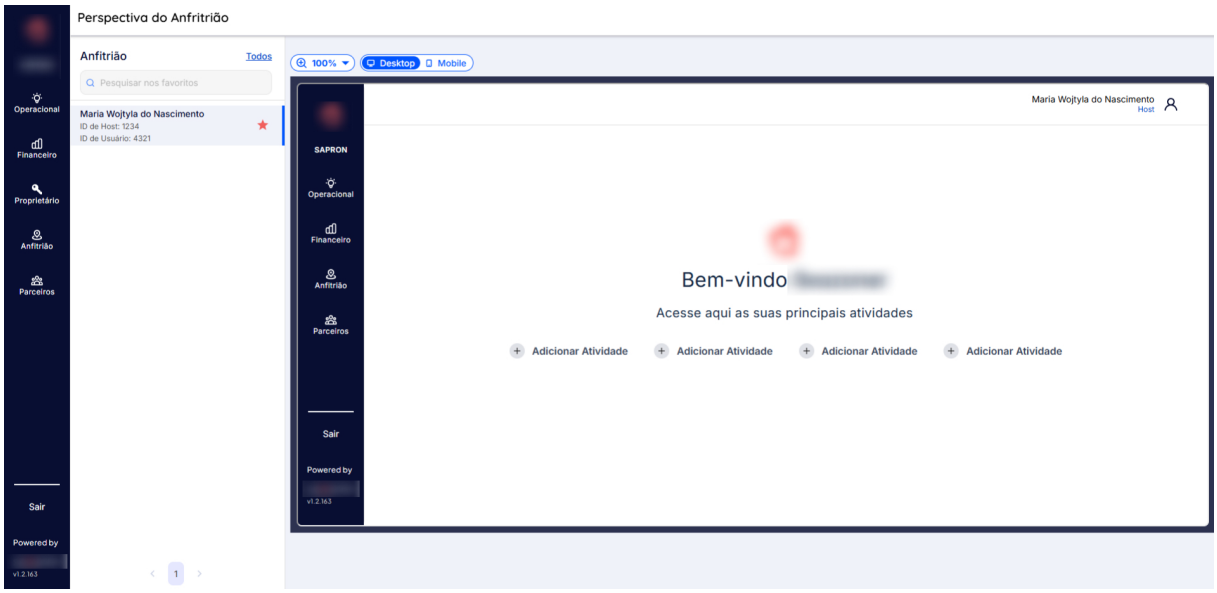
Figura 18 – Tela de carregamento da perspectiva do anfitrião



Fonte: AUTOR, 2025

Concluído o carregamento, abre-se uma nova sessão da aplicação que reproduz fielmente a visualização que o anfitrião teria ao acessar o sistema. Essa simulação, ilustrada na Figura 19, constitui o núcleo da ferramenta, pois possibilita ao colaborador experimentar a interface do cliente e, assim, compreender com maior precisão as dificuldades relatadas durante o atendimento.

Figura 19 – Tela de perspectiva do anfitrião autenticada

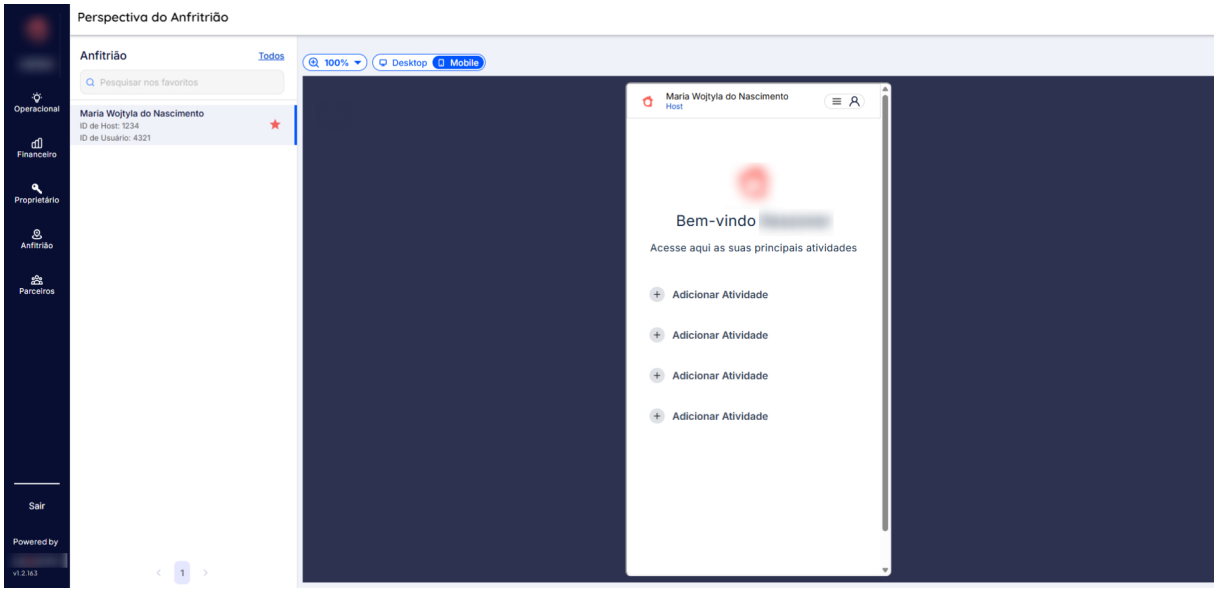


Fonte: AUTOR, 2025

No caso de o anfitrião atendido utilizar a aplicação por meio de um *smartphone*, a

ferramenta oferece também a opção de visualização na versão *mobile*. Esse recurso pode ser ativado ao clicar no botão localizado acima da tela de perspectiva, possibilitando que diferentes cenários de uso sejam contemplados e que os testes em modo de visualização se tornem mais completos e representativos da experiência real do usuário.

Figura 20 – Tela de perspectiva do anfitrião na versão *mobile*



Fonte: AUTOR, 2025

É importante destacar que as permissões disponibilizadas no modo de visualização da ferramenta são restritas ao formato somente leitura (*view-only*). Isso significa que quaisquer operações de edição, exclusão ou inserção de dados são bloqueadas por mecanismos de segurança implementados diretamente em código, assegurando que a integridade e a confidencialidade das informações dos clientes não sejam comprometidas. Nesse contexto, a tabela 2 apresenta uma síntese das principais funcionalidades acessíveis na "Perspectiva do Anfitrião", ainda que sob tais restrições de uso.

Tabela 2 – Funcionalidades acessíveis na Perspectiva do Anfitrião

Universo	**Funcionalidade**	**Permissões**
Operacional	Multicalendar	Consultar a disponibilidade dos imóveis, bloqueios e manutenções em formato de linha do tempo
	Controle	Consultar *check-ins* e *check-outs* em reservas da semana
	Danos de Hóspede	Consultar danos de hóspedes em reservas registrados pelo anfitrião
Financeiro	Pedido de Reembolso	Consultar o andamento dos pedidos de reembolso realizados pelo anfitrião
	Dashboard	Consultar o resumo financeiro da franquia e todo o detalhamento de faturamento, despesas e receitas em determinado período
Anfitrião	Score da Franquia	Consultar a principal métrica interna de desempenho da franquia visualizada

Fonte: AUTOR, 2025

Por fim, tendo o suporte realizado a triagem ou resolução do *ticket* do franqueado, um colaborador pode encerrar o acesso com segurança através do *logout* nativo da ferramenta, o que reforça as práticas de proteção de dados e evita usos indevidos da plataforma. Assim, a jornada do usuário foi desenhada para ser simples, objetiva e próxima da realidade operacional enfrentada pelas equipes de suporte, servindo como elemento central para a efetiva validação da solução proposta.

4.3 Síntese dos resultados

De forma geral, a validação funcional realizada demonstrou que a solução cumpre os requisitos previstos em sua concepção, traçando um limiar entre os resultados obtidos e interpretando-os à luz dos objetivos propostos neste trabalho. A discussão se orienta, portanto, pela análise de como a validação funcional confirma o atendimento aos objetivos específicos da pesquisa: compreender as dificuldades enfrentadas pela equipe de suporte, projetar um ambiente simulado seguro e realizar uma validação por meio de *checklist* dos requisitos propostos e executados. A partir disso, procura-se avaliar em que medida a ferramenta efetivamente contribui para a otimização dos processos internos, a redução do tempo de resposta e a melhoria da experiência do usuário.

Além disso, esta seção também busca problematizar pontos de atenção identificados durante o desenvolvimento e utilização inicial da solução, indicando aspectos que podem

ser aprimorados em futuras versões. Com isso, almeja-se abrir caminhos para a evolução contínua da ferramenta frente às regras de negócio e necessidades da equipe atendida, que se adaptam também aos requisitos dos clientes finais.

4.3.1 Contribuições para a eficiência do suporte técnico

A validação funcional da ferramenta demonstrou que o sistema permite à equipe de suporte visualizar o ambiente do anfitrião de forma fidedigna e segura, sem a necessidade de acesso direto às credenciais do cliente. Esse resultado representa um avanço significativo na rotina operacional, uma vez que elimina barreiras que dificultavam a resolução mais rápida de chamados relacionados a problemas específicos do ambiente do usuário.

Do ponto de vista prático, a utilização da perspectiva do anfitrião reduz o tempo necessário para compreender e reproduzir o cenário relatado no atendimento. Esse aspecto contribui diretamente para a agilidade no processo de suporte, permitindo que os colaboradores atuem de forma mais objetiva e direcionada à resolução do problema. Além disso, a integração de funcionalidades como a pesquisa de anfitriões e a lista de favoritos reforça a preocupação com o aspecto de navegação, já que otimiza o acesso aos casos mais recorrentes.

Esses achados dialogam com a proposta central da pesquisa de aprimorar a eficiência do atendimento e apoiar o trabalho das equipes de suporte em contextos de alta demanda. A ferramenta, ao centralizar e simplificar o acesso à perspectiva do cliente, consolida-se como um recurso estratégico para tornar o processo de suporte mais ágil e eficaz.

4.3.2 Segurança e conformidade

Outro aspecto relevante evidenciado pela validação funcional refere-se à segurança e conformidade da solução. Desde sua concepção, a ferramenta foi projetada para operar em modo *view-only*, o que impede qualquer tipo de modificação, exclusão ou inserção de dados durante a simulação da perspectiva do anfitrião. Essa característica assegura que a integridade das informações seja preservada, eliminando riscos de alterações indevidas que poderiam comprometer tanto o sistema quanto a experiência do cliente real.

A utilização de mecanismos de autenticação baseados em JWTs reforça ainda mais a confiabilidade do processo, uma vez que garante que apenas colaboradores devidamente autorizados tenham acesso às funcionalidades da ferramenta. Esse controle de permissões atende às exigências de segurança da informação, reduzindo a probabilidade de acessos não autorizados ou de vazamento de dados sensíveis.

Além disso, a solução mostra-se alinhada às diretrizes da Lei Geral de Proteção de Dados (LGPD), na medida em que evita o uso direto de credenciais dos clientes e promove a visualização mediada por *tokens* temporários e restritos. Ao eliminar a necessidade de

compartilhar ou manipular dados pessoais durante o suporte, a ferramenta contribui para a conformidade legal da empresa e fortalece a confiança dos usuários na plataforma.

Portanto, pode-se afirmar que a ferramenta não apenas alcança seu objetivo operacional de simulação, mas também cumpre um papel estratégico ao incorporar práticas seguras de desenvolvimento, em consonância com legislações e normas aplicáveis ao tratamento de dados pessoais.

4.4 Considerações finais sobre o capítulo

O presente capítulo apresentou os resultados da validação funcional da ferramenta desenvolvida, demonstrando, por meio de evidências práticas, que os fluxos de autenticação, acesso à perspectiva do anfitrião, navegação em modo *view-only* e adaptação para diferentes dispositivos operam conforme planejado. As telas e interações exibidas confirmam que as funcionalidades centrais foram corretamente implementadas e que a solução é capaz de reproduzir a experiência do anfitrião de forma segura e controlada.

De modo geral, os resultados atestam que a ferramenta cumpre os requisitos estabelecidos em sua concepção, validando sua aplicabilidade no contexto do suporte técnico a clientes de plataformas de *short-stay*. Essa constatação reforça o alinhamento entre o desenvolvimento realizado e os objetivos específicos da pesquisa, estabelecendo a base necessária para a análise crítica de seus impactos, a ser realizada no capítulo seguinte.

5 Conclusões

O presente trabalho teve como tema central o desenvolvimento de uma ferramenta de simulação planejada para melhorar o processo de suporte técnico à franqueados de uma empresa do setor de *short-stay*. A proposta consistiu em criar uma solução capaz de reproduzir a perspectiva do usuário franqueado de forma segura, visualmente idêntica e acessível à equipe de suporte, de modo a eliminar a necessidade de utilização de credenciais reais e, ao mesmo tempo, assegurar conformidade com as diretrizes da Lei Geral de Proteção de Dados (LGPD).

A relevância do tema manifesta-se em múltiplas dimensões. Do ponto de vista científico, este estudo contribui para a discussão sobre ambientes simulados como estratégia de suporte, apresentando uma aplicação prática que alia segurança da informação e eficiência operacional. Sob a perspectiva social e organizacional, a ferramenta desenvolvida agrega valor ao processo de atendimento, ao possibilitar respostas mais ágeis e direcionadas às demandas dos clientes, reduzindo frustrações e fortalecendo a relação entre usuário e empresa. No âmbito formativo e profissional, a pesquisa evidencia a aplicação prática de conceitos de engenharia de *software* em um contexto real de negócio, demonstrando a viabilidade de soluções tecnológicas voltadas à melhoria de processos de suporte.

Os objetivos inicialmente propostos foram alcançados. O objetivo geral — investigar e desenvolver uma solução que permitisse a simulação do ambiente do cliente para equipes de suporte — foi atendido com a implementação da ferramenta “Perspectiva do Anfitrião”. Quanto aos objetivos específicos, foi possível (i) analisar as dificuldades enfrentadas pelo suporte técnico na resolução de chamados relacionados ao ambiente do cliente; (ii) projetar e desenvolver um ambiente simulado que garante privacidade e segurança; e (iii) realizar testes funcionais em ambiente produtivo, os quais validaram a aplicabilidade e a eficácia esperada da solução.

A apresentação dos resultados evidenciou que a ferramenta cumpre seus requisitos centrais. O processo de autenticação validado assegura que apenas usuários autorizados acessem a plataforma; o acesso à perspectiva do anfitrião mostrou-se funcional e intuitivo; e as restrições de uso em modo *view-only* confirmaram a preservação da integridade dos dados. Além disso, recursos como a pesquisa de anfitriões, a lista de favoritos e a possibilidade de simulação em versão *mobile* reforçaram a utilidade prática da solução no cotidiano da equipe de suporte.

Embora o presente estudo tenha se concentrado na validação funcional da ferramenta, abre-se espaço para trabalhos futuros que possam aprofundar sua avaliação sob a ótica dos usuários. Recomenda-se, como desdobramento, a aplicação de instrumentos

qualitativos e quantitativos, tais como formulários, *surveys* e monitoramento de *Key Performance Indicators* (KPIs), a fim de mensurar de maneira mais precisa o impacto da ferramenta na rotina de suporte e em seus indicadores de desempenho.

Em síntese, o trabalho desenvolvido contribui de forma objetiva para o campo da engenharia de *software* aplicada ao atendimento ao cliente, oferecendo uma solução prática, segura e alinhada às necessidades de um setor em constante crescimento. A ferramenta "Perspectiva do Anfitrião" configura-se como uma inovação aplicável ao suporte técnico de plataformas de aluguel por temporada, reafirmando a importância de aproximar a prática tecnológica das demandas reais do mercado.

Referências

ANDRADE, R. S. d. A importância da excelência no atendimento ao cliente. *Faculdade de Tecnologia Assis (FATEC Assis)*, 2022. Citado 2 vezes nas páginas 6 e 7.

AWS, A. W. S. *Amazon EC2*. 2025. Disponível em: <https://aws.amazon.com/pt/ec2>. Citado na página 28.

AWS, A. W. S. *O que é Hospedagem na Web*. 2025. Disponível em: <https://aws.amazon.com/pt/what-is/web-hosting>. Citado na página 26.

AWS, A. W. S. *What is AWS?* 2025. Disponível em: <https://aws.amazon.com/pt/what-is-aws>. Citado na página 28.

BRASIL. *Lei nº 13.709, de 14 de agosto de 2018 - Lei Geral de Proteção de Dados Pessoais*. 2018. Disponível em: <https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm>. Citado na página 11.

CARMO, K. X.; FERREIRA, F.; FIGUEIREDO, E. Performance evaluation of back-end frameworks: A comparative study. In: *Proceedings of the 20th Brazilian Symposium on Information Systems*. [S.l.: s.n.], 2024. p. 1–9. Citado na página 22.

CAVALCANTI, D. *Short Stay: o que é e como aproveitar essa oportunidade*. 2023. Disponível em: <https://stays.net/blog/short-stay-como-aproveitar-essa-oportunidade>. Citado 2 vezes nas páginas 4 e 5.

CELERY. *Celery Documentation*. 2025. Disponível em: <https://docs.celeryq.dev/>. Citado na página 26.

CRESCENZO, A. W. de. Otimização de precificação dinâmica para aluguel de temporada. *Universidade Federal de Santa Catarina (UFSC)*, Florianópolis, SC, 2021. Citado na página 5.

DJANGO. *Django Documentation*. 2025. Disponível em: <https://www.djangoproject.com>. Citado 2 vezes nas páginas 23 e 24.

DOCKER. *Docker Documentation*. 2025. Disponível em: <https://docs.docker.com>. Citado na página 27.

FENTON, S. *TypeScript: Application-Scale JavaScript Development*. 2. ed. New York, NY: Apress, 2018. 281 p. Citado 2 vezes nas páginas 17 e 18.

FRAZÃO, A.; OLIVA, M. D.; TEPEDINO, G. *Lei Geral de Proteção de Dados Pessoais e suas repercussões no direito brasileiro*. [S.l.]: Thomson Reuters Brasil, 2019. Citado na página 12.

GOMES, P. G. G. N. et al. Vulnerabilidades: Panorama das legislações de proteção de dados pessoais gdpr, ccpa, lgpd e pipl. *Direito UNIFACS–Debate Virtual-Qualis A2 em Direito*, n. 297, 2025. Citado na página 11.

GORAI, S. et al. Web development barriers and challenges. *International Journal of Scientific Research in Engineering and Management (IJSREM)*, v. 09, n. 01, 2025. Citado 3 vezes nas páginas 17, 21 e 22.

IEEE, I. of E. . E. E. Ieee standard for system, software, and hardware verification and validation. *IEEE Std 1012-2016 (Revision of IEEE Std 1012-2012/ Incorporates IEEE Std 1012-2016/Cor1-2017)*, 2017. Citado na página 33.

KREMER, M. *Introduction to Full-Stack Web Development*. [S.l.]: University of California, 2023. v. 2. 466 p. Citado na página 17.

LATHKAR, M. *Modern Django Web Development*. [S.l.]: Apress, 2025. Citado 2 vezes nas páginas 23 e 24.

MANOR, D. et al. Containerization in cloud computing: A review. 2025. Citado 2 vezes nas páginas 9 e 27.

MASSé, M. *REST API Design Rulebook*. [S.l.]: O'Reilly Media, 2011. ISBN 978-1-449-31050-9. Citado 2 vezes nas páginas 22 e 23.

MDN, M. D. N. *Começando com React*. 2024. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn_web_development/Core/Frameworks_libraries/React_getting_started>. Citado na página 19.

MICROSOFT. *TypeScript Documentation*. 2024. Disponível em: <https://www.typescriptlang.org/docs>. Citado 2 vezes nas páginas 18 e 19.

MONTANHEIRO, L. S.; CARVALHO, A. M. M.; RODRIGUES, J. A. Utilização de json web token na autenticação de usuáriosem apis rest. *XIII Encontro Anual de Computação - EnAComp*, 2017. Citado na página 30.

MORCELLI, R. A. T. Modelagem de um serviço para gestão de imóveis de aluguel de temporada. *Universidade Federal de Santa Maria (UFSM)*, 2024. Citado na página 5.

MOURA, L. V. d.; COUTINHO, E. Lgpd e requisitos de software: Desafios e oportunidades de pesquisa. In: SBC. *Workshop sobre Aspectos Sociais, Humanos e Econômicos de Software (WASHES)*. [S.l.], 2024. p. 169–174. Citado na página 9.

NEVES, D. L. F. et al. A segurança da informação de encontro às conformidades da lgpd. *Revista Processando o Saber*, v. 13, p. 186–198, 2021. Citado na página 10.

OLIVEIRA, A. A. d. A importância dos softwares de simulação dentro da indústria 4.0: uma análise da inserção do digital twin nos contextos industriais. 2023. Citado na página 8.

PIHLAJANIEMI, N. Requirement gathering for an it project. *Haaga-Helia university of applied sciences*, 2023. Citado na página 15.

POSTGRESQL. *PostgreSQL Documentation*. [S.l.], 2025. Disponível em: <https://www.postgresql.org/docs>. Citado na página 24.

RAVSHANOVICH, A. R. Database structure: Postgresql database. *PSIXOLOGIYA VA SOTSIOLOGIYA ILMIY JURNALI*, v. 2, n. 7, p. 50–55, 2024. Citado na página 24.

REDIS. *Redis Documentation*. 2025. Disponível em: <https://redis.io/docs>. Citado na página 25.

SANTOS, R. B. d.; SILVA, T. B. P. e. Gestão da segurança da informação e comunicações análise ergonômica para avaliação de comportamentos inseguros. *RDBCI: Revista Digital de Biblioteconomia e Ciência da Informação*, SciELO Brasil, v. 19, p. e021024, 2021. Citado na página 11.

SCHON, J. R.; LIRA, P. A importância da gestão de atendimento ao cliente para o sucesso de uma organização. *Scientia y Humanity V. 2, n. 1, jul./dez. 2022*, v. 2, n. 1, p. 15, 2022. Citado na página 7.

SEAZONE. *Como funciona o aluguel por temporada?* 2023. Disponível em: <https://institucional.seazone.com.br/blog/como-funciona-aluguel-por-temporada>. Citado na página 5.

SÊMOLA, M. *Gestão da segurança da informação*. [S.l.]: Elsevier Brasil, 2014. v. 2. Citado na página 10.

SILVA, F. G. C. d. Levantamento de requisitos aplicado à pesquisa e desenvolvimento de produtos de software. Faculdade de Tecnologia de São Paulo, 2023. Citado na página 15.

SOUZA, D. *A importância da simulação de atendimento ao cliente em software*. 2024. Disponível em: <https://blog.casadodesenvolvedor.com.br/simulacao-de-atendimento-ao-cliente/>. Citado 2 vezes nas páginas 1 e 2.

STYLED-COMPONENTS. *Styled Components Documentation*. [S.l.], 2025. Disponível em: <https://styled-components.com/docs>. Citado 2 vezes nas páginas 20 e 21.

VITE. *Vite – Next Generation Frontend Tooling*. 2024. Disponível em: <https://vite.dev>. Citado na página 20.

ZENDESK. *Desafios no atendimento ao cliente na indústria de TI*. 2023. Disponível em: <https://www.zendesk.com.br/blog/atendimento-ao-cliente-industria-ti/>. Citado 2 vezes nas páginas 1 e 8.