

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO
RIO GRANDE DO SUL
CAMPUS CANOAS
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE
SISTEMAS

GABRIELE PESCHKE BAUM

**PumaJourney: Aplicação web para administração de hospedagem da Pousada Cabanas
Passos do Puma**

Canoas

2024

GABRIELE PESCHKE BAUM

PumaJourney: Aplicação web para administração de hospedagem da Pousada Cabanas
Passos do Puma

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas pelo Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul - Campus Canoas.

Prof. Dr. Dieison Soares Silveira
Orientador

Canoas

2024



Ministério da Educação
Secretaria de Educação Profissional, Científica e Tecnológica
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul
Campus Canoas

ATA DE DEFESA PÚBLICA DO TRABALHO DE CONCLUSÃO DE CURSO

Ao 01 dia do mês de agosto de 2024, às 14 horas, em sessão pública na sala <https://meet.google.com/mqp-mfjv-hxn> do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul, Campus Canoas, na presença da Banca Examinadora presidida pelo Professor:

Dr. Dieison Soares Silveira e composta pelos examinadores:

1. Prof. Dr. Sandro Silva
2. Prof. MSc. Marco Aurélio Schünke,

a aluna Gabriele Peschke Baum apresentou o Trabalho de Conclusão de Curso intitulado:

PumaJourney: Aplicação web para administração de hospedagem da Pousada Cabanas Passos do Puma como requisito curricular indispensável para a integralização do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas. Após reunião em sessão reservada, a Banca Examinadora deliberou e decidiu pela APROVAÇÃO do referido trabalho, divulgando o resultado formalmente a aluna e demais presentes e eu, na qualidade de Presidente da Banca, lavrei a presente ata que será assinada por mim, pelos demais examinadores e pela aluna.

Documento assinado digitalmente
gov.br DIEISON SOARES SILVEIRA
Data: 01/08/2024 14:54:29-0300
Verifique em <https://validar.itl.gov.br>

Presidente da Banca Examinadora

Documento assinado digitalmente
gov.br SANDRO JOSE RIBEIRO DA SILVA
Data: 01/08/2024 15:51:07-0300
Verifique em <https://validar.itl.gov.br>

Examinador 01

Documento assinado digitalmente
gov.br MARCO AURELIO SCHUNKE
Data: 01/08/2024 15:43:05-0300
Verifique em <https://validar.itl.gov.br>

Examinador 02

Documento assinado digitalmente
gov.br GABRIELE PESCHKE BAUM
Data: 02/08/2024 10:58:14-0300
Verifique em <https://validar.itl.gov.br>

Aluna

RESUMO

Com o aumento do uso da tecnologia no setor hoteleiro, as aplicações de gestão interna para hotéis têm ganhado cada vez mais espaço e importância. O presente trabalho tem como objetivo criar um sistema web de gestão de reservas e consumos para a Pousada Cabanas Passos do Puma, uma pousada que iniciou suas atividades em 2020 na cidade de São José dos Ausentes e tem aumentado suas demandas devido ao crescimento no número de cabanas disponíveis. A aplicação visa aumentar a eficiência na gestão, oferecendo funcionalidades como calendário de reservas, criação de reservas, gestão de consumo dos hóspedes e cálculos automáticos dos gastos. Para o desenvolvimento, foram utilizadas tecnologias como TypeScript, React.js, Node.js, Fastify e PostgreSQL. A hospedagem foi realizada na plataforma Vercel, enquanto o banco de dados foi gerenciado pelo serviço AWS RDS (*Relational Database Service*). Para validação da aplicação, uma versão de teste foi disponibilizada, resultando na aprovação satisfatória do cliente final, o que demonstra que o trabalho atingiu seu objetivo principal.

Palavras-Chave: Aplicação web. Sistema de Gestão de Propriedades. Gestão de reservas. Gestão de consumo de hóspedes.

ABSTRACT

With the increasing use of technology in the hospitality industry, internal management applications for hotels have gained more space and importance. This work aims to create a web-based system for managing reservations and consumption for the Guesthouse Cabanas Passos do Puma, an guesthouse that started its activities in 2020 in the São José dos Ausentes city and has been increasing its demands due to the growth in the number of available huts. The application aims to enhance management efficiency by offering features such as reservation calendar, reservation creation, guest consumption management, and automatic expense calculations. For development, technologies such as TypeScript, React.js, Node.js, Fastify, and PostgreSQL were used. Deploy was performed on the Vercel platform, while the database was managed using AWS RDS (Relational Database Service). For application validation, a test version was made available, resulting in satisfactory approval from the end client, demonstrating that the work achieved its main objective.

Keywords: Web Application. Property Management System. Reservation Management. Guest Consumption Management.

LISTA DE FIGURAS

Figura 1: Diagrama de casos de uso para um sistema de aluguel de carros.....	15
Figura 2: Processo Scrum.....	16
Figura 3: Exemplo de arquitetura cliente-servidor.....	18
Figura 4: Seções principais de um projeto Git.....	21
Figura 5: Tela de calendário do Cloudbeds.....	23
Figura 6: Calendário de reservas do Sirvoy.....	24
Figura 7: Tela principal do RoomRaccon.....	25
Figura 8: Lista de itens no sistema Jira.....	30
Figura 9: Exemplo de item no sistema Jira.....	31
Figura 10: Exemplo de User Story.....	32
Figura 11: Digrama de casos de uso do sistema PumaJourney.....	33
Figura 12: Diagrama entidade-relacionamento do sistema PumaJourney.....	35
Figura 13: Tela de login.....	36
Figura 14: Tela inicial.....	37
Figura 15: Opções disponíveis para uma reserva no calendário.....	38
Figura 16: Escolha do hóspede existente.....	39
Figura 17: Criação de novo hóspede.....	40
Figura 18: Captura de tela da validação do hóspede.....	41
Figura 19: Etapa de reserva sem dados preenchidos.....	42
Figura 20: Etapa da reserva com dados preenchidos.....	43
Figura 21: Etapa de reserva com o valor total apresentado.....	44
Figura 22: Etapa de consumo.....	44
Figura 23: Resumo da reserva criada.....	45
Figura 24: Edição da reserva.....	46
Figura 25: Visualização da reserva.....	46
Figura 26: Checkout de uma reserva.....	47
Figura 27: Cancelamento da reserva.....	48
Figura 28: Teste unitário de cancelamento de reserva.....	49
Figura 29: Executando todos os testes unitários.....	49
Figura 30: Teste E2E de criação de reserva.....	50
Figura 31: Executando todos os testes E2E.....	50

LISTA DE TABELAS

Tabela 1: Comparativo dos sistemas.	25
--	----

LISTA DE ABREVIATURAS E SIGLAS

API	<i>Application Programming Interface</i>
AWS	<i>Amazon Web Services</i>
CPF	Cadastro de Pessoa Física
CSS	<i>Cascading Style Sheets</i>
E2E	<i>End-to-end</i>
ER	Entidade-Relacionamento
HTML	<i>Hypertext Markup Language</i>
JS	JavaScript
MVP	Produto Mínimo Viável
PMS	<i>Property Management System</i>
RDS	<i>Relational Database Service</i>
RS	Rio Grande do Sul
SGBD	Sistema de Gerenciamento de Banco de dados
SQL	<i>Structured Query Language</i>
SSR	<i>Server-Side Rendering</i>
UML	<i>Unified Modeling Language</i>
WWW	<i>World Wide Web</i>

SUMÁRIO

1. INTRODUÇÃO	11
1.1. MOTIVAÇÃO.....	12
1.2. OBJETIVOS.....	13
1.2.1. OBJETIVO GERAL.....	13
1.2.2. OBJETIVOS ESPECÍFICOS	13
2. REFERENCIAL TEÓRICO	14
2.1. UML	14
2.2. METODOLOGIA ÁGIL	15
2.3. BANCO DE DADOS	16
2.4. ARQUITETURA CLIENTE-SERVIDOR.....	17
2.5. TECNOLOGIAS FRONTEND.....	18
2.6. TECNOLOGIAS BACKEND.....	19
2.7. VERSIONAMENTO DE CÓDIGO	20
3. ESTADO DA ARTE.....	22
3.1. CLOUDBEDS	22
3.2. SIRVOY	23
3.3. ROOMRACCOON.....	24
3.4. COMPARATIVO	25
4. METODOLOGIA	27
5. APLICAÇÃO PUMAJOURNEY	29
5.1. LEVANTAMENTO DE REQUISITOS	29
5.1.1. USER STORIES.....	31
5.1.2. DIAGRAMA DE CASO DE USO	33
5.2. MODELAGEM DO BANCO DE DADOS	34
5.3. DESENVOLVIMENTO DA APLICAÇÃO	35
5.3.1. CONEXÃO E HOSPEDAGEM.....	35
5.3.2. TELA DE LOGIN	36
5.3.3. TELA INICIAL	37
5.3.4. CADASTRO DE RESERVA DE HOSPEDAGEM	38
5.3.5. EDIÇÃO DE RESERVA DE HOSPEDAGEM.....	45
5.3.6. VISUALIZAÇÃO DE RESERVA DE HOSPEDAGEM	46
5.3.7. CHECKOUT DE RESERVA.....	47
5.3.8. CANCELAMENTO DE RESERVA	47

5.3.9. TESTES	48
6. CONSIDERAÇÕES FINAIS	52
REFERÊNCIAS	53

1. INTRODUÇÃO

Atualmente, a tecnologia vem crescendo cada vez mais e as demandas dentro desta área tornam-se cada vez mais importantes em diversas áreas da sociedade. A constante evolução tecnológica tem incentivado as organizações a buscarem formas inovadoras de se adaptarem a essa nova realidade e atenderem às demandas cada vez mais complexas do público.

A procura por hospedagens pós pandemia também tem aumentado significativamente. De acordo com um estudo realizado pela Booking, muitos brasileiros planejam viajar entre duas e quatro vezes por ano. Além disso, o estudo também mostrou que os brasileiros estão mais interessados em viagens domésticas e priorizam bastante as medidas sanitárias de suas hospedagens. (Booking, 2021)

Diante do cenário promissor no setor hoteleiro, a Pousada Cabanas Passos do Puma, localizada em São José dos Ausentes, não é exceção. A pousada começou a procurar alguma forma para tornar suas atividades mais práticas e ter maior facilidade em relação à organização, considerando que atualmente tudo é feito de maneira manual, o que torna tarefas simples, como por exemplo o cálculo final das despesas dos hóspedes, extremamente trabalhosas.

A pousada iniciou suas atividades em janeiro de 2020 na cidade de São José dos Ausentes, sendo gerenciada pelo casal proprietário, que também atua como funcionário da pousada. Desde o início, todas as reservas, desde hospedagens à alimentação, vêm sendo conduzidas pelos proprietários através das redes sociais, com o auxílio de calendários e agendas físicas, onde todas as informações são anotadas à mão. O cálculo final das despesas dos hóspedes também é manual e demanda tempo. Esse processo, embora tenha funcionado até o momento, tem apresentado algumas limitações significativas.

Dentre os desafios enfrentados, destaca-se a demanda crescente de clientes, o que tem sobrecarregado a administração e tornado tarefas simples, como o cálculo final das despesas dos hóspedes, extremamente trabalhosas e suscetíveis a erros. Diante dessa realidade, os proprietários sentiram a necessidade de otimizar e aprimorar suas operações, permitindo uma gestão mais ágil, precisa e eficiente.

Assim, este trabalho se justifica pela urgência em atender às necessidades da Pousada Cabanas Passos do Puma, proporcionando-lhe uma solução personalizada e automatizada que facilite a administração de suas atividades. A aplicação web responsiva proposta, que será chamada PumaJourney, visa simplificar a experiência dos proprietários, proporcionando uma

plataforma intuitiva que centralize e agilize os processos de gestão, controle de disponibilidade e cálculo das despesas.

Este projeto tem como abordagem uma pesquisa qualitativa, de natureza aplicada e com objetivo exploratório. Com o objetivo de implementar uma aplicação web responsiva que visa servir os proprietários, serão utilizadas tecnologias como Typescript e React.js, para o frontend, e Fastify e PostgreSQL, para o backend. Nos próximos capítulos deste trabalho serão apresentados os objetivos, a motivação e a metodologia que foram pensados para este trabalho.

1.1. MOTIVAÇÃO

A Pousada Cabanas Passos do Puma é uma propriedade situada na cidade de São José dos Ausentes, RS. Atualmente, conta com duas cabanas em funcionamento e outras duas em fase de construção. A Pousada deu início às suas atividades no começo do ano de 2020. No momento, opera durante todo o ano, mas exige reserva antecipada devido aos proprietários residirem em outra cidade, o que requer planejamento prévio para a viagem até a propriedade. Os proprietários, um casal apaixonado pela região, são responsáveis por toda a gestão do empreendimento, desde a alimentação, reservas e passeios pela área até a construção e organização das cabanas.

Atualmente, a parte administrativa da Pousada é conduzida de forma inteiramente manual. Para efetuar uma reserva, o hóspede entra em contato diretamente com um dos proprietários, por meio de redes sociais. Essa abordagem não representa um problema significativo, uma vez que os donos da pousada necessitam se organizar antecipadamente para estarem presentes na propriedade e atenderem aos clientes, o que é valorizado pelos visitantes. Durante o processo de reserva, o inquilino deve fornecer informações sobre a data desejada e, se for o caso, solicitar serviços de alimentação adicionais. Uma vez na propriedade, o hóspede é atendido pessoalmente pelos proprietários, com a possibilidade de fazer pedidos de alimentos ou passeios que não tenham sido previamente solicitados. Por fim, todas as solicitações do cliente são registradas em papel, e o cálculo das despesas é realizado manualmente.

É imprescindível estabelecer uma comunicação eficiente entre o hóspede e o proprietário para tratar de questões relacionadas a pedidos de alimentação, agendamento de eventos e cálculos de despesas. Contudo, algumas ocorrências mal-entendidas têm surgido, como, por exemplo, pedidos de alimentação esquecidos, situações que poderiam ser facilmente evitadas. Diante disso, surgiu a ideia de desenvolver uma aplicação web, proporcionando aos proprietários a possibilidade de registrar reservas, alterá-las e gerenciar os consumos dos

hóspedes. Vale destacar que o sistema não será responsável por permitir que os hóspedes façam reservas diretamente. Todas as reservas continuarão sendo realizadas através do contato direto com os proprietários. A aplicação também fornecerá um cálculo final das despesas do hóspede, considerando o valor já pago no momento da reserva.

1.2. OBJETIVOS

Nesta seção serão apresentados o objetivo geral e os objetivos específicos deste trabalho.

1.2.1. OBJETIVO GERAL

Desenvolver uma aplicação web responsiva que facilite a gestão de hospedagens na Pousada Cabanas Passos do Puma.

1.2.2. OBJETIVOS ESPECÍFICOS

- Definir todos os requisitos mínimos e desejáveis do sistema com os proprietários.
- Definir tecnologias a serem utilizadas.
- Definir a modelagem do sistema.
- Desenvolver a aplicação com perfil administrador, que será utilizado pelos proprietários.
- Realizar testes de funcionalidade.
- Realizar testes unitários.
- Validar com o usuário final, liberando a aplicação para testes.

2. REFERENCIAL TEÓRICO

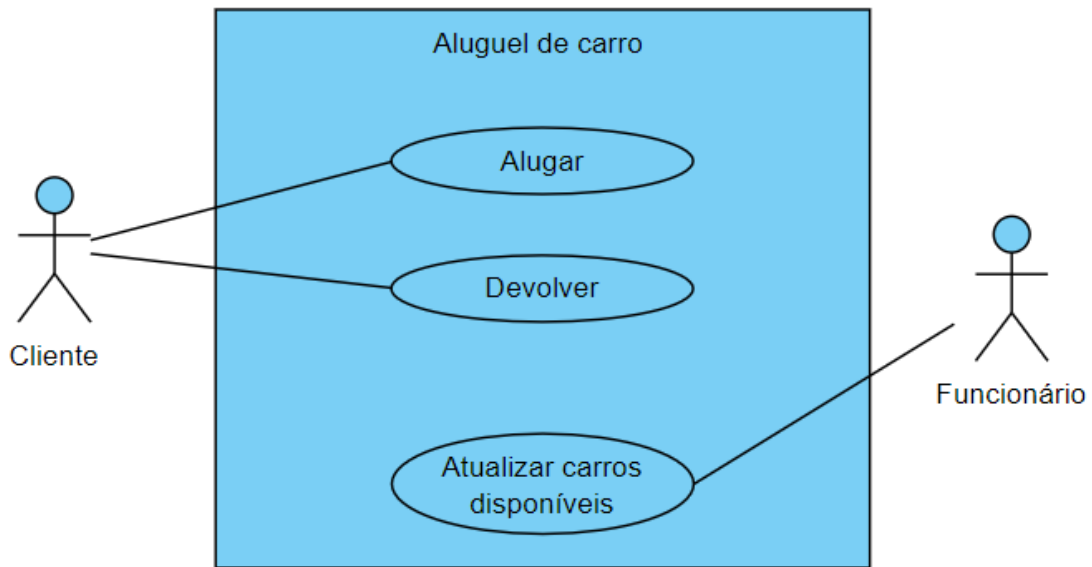
A aplicação desenvolvida neste projeto tem como objetivo facilitar a gestão de hospedagens e o controle de consumos dos hóspedes na Pousada Cabanas Passos do Puma. Para isso, foi desenvolvida uma aplicação web que funciona tanto na versão desktop quanto na versão mobile. Durante o processo de desenvolvimento, foram utilizadas diversas tecnologias e ferramentas, como, por exemplo, UML, Git, React.js e Node.js. Neste capítulo, serão apresentadas as principais tecnologias e ferramentas empregadas no projeto.

2.1. UML

A Unified Modeling Language (UML) foi desenvolvida por volta de 1995 em resposta a necessidade crescente de uma linguagem de modelagem que pudesse lidar com as demandas cada vez mais complexas das aplicações desenvolvidas com linguagens orientadas a objetos. Ela é “uma linguagem gráfica para visualização, especificação, construção e documentação de artefatos de sistemas complexos de software” (BOOCH; RUMBAUGH; JACOBSON, 2006, p.8). Com a introdução da UML, houve a unificação de métodos de modelagem previamente utilizados, oferecendo uma linguagem unificada e padronizada que facilita a construção de projetos orientados a objetos.

A UML engloba vários tipos de diagramas, sendo utilizado um específico para este projeto: o diagrama de caso de uso, que é um diagrama de comportamento. O diagrama de caso de uso é considerado o mais geral e informal da UML, de acordo com Guedes (2018). Ele é utilizado para identificar os usuários, chamados de atores, e descreve as possíveis interações deles com o sistema, porém sem muitos detalhes. Na Figura 1, é apresentado um exemplo simples de diagrama de caso de uso. Os atores do sistema, neste caso o cliente e o funcionário, são representados por bonecos palitos. As elipses representam os casos de uso, que correspondem às funcionalidades do sistema, e são visualmente relacionadas aos atores por meio de uma linha.

Figura 1: Diagrama de casos de uso para um sistema de aluguel de carros.



Fonte: Autoria própria

2.2. METODOLOGIA ÁGIL

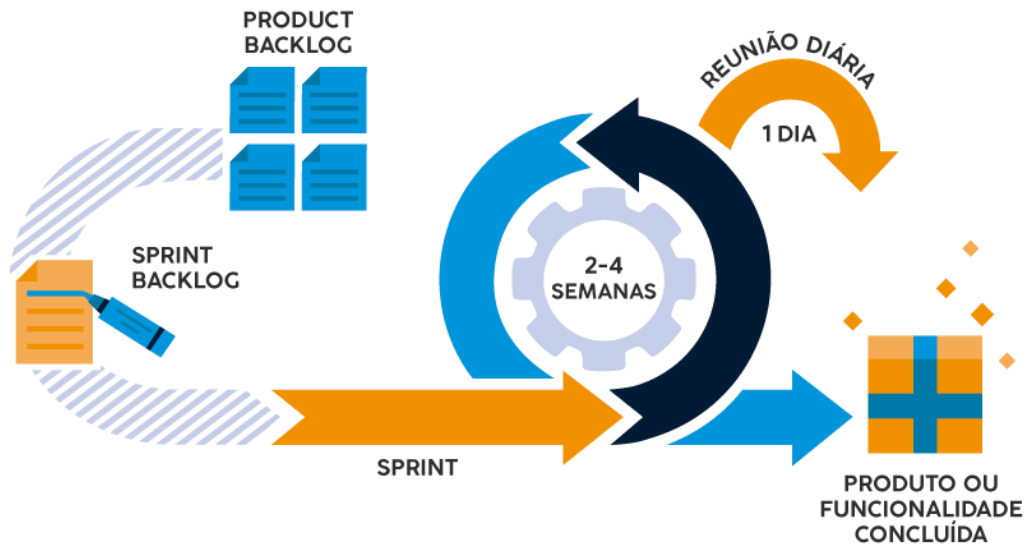
As metodologias ágeis têm como objetivo acelerar as entregas de um projeto e melhorar sua qualidade, fornecendo um processo mais eficiente e que priorize as necessidades dos clientes. Elas consistem na divisão das entregas ao cliente final em ciclos menores, conhecidos como iterações, permitindo que problemas sejam identificados e corrigidos mais rapidamente. Segundo Sbrocco e Macedo (2012, p.13), “cada iteração é considerada um “miniprojeto”, uma vez que inclui todas as tarefas necessárias para implementar a parte que está sendo desenvolvida, como planejamento, análise de requisitos, projeto, codificação, teste e documentação.”.

Uma das metodologias ágeis mais conhecidas é o Scrum. Ela é composta por várias cerimônias, que consistem em planejamento de *sprint*, *sprint*, Scrum diário, revisão de *sprint* e retrospectiva do *sprint*. A fase inicial, planejamento de *sprint*, é uma reunião com os membros da equipe destinada a determinar e estimar as tarefas que devem ser desenvolvidas e concluídas durante o próximo *sprint*. O *sprint* é um período, geralmente de duas semanas, em que ocorre o desenvolvimento das tarefas planejadas. Durante o *sprint*, ocorre o chamado Scrum diário, uma reunião breve com o objetivo de planejar o dia e identificar qualquer possível impedimento que possa dificultar a conclusão das metas do *sprint*. Ao final do *sprint*, ocorrem duas reuniões: a revisão, com o propósito de revisar e apresentar o trabalho realizado, e a retrospectiva, em

que a equipe discute o andamento do *sprint* concluído, identificando possíveis melhorias para os próximos ciclos. (AWS, 2022).

A Figura 2 demonstra como é o básico de um processo Scrum, por meio de uma ilustração.

Figura 2: Processo Scrum



Fonte: Tecnicon (2019)

2.3. BANCO DE DADOS

Um banco de dados é um conjunto sistematizado de informações ou dados estruturados, tipicamente armazenados de forma eletrônica em um sistema computacional. Geralmente, é utilizado um Sistema de Gerenciamento de Banco de dados (SGBD) para administrá-los, facilitando diversas operações com dados, como acesso, gerenciamento, modificação, atualização, controle e organização. (ORACLE, 2014).

Neste trabalho foi utilizado o SGBD PostgreSQL¹. Ele gerencia bancos de dados relacionais, que são organizados em tabelas onde cada linha representa um registro com uma chave única. De acordo com PostgreSQL (2024), ele

[...] é um poderoso sistema de banco de dados objeto-relacional de código aberto que usa e estende a linguagem SQL combinada com muitos recursos que armazenam e dimensionam com segurança as cargas de trabalho de dados mais complicadas (PostgreSQL, 2024).

¹ <https://www.postgresql.org/>

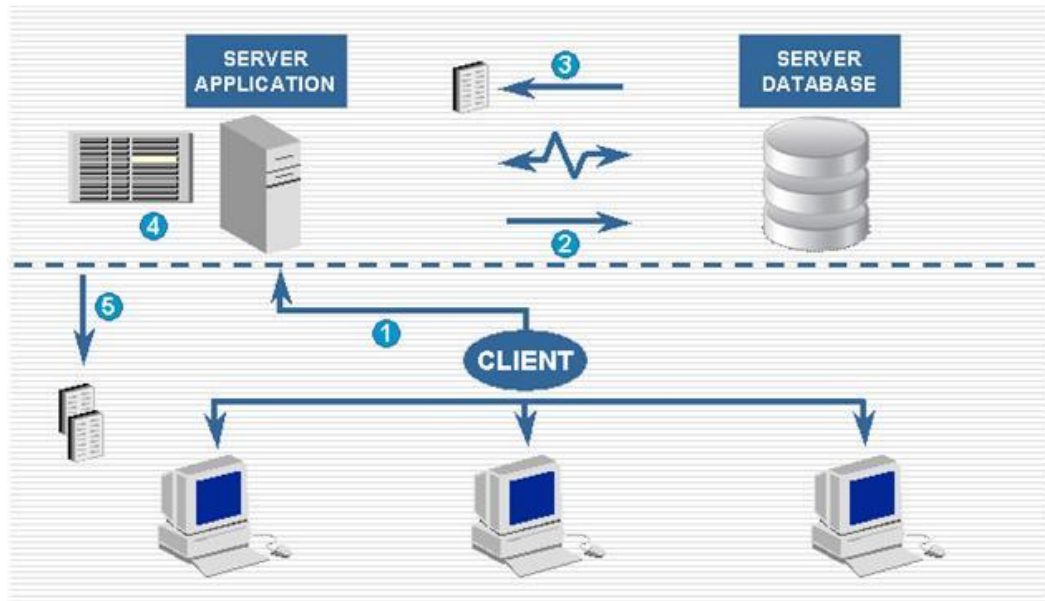
2.4. ARQUITETURA CLIENTE-SERVIDOR

O sistema PumaJourney utiliza a arquitetura Cliente-Servidor. Esse modelo é amplamente utilizado na indústria tecnológica e consiste na interação entre duas partes principais: o cliente e o servidor. O cliente é responsável pela interação direta com o usuário e nele ocorre o envio de requisições para o servidor, solicitando operações específicas. O servidor, por sua vez, recebe essas requisições, processa as operações solicitadas e retorna uma resposta de volta ao cliente após a conclusão. Esse modelo favorece a distribuição de tarefas entre dois componentes, fornecendo uma arquitetura mais escalável e robusta (COULOURIS, 2013).

O modelo cliente-servidor é viável tanto em cenários onde o cliente e o servidor estão próximos fisicamente, como dentro do mesmo prédio, quanto em situações em que estão separados por grandes distâncias. Por exemplo, quando alguém em casa acessa uma página na *World Wide Web* (WWW), o mesmo modelo é utilizado: o servidor web remoto atua como o servidor, enquanto o computador pessoal do usuário funciona como o cliente (TANENBAUM, 2011).

Na Figura 3 podemos observar um exemplo de arquitetura cliente-servidor. Quando o usuário realiza uma ação, o cliente envia uma requisição para o servidor (Passo 1). O servidor, ao receber essa requisição, interage com o banco de dados (Passo 2), acionando o Sistema de Gerenciamento de Banco de Dados. O SGBD executa a solicitação e retorna os dados solicitados para o servidor (Passo 3). Em seguida, o servidor processa esses dados, realizando cálculos ou aplicando regras de negócio conforme necessário, e por fim, envia a resposta de volta para o cliente.

Figura 3: Exemplo de arquitetura cliente-servidor



Fonte: DevMedia (2007)

2.5. TECNOLOGIAS FRONTEND

No projeto em questão, as tecnologias TypeScript, React.js e Next.js foram utilizadas para implementar o frontend, também conhecido como lado do cliente. Essas ferramentas desempenham um papel crucial na criação da interface com a qual os usuários têm acesso e interagem diretamente.

JavaScript², ou JS, é uma linguagem de programação de alto nível amplamente reconhecida como a mais popular no mundo. Sua popularidade se deve ao fato de ser a linguagem padrão interpretada pelos navegadores, trabalhando em conjunto com HTML (Hypertext Markup Language) e CSS (Cascading Style Sheets) para formar a base da Web. Enquanto HTML³ define a estrutura e o conteúdo da página, e CSS⁴ controla o estilo e o visual, JavaScript torna as páginas dinâmicas (ALURA, 2023). No entanto, a linguagem não possui uma tipificação de valores, o que pode causar erros graves e prejudicar o desenvolvimento. Para isso surgiu o TypeScript⁵, que nada mais é do que “uma linguagem de programação fortemente tipada que se baseia em JavaScript, oferecendo melhores ferramentas em qualquer escala” (TypeScript, 2020).

² <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>

³ <https://developer.mozilla.org/pt-BR/docs/Web/HTML>

⁴ <https://developer.mozilla.org/pt-BR/docs/Web/CSS>

⁵ <https://www.typescriptlang.org/>

React.js⁶ é uma biblioteca JavaScript, que também tem suporte a TypeScript, de código aberto utilizada para facilitar a criação de interfaces de usuário interativas e reativas. Ela permite a criação de interfaces utilizando componentes, que são partes isoladas da interface. Esses componentes podem variar de um simples botão a uma complexa tela de formulário (React, 2015). Essa abordagem ajuda a manter o código mais organizado, facilitando tanto a manutenção quanto o reaproveitamento de código.

Next.js⁷ é um *framework*⁸ que fornece uma estrutura básica para projetos React.js. Com ele, podemos ter um acesso facilitado a diversas funcionalidades, como roteamento e *Server-Side Rendering* (SSR). O SSR permite que o conteúdo seja renderizado no servidor e entregue pronto ao usuário, trazendo benefícios para a indexação por mecanismos de busca, melhorando a visibilidade do sistema na web. Outro benefício que o *framework* proporciona é o roteamento automático através *do file-system routing*, ou seja, a estrutura de pastas e arquivos determina as rotas da aplicação. (CLEMENTE, 2023)

2.6. TECNOLOGIAS BACKEND

Para o backend, ou seja, lado do servidor, foram utilizadas tecnologias como Node.js, Fastify, PostgreSQL. Elas garantem uma base sólida e eficiente tanto para o desenvolvimento da API (*Application Programming Interface*) quanto para o gerenciamento de dados.

Como citado anteriormente, JavaScript é uma linguagem de programação muito popular, com foco principal no lado do cliente. No entanto, com a evolução das tecnologias web, tornou-se possível utilizar essa linguagem também no backend, como é o caso do Node.js⁹, um ambiente de execução de código JavaScript do lado do servidor criado em 2009 (ALURA, 2023). Segundo Pereira (2016, p.6) este ambiente encanta muitos desenvolvedores pois, além de possuir uma baixa curva de aprendizado, ele

[...] provou ser uma plataforma excelente na solução de diversos problemas, principalmente para construção de APIs RESTful. Sua arquitetura Single Thread que realiza I/O não bloqueante rodando em cima do JavaScript – que é uma linguagem muito presente em praticamente todos os browsers atuais – demonstrou uma boa eficiência no processamento de muitas aplicações atuais (PEREIRA, 2016, p.6).

⁶ <https://pt-br.legacy.reactjs.org/>

⁷ <https://nextjs.org/>

⁸ Definem a estrutura de um projeto e fornecem ferramentas essenciais que servem como blocos de construção, facilitando o desenvolvimento (LENCINA, 2023)

⁹ <https://nodejs.org/pt>

Neste projeto, para aproveitar o ecossistema Node.js, foi utilizado o Fastify¹⁰. Ele é um *framework* backend para JavaScript que promete uma maior velocidade nas requisições de aplicações, utilizando uma arquitetura baseada em plugins. Essa abordagem reduz a sobrecarga de ferramentas desnecessárias, oferecendo um melhor desempenho.

Para o banco de dados, foi utilizado PostgreSQL. Ele é um sistema de gerenciamento de banco de dados relacional de objeto de código aberto, que é reconhecido por sua confiabilidade e capacidade de lidar com altas cargas de trabalho. Com mais de 35 anos de mercado, foi o pioneiro ao introduzir conceitos que se tornaram disponíveis em outros sistemas comerciais muito mais tarde (POSTGRESQL, 2024). Segundo Carvalho (2017, p.6), o PostgreSQL, como um banco de dados de nível corporativo, oferece recursos sofisticados como:

- O controle de concorrência multiversionado (MVCC, em inglês);
- Recuperação em um ponto no tempo (PITR, em inglês), tablespaces;
- Replicação assíncrona;
- Transações agrupadas (savepoints);
- Cópias de segurança quente (online/hot backup);
- Um sofisticado planejador de consultas (otimizador) e registrador de transações sequencial_ (WAL) para tolerância a falhas,
- Suporta conjuntos de caracteres internacionais;
- Codificação de caracteres multibyte, Unicode e sua ordenação por localização;
- Sensibilidade a caixa (maiúsculas e minúsculas) e formatação;
- É altamente escalável, tanto na quantidade enorme de dados que pode gerenciar quanto no número de usuários concorrentes que pode acomodar. Existem sistemas ativos com o PostgreSQL em ambiente de produção que gerenciam mais de 4TB de dados. (CARVALHO, 2017, p.6)

2.7. VERSIONAMENTO DE CÓDIGO

No dia a dia do desenvolvimento de software, as mudanças são constantes: novas funcionalidades são adicionadas, funcionalidades existentes são modificadas ou removidas. Além disso, é raro que um sistema seja construído por uma única pessoa, o que torna essencial detectar e mesclar alterações no mesmo código feitas por diferentes colaboradores. Essa necessidade levou ao surgimento dos sistemas de versionamento, que agem como uma máquina do tempo e robôs de integração, mantendo um histórico das diferentes versões do código (AQUILES; FERREIRA, 2014, p.2).

Um sistema de controle de versão amplamente conhecido é o Git¹¹. Sendo gratuito, de código aberto e com uma curva de aprendizado baixa, ele foi projetado para lidar com projetos

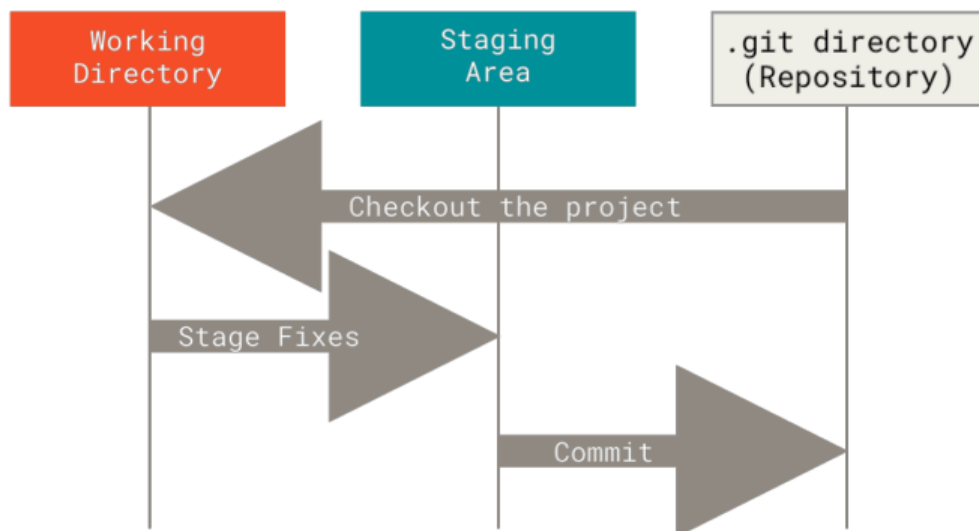
¹⁰ <https://fastify.dev/>

¹¹ <https://git-scm.com/>

de pequena a grande escala de maneira rápida e eficiente (Git, 2024). Para ele os arquivos passam por três estados: modificado (*modified*), preparado (*staged*), e *committed*. O primeiro estado, modificado, é quando o arquivo foi alterado, mas ainda não foi realizado o *commit* no seu banco de dados. Preparado é quando o desenvolvedor marcou o arquivo para fazer parte do próximo *commit*. O último estado, *committed*, é quando o arquivo fez parte de algum *commit*, ou seja, ele foi armazenado corretamente no banco de dados local. (CHACON; STRAUB, 2014, p.16).

Na Figura 4 é apresentado as três seções principais de um projeto Git. A área de trabalho (*working directory*) é uma cópia de uma versão do projeto armazenada no computador local do desenvolvedor, possibilitando alterações. A área de preparo (*staging area*) é um arquivo, localizado geralmente no diretório Git, que armazena o que será incluído no próximo *commit*. O diretório Git (*git directory*) é a seção mais importante, pois armazena todos os metadados e o banco de dados completo do projeto (CHACON; STRAUB, 2014, p.17).

Figura 4: Seções principais de um projeto Git



Fonte: Chacon e Straub, (2014, p.17)

3. ESTADO DA ARTE

Durante o início do projeto, foram realizadas pesquisas sobre as aplicações de PMS (*Property Management System*) já existentes no mercado que compartilham objetivos semelhantes ao PumaJourney. Essas pesquisas foram essenciais durante a análise de requisitos do projeto, possibilitando a identificação de funcionalidades e características diferenciais para nossa aplicação. Neste capítulo serão apresentadas algumas das aplicações comerciais encontradas.

3.1. CLOUDBEDS

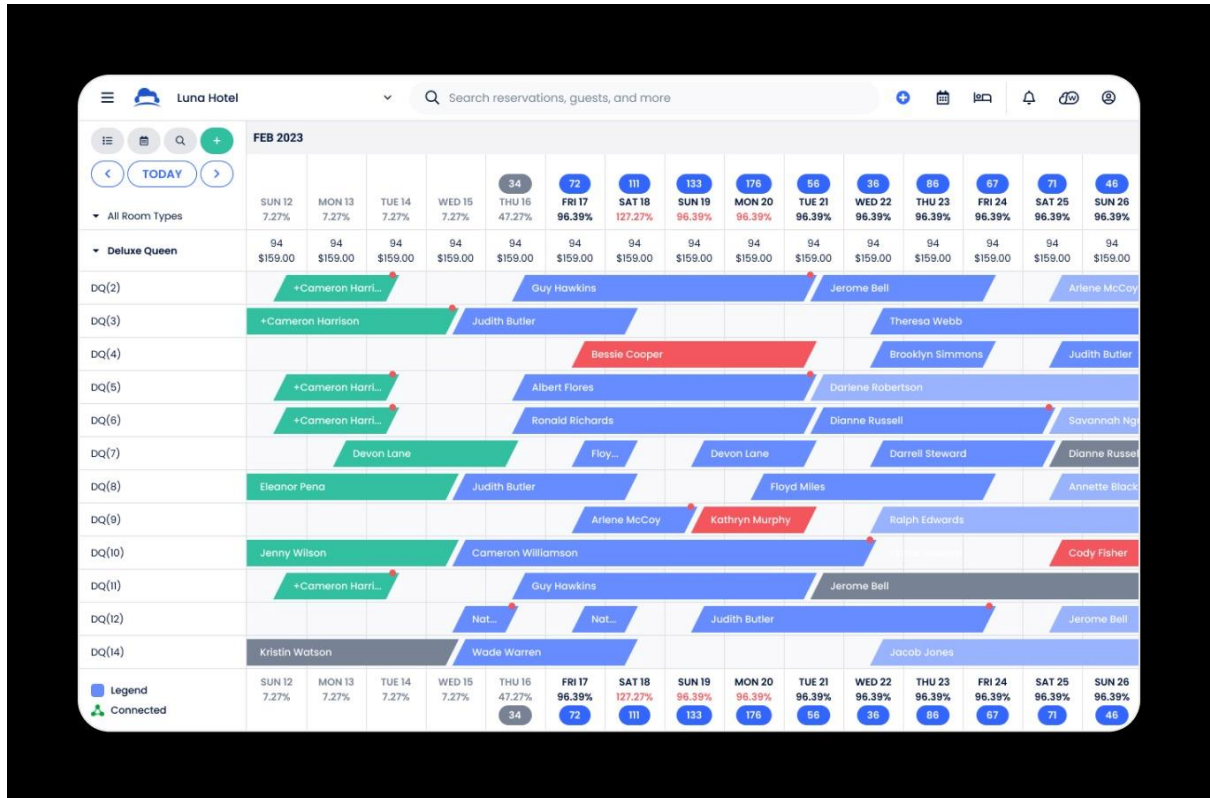
Desde seu lançamento em 2012, o Cloudbeds¹² tem sido uma aplicação que revolucionou o setor de gestão de propriedades. Projetada para simplificar as operações rotineiras e diárias de hotéis, independentemente do tamanho, desde o *check-in* dos hóspedes até análises financeiras detalhadas, a plataforma oferece uma ampla lista de soluções integradas em apenas uma interface. Essa abordagem não apenas aumenta a eficiência operacional, como também eleva a experiência dos hóspedes, proporcionando um atendimento mais ágil.

Além disso, a aplicação se destaca pela sua capacidade de integração com uma variedade de serviços de terceiros, como sistemas de pagamento, ferramentas de marketing e canais de distribuição. Isso permite que os usuários personalizem e ampliem suas funcionalidades, criando um ambiente altamente flexível.

A Figura 5 apresenta uma das telas principais do Cloudbeds na versão web, a tela de calendário das reservas, onde é possível visualizar todas as reservas de acordo com os quartos disponíveis.

¹² <https://www.cloudbeds.com/pt-br/>

Figura 5: Tela de calendário do Cloudbeds



Fonte: Cloudbeds (2022).

3.2. SIRVOY

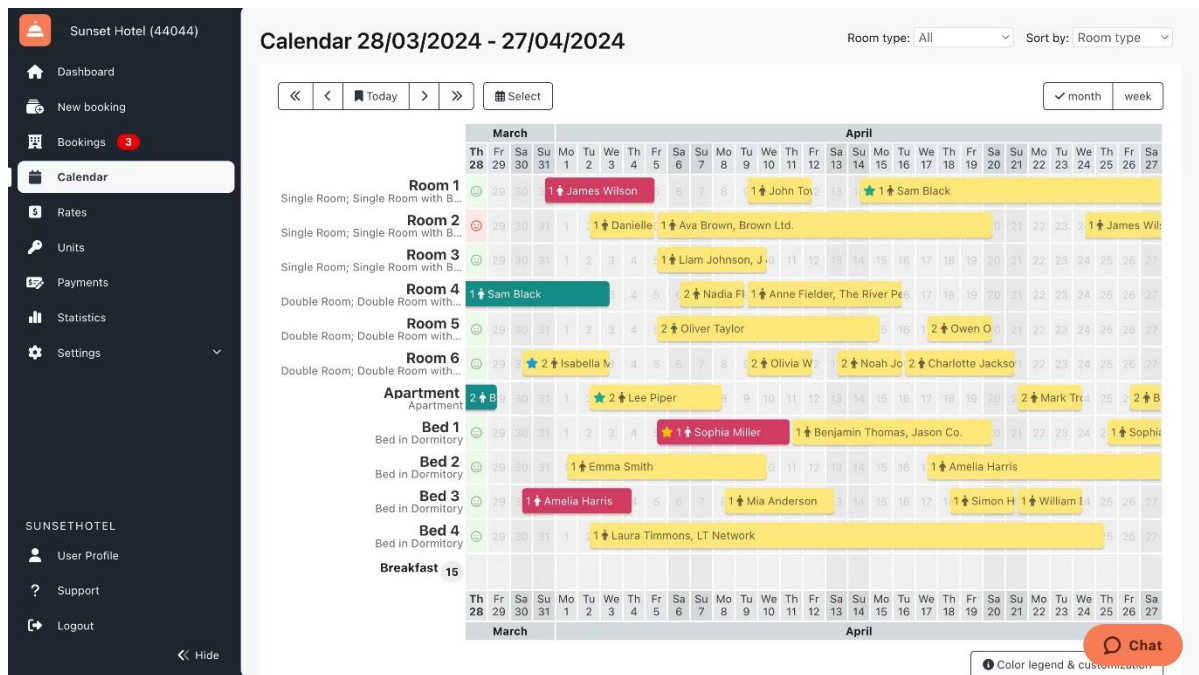
Sirvoy¹³ é um sistema fácil de usar e muito intuitivo, voltado para hotéis e propriedades de pequeno a médio porte, oferecendo funcionalidades básicas de gestão de reservas. Ele se destaca pelos seus diferentes planos de preços, que o tornam mais acessível e adaptável às diversas necessidades dos estabelecimentos. A flexibilidade dos planos permite que proprietários escolham a solução mais adequada ao seu orçamento e às suas necessidades.

Entre as principais funcionalidades, destaca-se a gestão de propriedades, que inclui calendário de reservas, histórico de hóspedes e pagamentos. O motor de reservas permite a realização de reservas online de forma fácil para os hóspedes, e o sistema oferece integrações com diversos serviços de terceiros, ampliando suas capacidades e centralizando a gestão das operações hoteleiras. A aplicação também permite que, no momento da reserva, seja possível selecionar consumos adicionais que o hóspede terá, como café da manhã, aluguel de bicicleta ou outras atividades. No entanto, essa funcionalidade não é o foco principal da aplicação, sendo não muito aprofundada.

¹³ <https://sirvoy.com/>

A Figura 6 apresenta a tela de calendário de reservas do Sirvoy, em sua versão web.

Figura 6: Calendário de reservas do Sirvoy



Fonte: Sirvoy (2023).

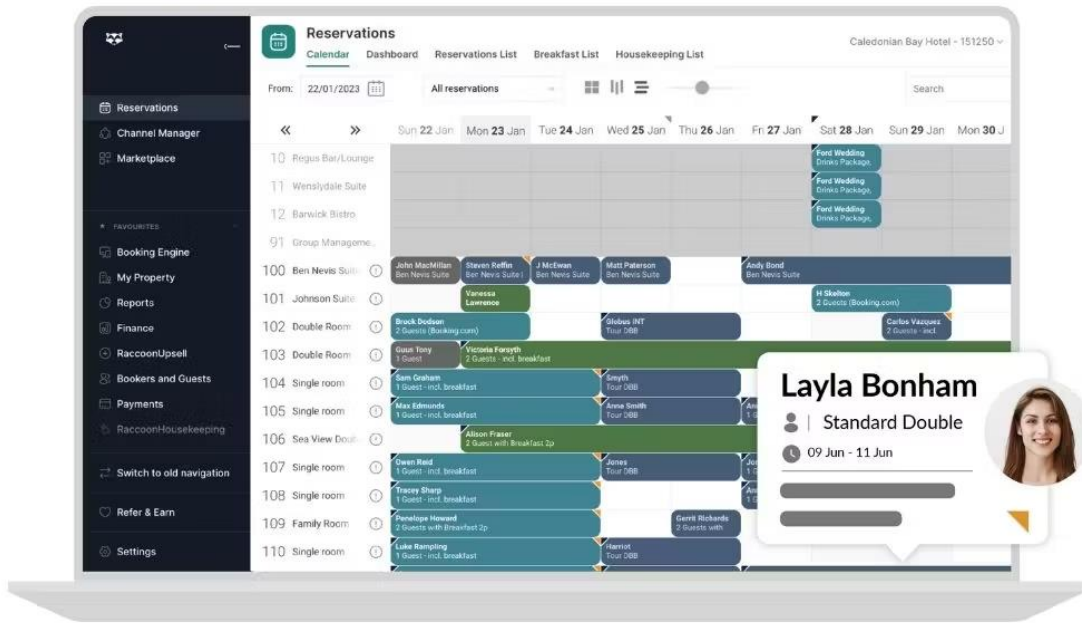
3.3. ROOMRACCOON

Esta aplicação se diferencia das mencionadas anteriormente principalmente devido ao seu foco em automações e integrações. Ela inclui diversas automações que agilizam os processos de *check-in*, *check-out* e comunicação com os hóspedes. Oferece a possibilidade de integrar-se com uma ampla variedade de serviços de terceiros, como sistemas de pagamento e marketing, entre outros. Além disso, o RoomRaccoon¹⁴ oferece funcionalidades avançadas para gerenciamento e ajuste personalizado de tarifas, tornando-se uma boa escolha para hotéis que necessitam de flexibilidade nos cálculos tarifários.

A Figura 7 apresenta a tela principal da aplicação RoomRaccoon no formato web. Podemos ver o calendário de reservas e outras opções disponíveis.

¹⁴ <https://roomraccoon.com/>

Figura 7: Tela principal do RoomRaccon



Fonte: RoomRaccoon (2023)

3.4. COMPARATIVO

Os sistemas mencionados compartilham diversas funcionalidades semelhantes, cada um com seu foco principal definido. Por exemplo, o Cloudbeds é reconhecido por sua abrangência e capacidade de atender às necessidades complexas de propriedades de grande porte. Por outro lado, o Sirvoy tem funcionalidades básicas que visam atender propriedades menores. No entanto, todos eles não oferecem uma versão gratuita.

Abaixo, na Tabela 1, é apresentado um comparativo de todos os sistemas mencionados, incluindo a aplicação desenvolvida neste projeto, PumaJourney.

Tabela 1: Comparativo dos sistemas.

Funcionalidade	Cloudbeds	Sirvoy	RoomRaccoon	PumaJourney
Gestão de reservas	Sim	Sim	Sim	Sim
Gestão de consumo	Sim	Parcialmente	Sim	Sim

Histórico de reservas	Sim	Sim	Sim	Entregas futuras
Relatório mensal/anual	Sim	Sim	Sim	Entregas futuras
Responsivo mobile	Sim	Sim	Sim	Sim
Gratuito	Não	Não	Não	Sim

Fonte: Autoria própria

4. METODOLOGIA

Este projeto tem uma abordagem de pesquisa qualitativa, ou seja, busca entender exatamente como a pousada funciona, quais os aspectos a melhorar e como aplicar mudanças. Quanto à natureza, classifica-se como aplicada, pois utiliza conhecimentos básicos para compor uma solução. Por fim, quanto ao objetivo, classifica-se como exploratória, investigando os problemas e possíveis soluções (GERHARDT e SILVEIRA, 2009).

Foram utilizadas diversas técnicas de pesquisa para construção deste projeto, tal como entrevistas, questionários e observações. Antes de tudo, foi realizado um estudo sobre quais as questões que seriam abordadas nessas pesquisas e entrevistas. Em relação a amostragem, ou seja, os participantes dessas pesquisas, foi composta pelos proprietários das cabanas, em razão de serem os usuários do sistema desenvolvido.

Durante a parte inicial deste projeto foi feita a análise de requisitos do sistema. Essa é uma etapa extremamente importante para a organização e levou em consideração os objetivos principais e secundários da aplicação e, também, o tempo estipulado para a finalização do projeto. Para isso, foram realizadas várias entrevistas e reuniões com os proprietários da Pousada, resultando em uma lista de funcionalidades cruciais e desejadas. A partir dessa lista, foi possível definir o MVP (Produto Mínimo Viável) do sistema. O MVP é uma versão enxuta de uma solução que inclui apenas as funcionalidades essenciais para atender às necessidades básicas dos usuários, permitindo que um pequeno grupo teste e opine sobre a produto (FIA, 2022). Com base no resultado dessa análise foram implementados diagramas de caso de uso, que têm como objetivo descrever a aplicação pela visão do usuário.

Para este projeto, foi utilizado a metodologia ágil Scrum. Esta, como explicado anteriormente no capítulo “Referencial teórico”, é utilizada para a organização e gerência do trabalho como um todo, utilizando *sprints* para a separação do trabalho em determinado período (SUTHERLAND, JEFF, 2016). As *sprints* foram separadas em um intervalo de quinze dias e organizadas utilizando ferramentas como Notion e Jira.

Notion¹⁵ é um aplicativo estilo *workspace*, que permite que o usuário customize do seu próprio jeito, adicionando desde páginas com várias camadas até planilhas e calendários (KOVACS, 2023). Neste projeto, o Notion foi utilizado para armazenar e organizar principalmente a documentação das funcionalidades, diagramas implementados e outras informações importantes.

¹⁵ <https://www.notion.so/pt-br>

O Jira¹⁶ é uma aplicação de gerenciamento de projetos que permite planejar e monitorar tarefas de forma eficiente. Nele, todas as funcionalidades da aplicação foram listadas e organizadas em *sprints* de duas semanas. Em cada nova *sprint*, foi realizado um planejamento detalhado, onde essas funcionalidades foram divididas em tarefas específicas.

Passada a etapa de organização e modelagem do sistema, foi iniciado o desenvolvimento das telas e funcionalidades da aplicação. A linguagem de programação utilizada tanto para o frontend, quanto para o backend, foi o Typescript, em razão de que a tipificação de valores que a linguagem oferece garante redução de erros, se comparado com Javascript (Typescript. O Javascript Moderno Para Criação de Aplicações, 2017). No frontend também foi utilizado a biblioteca React.js, para a criação de interfaces, e *Context API*¹⁷, utilizado na persistência de dados. Para o backend foi utilizado o Fastify Framework e, para o banco de dados da aplicação, foi utilizado o PostgreSQL.

Para garantir o funcionamento esperado da aplicação, foram realizados diversos testes ao longo do desenvolvimento. Incluíram-se testes unitários para verificar o funcionamento isolado de cada componente, testes *end-to-end* (E2E) para validar o fluxo completo da aplicação e testes funcionais para assegurar que todas as funcionalidades atendem os requisitos.

¹⁶ <https://www.atlassian.com/br/software/jira>

¹⁷ <https://www.loginradius.com/blog/engineering/react-context-api/>

5. APLICAÇÃO PUMAJOURNEY

Nesta seção serão apresentados detalhes do desenvolvimento da aplicação, desde levantamento de requisitos até os testes.

5.1. LEVANTAMENTO DE REQUISITOS

Para o levantamento de requisitos do sistema desenvolvido, foram conduzidas várias entrevistas com os proprietários da pousada. Isso proporcionou uma compreensão mais profunda das funcionalidades essenciais que otimizariam as operações diárias e economizariam tempo significativo para os proprietários.

Ficou evidente a necessidade de priorizar funcionalidades como a gestão de reservas e o controle de consumo dos hóspedes. Um sistema com um calendário individual para cada cabana, que permita adicionar, modificar, visualizar e cancelar reservas, além de registrar os consumos dos hóspedes e realizar todos os cálculos do total da reserva, seria crucial para agilizar o processo da administração.

Durante esse processo, para auxiliar na organização e priorização dessas funcionalidades, foi utilizado o sistema Jira. Essa ferramenta se mostrou fundamental para gerenciar de forma eficiente as demandas, garantindo uma abordagem estruturada e transparente no desenvolvimento do sistema. Além disso, para o processo de gestão de projeto foi utilizado a metodologia Scrum, proporcionando uma estrutura clara e facilitando o acompanhamento do progresso. Na Figura 8, podemos observar a visão de listagem de itens no Jira, que apresenta os status de cada item e as *sprints* às quais eles estão atribuídos.

Figura 8: Lista de itens no sistema Jira

Projeto de software

PLANEJAMENTO

- Linha do tempo
- Backlog
- Painel
- Listar**
- Objetivos
- Itens

DESENVOLVIMENTO

- Código
- Páginas de projeto
- Adicionar atalho
- Configurações do proj...

Você está em um projeto gerenciado por uma equipe

Saiba mais

Projeto de software / Gestão de hospedagens

Listar

Pesquisar lista

	Tipo	# Chave	Resumo	Status	Sprint	Responsável
<input type="checkbox"/>	▼	GDH-7	Login	CONCLUÍDO		Gabi
<input type="checkbox"/>	>	GDH-20	Página de login	CONCLUÍDO	Sprint 2	Gabi
<input type="checkbox"/>	>	GDH-8	Avaliar tecnologias a serem utilizadas para o frontend e inici...	CONCLUÍDO	Sprint 2	Gabi
<input type="checkbox"/>	▼	GDH-4	Gestão de Reservas	CONCLUÍDO		Gabi
<input type="checkbox"/>	>	GDH-23	Calendário com as reservas	CONCLUÍDO	Sprint 3	Gabi
<input type="checkbox"/>	>	GDH-28	Criação de reserva	CONCLUÍDO	Sprint 5	Gabi
<input type="checkbox"/>	>	GDH-46	[Criação de Reserva] Adicionar modal ao excluir um item de ...	CONCLUÍDO	Sprint 5	Gabi
<input type="checkbox"/>	>	GDH-30	Visualização de reserva	CONCLUÍDO	Sprint 6	Gabi
<input type="checkbox"/>	>	GDH-31	Cancelamento de reserva	CONCLUÍDO	Sprint 6	Gabi
<input type="checkbox"/>	>	GDH-29	Edição de reserva	CONCLUÍDO	Sprint 7	Gabi
<input type="checkbox"/>	>	GDH-80	Checkout de reserva	CONCLUÍDO	Sprint 9	Gabi
<input type="checkbox"/>	>	GDH-47	[Front] Refatorar CRUD de reserva para utilizar Context API	CONCLUÍDO	Sprint 5	Gabi
<input type="checkbox"/>	>	GDH-70	Opção para alterar consumo (modal ou página? Melhor pági...	CONCLUÍDO		Gabi

+ Criar

Fonte: Autoria própria

Cada item pode conter vários subitens, o que ajuda significativamente na organização durante o desenvolvimento. Além disso, é possível vincular diversos links a cada item, sendo que neste projeto foi utilizado essa funcionalidade para conectar a documentação armazenada em uma página do Notion. Na Figura 9, podemos visualizar um exemplo de um item no Jira.

Figura 9: Exemplo de item no sistema Jira

GDH-4 / GDH-28

Criação de reserva

Anexar Adicionar item filho Vincular item

Descrição
Editar descrição

Itens filho Ordenar por 100% concluído

GDH-36	Diagramação front	CONCLUÍDO
GDH-39	Criar endpoint de obter hóspedes	CONCLUÍDO
GDH-37	Criar endpoint de calcular 30 %	CONCLUÍDO
GDH-38	Criar endpoint de criar reserva	CONCLUÍDO
GDH-40	Consumo de endpoints no front	CONCLUÍDO
GDH-45	Criar endpoint para obter consumos disponíveis	CONCLUÍDO
GDH-50	Testes	CONCLUÍDO

Links da web
documentação

Adicionar comentário...

Dica de ouro: aperte **M** para fazer comentários

Fonte: Autoria própria

5.1.1. USER STORIES

Uma *User Story*, ou história de usuário, é uma explicação e descrição de uma funcionalidade escrita da perspectiva do usuário final, com o objetivo de mostrar como esse recurso de software agregará valor ao cliente (REHKOPF, 2024).

Na Figura 10 é apresentado uma das várias *user stories* criadas para o sistema PumaJourney. Nela, está detalhada a história de usuário da funcionalidade de *checkout* de reserva, acompanhada dos seus critérios de aceitação.

Figura 10: Exemplo de User Story

Checkout de reserva

Como usuário com acesso ao sistema,
quero poder acessar o calendário
para realizar o checkout de uma reserva

Critérios de Aceitação

Dado que sou um usuário do sistema,
quando entrar no calendário e houver uma reserva em andamento ou finalizada
então poderei realizar o checkout

Cenário: Reserva Não iniciada e Checkout realizado

Dado que estou no calendário
quando clico em uma reserva com o status Não iniciada ou Checkout realizado
então a opção checkout estará habilitada e ao selecionar essa opção uma modal com informações
do checkout abrirá com o botão "Confirmar" desabilitado

Cenário: Reserva Em andamento ou Finalizada

Dado que estou no calendário
quando clico em uma reserva com o status Em andamento ou Finalizada
então a opção checkout estará habilitada e ao selecionar essa opção uma modal com informações
do checkout abrirá com o botão "Confirmar" habilitado

Regras de negócio

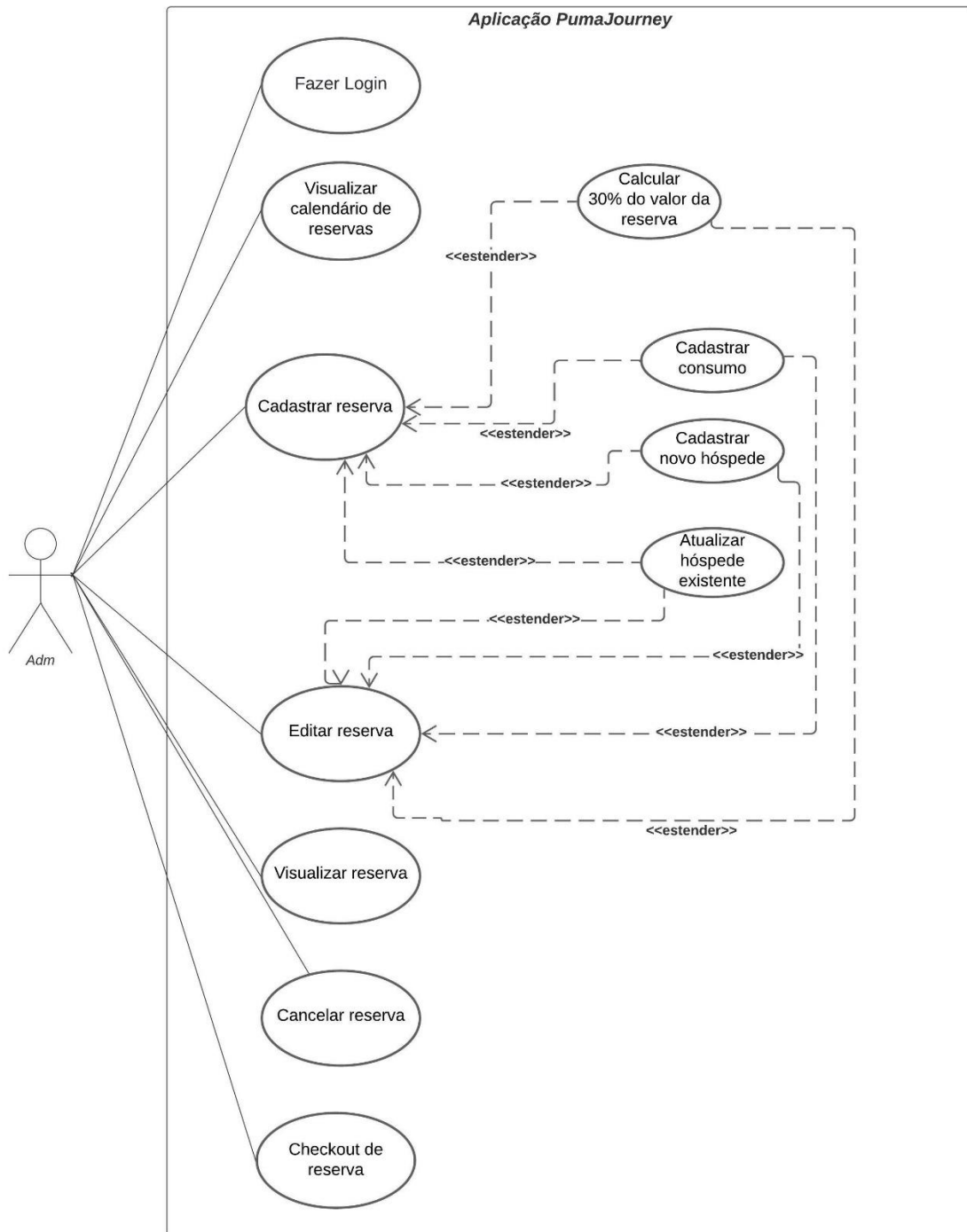
- O título da modal será "Checkout"
- Será mostrado "Valor total da hospedagem" com o valor total.
- Será mostrado "Valor de entrada" com o valor de entrada.
- Será mostrado uma seção "Itens consumidos" que será um accordion e aparecerá fechado no início. Ao clicar no accordion aparecerá uma lista de todos os itens consumidos com seus preços.
- Abaixo do accordion será mostrado o valor total do consumo.
- Será mostrado a informação "Valor a pagar"
- Será mostrado um botão "Confirmar", caso seja possível realizar o checkout (Reserva com status diferente de Não iniciada e Cancelada).
- Ao clicar em confirmar, deve ser gravado na reserva a data de checkout.

Fonte: Autoria própria

5.1.2. DIAGRAMA DE CASO DE USO

Para criar uma visão organizada do projeto PumaJourney, foi elaborado um diagrama de caso de uso que engloba todas as funcionalidades prioritárias do sistema, como pode ser visto na Figura 11.

Figura 11: Diagrama de casos de uso do sistema PumaJourney



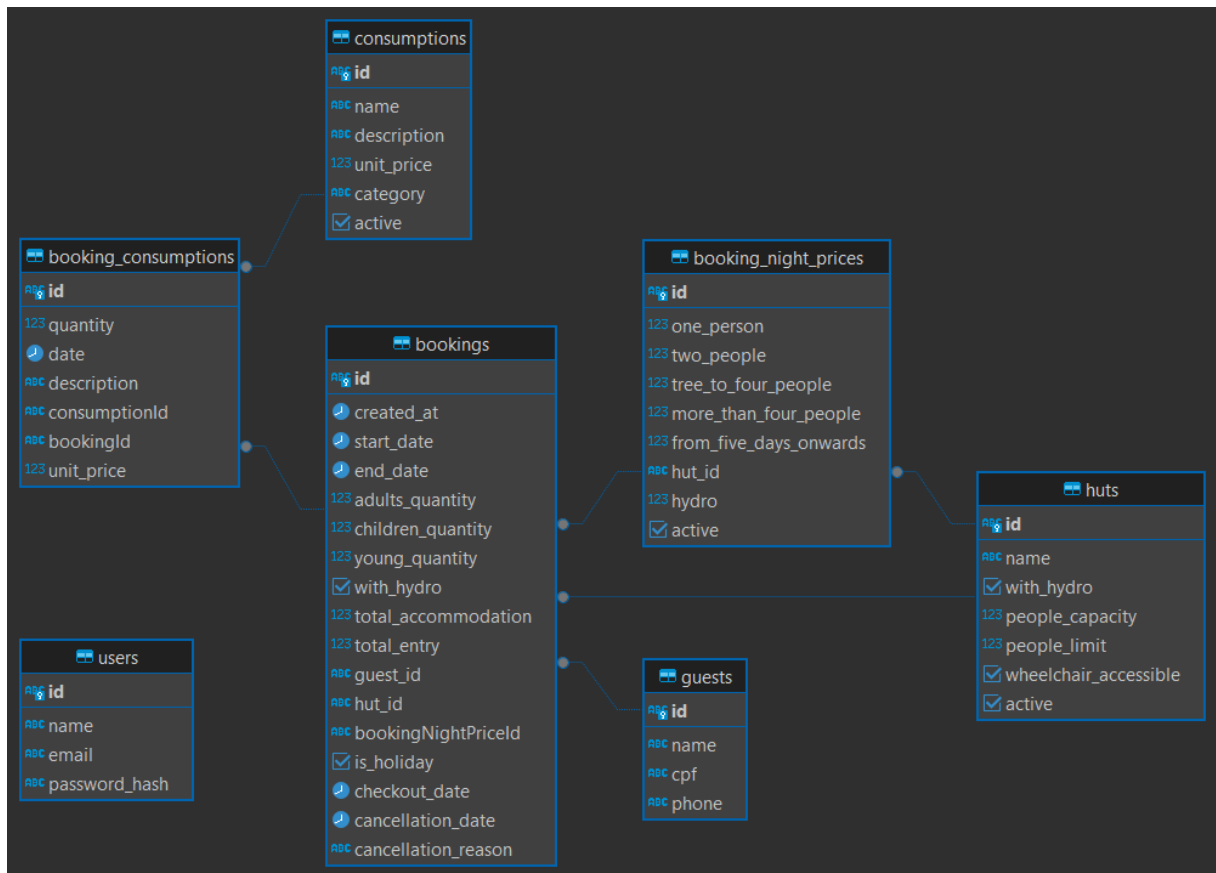
Atualmente, há apenas um ator principal identificado, que é o administrador da pousada. Ele pode fazer login, visualizar o calendário de reservas, cadastrar novas reservas, editar reservas existentes, visualizar detalhes das reservas, cancelar reservas e realizar o checkout das reservas. As funcionalidades de cadastro e edição de reservas incluem também o cálculo do valor de entrada padrão, que corresponde a 30% do total da reserva, o gerenciamento de consumos associados à reserva e o cadastro e edição dos hóspedes.

5.2. MODELAGEM DO BANCO DE DADOS

Para realizar uma modelagem eficiente do banco de dados, foi criado um diagrama para representar a estrutura. Esse diagrama é o entidade-relacionamento (ER), que funciona como um fluxograma que tem como objetivo ilustrar como as entidades estão relacionadas umas com as outras dentro do sistema. Na Figura 12, está apresentado o diagrama entidade-relacionamento que mapeia todas as entidades e seus relacionamentos neste projeto.

A tabela principal é a “bookings”, que registram as informações de todas as reservas do sistema. Cada reserva está associada a um hóspede na tabela “guests” e a uma cabana na tabela “huts”. Os preços das cabanas são gerenciados pela tabela "booking_night_prices", que são vinculados a cada cabana e a reserva. Essa abordagem foi adotada para manter um histórico consistente no valor da reserva, visto que os preços podem ser alterados no futuro. Além disso, as reservas podem ser associadas aos consumos na tabela "booking_consumptions", os quais estão vinculados aos diferentes tipos de consumos na tabela "consumptions".

Figura 12: Diagrama entidade-relacionamento do sistema PumaJourney



Fonte: Autoria própria

5.3. DESENVOLVIMENTO DA APLICAÇÃO

Essa seção aborda a implementação da aplicação PumaJourney. Inicialmente, a seção 5.3.1 apresenta detalhes da hospedagem da aplicação e nas seções que seguem são apresentadas todas as funcionalidades e os testes realizados na aplicação.

5.3.1. CONEXÃO E HOSPEDAGEM

Para a hospedagem do servidor e do cliente da aplicação, foi escolhida a plataforma Vercel, que proporciona uma solução eficiente para o *deploy* e gerenciamento de aplicações web. A Vercel¹⁸ não só disponibiliza um plano gratuito, mas também suporta *deploy* automático a partir do GitHub, o que facilita a integração contínua e o gerenciamento das versões. Além disso, a plataforma permite a configuração de variáveis de ambiente, que foram necessárias principalmente para a conexão com o banco de dados.

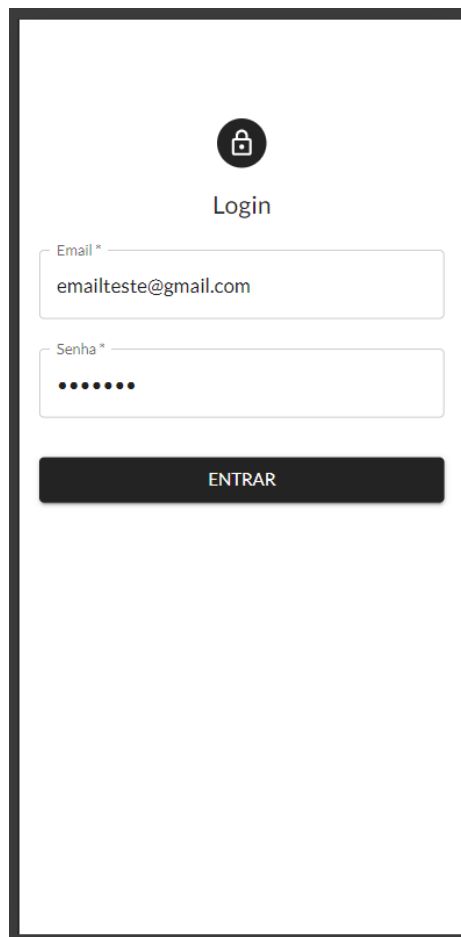
¹⁸ <https://vercel.com/>

Para o armazenamento do banco de dados, foi utilizado o serviço AWS RDS (*Relational Database Service*). Este serviço oferece uma solução escalável que garante alta disponibilidade. Além disso, o AWS RDS oferece um plano gratuito no primeiro ano, facilitando a implementação inicial sem custos adicionais.

5.3.2. TELA DE LOGIN

Na Figura 13, está ilustrada a tela de login da aplicação. É possível verificar que essa tela é bem simples, contendo apenas o título “Login”, os campos de entrada para e-mail e senha, e o botão principal para entrar na aplicação. A tela foi desenvolvida de maneira minimalista e não há nenhuma opção para se registrar no sistema, visto que o objetivo do sistema é ser utilizado internamente pela pousada.

Figura 13: Tela de login



A imagem mostra a interface de login de uma aplicação. No topo, há um ícone de cadeado dentro de um círculo. Abaixo dele, o texto "Login" é exibido. Em seguida, há dois campos de entrada: o primeiro é rotulado "Email *" e contém o texto "emailteste@gmail.com"; o segundo é rotulado "Senha *" e contém sete pontos para ocultar a senha. Abaixo dos campos, há um botão preto com o texto "ENTRAR" em branco.

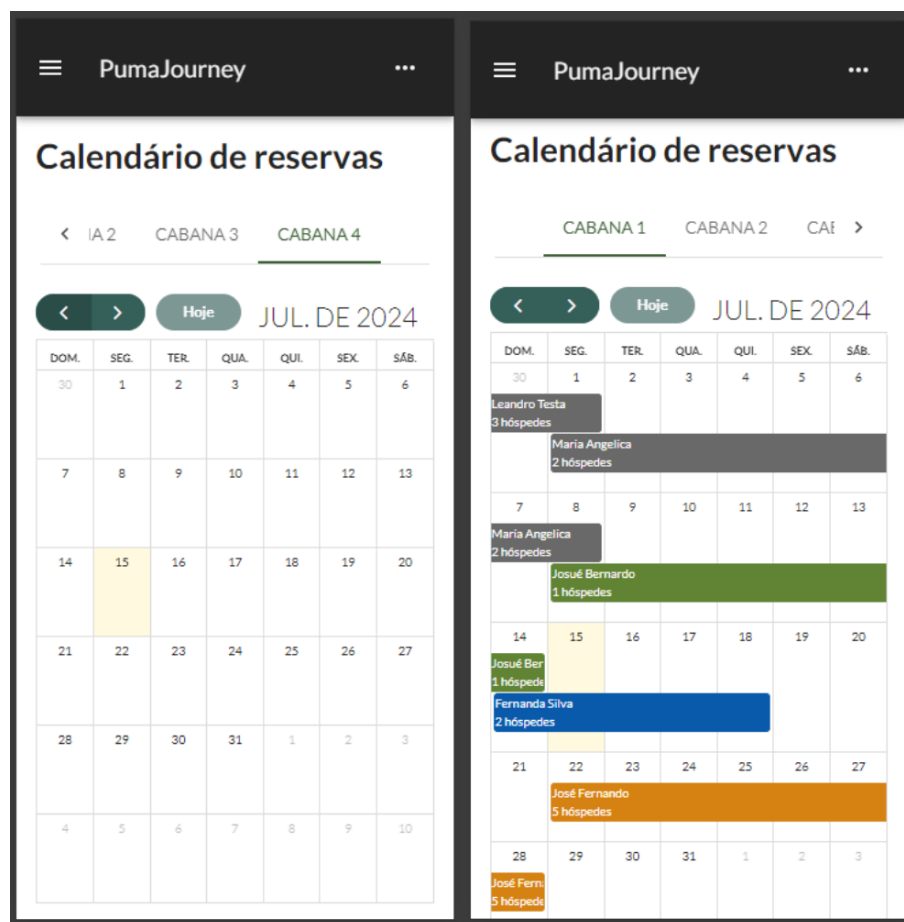
Fonte: Autoria própria

5.3.3. TELA INICIAL

A tela inicial apresenta o calendário das reservas. A primeira seção permite a seleção da cabana cujas reservas serão exibidas, garantindo uma visualização organizada. A lista de cabanas é obtida dinamicamente do backend, possibilitando futuras adições de maneira fácil e eficiente. Em seguida, é apresentado o mês ao qual as reservas correspondem, juntamente com botões que permitem navegar rapidamente entre os meses.

No calendário, para cada reserva é exibido o nome do hóspede principal e o número total de hóspedes. As reservas são diferenciadas por cores, indicando o status de cada uma. Essa separação por cores facilita a identificação rápida e visual do estado das reservas, melhorando a eficiência na gestão e monitoramento. As cores possuem os seguintes significados: laranja para reservas não iniciadas, azul para reservas em andamento, verde para reservas finalizadas sem checkout realizado, e cinza para reservas com checkout concluído.

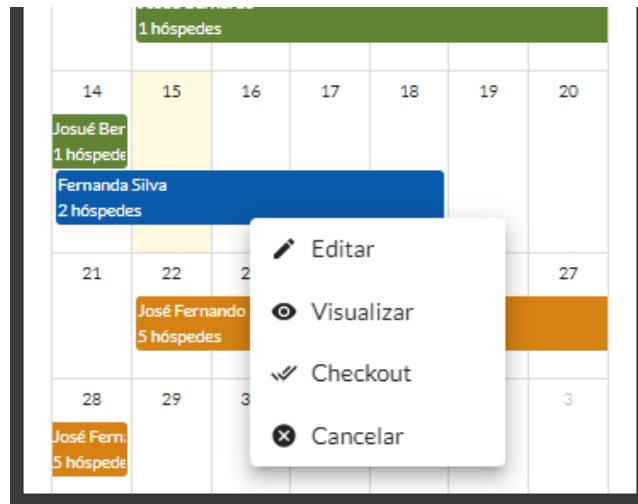
Figura 14: Tela inicial



Fonte: Autoria própria

Ao clicar em uma reserva, uma caixa é exibida, como apresentada na Figura 15, com várias opções de ações que podem ser realizadas para essa reserva: editar, visualizar, realizar o checkout e cancelar. Os detalhes de cada uma dessas opções serão abordados nos capítulos seguintes.

Figura 15: Opções disponíveis para uma reserva no calendário



Fonte: Autoria própria

5.3.4. CADASTRO DE RESERVA DE HOSPEDAGEM

Quando o usuário seleciona algum dia do calendário, ele é redirecionado para a tela de cadastro de reserva. Essa tela é estruturada em três passos principais: Hóspede, Reserva e Consumo.

Na primeira etapa, o usuário seleciona qual hóspede será vinculado à reserva. Para isso, ele pode escolher um hóspede já cadastrado previamente ou criar um hóspede preenchendo obrigatoriamente o nome e o telefone, e opcionalmente, o CPF.

Na Figura 16, é demonstrado como a escolha de um hóspede existente funciona. Inicialmente, o usuário realiza uma pesquisa pelo nome, número ou CPF do hóspede no campo "Hóspede existente". Se houver correspondências, os resultados da pesquisa são exibidos. Ao selecionar uma opção, os campos de nome, telefone e CPF são automaticamente preenchidos. Além disso, a ferramenta possibilita a edição desses hóspedes, permitindo que o usuário selecione e faça alterações nos campos conforme necessário.

Figura 16: Escolha do hóspede existente

The figure displays three sequential screenshots of the 'Nova Reserva' (New Reservation) screen in the PumaJourney application, illustrating the process of selecting an existing guest.

- Left Screenshot:** Shows the 'Hóspede existente (opcional)' dropdown menu. The progress indicator shows '1 Hóspede' selected, '2 Reserva' in progress, and '3 Consumo' as the next step. Below the dropdown are input fields for 'Nome do Hóspede *', 'Telefone do Hóspede *', and 'CPF do Hóspede'. A 'PRÓXIMO' button is at the bottom.
- Middle Screenshot:** Shows the dropdown menu with the text 'an|' entered. A list of existing guests is displayed:
 - Maria Angelica 638.497.890-39 (61) 2 3123-1231
 - Leandro Testa - (51) 9 7128-3724
 - Fernanda Silva - (51) 9 9278-2678
 - José Fernando - (51) 9 8293-4823
 A 'PRÓXIMO' button is at the bottom.
- Right Screenshot:** Shows the dropdown menu with 'Josué Bernardo 482.420.270-13 (51) 9 2394-2934' selected. The input fields are now populated with the guest's information:
 - Nome do Hóspede *: Josué Bernardo
 - Telefone do Hóspede *: (51) 9 2394-2934
 - CPF do Hóspede: 482.420.270-13
 A 'PRÓXIMO' button is at the bottom.

Fonte: Autoria própria

Para adicionar um novo hóspede, o usuário deve preencher obrigatoriamente o nome e o telefone. O preenchimento do CPF é opcional, uma vez que, em alguns casos, essa informação pode não estar disponível. Na Figura 17, é exibida a tela durante o preenchimento das informações de um novo hóspede.

Figura 17: Criação de novo hóspede

The image displays two side-by-side screenshots of a mobile application interface for creating a new guest. Both screens show a form titled 'Nova Reserva' with a 'Voltar' button. The form has three steps: 1. Hóspede, 2. Reserva, and 3. Consumo. The left screenshot shows the form with valid data: 'Nome do Hóspede' filled with 'Nome Hóspede' and 'Telefone do Hóspede' filled with '(51) 9 9999-9999'. The right screenshot shows the same form but with a red border around the phone number field and a red error message 'Campo inválido' below it. A red notification banner at the top right of the right screenshot says 'Dados inválidos'.

Fonte: Autoria própria

É importante destacar que todos os formulários da aplicação são devidamente validados, tanto no frontend quanto no backend. No frontend, foram utilizadas as bibliotecas React Hook Form¹⁹ e Yup²⁰ para garantir essa validação. Na Figura 18, é apresentada a validação do formulário de hóspedes no frontend.

¹⁹ <https://react-hook-form.com/>

²⁰ <https://www.npmjs.com/package/yup>

Figura 18: Captura de tela da validação do hóspede

```
6  export type GuestForm = {
7    existingGuest?: ItemIdName
8    guestName?: string
9    guestCPF?: string
10   guestPhone?: string
11 }
12
13 yup.setLocale({
14   mixed: {
15     required: MESSAGE_REQUIRED_FIELD,
16   },
17 })
18
19 export const GUEST_FORM_SCHEMA: yup.Schema<GuestForm> = yup.object({
20   existingGuest: yup.object().nullable() as yup.Schema<ItemIdName>,
21   guestName: yup.string().required(),
22   guestCPF: yup
23     .string()
24     .nullable()
25     .test('is-valid-cpf', MESSAGE_INVALID_FIELD, (value) => {
26       return value === null || value === '' || value.length >= 14
27     }),
28   guestPhone: yup.string().min(16, MESSAGE_INVALID_FIELD).required(),
29 })
30
31 export const GUEST_FORM_DEFAULT_VALUES: DeepPartial<GuestForm> = {
32   existingGuest: null,
33   guestName: '',
34   guestCPF: '',
35   guestPhone: '',
36 }
```

Fonte: Autoria própria

Ao clicar no botão “Próximo” o usuário avança para a etapa de reserva, desde que todos os dados do hóspede estejam válidos. Essa etapa, ilustrada na Figura 19, é responsável por obter as informações principais da reserva.

Figura 19: Etapa de reserva sem dados preenchidos

A captura de tela mostra a interface de reserva do aplicativo PumaJourney. No topo, há um menu hambúrguer e o nome 'PumaJourney'. Abaixo, uma barra de progresso indica três etapas: 'Hóspede' (com um checkmark verde), 'Reserva' (destacada com um círculo verde e o número 2) e 'Consumo' (com um círculo cinza e o número 3). O formulário contém os seguintes campos: 'Data entrada *' com o valor '10/07/2024' e 'Data saída *' (vazio); 'Cabana' com um menu suspenso selecionando 'Cabana 1'; 'Adultos:' com um campo numérico contendo '1'; 'Crianças entre 6 e 12 anos' com um campo numérico contendo '0'; 'Crianças até 5 anos' com um campo numérico contendo '0'; duas opções de checkbox desativadas: 'Com hidromassagem' e 'Feriado'; 'Valor de entrada *' com o valor '0,00' e um botão azul 'CALCULAR 30%'; e dois botões de navegação verde: 'ANTERIOR' e 'PRÓXIMO'.

Fonte: Autoria própria

Ao acessar a tela de criação, a data de entrada e a cabana serão preenchidas automaticamente com o dia selecionado pelo usuário no calendário e a cabana que estava sendo visualizada. No entanto, o usuário ainda pode alterar essas informações manualmente conforme desejar.

Os primeiros campos de entrada referem-se às datas de entrada e saída da reserva. Ao selecionar a data de saída, o sistema calcula automaticamente a quantidade de diárias e exibe essa informação na tela. Abaixo desses campos, há opções para selecionar a cabana, a quantidade de adultos, crianças entre 6 e 12 anos e crianças até 5 anos. As informações detalhadas sobre a quantidade de crianças, separadas por faixa etária, são necessárias para o cálculo da reserva. Além disso, há campos para indicar se a reserva inclui hidromassagem e se é em período de feriado. A Figura 20 apresenta esses campos preenchidos.

Figura 20: Etapa da reserva com dados preenchidos

The image displays two side-by-side screenshots of the PumaJourney reservation form. Both screenshots show the 'Reserva' step selected in the progress bar (Hóspede, Reserva, Consumo). The left screenshot shows the form with the following data: Data entrada: 29/07/2024, Data saída: 31/07/2024, Cabana: Cabana 1, Quantidade de diárias: 2, Adultos: 2, Crianças entre 6 e 12 anos: 1, Crianças até 5 anos: 0, Com hidromassagem: , Feriado: . The right screenshot shows the form with the following data: Data entrada: 29/07/2024, Data saída: 03/08/2024, Cabana: Cabana 1, Quantidade de diárias: 5, Adultos: 2, Crianças entre 6 e 12 anos: 0, Crianças até 5 anos: 0, Com hidromassagem: , Feriado: . Both screenshots show a 'CALCULAR PREÇO' button and 'ANTERIOR' and 'PRÓXIMO' navigation buttons.

Fonte: Autoria própria

O último campo dessa tela é o valor de entrada, que os hóspedes devem pagar no momento da reserva. O valor padrão de entrada é 30% do valor total da hospedagem, e um botão foi incluído para fazer uma requisição ao backend e calcular automaticamente esse valor. No entanto, o valor ainda pode ser alterado manualmente. Ao clicar neste botão, o valor total da hospedagem também é exibido na tela, conforme é mostrado na Figura 21.

Figura 21: Etapa de reserva com o valor total apresentado

The screenshot shows the 'PumaJourney' app interface for the reservation stage. At the top, there's a header with a menu icon, the text 'PumaJourney', and a three-dot menu icon. Below the header, a dropdown menu is set to 'Cabana 1'. A green pill-shaped button indicates 'Quantidade de diárias: 5'. Underneath, there are three rows for guest counts: 'Adultos' with a value of 2, 'Crianças entre 6 e 12 anos' with a value of 0, and 'Crianças até 5 anos' with a value of 0. Each row has minus and plus buttons. Two checkboxes are checked: 'Com hidromassagem' and 'Feriado'. A green pill-shaped button displays 'Valor total: R\$ 1.750,00'. Below this, there's a 'Valor de entrada' field with '525,00' and a blue 'CALCULAR 30%' button. At the bottom, there are two green buttons: 'ANTERIOR' and 'PRÓXIMO'.

Fonte: Autoria própria

Ao avançar para a próxima e última etapa, o usuário encontrará um formulário básico para adicionar consumos à reserva. Para isso, é necessário escolher um item, a data em que ocorrerá o consumo e a quantidade. Também é possível adicionar uma descrição para o consumo. Após adicionado, é apresentado uma lista abaixo do formulário, onde o usuário pode alterar as quantidades de cada consumo.

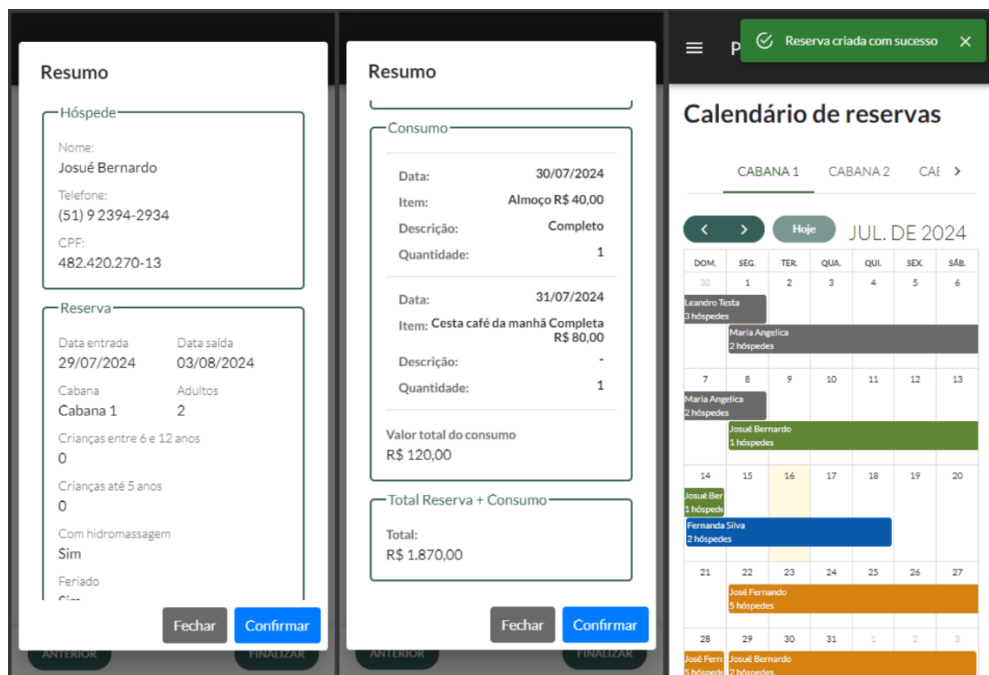
Figura 22: Etapa de consumo

The figure consists of three side-by-side screenshots of the 'PumaJourney' app showing the consumption stage. The first screenshot shows the date range '29/07/2024 - 03/08/2024' and progress indicators for 'Hóspede', 'Reserva', and 'Consumo'. A blue button 'Adicionar novo consumo' is visible. Below it are input fields for 'Item de consumo', 'Data do consumo', and 'Descrição (opcional)', along with a quantity selector set to 1. A message at the bottom states 'Nenhum consumo foi cadastrado para esta reserva.' The second screenshot shows the 'Adicionar novo consumo' form filled out with 'Almoço R\$ 40,00', '30/07/2024', and 'Completo', with a quantity of 2. The third screenshot shows a list of added consumos: 'Almoço R\$ 40,00' on '30/07/2024' with a quantity of 2, and 'Cesta café da manhã Completa R\$ 80,00' on '31/07/2024' with a quantity of 1. Each screenshot has 'ANTERIOR' and 'FINALIZAR' buttons at the bottom.

Fonte: Autoria própria

Ao clicar no botão “Finalizar”, uma modal é exibida com o resumo da reserva para uma última validação. O usuário pode visualizar o hóspede selecionado, os dados básicos da reserva incluindo o valor total da hospedagem, uma lista de todos os consumos adicionados com o valor total dos consumos, e o cálculo total da reserva, que inclui o valor da hospedagem e dos consumos. Após o usuário confirmar o cadastro da reserva e se houver disponibilidade na data selecionada, uma mensagem de confirmação é exibida. Em seguida, o usuário é redirecionado de volta ao calendário de reservas, com a mesma cabana e mês selecionados onde a reserva foi cadastrada. A Figura 23 ilustra esses cenários.

Figura 23: Resumo da reserva criada

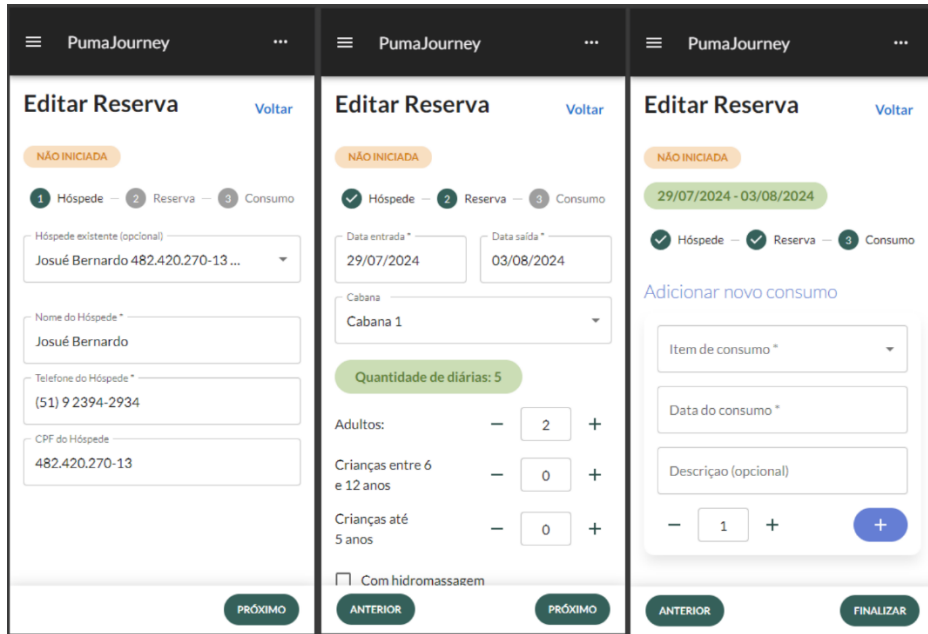


Fonte: Autoria própria

5.3.5. EDIÇÃO DE RESERVA DE HOSPEDAGEM

A edição de uma reserva segue um fluxo muito semelhante ao da criação. Após o usuário selecionar uma reserva e clicar na opção de editar, ele é redirecionado para um formulário onde todos os campos estão preenchidos com as informações da reserva previamente informadas. Como o sistema é utilizado apenas internamente na pousada e apenas administradores têm acesso às alterações, é possível modificar qualquer dado conforme necessário. A alteração visualmente mais perceptível, comparando com a tela de cadastro, é a inclusão da informação de status que indica o estado atual da reserva. A Figura 24 mostra essa tela.

Figura 24: Edição da reserva

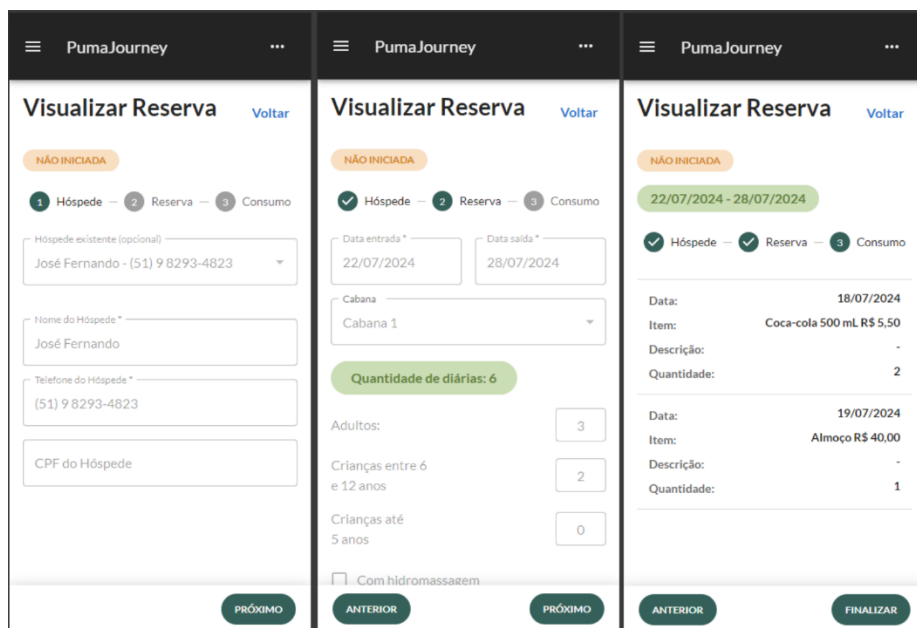


Fonte: Autoria própria

5.3.6. VISUALIZAÇÃO DE RESERVA DE HOSPEDAGEM

Para visualizar uma reserva, é necessário clicar em uma reserva no calendário e selecionar a opção “Visualizar”. Após isso, o usuário será redirecionado para o formulário da reserva, onde todos os campos estarão desabilitados, permitindo apenas a visualização dos detalhes da reserva. A Figura 25 ilustra essa tela.

Figura 25: Visualização da reserva

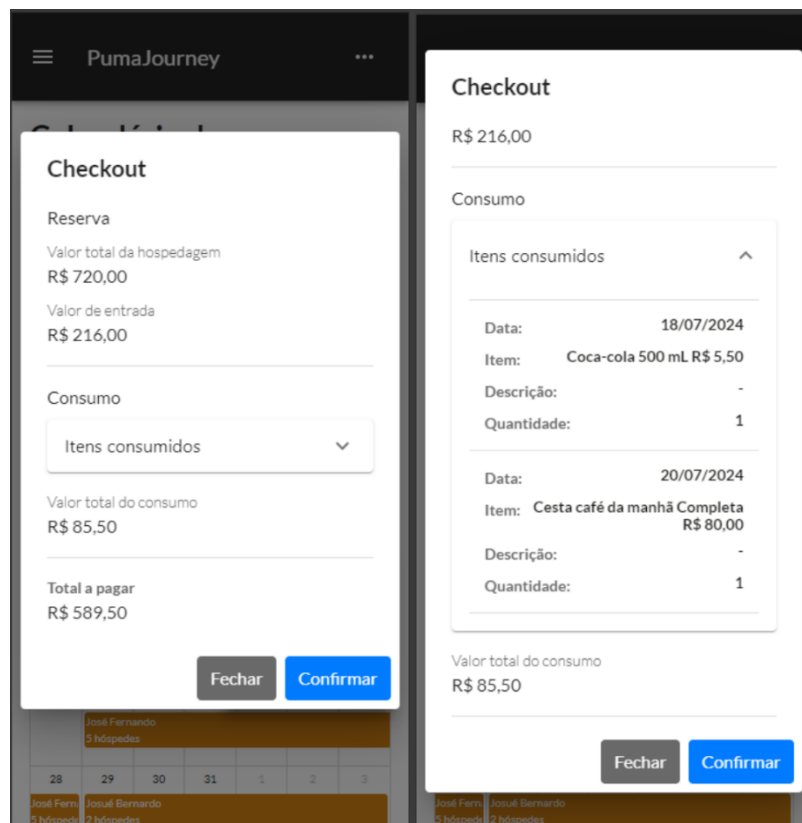


Fonte: Autoria própria

5.3.7. CHECKOUT DE RESERVA

No último dia da estadia, o hóspede deve pagar o valor restante da hospedagem, bem como o valor total dos consumos adicionais, se houverem. Para realizar esse cálculo, o sistema possui a funcionalidade de checkout, que pode ser utilizada apenas quando a reserva está em andamento ou finalizada. Ao selecionar uma reserva e clicar na opção “Checkout”, o usuário terá acesso a um modal com as seguintes informações: total da hospedagem, valor de entrada já pago pelo hóspede, valor total dos consumos, com a opção de visualizar todos esses consumos detalhadamente, e, ao final, o total a pagar. A Figura 26 ilustra a modal de checkout com todos os detalhes. A imagem mostra a modal tanto com os itens de consumo ocultos quanto com os itens de consumo exibidos.

Figura 26: Checkout de uma reserva



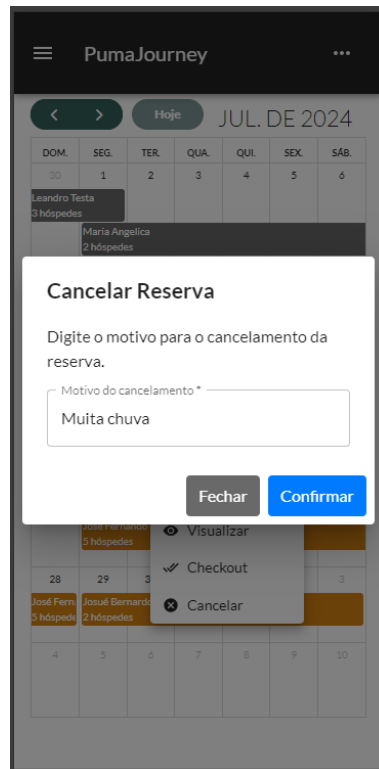
Fonte: Autoria própria

5.3.8. CANCELAMENTO DE RESERVA

O sistema também possui a funcionalidade de cancelamento de reservas. Para cancelar uma reserva, o usuário deve selecioná-la e clicar na opção “Cancelar”. Em seguida, uma modal será exibida, solicitando que o usuário insira o motivo do cancelamento, campo que é

obrigatório. Ao confirmar o cancelamento, a reserva será ocultada do calendário. A Figura 27 mostra essa modal em detalhe.

Figura 27: Cancelamento da reserva



Fonte: Autoria própria

5.3.9. TESTES

Cada funcionalidade desenvolvida no sistema seguiu o mesmo fluxo: desenvolvimento no backend, implementação de testes unitários no backend, implementação de testes E2E no backend, desenvolvimento no frontend e, por último, testes funcionais. Esse processo, que demandou um tempo considerável, foi essencial para assegurar a qualidade e o funcionamento da aplicação.

Os testes unitários e E2E foram desenvolvidos utilizando o Vitest²¹, um *framework* que simplifica a criação e execução de testes. Na Figura 28, são apresentados alguns testes unitários relacionados ao cancelamento de reservas. Estes testes abrangem cenários onde a reserva não existe, onde não é possível cancelar e um cenário de sucesso. Para executar todos os testes unitários da aplicação, utilizamos o comando “npm run test”, que nos permite verificar

²¹ <https://vitest.dev/>

rapidamente se todos os testes estão passando. Na Figura 29, é possível ver esse comando sendo executado e todos os testes sendo realizados com sucesso.

Figura 28: Teste unitário de cancelamento de reserva

```

12 describe('Cancel Booking Service', () => {
13   beforeEach(() => {
14     bookingsRepository = new InMemoryBookingsRepository()
15     sut = new CancelBookingService(bookingsRepository)
16   })
17
18   it('should get error when trying to cancel unexistent booking', async () => {
19     await expect(async () => sut.cancelBooking({ id: '882', cancellationReason: 'Razão do cancelamento' }))
20       .rejects.toBeInstanceOf(ResourceNotFoundError)
21   })
22
23   it('should get error when trying to cancel a already checkouted out booking', async () => {
24     > const bookingInDb: Prisma.BookingGetPayload<{...
33 > }> = {...
78   }
79
80   bookingsRepository.itemsAllPropsIncluded.push(bookingInDb)
81
82   await expect(async () => sut.cancelBooking({ id: '123', cancellationReason: 'Razão do cancelamento' }))
83     .rejects.toBeInstanceOf(ValidationError)
84   })
85
86 > it('should be able to cancel booking per id', async () => {...
164   })
165 })

```

Fonte: Autoria própria

Figura 29: Executando todos os testes unitários

```

PS C:\Users\gabip\Documents\TCC\gestao-pousada-api> npm run test

> gestao-pousada-api@1.0.0 test
> vitest run --dir src/tests/unitTests

RUN v0.33.0 C:/Users/gabip/Documents/TCC/gestao-pousada-api
✓ src/tests/unitTests/users/register.spec.ts (3)
✓ src/tests/unitTests/users/get-user-profile.spec.ts (2)
✓ src/tests/unitTests/users/authenticate.spec.ts (3)
✓ src/tests/unitTests/bookings/booking.spec.ts (15)
✓ src/tests/unitTests/bookings/get-booking-price.spec.ts (18)
✓ src/tests/unitTests/bookings/create-booking.spec.ts (13)
✓ src/tests/unitTests/bookings/edit-booking.spec.ts (21)
✓ src/tests/unitTests/consumptions/get-consumptions.spec.ts (4)
✓ src/tests/unitTests/bookings/get-booking.spec.ts (2)
✓ src/tests/unitTests/huts/get-huts.spec.ts (2)
✓ src/tests/unitTests/guests/get-guests.spec.ts (2)
✓ src/tests/unitTests/bookings/checkout.spec.ts (6)
✓ src/tests/unitTests/bookings/get-all-bookings.spec.ts (3)
✓ src/tests/unitTests/bookings/cancel-booking.spec.ts (3)

Test Files 14 passed (14)
Tests      97 passed (97)
Start at   18:54:50
Duration   2.42s (transform 1.19s, setup 1ms, collect 6.11s, tests 355ms, environment 4ms, prepare 3.82s)

```

Fonte: Autoria própria

Como mencionado anteriormente, os testes E2E, que têm a finalidade de validar o fluxo completo de uma funcionalidade, também foram realizados utilizando o Vitest. Na Figura 30, podemos observar um exemplo de teste E2E para a criação de uma reserva. Devido à maior lentidão desses testes, optou-se por realizar apenas o cenário de sucesso das funcionalidades. Para executar todos os testes E2E, utiliza-se o comando “npm run test :e2e”. A Figura 31 mostra o resultado da execução desses testes.

Figura 30: Teste E2E de criação de reserva

```
8 describe('Create Booking (e2e)', () => {
12
13   afterAll(async () => {
14     await app.close()
15   })
16
17   it('should be able to create a booking', async () => {
18     const { token } = await createAndAuthenticateUser(app)
19     await createHuts()
20     await createBookingNightPrices()
21
22     const createBookingResponse = await request(app.server)
23       .post('/bookings')
24       .send({
25         startDate: new Date(2023, 8, 20),
26         endDate: new Date(2023, 8, 26),
27         adultsQuantity: 2,
28         youngQuantity: 1,
29         childrenQuantity: 0,
30         withHydro: false,
31         isHoliday: true,
32         totalEntry: 222,
33         total: 1,
34         hutId: 'b920dbb0-3e5a-46d4-8031-243e98ccd459',
35         guest: { name: 'Joana Silva', cpf: '881.388.500-86', phone: '(51) 9 93132232' }
36       })
37       .set('Authorization', `Bearer ${token}`)
38
39     expect(createBookingResponse.statusCode).toEqual(201)
40     expect(createBookingResponse.body).toEqual(expect.objectContaining({
41       id: expect.any(String)
42     }))
43   })
44 })
```

Fonte: Autoria própria

Figura 31: Executando todos os testes E2E

```
✓ src/tests/e2eTests/users/profile.spec.ts (1)
✓ src/tests/e2eTests/bookings/edit-booking.controller.spec.ts (1) 357ms
✓ src/tests/e2eTests/bookings/create-booking.controller.spec.ts (1) 326ms
✓ src/tests/e2eTests/bookings/checkout.controller.spec.ts (1) 346ms
✓ src/tests/e2eTests/bookings/get-booking.controller.spec.ts (1) 321ms
✓ src/tests/e2eTests/bookings/get-all-bookings.spec.ts (1) 309ms
✓ src/tests/e2eTests/bookings/get-checkout-data.controller.spec.ts (1) 328ms
✓ src/tests/e2eTests/consumptions/get-consumptions.controller.spec.ts (2)
✓ src/tests/e2eTests/users/refresh.spec.ts (1)
✓ src/tests/e2eTests/consumptions/get-active-consumptions.controller.spec.ts (2)
✓ src/tests/e2eTests/bookings/get-booking.price.controller.spec.ts (1)
✓ src/tests/e2eTests/huts/get-huts.spec.ts (1)
✓ src/tests/e2eTests/users/authenticate.spec.ts (1)
✓ src/tests/e2eTests/guests/get-guests.spec.ts (1)

Test Files 14 passed (14)
Tests      16 passed (16)
Start at   19:00:38
Duration   18.18s (transform 276ms, setup 6ms, collect 16.62s, tests 3.18s, environment 87.24s, prepare 3.69s)
```

Fonte: Autoria própria

Ao final do desenvolvimento, foram realizados testes de aceitação com os usuários do sistema. Esses testes têm como objetivo verificar se o sistema atende aos critérios de aceitação e está alinhado com as expectativas dos usuários finais.

Para isso, após a hospedagem do sistema, ele foi liberado para que os proprietários da pousada pudessem validá-lo. Após alguns dias de testes, os proprietários consideraram o sistema muito útil, especialmente pelo cálculo automático das reservas e do checkout. Foram solicitadas algumas pequenas alterações relacionadas ao layout, que já foram corrigidas.

6. CONSIDERAÇÕES FINAIS

Com a demanda crescente de reservas e o aumento do número de cabanas disponíveis, a Pousada Cabanas Passos do Puma teve dificuldades na gestão interna. O uso de calendários e planilhas físicas estava causando atrasos significativos, seja pela necessidade constante de verificar disponibilidade de reservas com a pessoa responsável pela planilha, seja pelos cálculos complexos das reservas, que variam de acordo com o número de hóspedes, quantidade de dias, entre outros fatores.

Este trabalho teve como objetivo transformar a gestão da pousada, tornando-a mais eficiente e rápida. A aplicação desenvolvida facilitará o cadastro e visualização de reservas, com uma funcionalidade de calendário separado por cabanas, permitindo incluir ou alterar reservas conforme necessário. A funcionalidade de cadastro e gerenciamento dos consumos das reservas torna o processo de registro dos consumos dos hóspedes muito mais rápido e eficiente. E, uma das funcionalidades que se destacam, a facilidade de cálculo das reservas, tanto no momento do cadastro quanto no checkout, considerando vários fatores simultaneamente e incluindo o valor do consumo dos hóspedes.

Além disso, a escolha pela implementação de uma aplicação web permite o acesso em diferentes dispositivos, tanto smartphones quanto computadores. Isso proporciona maior flexibilidade aos usuários, que podem optar por utilizar ambas as opções conforme suas necessidades.

Os testes foram uma parte crucial do desenvolvimento. Eles asseguraram a qualidade da aplicação e garantiram que funcionalidades desenvolvidas posteriormente não afetassem as já existentes.

Neste trabalho foi desenvolvido o MVP da aplicação. Embora diversas funcionalidades adicionais tenham sido pensadas, a falta de tempo impediu sua implementação. As próximas etapas previstas incluem o desenvolvimento de uma página para visualizar um histórico completo de reservas, com diversas opções de filtragem. Em seguida, serão criados relatórios mensais, possibilitada o uso da aplicação pelos próprios hóspedes para solicitação de consumos, e uma página pública da pousada com informações detalhadas sobre ela.

REFERÊNCIAS

ABREU, L. Typescript. O Javascript Moderno Para Criação de Aplicações. 1ª ed. Lisboa: Editora FCA, 2017.

AQUILES, Alexandre; FERREIRA, Rodrigo. Controlando Versões com Git e GitHub. São Paulo: Casa do código, 2014.

Arquitetura Cliente / Servidor e Three-tier. DevMedia, 2007. Disponível em: <<https://www.devmedia.com.br/arquitetura-client-server-e-three-tier/5865>>. Acesso em: 7 jul. 2024.

BOOCH, G; RUMBAUGH, J e JACOBSON, I. UML, Guia do Usuário: tradução; Fábio Freitas da Silva, Rio de Janeiro, Campus, 2000.

CALDEIRA, C.P. PostgreSQL Guia Fundamental. 1ª ed. Lisboa: Edições Sílabo, 2015.

CARVALHO, Vinícius. PostgreSQL - Banco de dados para aplicações web modernas. São Paulo: Casa do código, 2017

CHACON, Scott; STRAUB, Ben. Pro Git. 2a ed. Apress, 2014.

COULOURIS, George; DOLLIMORE, Jean; KINDERBERG, Tim; GORDON, Blair. Sistemas Distribuídos: Conceitos e Projetos. 5a ed. Porto Alegre: Bookman, 2013.

DOCUMENTAÇÃO DO FASTIFY. Fastify, 2024. Disponível em: <<https://fastify.dev/docs/latest/>>. Acesso em 2 de Março de 2024.

GERHARDT, Tatiana Engel; SILVEIRA, Denise Tolfo (Org.). Métodos de pesquisa. Porto Alegre: UFRGS, 2009. Disponível em: <<http://www.ufrgs.br/cursopgdr/downloadsSerie/derad005.pdf>>. Acesso em: 1 de Agosto de 2023.

Guia de JavaScript: o que é e como aprender a linguagem mais popular do mundo? Alura, 2023. Disponível em: <<https://www.alura.com.br/artigos/javascript>>. Acesso em: 7 jul. 2024.

Introdução ao Next.js - Um Framework para Desenvolvedores React. CLEMENTE, P., 2023. Disponível em: <<https://blog.rocketseat.com.br/introducao-ao-next-js/>>. Acesso em: 7 jul. 2024.

JavaScript With Syntax For Types. TypeScript, 2020. Disponível em:<<https://www.typescriptlang.org/pt/>>. Acesso em: 7 jul. 2024.

Metodologia Scrum para a gestão de processos ágeis na indústria. Tecnicon, 2019. Disponível em: <<https://www.tecnicon.com.br/blog/411->

Metodologia_Scrum_para_a_gestao_de_processos_ageis_na_industria>. Acesso em: 6 jul. 2024.

Node.JS: o que é, como funciona esse ambiente de execução JavaScript e um Guia para iniciar. Alura, 2023. Disponível em: <<https://www.alura.com.br/artigos/javascript>>. Acesso em: 8 jul. 2024.

O que é o Scrum? | Explicação sobre a metodologia Scrum. AWS, 2022. Disponível em: <<https://aws.amazon.com/pt/what-is/scrum/#:~:text=O%20Scrum%20%C3%A9%20um%20framework%20para%20fazer%20o%20trabalho%20no,Agile%20para%20gerenciamento%20de%20projetos.>>. Acesso em: 6 jul. 2024.

O que é um banco de dados? Oracle, 2014. Disponível em: <[https://www.oracle.com/br/database/what-is-database/#:~:text=Um%20banco%20de%20dados%20%C3%A9,banco%20de%20dados%20\(DBMS\).](https://www.oracle.com/br/database/what-is-database/#:~:text=Um%20banco%20de%20dados%20%C3%A9,banco%20de%20dados%20(DBMS).>)>. Acesso em: 6 jul. 2024.

O que é um framework e para que serve? LENCINA, W. Disponível em: <<https://ebaonline.com.br/blog/framework-seo#:~:text=Um%20framework%20define%20a%20estrutura,%20seguran%C3%A7a%20C%20autentic%C3%A7%C3%A3o%20etc.>>. Acesso em: 21 jul. 2024.

PEIREIRA, Caio Ribeiro. Construindo APIs REST com Node.js. São Paulo: Casa do código, 2016.

PÓS A PANDEMIA, MAIORIA DOS VIAJANTES BRASILEIROS PRETENDE VIAJAR ENTRE DUAS E QUATRO VEZES POR ANO. Booking.com, 2021. Disponível em: <<https://news.booking.com/pt-br/apos-a-pandemia-maioria-dos-viajantes-brasileiros-pretende-viajar-entre-duas-e-quatro-vezes-por-ano/>>. Acesso em 1 de Agosto de 2023.

React. React, 2015. Disponível em: <<https://pt-br.react.dev/>>. Acesso em: 7 jul. 2024.

SBROCCO, José Henrique Teixeira de Carvalho; MACEDO, Paulo Cesar de. Metodologias ágeis: engenharia de software sob medida. São Paulo: Érica : Saraiva, 2012.

SMITH, J. K. A ciência do exemplo: Guia prático de formatação. 2ª ed. São Paulo: Editora ABC, 2021.

Sobre. Git, 2024. Disponível em: <<https://git-scm.com/about>>. Acesso em: 9 jul. 2024.

SUTHERLAND, Jeff. Scrum. A Arte de Fazer o Bom do Trabalho na Metade do Tempo. Editora LeYa, 2016.

TANENBAUM, Andrew S. Redes de computadores. 5ª ed. São Paulo: Pearson, 2011.

What Is PostgreSQL? PostgreSQL, 2024. Disponível em:
<<https://www.postgresql.org/docs/current/intro-what-is.html>>. Acesso em: 9 jul. 2024.