

UNIVERSIDADE PAULISTA

**BRUNO PADOVEZ TAVANTE
DIEGO BARBOSA ROCHA
FABIANA IVO SOUZA
MATHEUS DURÃES PIRES
MAYARA KRYSTINA DOS S OLIVEIRA**

**DESENVOLVIMENTO DE SOFTWARE:
SISTEMA DE GESTÃO HOTELEIRA**

**SÃO PAULO
2021**

BRUNO PADOVEZ TAVANTE
DIEGO BARBOSA ROCHA
FABIANA IVO SOUZA
MATHEUS DURÃES PIRES
MAYARA KRYSTINA DOS S OLIVEIRA

DESENVOLVIMENTO DE SOFTWARE:
SISTEMA DE GESTÃO HOTELEIRA

Projeto Integrado Multidisciplinar para
conclusão de curso de Tecnologia em
Análise e Desenvolvimento de Sistemas
apresentado à Universidade Paulista –
UNIP.

Orientador: Prof. Me. José Cassiano
Grassi Gunji.

SÃO PAULO
2021

CIP - Catalogação na Publicação

Desenvolvimento de software : sistema de gestão hoteleira / Diego
Barbosa Rocha...[et al.]. - 2021.

128 f. : il. color

Trabalho de Conclusão de Curso (Graduação) apresentado ao Instituto
de Ciência Exatas e Tecnologia da Universidade Paulista, São Paulo,
2021.

Área de Concentração: Tecnologia.

Orientador: Prof. Me. José Cassiano Grassi Gunji.

1. Software. 2. Desenvolvimento. 3. Sistema. I. Rocha, Diego Barbosa.
- II. Gunji, José Cassiano Grassi (orientador).

BRUNO PADOVEZ TAVANTE
DIEGO BARBOSA ROCHA
FABIANA IVO SOUZA
MATHEUS DURÃES PIRES
MAYARA KRYSTINA DOS S OLIVEIRA

DESENVOLVIMENTO DE SOFTWARE:
SISTEMA DE GESTÃO HOTELEIRA

Projeto Integrado Multidisciplinar para
conclusão de curso de Tecnologia em
Análise e Desenvolvimento de Sistemas
apresentado à Universidade Paulista –
UNIP.

Aprovado em:

BANCA EXAMINADORA

_____/_____/_____

Prof. Nome do Professor

Universidade Paulista – UNIP

_____/_____/_____

Prof. Nome do Professor

Universidade Paulista – UNIP

_____/_____/_____

Prof. Nome do Professor

Universidade Paulista – UNIP

RESUMO

Este trabalho foca em documentar o desenvolvimento de softwares objetivos, facilitando as principais operações de um hotel. Busca isso através de um design adequado, com as opções bem distribuídas e práticas para uso diário, com interfaces de usuário que possibilitam a automatização dos serviços selecionados, atendendo as necessidades rotineiras do cliente. Propõe-se a estabelecer um modelo de processo valorizando as habilidades da equipe nos métodos de desenvolvimento de software, para adesão de boas práticas e garantia da qualidade do produto. Para isso foram utilizadas metodologias para guiar projeto. Os envolvidos, respeitando as definições e com os métodos e ferramentas específicas, podem implementar as propriedades desejadas. A integração de frameworks juntamente com o modelo do projeto resultou na arquitetura em si, estruturando um fluxo de desenvolvimento bem definido.

Palavras-chave: Sistema. Software. Desenvolvimento.

ABSTRACT

This final paper focus on documentate the development of objective softwares, improvindo the main operations of a hotel. Looks for it by using a suitable design, with pretty disposed and practice options for daily use, with user interfaces that makes possible the automatization of the selected services, attending client's common necessities. Proposes on establish a process model that inhances team's qualities on software development methods, for good practices approach and guaranteeing the product quality. Therefore, were used methodologies to guide the project. The involved, respecting the definitions and with the specific methods and tools, can implement the desired properties. Frameworks integration alongside with the project model resulted on the archtecture itself, structuring a well defined development flow.

Key-words: System. Software. Development.

LISTA DE FIGURAS

Figura 1 - Colaboradores da organização Quiet Time.....	15
Figura 2 - Metas e atividades do processo.....	17
Figura 3 - Processos focados pela organização e como realizá-los.....	21
Figura 4 – Processos da ISO 12207 e como o projeto os aborda essencialmente ...	21
Figura 5 - Características da ISO 9126 para o projeto	22
Figura 6 - EAP do projeto	24
Figura 7 - Cronograma de tarefas do projeto	25
Figura 8 - Diagrama de rede do cronograma	25
Figura 9 - Modelo conceitual do banco de dados.....	38
Figura 10 - Modelo lógico do banco de dados.....	38
Figura 11 - Casos de Uso gerais do sistema desktop	44
Figura 12 - Caso de Uso para login no sistema	44
Figura 13 - Casos de Uso do módulo de Gerência	45
Figura 14 - Casos de Uso do módulo de Governança	45
Figura 15 - Casos de Uso do módulo de Recepção.....	46
Figura 16 - Casos de Uso dos sistemas web e mobile.....	46
Figura 17 - Atividades de recepção do hotel	59
Figura 18 - Atividades de governança do hotel	60
Figura 19 - Atividades de gerência do hotel	61
Figura 20 - Atividades realizadas pelos hóspedes	63
Figura 21 - Classes de entidades.....	64
Figura 22 - Classes de acesso ao banco de dados e suas interfaces (contrato)	64
Figura 23 - Classes da interface Desktop	65
Figura 24 - Classes da interface web	65
Figura 25 - Diagrama de pacotes	66
Figura 26 - Diagrama de comunicação completo	67
Figura 27 - Diagrama de comunicação com zoom 1	67
Figura 28 - Diagrama de comunicação com zoom 2	68
Figura 29 - Diagrama de comunicação com zoom 3	68
Figura 30 - Diagrama de comunicação com zoom 4	69
Figura 31 - Diagrama de comunicação com zoom 5	69
Figura 32 - Máquina de estados para cadastro de pessoas.....	70
Figura 33 - Máquina de estados para atualizar pessoas	70

Figura 34 - Máquina de estados para manipular reserva	71
Figura 35 - Máquina de estados para alterar preços	71
Figura 36 - Máquina de estados para login no sistema.....	72
Figura 37 - Sequência para registro de reserva	73
Figura 38 - Sequência para iniciar hospedagem	73
Figura 39 - Sequência para adicionar consumo	74
Figura 40 – Interface nova reserva em tela cheia (desktop responsivo).	86
Figura 41 - Interface para registrar nova reserva.	87
Figura 42 - Página inicial web seção de pesquisar reserva	87
Figura 43 - Cards informativos em tamanho normal	88
Figura 44 - Cards informativos em tamanho reduzido responsivo	88
Figura 45 - Página de login em tamanho normal.....	89
Figura 46 - Página de login em tamanho reduzido responsivo	89
Figura 47 - Tela de login android responsiva	90
Figura 48 – Alteração de dados em android tela responsiva	90
Figura 49 - Lógica das camadas do sistema desktop e web	91
Figura 50 - Gerenciador de solução com os projetos.....	92
Figura 51 - Código com tratamento de erro	92
Figura 52 - string de conexão com banco de dados.....	93
Figura 53 - Uso de polimorfismo	93
Figura 54 - Lógica das páginas web.....	94
Figura 55 - Página inicial do site	94
Figura 56 - Uso de C Sharp junto de HTML	95
Figura 57 - Conexão com banco em página HTML.....	95
Figura 59 - Constraints de interface android	96
Figura 60 - strings para uso em android.....	96

LISTA DE QUADROS

Quadro 1 - Tipos de Quarto do hotel.....	16
Quadro 2 - Matriz de responsabilidades.....	20
Quadro 3 - Conceitos do Programa 5S	23
Quadro 4 - Regras de negócio para o sistema.....	26
Quadro 5 - Requisitos de Usuário	27
Quadro 6 - Requisitos de Sistema.....	28

Quadro 7 - Requisitos Funcionais do sistema desktop	30
Quadro 8 - Requisitos funcionais do sistema web.....	31
Quadro 9 - Requisitos funcionais do sistema mobile.....	31
Quadro 10 - Casos de teste para verificação	39
Quadro 11 - Casos de teste do sistema	42
Quadro 12 - Descrição de caso de uso de login.....	47
Quadro 13 - Descrição de caso de uso relatório	47
Quadro 14 - Descrição de caso de uso alterar dados hóspede.....	48
Quadro 15 - Descrição de caso de uso alterar dados funcionário.....	48
Quadro 16 - Descrição de caso de uso cadastrar funcionário.....	49
Quadro 17 - Descrição de caso de uso alterar preços	49
Quadro 18 - Descrição de caso de uso conferir quartos	50
Quadro 19 - Descrição de caso de uso alterar quartos	50
Quadro 20 - Descrição de caso de uso manter consumo	51
Quadro 21 - Descrição de caso de uso adicionar consumo	51
Quadro 22 - Descrição de caso de uso manter produtos	51
Quadro 23 - Descrição de caso de uso adicionar produto	52
Quadro 24 - Descrição de caso de uso nova reserva	52
Quadro 25 - Descrição de caso de uso ver reservas	53
Quadro 26 - Descrição de caso de uso check-in.....	53
Quadro 27 - Descrição de caso de uso check-out.....	54
Quadro 28 - Descrição de caso de uso pagamento	54
Quadro 29 - Descrição de caso de uso cancelar reserva.....	55
Quadro 30 - Descrição de caso de uso ver informações.....	55
Quadro 31 - Descrição de caso de uso login hóspede	56
Quadro 32 - Descrição de caso de uso reserva web.....	56
Quadro 33 - Matriz de Rastreabilidade.....	57
Quadro 34 - Usuários primários do sistema	78
Quadro 35 - Usuários secundários do sistema.....	78
Quadro 36 - Prioridades da interface	79
Quadro 37 - Prioridades de tarefas de usuários.....	80
Quadro 38 - Requisitos de Usabilidade	81
Quadro 39 - Perfil de usuário para Gerente Geral.....	82
Quadro 40 - Perfil de usuário para Gerente de Governança	83

Quadro 41 - Perfil de usuário para Repcionista.....	83
Quadro 42 - Perfil de usuário para Hóspedes	84
Quadro 43 - Persona de Gerente Geral	85
Quadro 44 - Persona de Hóspede.....	85

SUMÁRIO

1 INTRODUÇÃO	13
2 MODELAGEM DO NEGÓCIO	14
2.1 Compreensão do negócio	14
2.1.1 Cargos e funções dos colaboradores envolvidos.	15
2.1.2 Organograma com colaboradores essenciais.	15
2.2 Características estruturais dos quartos	16
2.3 Atores do sistema	16
3 PROCESSO DE DESENVOLVIMENTO COM FOCO NA QUALIDADE	17
3.1 Métodos do arcabouço.	17
3.1.1 Ciclo de vida de desenvolvimento.	18
3.1.2 Gerenciamento da equipe de desenvolvimento.	19
3.1.2.1 Matriz de Responsabilidades.	19
3.1.2.2 Cálculo de responsabilidade.	20
3.1.3 Gerência do processo com CMMI.	20
3.1.4 Qualidade do ciclo de vida do software com ISO 12207.	21
3.1.5 Garantia da qualidade funcional com ISO 9126.	22
3.1.6 Gestão da equipe para qualidade do projeto	22
3.2 Ferramentas do arcabouço.	23
3.3 Gestão do projeto	24
3.3.1 Estrutura Analítica do Projeto (EAP)	24
3.3.2 Análise de cronograma	24
3.3.3 Diagrama de Setas do cronograma com eventos e tarefas.	25
4 ESPECIFICAÇÃO DE REQUISITOS	25
4.1 Regras de negócio	26
4.2 Requisitos de Usuário e Requisitos de Sistema.	27
4.3 Requisitos Funcionais.	30

4.4 Requisitos Não Funcionais.	32
4.4.1 Adaptação à LGPD.	34
5 ELABORAÇÃO DO BANCO DE DADOS	35
5.1 Diagramas MER	37
5.1.1 Modelo Conceitual	37
5.1.2 Modelo Lógico	38
5.1.3 Modelo Físico	39
5.2 Planilhas de testes	39
5.2.1 Verificação do código por meio de Casos de Teste.	39
5.2.2 Testes do banco de dados	42
6 MODELAGEM DO SISTEMA	43
6.1 Diagramas de Casos de Uso.	43
6.1.1 Descrição dos Casos de Uso.	47
6.1.2 Matriz de Rastreabilidade.	56
6.2 Diagramas de Atividades.	58
6.3 Diagrama de Classes	63
6.4 Diagrama de pacotes.	66
6.4 Diagramas de comunicação.	67
6.5 Diagramas de máquina de estados.	69
6.4 Diagramas de Sequência	72
6.5 Diagrama de Implantação	75
7 ESTUDO DO USUÁRIO E ESTUDO DE INTERFACE	75
7.1 Estudo do usuário.	76
7.1.1 Documento de visão.	76
7.1.2 Requisitos de usabilidade.	81
7.1.3 Perfis de usuário e Personas.	82
7.2 Protótipos de interface.	86

7.2.1 Interfaces dos sistemas Desktop, Web e Mobile.	86
8 CODIFICAÇÃO DOS SISTEMAS	91
8.1 Aplicação Desktop	91
8.2 Aplicação Web.	94
8.3 Aplicação Android.	96
CONCLUSÃO	97
REFERÊNCIAS	99
GLOSSÁRIO	100
APÊNDICE A – DICIONÁRIO DE DADOS	101
APÊNDICE B – MANUAL DE USO DOS SOFTWARES	105
APÊNDICE C – LINKS PARA RELEASES COM CÓDIGOS E PROJETOS	111
APÊNDICE D – SCRIPTS DE CRIAÇÃO DO BANCO DE DADOS	112
APÊNDICE E – CÓDIGOS DE PRINCIPAIS ARTEFATOS	124

1 INTRODUÇÃO

Problema de pesquisa:

O funcionamento de hotéis envolve algumas atividades que se espera que todos executem, como registro de hóspedes, de reservas, gestão das hospedagens e do hotel como empresa. O Hotel Quiet Time de fato as executa, utilizando livros de controle, prática que advém de sua criação décadas atrás. Porém, esse modelo têm se mostrado cada vez mais ultrapassado, visto o avanço tecnológico, recorrência de erros, lentidão e pouca praticidade nas atividades e mudanças dos dados.

Hipótese de pesquisa:

Com a entrada da novos herdeiros na gerência, esses decidiram investir em tecnologia e implementar sistemas (softwares) para controlar as atividades, substituindo os livros de controle. Para isso contrataram a empresa SofTech Desenvolvimento, a qual designou uma equipe de desenvolvimento de software para realizar a tarefa de construção de um sistema de gestão hoteleira.

Escopo:

Após o reconhecimento de cenário realizado pela equipe, concluiu-se que para contemplar todas as atividades do hotel no sistema, seriam necessários 4 anos. Assim sendo, foi acordado que no primeiro ano fossem contempladas apenas as funcionalidades essenciais, definidas pela alta gerência. São elas:

1. Cadastrar e atualizar dados de hóspedes.
2. Registrar e manipular reservas de hospedagem, efetuando check-in, check-out, cancelamento e pagamentos.
3. Gerenciar o estado dos quartos, para serviços de governança.
4. Controlar consumos do frigobar dos quartos durante hospedagem, inclusive o estoque desses produtos consumíveis.
5. Cadastrar e atualizar dados de funcionários.
6. Obter relatórios gerenciais, relacionados aos dados geridos pelo sistema.
7. Possibilitar manipulação dos preços de estadia.
8. Prover aos hóspedes informações sobre o hotel.

Objetivo geral:

Estando definidas as atividades, a equipe montou um projeto com o seguinte objetivo: **Realizar análise, documentação e desenvolvimento dos seguintes itens:**

- I. Aplicação desktop **completa**, restrito apenas à funcionários do hotel, com todas as funcionalidades acordadas.
- II. Aplicação web simplificada, para uso dos hóspedes, que possibilite conhecer o hotel, cadastro do hóspede, registro de nova reserva e acesso aos próprios dados. Esta será uma **versão para testes** do futuro site comercial do hotel.
- III. Protótipos de telas para aplicação para dispositivos móveis, com algumas funcionalidades da versão web, em sistema operacional Android.

Objetivos específicos:

A execução desse projeto demanda algumas tarefas cruciais como:

- Analisar o negócio do Hotel Quiet Time.
- Definir metodologia de desenvolvimento.
- Estruturar requisitos.
- Estruturar o banco de dados.
- Modelar o sistema.
- Modelar as interfaces de usuário.
- Programar o sistema.
- Testar e validar o sistema gerando o manual de uso.

Justificativa:

Utilizar um sistema de gestão hoteleira tende a resultar em inúmeros benefícios para um hotel. A execução de suas atividades ficam mais rápidas e com menos chances de erros, garantindo qualidade no atendimento ao cliente. O treinamento de novos funcionários é rápido e fácil. O sistema é mutável, e pode contemplar futuras alterações nas funcionalidades ou inclusão de novas. Por fim, esses e demais fatores consequentes contribuem para a reputação do hotel, aumentando sua vantagem competitiva.

2 MODELAGEM DO NEGÓCIO

É importante que antes do início dos processos de desenvolvimento, se entenda o funcionamento do negócio no qual o software será inserido. A seguir serão apresentadas diferentes visões das atividades do hotel que serão relacionadas no sistema.

2.1 Compreensão do negócio

2.1.1 Cargos e funções dos colaboradores envolvidos.

O Gerente Geral é responsável por supervisionar todo o hotel, analisar relatórios e administrar dados dos funcionários. Também tem permissão para alterar preços das diárias.

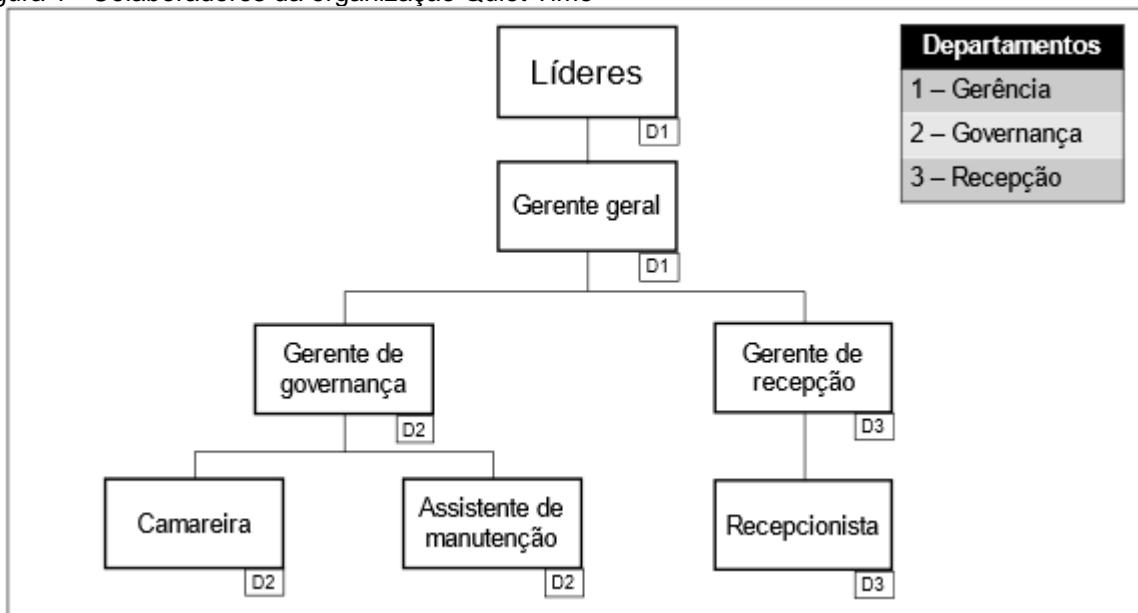
O Gerente de Governança administra as hospedagens. É quem verifica os status dos quartos para delegar limpezas ou reparos, registra consumos dos hóspedes seja por contato dos ou por verificação da camareira e também faz o controle do estoque desses poucos produtos.

O Recepção é quem faz contato direto com os hóspedes. No atendimento, informa-os sobre as regras do hotel. É quem registra as reservas de hospedagem, coletando dados pessoais e do período de estadia. Manipula as reservas podendo iniciar, finalizar, cancelar ou adicionar pagamentos.

2.1.2 Organograma com colaboradores essenciais.

Para uma compreensão mais ampla, o organograma a seguir ilustra os relacionamentos entre colaboradores envolvidos no projeto. Vale ressaltar que apenas alguns serão usuários do sistema gerencial em versão desktop.

Figura 1 - Colaboradores da organização Quiet Time



Fonte: Autor, 2021.

2.2 Características estruturais dos quartos

A recepção e demais áreas não afetam o sistema. Já os quartos sim, pois seus diferentes tipos diferem os preços da estadia. São trinta quartos no total, seis de cada tipo, detalhados a seguir:

Quadro 1 - Tipos de Quarto do hotel

TIPO	QTDE	ÁREA m ²	CAMAS
Único	10	18m ²	1 Solteiro
Dupla	10	20m ²	2 Solteiro
Casal	10	22m ²	1 Casal
Mix	10	26m ²	1 Casal e 1 Solteiro
Máximo	10	28m ²	1 Casal e 2 Solteiro

Fonte: Autor, 2021.

Exceto essas particularidades, todos os quartos possuem:

- Varanda
- Banheiro privativo.
- TV à cabo.
- Ar-condicionado.
- Telefone para contato com recepção e governança.
- Frigobar.
- Escrivaninha.

2.3 Atores do sistema

Nesse primeiro estágio do software, as pessoas que terão acesso e o utilizarão, ou seja, seus atores ou usuários, serão:

- Na versão Desktop: Gerente Geral, Gerente de Governança, Repcionista.
- Nas versões Web e Mobile: Hóspedes (clientes do hotel).

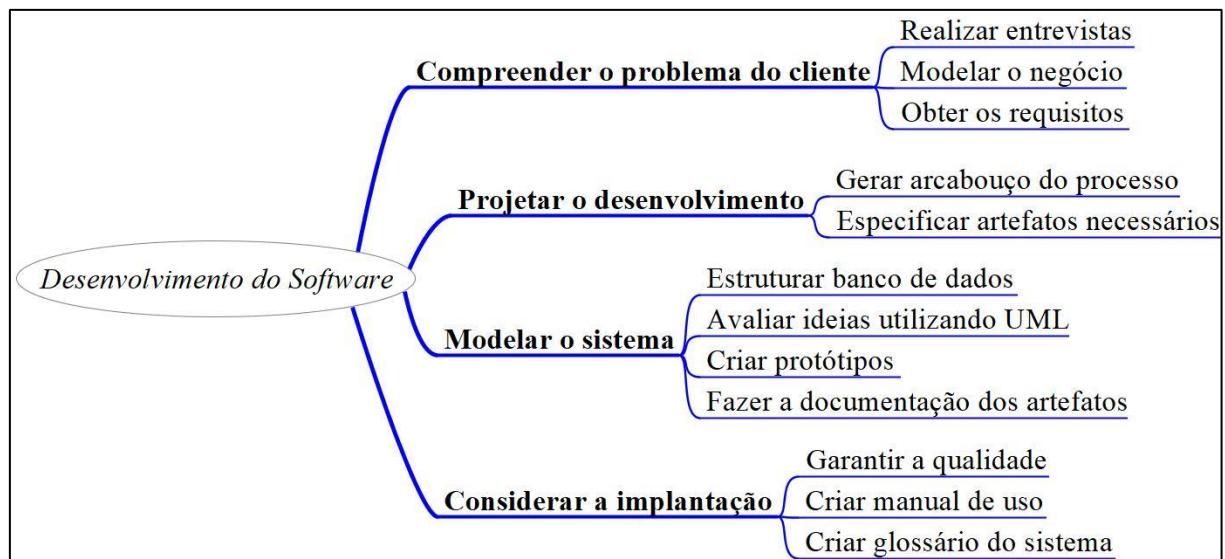
Para facilitar a interação dos usuários nessa primeira versão, os donos do hotel devem ser classificados como integrantes do departamento de gerência, com cargos de gerente geral, garantindo assim seu acesso completo ao sistema.

3 PROCESSO DE DESENVOLVIMENTO COM FOCO NA QUALIDADE

O processo do desenvolvimento de software envolve o conceito de obter dados como entrada, processá-los e entregar o resultado como saída. No projeto em questão, o cliente (hotel) fornece informações, que serão utilizadas pela equipe de desenvolvimento para desenvolver um software, e como resultado entregar um sistema funcional.

Seguindo os moldes do Capability Maturity Model Integration (CMMI), um processo pode conter metas e cada meta conter atividades relacionadas. Para este projeto, o processo pode ser definido conforme a Figura 1.

Figura 2 - Metas e atividades do processo



Fonte: Autor, 2021.

A qualidade é uma questão fundamental na vida de todos quando o assunto é tecnologia. Para que um Software seja elaborado é necessária a abordagem de uma sequência de passos bem definidos que vão influenciar em todo o processo de criação.

O **Arcabouço do Processo** comporta os métodos e ferramentas que caracterizam o processo de desenvolvimento de um software.

3.1 Métodos do arcabouço.

Os métodos envolvem metodologias e normas que contém instruções sobre como realizar as atividades do processo de desenvolvimento de software.

3.1.1 Ciclo de vida de desenvolvimento.

Para realizar esse desenvolvimento, atualmente existem diversas metodologias que facilitam essa concepção e tornam esse processo mais simples e organizado. À vista disso, durante o desenvolvimento do sistema para gerência do hotel foi decidido o uso de uma metodologia ágil chamada DevOps que partilhará de alguns recursos do framework Scrum e Kanban.

O DevOps é uma nova tendência na indústria de TI, ele surgiu por volta de 2009 quando dois amigos do grupo Flickr apontaram um fato que ocorreu em seu ambiente de trabalho. Quando foi apontado uma falha para uma equipe, eles perceberam que a equipe quis responsabilizar uns aos outros pela falha. O DevOps acredita que qualquer erro ocasionado, pode ser considerado um erro sistêmico e no lugar de encontrar um responsável pela falha, ele busca uma solução rápida para que esse mesmo erro não se repita, unindo o time de desenvolvimento com a equipe operacional a fim de que contribuam juntamente uma vez que o objetivo de ambos é o mesmo, o que diminui prazos de entrega.

O Scrum torna-se grande aliado do DevOps pois compartilha algumas ideias de desenvolvimento, como por exemplo, a entrega contínua. O DevOps além de realizar práticas de CI/CD, faz uso de ferramentas eficientes de gerenciamento e automação. No desenvolvimento desse software, foi definido que:

- Aproveitando do Scrum a forma de coleta de requisitos e documentação, prioriza-se que seja registrado somente o essencial.
- Uso do Kanban para controlar tarefas.
- Foi definido que cada Sprint deverá durar duas semanas.
- Durante o desenvolvimento do software, os canais de feedback do DevOps ficam aberto o tempo todo, permitindo que os usuários realizem comentários que são convertidos em códigos, que passam por testes e que após aprovados, retornam ao cliente como uma nova implementação.

A ideia é que ao final de cada Sprint o cliente tenha um executável para que ele possa apontar o que deve ser melhorado e o que está de acordo com suas necessidades, mas isso não impede que o cliente realize comentários e solicitações durante todo o andamento do projeto.

3.1.2 Gerenciamento da equipe de desenvolvimento.

Os Stakeholders, também chamados envolvidos, podem ser definidos, segundo Sommerville e Sawyer (2016, p. 139), como “qualquer pessoa que se beneficie de forma direta ou indireta do sistema que está sendo desenvolvido”. Para o sistema do hotel, serão considerados a equipe de desenvolvimento e a equipe do hotel, essa última, com uma representante específica para manter contato com a equipe de desenvolvimento.

A seguir, constam os stakeholders do projeto, seus cargos e funções. Os cargos que contém a indicação “(T)” significam pertencimento ao Time de Desenvolvimento (Team) do Scrum como programador.

- **Cliente:** Sandra Muniz Bozolan. Representa o hotel, fornece informações e aprova artefatos gerados no desenvolvimento.
- **(T) Gerente do Projeto:** Diego Barbosa Rocha. Atua como Product Owner, contatando o cliente e gerando material para o time.
- **(T) Gerente do Sistema:** Mayara Krystina Dos S Oliveira. Atua como Scrum Master, monitorando o time e analisando os artefatos gerados.
- **(T) Analista de Sistemas:** Matheus Durães Pires. Responsável pela modelagem do sistema e direção do desenvolvimento.
- **(T) DBA¹:** Fabiana Ivo Souza. Responsável pela estrutura e controle do banco de dados. Também atua como analista DevOps.
- **(T) Programador Sênior:** Bruno Padovez Tavante. Responsável pela arquitetura e codificação do sistema.

3.1.2.1 Matriz de Responsabilidades.

A Matriz de Responsabilidades do Project Management Body of Knowledge (PMBOK) é um método utilizado para dispor as atividades do processo em conjunto com os integrantes que as realizarão, sendo assim possível obter informações como qual stakeholders tem mais influência no projeto, ou se algum deles está sobrecarregado. Os atributos do processo representam o papel do stakeholder na atividade. Estão dispostos com as letras RASP, que significam respectivamente: Responsável, Aprova, Suporte e Participante.

¹ DBA (Database Administrator) é a abreviação de Administrador(a) de Bancos de Dados.

Quadro 2 - Matriz de responsabilidades

Item	Cargo Atividade	Cliente	G. Projeto	G. Sistema	A. Sistema	DBA	Pro.
A	Objetivos do sistema	R	S	P	P	P	P
B	Requisitos do sistema	A	R	S	P	P	P
C	Gerência da Equipe	P	P	R	P	A	P
D	Arquitetura do BD	-	P	P	S	R	A
E	Diagramas UML	-	P	A	R	P	S
G	Protótipo de Interface	A	P	P	S	S	R
H	Revisão da Qualidade	P	A	R	S	P	P
I	Manual de Uso	P	A	S	R	P	S
J	Glossário do sistema	P	R	S	A	P	P

Fonte: Autor, 2021.

3.1.2.2 Cálculo de responsabilidade.

Através dos atributos do processo da matriz, o cálculo sobre o peso de cada atributo revela a divisão de influências. Serão atribuídos os valores: R = 40; A = 30; S = 20; P = 10. A seguir os cálculos de cada integrante.

- Cliente = 1R, 2A, 4P = $1 \times 40 + 2 \times 30 + 4 \times 10 = 140$.
- G. Projeto = 2R, 2A, 1S, 4P = $2 \times 40 + 2 \times 30 + 1 \times 20 + 4 \times 10 = 200$.
- G. Sistema = 2R, 1A, 3S, 3P = $2 \times 40 + 1 \times 10 + 3 \times 20 + 3 \times 10 = 180$.
- A. Sistema = 2R, 1A, 3S, 3P = $2 \times 40 + 1 \times 10 + 3 \times 20 + 3 \times 10 = 180$.
- DBA = 1R, 1A, 1S, 6P = $1 \times 40 + 2 \times 20 + 6 \times 10 = 150$.
- Programador = 1R, 1A, 2S, 5P = $1 \times 40 + 1 \times 30 + 2 \times 20 + 5 \times 10 = 160$.

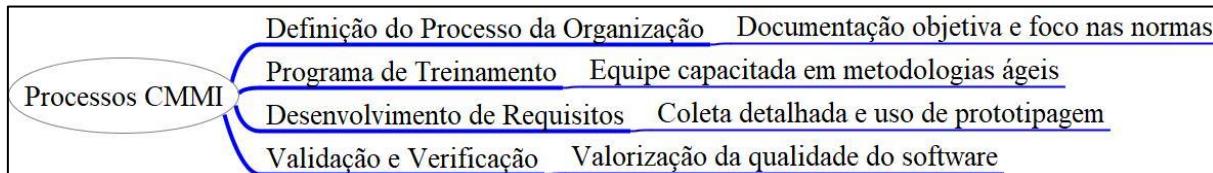
Como esperado, os cargos gerenciais possuem uma influência maior sobre o projeto. Para a equipe, a distribuição de carga está justa, sem sobrecarga.

3.1.3 Gerência do processo com CMMI.

O CMMI é um modelo de maturidade que determina boas práticas para se cumprir durante o desenvolvimento de software. Basicamente, divide em cinco níveis os processos recomendados, e à medida da adesão desses processos, uma organização é qualificada no nível correspondente.

Visando atingir o nível 3, que representa um processo de desenvolvimento de software “definido”, a SofTech Desenvolvimento busca cumprir essencialmente os seguintes itens:

Figura 3 - Processos focados pela organização e como realizá-los

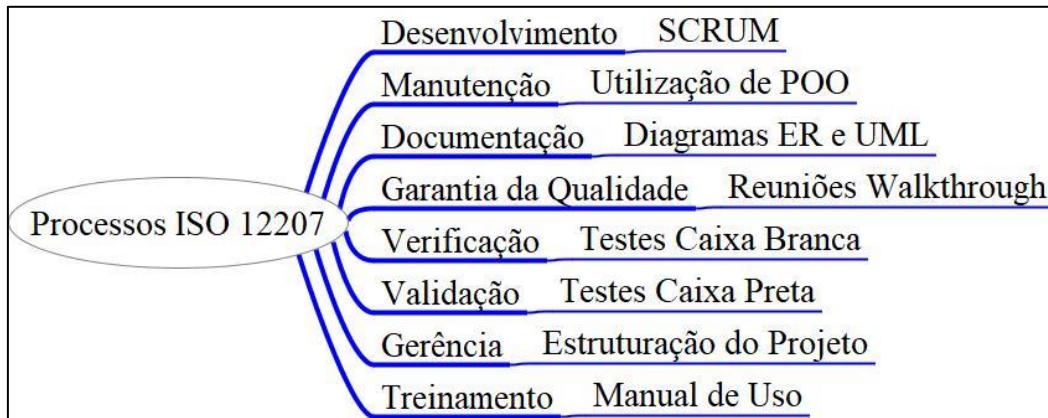


Fonte: Autor, 2021.

3.1.4 Qualidade do ciclo de vida do software com ISO 12207.

International Organization for Standardization (ISO) é uma entidade de padronização. A ISO 12207 busca organizar o momento de desenvolvimento do software, ou seja, seu ciclo de vida, de modo que todos os envolvidos possam compreender sua estrutura e desenvolvimento. Para isso, define processos que podem ser classificados em Fundamentais, De Apoio e Organizacionais. Abaixo, constam os principais processos dessa norma que esse projeto busca realizar.

Figura 4 – Processos da ISO 12207 e como o projeto os aborda essencialmente



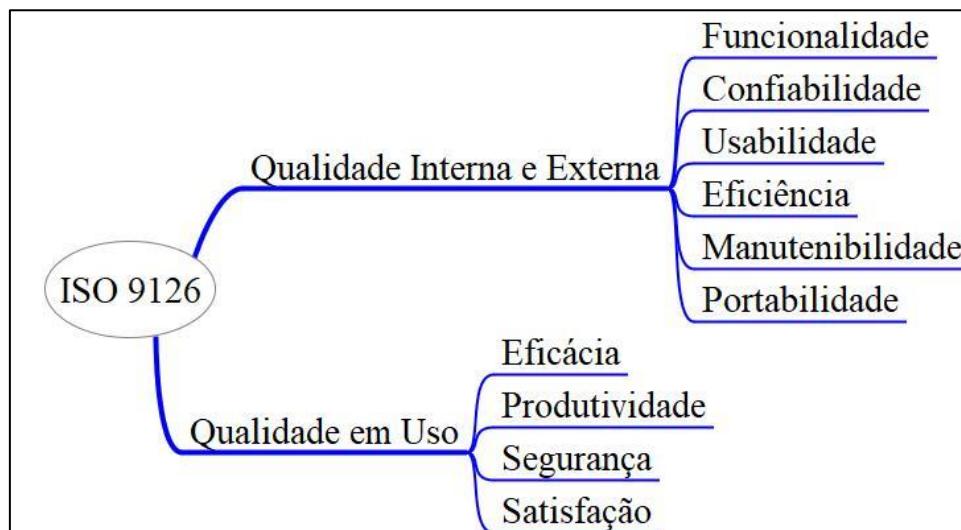
Fonte: Autor, 2021.

3.1.5 Garantia da qualidade funcional com ISO 9126.

A ISO 9126 é uma norma internacional que visa a qualidade do produto de software. Segundo Molinari (2003), a norma ISO 9126 define qualidade de software como a totalidade de características de um produto de software que o torne capaz de satisfazer necessidades implícitas e explícitas. Necessidades explícitas estão relacionadas aos responsáveis pelo desenvolvimento, enquanto as implícitas são subjetivas ao usuário.

A definição da norma ISO 9126 aborda essas características como fatores da qualidade interna, qualidade externa e qualidade em uso do produto de software. Essas características também podem representar métricas da qualidade de software. As métricas que o projeto deverá considerar são:

Figura 5 - Características da ISO 9126 para o projeto



Fonte: Autor, 2021.

Essas características ou métricas serão novamente abordadas adiante nas seções de Requisitos Não Funcionais e Análise da Interface.

3.1.6 Gestão da equipe para qualidade do projeto

O Programa 5S contém um conjunto de técnicas simples, que podem mudar humor e atitude dos colaboradores de uma empresa, influenciando o desempenho em projetos. Ele é valorizado e aplicado pela SofTech Desenvolvimento. Resumidamente, esse programa utiliza de cinco palavras em japonês e alguns de seus significados para agrupar seus conceitos a serem aplicados. O quadro abaixo relaciona de forma simplificada esses conceitos:

Quadro 3 - Conceitos do Programa 5S

Palavra	Significado	Reflexão a ser feita	Consequências
Seiri	Organização	Reutilização	Mais compartilhamento
Seiton	Classificação	Visão espacial	Melhor planejamento
Seisou	Zelo	Evitar sujar ambientes	Valorização do simples
Seiketsu	Higiene	Bons hábitos	Felicidade
Shitsuke	Compromisso	Respeito	Responsabilidade

Fonte: Autor, 2021.

3.2 Ferramentas do arcabouço.

As ferramentas envolvem softwares e semelhantes que sejam utilizados no processo de desenvolvimento. A seguir, as ferramentas que foram selecionadas e suas finalidades:

- Linguagem de programação C# para desenvolvimento do sistema desktop.
- Tecnologia ASP.Net e linguagem C# para desenvolvimento do sistema web.
- Linguagem de programação Java para desenvolvimento do sistema mobile (Android)
- Microsoft Word para criação e edição de textos.
- Microsoft PowerPoint para criação de slides e ilustrações.
- Microsoft Visual Studio Community para codificação em C#.
- Draw.io para criação de diagramas Entidade Relacionamento (banco de dados) e diagramas UML.
- BrModelo para análise de diagramas Entidade Relacionamento.
- Microsoft SQL Server, utilizado como Sistema Gerenciador de Banco de Dados (SGBD).
- SQL Server Management Studio, para manipulação do banco de dados.
- StarUML e ArgoUML para criação de diagramas UML, utilizados nas modelagens do sistema.
- FreeMind, para criação de mapas mentais que representem sequências e conexões de determinados elementos.
- Figma, para criação de protótipos de interface.
- Codepen: Para codificação inicial das páginas web.

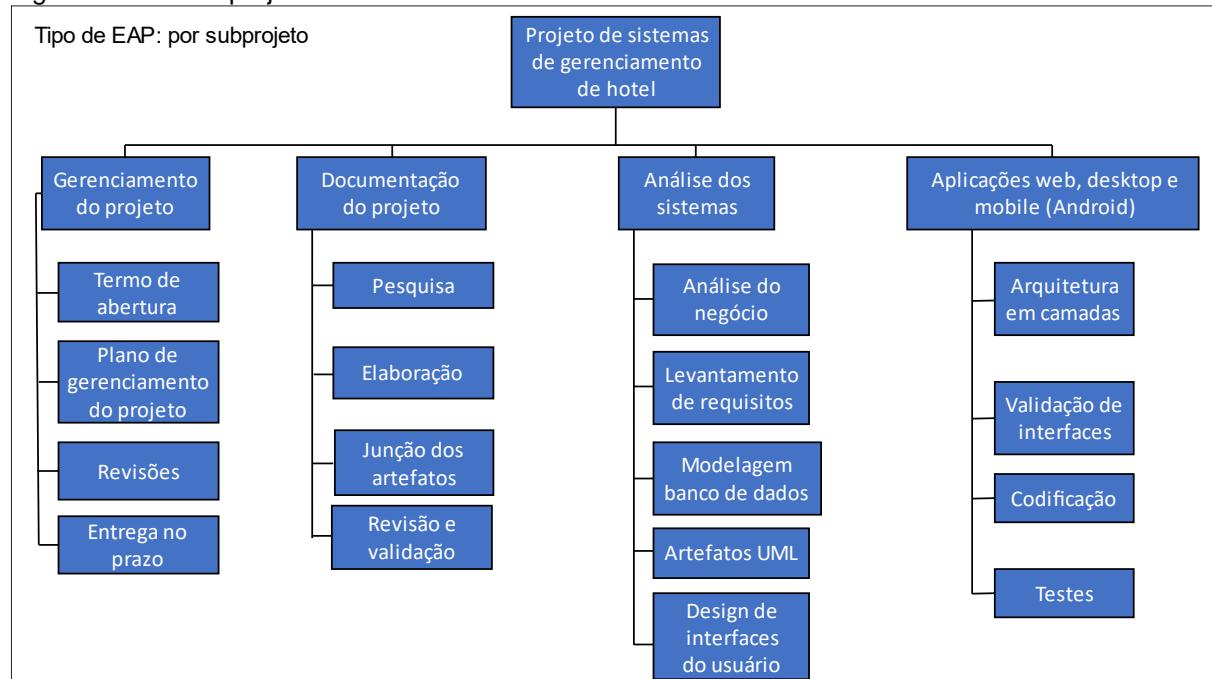
3.3 Gestão do projeto

Projetos, segundo o guia PMBOK, devem cumprir realizar seis definições previamente feitas no momento de entrega: Risco, Cronograma, Orçamento, Escopo, Qualidade e Recursos. A falha em um desses tende a significar fracasso do projeto.

3.3.1 Estrutura Analítica do Projeto (EAP)

Para auxiliar no controle do escopo, foi utilizada a Estrutura analítica do Projeto (EAP). Utilizando o tipo por subprojeto, é feita uma divisão por categoria entre as tarefas que serão realizadas no projeto, e que no fim estão dentro do conjunto principal, localizado no topo, que representa o projeto como um todo.

Figura 6 - EAP do projeto



Fonte: Autor, 2021.

3.3.2 Análise de cronograma

O cronograma consiste em organizar o tempo e a rotina de modo a entregar o resultado no prazo. A figura abaixo ilustra uma tabela com oito principais tarefas que comportam todo o escopo no projeto. O tempo estimado foi reduzido de 1 ano para oito meses, com o objetivo de evitar atrasos.

Figura 7 - Cronograma de tarefas do projeto

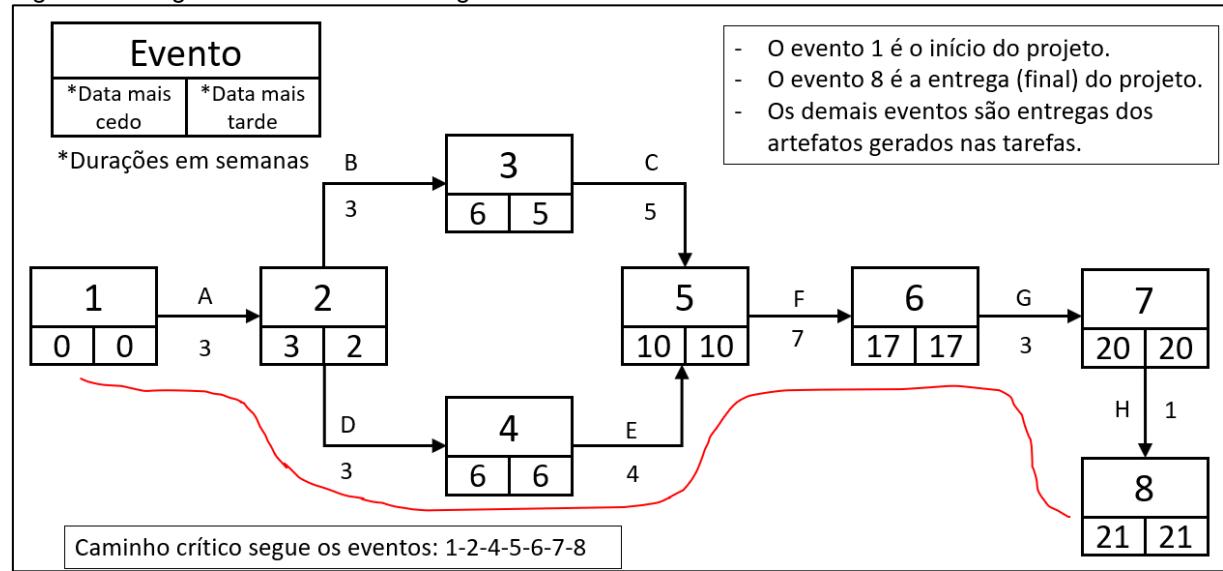
Tempo estimado total: 8 meses (32 semanas)				
Tarefa	Sigla	Precedente	Sucessora	Duração (semanas)
Documentar requisitos	A	-	B	3
Modelar banco de dados	B	A	C	3
Modelar sistema com UML	C	B	F	5
Estudar usuário	D	A	E	3
Criar protótipos de interfaces	E	D	F	4
Programar sistemas	F	C, E	G	7
Revisar e realizar alterações	G	F	H	3
Testar sistemas	H	G	-	1

Fonte: Autor, 2021.

3.3.3 Diagrama de Setas do cronograma com eventos e tarefas.

Para calcular possibilidades de prazo entre tarefas antes mesmo da entrega final, foi utilizado o diagrama de redes no modelo de setas. As legendas da figura detalham o diagrama e seus componentes. O **caminho crítico** obtido da análise é útil para identificar tarefas que merecem mais atenção, pois haver atraso nelas significaria maior risco de comprometer os demais prazos do projeto.

Figura 8 - Diagrama de rede do cronograma



Fonte: Autor, 2021.

4 ESPECIFICAÇÃO DE REQUISITOS

Neste tópico, consta a análise dos requisitos, elementos que representam partes ou funções que estarão presentes no software. O método utilizado para a

obtenção dos mesmos foi a entrevista junto de observação direta, onde o Product Owner observou o estilo do trabalho do hotel, enquanto se hospedava no mesmo durante o período desse processo.

4.1 Regras de negócio

As regras de negócio do hotel, informações que afetam o fluxo de atividades, que serão contempladas pelo sistema estão dispostas a seguir.

Quadro 4 - Regras de negócio para o sistema

ID Regra	Descrição
RN01	Uma reserva de hospedagem só pode ocupar um único quarto.
RN02	Não deve ser permitida a realização de uma reserva sem que o hóspede tenha fornecido seus dados pessoais, nem a finalização caso haja despesas pendentes.
RN03	Hóspedes precisam fornecer CPF ou Passaporte, do contrário não podem efetuar reserva.
RN04	Hóspedes podem fazer inúmeros pagamentos até quitar sua reserva. As formas para pagamento são dinheiro, crédito e débito.
RN05	O pagamento total da estadia deve ser feito antes do check-out.
RN06	Reservas podem ser canceladas gratuitamente antes do início.
RN07	Se o hóspede não fizer check-in, sua reserva será cancelada.
RN08	Independentemente da quantidade de hóspedes, serão coletados apenas os dados pessoais do adulto responsável pela reserva.
RN09	Os valores das diárias de cada quarto, reservas com e sem crianças diferem (será considerado criança todo hóspede menor de 18 anos).
RN10	Os valores cobrados são variáveis de acordo com o setor de finanças do hotel e a temporada de viagens. Devem poder ser alterados.
RN11	O custo do quarto é por dia (diária). Reservas tem despesa obtida com a fórmula de cálculo (1) após o quadro.

RN12	O custo dos hóspedes é por quantidade (de adultos e crianças). Crianças não podem ter preços maiores que adultos.
RN13	O cálculo do total das despesas de uma reserva é feito pela soma das diárias e os consumos existentes.
RN15	Ao se realizar um check-out, o quarto recém liberado deve ser identificado para limpeza.
RN16	Ao identificar irregularidades nos quartos, a camareira deve contatar seu gerente, para a restrição do quarto e início dos reparos.
RN17	O hotel tem obrigação de armazenar dados das reservas e seu hóspede para possíveis ocorrências judiciais. Por isso dados pessoais não são excluídos, mas podem ser desativados.
RN18	Hóspedes inativos não poderão fazer login na conta pelo sistema web. Funcionários inativos não podem fazer login no sistema desktop.

Fonte: Autor, 2021.

Legenda dos cálculos:

- DQuar = Valor da diária do quarto.
- hA = Valor da hospedagem de um adulto.
- hC = Valor da hospedagem de uma criança.
- nA = número de adultos.
- nC = número de crianças.
- dias = quantidade de dias que durará a hospedagem
- totD = total das despesas.

$$(DQuar * dias) + (hA * nA) + (hC * nC) = totD \quad (1)$$

4.2 Requisitos de Usuário e Requisitos de Sistema.

As atividades dos atores estão representadas nos requisitos de usuário, enquanto os detalhes e restrições sobre essas atividades que o sistema deverá conter, encontram-se nos requisitos de sistema.

Quadro 5 - Requisitos de Usuário

Requisito	Descrição
RU01	O recepcionista deve ser capaz de registrar reservas de hospedagem.

RU02	O recepcionista deve ser capaz de iniciar, finalizar ou cancelar a estadia, bem como registrar pagamentos do hóspede.
RU03	O gerente de governança pode verificar e alterar o status dos quartos, para gerir as atividades de governança, como limpeza dos quartos.
RU04	O gerente de governança deve ser capaz de registrar consumos do frigobar realizados pelos hóspedes.
RU05	O gerente de governança deve gerir o estoque de produtos do frigobar.
RU06	O gerente geral deve analisar relatórios referentes aos dados registrados.
RU07	O gerente geral deve ter acesso total aos dados pessoais, podendo verificar, alterar ou desativar.
RU08	O gerente geral deve poder alterar os preços de custo dos serviços do hotel (diárias de quarto e custo de hóspede)
RU09	Os clientes devem poder obter informações sobre o hotel e suas regras.
RU10	Os clientes devem poder acessar seus dados e alterá-los.
RU11	Os clientes devem poder pesquisar e registrar reservas de hospedagem.

Fonte: Autor, 2021.

Quadro 6 - Requisitos de Sistema

Requisito	Descrição
RS01	<p>1.1 No momento de cadastrar os dados do hóspede, deve-se fazer uma checagem para garantir que não ocorra cadastro duplo.</p> <p>1.2 O sistema deve calcular o total de despesas da estadia simultaneamente à inserção dos dados de hospedagem.</p>

RS02	2.1 As reservas devem ser exibidas em lista, possibilitando escolha de qual manipular. 2.3 Os pagamentos devem ser listados junto às demais informações da reserva, com opção para inserir novo. 2.4 O sistema deve impedir que uma estadia com pagamentos pendentes seja finalizada. 2.5 O sistema deve impedir pagamento superior às despesas.
RS03	3.1 Os quartos devem ser exibidos em lista, possibilitando alterar seu status. 3.2 O sistema deve impedir que se altere o status de um quarto com status “ocupado”, significando hóspede ativo.
RS04	4.1 Deve ser exibida lista com os consumos. 4.2 Deve haver campos para adicionar consumo. 4.3 A adição de um consumo deve atualizar as despesas da reserva relacionada.
RS05	5.1 Os produtos devem ser exibidos em lista. 5.2 Deve haver campos para adicionar produto.
RS06	6.1 Dispor relatórios de ocupação atual e financeiro.
RS07	7.1 Os dados pessoais devem ser acessíveis através de pesquisa por CPF. 7.2 Devem haver campos para editar dados. 7.3 Funcionários ativos devem ser apresentados em lista.
RS08	8.1 Os referenciais de custo devem ser exibidos com opção para alterar e salvar mudanças.
RS09	9.1 Deve ser exibido textos, imagens e demais itens necessários para a apresentação do hotel, suas características e regras.
RS11	10.1 Deve haver funcionalidade para criar ou acessar conta. 10.2 Devem ser apresentados seus dados pessoais. 10.3 Deve ser possível alterar os dados pessoais, exceto desativação.
RS10	10.1 Deve haver área na interface com campos de dados para que os clientes possam pesquisar reservas.

	10.2 Para realizar reservas, clientes precisam ter cadastrado seus dados pessoais. 10.3 Clientes devem estar logados para efetuar registro de reserva.
--	---

Fonte: Autor, 2021.

4.3 Requisitos Funcionais.

As funções de processamento de dados relacionadas às atividades do hotel que o sistema deverá realizar. Para o sistema desktop, que conta com todas as funcionalidades acordadas e tem acesso restrito a funcionários, os requisitos funcionais são:

Quadro 7 - Requisitos Funcionais do sistema desktop.

ID Requisito	Descrição
RF01	O sistema desktop deve diferenciar acessos de acordo com o tipo de usuário logado.
RF02	O sistema desktop deve possibilitar o cadastro, pesquisa e alteração de dados de pessoas, sejam hóspedes ou funcionários. 2.1. Recepcionistas podem cadastrar, enquanto somente Gerentes Gerais podem pesquisar e alterar.
RF03	O sistema desktop deve possibilitar registro, pesquisa e manipulação de reservas para hóspedes. 3.1 Reservas devem ser identificadas com diferentes status para possibilitar a manipulação correta.
RF04	O sistema desktop deve possuir rotinas que identifiquem Cancelamento de reservas, Check-in ou Check-out do hóspede.
RF05	O sistema desktop deve possibilitar a adição de pagamentos dos hóspedes referentes às reservas. 5.1 Deve ser possibilitada a adição de indeterminados pagamentos.
RF06	O sistema desktop deve possibilitar a verificação das informações dos quartos e alteração de seu status.
RF07	O sistema desktop deve possibilitar visualização e adição de consumos dos hóspedes durante sua estadia.

RF08	O sistema desktop deve permitir cadastro de produtos do frigobar para controle de estoque.
RF09	O sistema desktop deve fornecer relatórios sobre os dados registrados.
RF10	O sistema desktop deve permitir a alteração dos preços de estadia.

Fonte: Autor, 2021.

Para o sistema web, que servirá como teste para o futuro site completo do hotel, cujo usuários serão hóspedes, os requisitos funcionais são:

Quadro 8 - Requisitos funcionais do sistema web

ID Requisito	Descrição
RF11	O sistema deve conter exibição de informações sobre o hotel.
RF12	O sistema deve possibilitar cadastro de usuário e acesso aos dados de usuário.
RF13	Os sistemas deve possibilitar pesquisa e registro de reservas. 13.1 Enquanto pesquisas são livres, registro de reservas deve exigir login do usuário.

Fonte: Autor, 2021.

Para o sistema mobile em android, seu foco seria em hóspedes recorrentes, por isso, nessa versão inicial, não contém função de cadastro, apenas login e registro de reservas. Hóspedes deveriam fazer seu cadastro no site para então utilizar o aplicativo. Seus requisitos funcionais derivam da versão web, são eles:

Quadro 9 - Requisitos funcionais do sistema mobile

ID Requisito	Descrição
RF12.1	O sistema deve possibilitar acesso e edição dos dados de usuário (hóspede).
RF13.1	Os sistemas deve possibilitar pesquisa e registro de reservas. 13.1 Enquanto pesquisas são livres, registro de reservas deve exigir login do usuário.

Fonte: Autor, 2021.

4.4 Requisitos Não Funcionais.

Os requisitos não funcionais são compostos por restrições de execução do sistema e métricas da qualidade do mesmo. Segundo principalmente as métricas essenciais definidas pela ISO 9126, os requisitos não funcionais que o sistema deverá contemplar são:

1. RNF01 – Confiabilidade.
 - a. O código deve conter captura de falhas para evitar erros de execução.
 - b. O tempo de resposta para erros de execução deve ser de no máximo vinte segundos.
 - c. O sistema deve identificar os diferentes tipos de usuário no momento de login, permitindo níveis de acesso diferentes.
2. RNF02 – Eficácia.
 - a. O sistema deve passar por testes de validação das atividades com os usuários finais.
 - b. As atividades devem ser realizáveis com no mínimo 90% de acurácia.
3. RNF03 – Eficiência.
 - a. O código do software deverá abstrair as classes o máximo possível, para evitar redundâncias, utilizando para isso a estrutura do banco de dados como suporte.
 - b. Diferentes tipos de validações deverão ser agrupadas em uma classe estática.
 - c. Os usuários deverão ter a possibilidade de realizar uma determinada tarefa em no máximo cinco minutos.
4. RNF04 – Funcionalidade.
 - a. O software deverá atender totalmente aos seus requisitos de usabilidade.
 - b. A navegação entre atividades deve ser garantida e facilmente perceptível.
5. RNF05 – Manutenibilidade.
 - a. Uso de Orientação a Objetos e padrão de projeto Modelo, Visão e Controle (MVC) para promover estrutura sólida e ágil para eventuais manutenções.
 - b. Uso do framework .Net para proporcionar uma interoperabilidade acessível a outros sistemas operacionais caso necessário futuramente.
6. RNF06 – Modularidade.
 - a. A linguagem de programação a ser utilizada para o sistema Desktop será C#.

- b. A linguagem de programação a ser utilizada para o sistema web será C#, utilizando tecnologia ASP.Net, HTML e CSS.
 - c. A linguagem de programação a ser utilizada para o sistema Mobile será Java, visando os sistemas operacionais Android (90% dos clientes).
 - d. O Sistema Gerenciador de Banco de Dados a ser utilizado será o Microsoft SQL Server.
 - e. O banco de dados deverá ser hospedado em um servidor Windows Server.
 - f. A versão desktop deve ser executada em Sistema Operacional Windows (preferencialmente Windows 10).
 - g. A versão mobile deve ser executada em plataformas Android.
7. RNF06 – Portabilidade.
- a. Implementar responsividade da versão web do sistema para garantir seu uso em diferentes ambientes com diferentes telas.
 - b. Reaproveitar da documentação do sistema desktop as classes necessárias para os sistemas web e mobile.
8. RNF07 – Produtividade.
- a. Priorizar interfaces de fácil entendimento a interfaces complexas, para evitar confusões e demora na realização de atividades.
 - b. Proporcionar sequências de ações pequenas e objetivas para que o usuário possa se acostumar aos padrões das atividades rapidamente.
9. RNF08 – Segurança.
- a. Proteger o acesso ao sistema desktop para que apenas funcionários do hotel possam acessá-lo por meio de um login próprio.
 - b. Aderir à Lei Geral de Proteção de Dados (LGPD) para garantir a segurança de dados potencialmente sensíveis.
 - c. Fazer análise de integridade de dados para distribuir corretamente a visibilidade dos mesmos entre os diferentes usuários do sistema desktop.
 - d. Proteger o acesso a dados e reservas pessoais nos sistemas web e mobile por meio de login em conta do usuário.
10. RNF09 – Satisfação.
- a. Proporcionar aos usuários a criação de um modelo mental apropriado do sistema por meio de interfaces objetivas, evitando assim a frustração durante o uso.

- b. Utilizar mensagens de alerta ou pop-up para comunicar ao usuário o que houve após sua realização de ações.

11.RNF10 – Usabilidade.

- a. Realizar estudo de interface por meio de estudo do usuário, criação de documento de visão e requisitos de usabilidade (estão presentes adiante).
- b. Realizar testes interativos com protótipos de baixa ou alta fidelidade para obter feedbacks.

4.4.1 Adaptação à LGPD.

No âmbito da hotelaria, é inevitável a necessidade de dados pessoais dos hóspedes, pois é preciso utilizar, arquivar e enviar a FNRH (Ficha Nacional de Registro de Hóspedes) ao governo, por questões de turismo. Entretanto, nem todos os dados requisitados por esse documento são essenciais para o software, e nesse contexto é aplicada a LGPD. A lei define regras para organizações que coletam e manipulem dados de cidadãos que estejam em território nacional. Resumidamente, é crucial que o possidente dos dados tenha conhecimento da coleta e das possibilidades de utilização dos mesmos. A LGPD possui 10 princípios que devem ser respeitados quando se trabalhar com dados pessoais. A seguir, constam os métodos definidos para o cumprimento desses princípios nos softwares desse projeto:

1. **Finalidade** – Por qual motivo os dados são necessários.

O armazenamento dos dados digitalmente melhora a produtividade geral. Colaboradores realizarão suas tarefas com mais velocidade, segurança e eficácia, enquanto a implantação dos sistemas acessíveis aos hóspedes irá permiti-los contatar e usufruir melhor dos serviços do hotel.

2. **Adequação** – Como os dados serão utilizados em cumprimento da finalidade.

Os dados serão anexados aos registros de reservas de hospedagem, permitindo a relação exata de serviços à reserva e hóspede corretos. Servirão também para a análise gerencial dos administradores do hotel, para que possam melhorar seus serviços de acordo com o estilo de seus hóspedes.

3. **Necessidade** – Devem ser coletados e utilizados o mínimo de dados possível.

Os únicos dados pessoais coletados serão os identificadores e de contato, utilizados para a distinção entre indivíduos e contato com os mesmos.

4. **Transparência** – Informações aos possidentes dos dados sobre seu tratamento.

Os sistemas web e mobile contarão com informações a respeito do tratamento de dados logo nas páginas iniciais, facilitando o conhecimento sobre. Os

recepção e os recepcionistas também descreverão como é feito esse tratamento para atendimento no local.

5. Não discriminação – Impossibilidade de utilização para tais fins.

No sistema desktop, os dados só serão acessíveis nas conexões dos serviços, impossibilitando o livre uso ou estímulo de exposição. Quanto aos sistemas web e mobile, apenas o usuário possuidor dos dados poderá acessá-los, não havendo funcionalidade de exposição na própria aplicação.

6. Livre acesso – Acesso fácil e gratuito aos dados por seus possuidores.

O possuidor dos dados poderá contatar o hotel para realizar consulta de seus dados, ou acessá-los de forma independente através dos sistemas web e mobile.

7. Qualidade dos dados – Permissão aos possuidores para que atualizem livremente seus dados.

O possuidor dos dados poderá contatar o hotel para realizar atualização ou inativação de seus dados, ou realizá-los ele mesmo através dos sistemas web e mobile.

8. Segurança – Garantias de que os dados estarão seguros.

Os dados estarão armazenados em sistemas desenvolvidos com documentação, testes e atenção à qualidade, com uma estrutura sólida e segura.

O sistema desktop estará funcional em uma rede privada local no hotel, enquanto os servidores de suporte web e mobile contarão com proteções adequadas.

Todas as áreas dos sistemas que envolvam exibição de dados pessoais estarão sob necessidade de login.

9. Prevenção – Garantia de que os colaboradores da organização têm conhecimento sobre a importância da proteção dos dados.

Será realizado treinamento completo para uso do software desktop, com tópico focado na segurança de dados pessoais e respeito ao acesso somente quando necessário.

10. Responsabilização – Meios para comprovação da adesão à LGPD.

A documentação dos softwares contém referência à lei e eventuais checagens dos códigos comprovarão o uso de padrões e boas práticas para proteção dos dados.

5 ELABORAÇÃO DO BANCO DE DADOS

Antes da modelagem do sistema com UML, será abordada a modelagem do banco de dados. Foi dada preferência ao banco de dados pois, mesmo que o sistema

venha a mudar, os dados utilizados serão muito provavelmente os mesmos, portanto a estrutura do banco de dados possui fundamental importância.

A UML prega a utilização da criação dos diagramas de casos de uso como primeira etapa de um processo metodológico de desenvolvimento de aplicações. Porém, é um cacoete derivado dos conceitos de análise estruturada e análise essencial, em que destaque e importância maior sempre foram dados aos processos e não aos dados. Voltamos a lembrar que dados são sempre mais estáveis do que processos. (MACHADO, 2020, p. 64).

O minimundo é uma forma de abstrair o mundo real, pensando essencialmente em dados importantes que devem ser armazenados para o cumprimento do objetivo da abstração. A seguir, representação do negócio do hotel com os dados que devem ser armazenados:

- Hóspedes precisam informar seus dados pessoais (nome, data de nascimento, sexo, e-mail, telefone e CPF ou Passaporte) para que possam se hospedar. É preciso também controlar o status de um hóspede (ativo, inativo).
- Hóspedes podem realizar reservas, que contém dados de hospedagem (data de entrada, data de saída, tipo de quarto escolhido, quantidade de adultos e de crianças), status (reservada, iniciada, finalizada, cancelada) e o total das despesas, que pode ser pago em várias vezes até o fim da hospedagem. Consequentemente, deve ser feito o controle desses pagamentos, com o valor pago, momento e forma de pagamento.
- Quanto à governança do hotel, os quartos devem ser identificados com seu número, status e deve estar conectado à um tipo de custo.
- Os custos da hospedagem advém das diárias do quarto, custo por hóspede e possíveis consumos. Os tipos de custo garantidos (diária, hóspede) são armazenados com nome do tipo e preço.
- Os consumos de produtos dos hóspedes possuem reserva relacionada, quantidade e data e hora do consumo. Os próprios produtos por sua vez, possuem nome, data de validade e quantidade em estoque.
- Os funcionários do hotel são cadastrados com os mesmos dados pessoais dos hóspedes (herança), com a adição do salário, cargo e status como funcionário (ativo, inativo).
- Hóspedes poderão ter contas de usuário que servirá para ambos os sistemas web e mobile, assim como funcionários terão login para o sistema desktop. Um

login deverá ter nome de usuário e senha, e poderá estar conectado a um hóspede ou a um funcionário.

- Para garantir controle de mudança, as entidades hóspede, funcionário, conta de login e tipo de custo, deverão ter um atributo para a data de atualização.

Recordando que hotéis têm o dever de enviar dados de hospedagem por meio da FNRH ao governo para fim de análise do turismo, e que devem também manter registros próprios dessas hospedagens, a exclusão de dados referentes à reservas, não deve ser possibilitada. Para garantir a proteção de dados sensíveis, o uso do atributo de status permite que somente usuários com permissão total (nesse caso, Gerente Geral) possam acessá-los livremente. Os demais usuários desktop só visualizarão dados pessoais que estejam ativos.

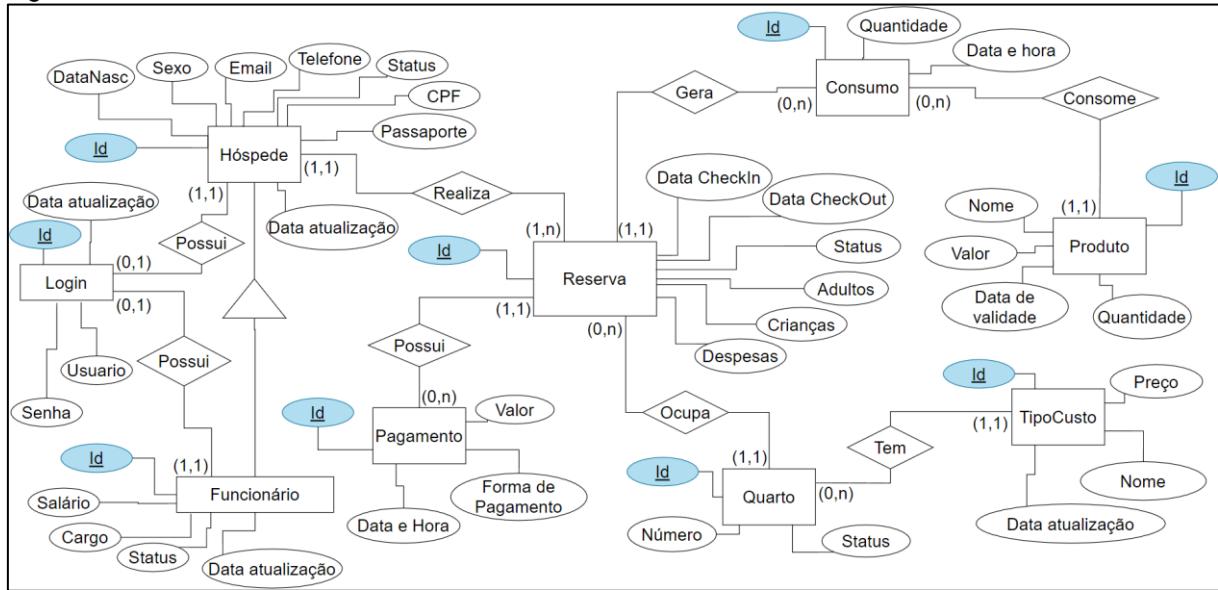
5.1 Diagramas MER

O modelo relacional de banco de dados faz uma espécie de alusão à Teoria dos Conjuntos da matemática. É uma abordagem que permite ao usuário final (cliente) entender a estrutura do banco de dados. O Modelo Entidade Relacionamento (MER) é uma forma de organizar os dados, onde as Entidades são elementos do mundo real que terão dados armazenados como Atributos da entidade, e podem ou não ocorrer relacionamentos entre as entidades.

5.1.1 Modelo Conceitual

O objetivo do modelo conceitual do banco de dados é representar a estrutura que será criada em um diagrama simples, para que o usuário final possa visualizar e compreender o banco de dados.

Figura 9 - Modelo conceitual do banco de dados

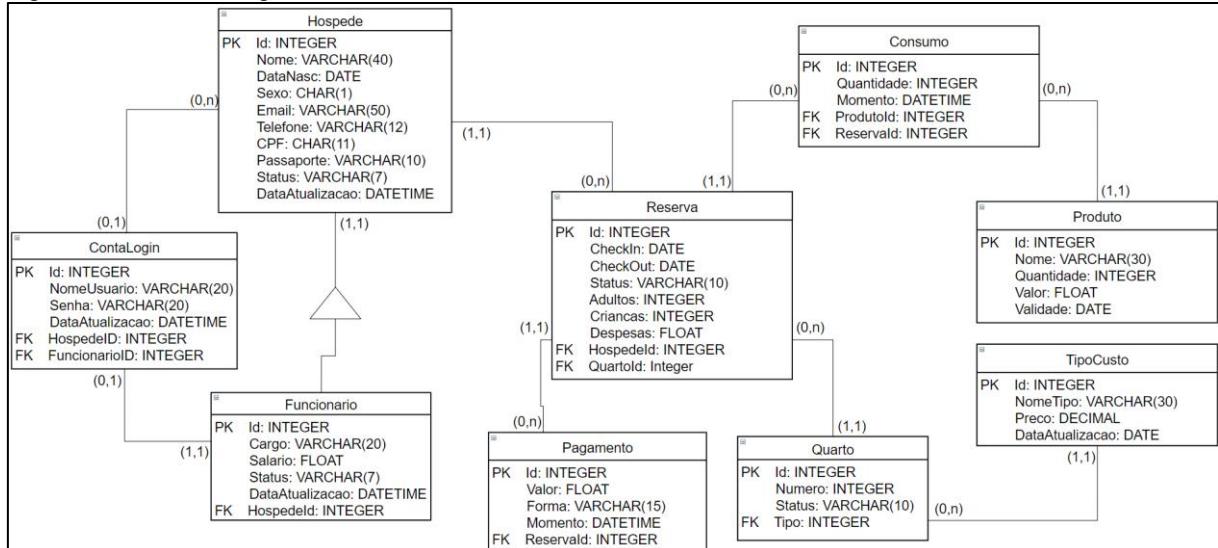


Fonte: Autor, 2021.

5.1.2 Modelo Lógico

O objetivo do modelo lógico do banco de dados é exibir as entidades com todos os atributos que deverão ser criados no esquema já com seus devidos tipos de dado e chave, para que possa ser feita uma análise primária e evitar criações de esquemas incorretos.

Figura 10 - Modelo lógico do banco de dados



Fonte: Autor, 2021.

5.1.3 Modelo Físico

O modelo físico do banco de dados é a sua efetiva criação em um SGBD, por meio de linguagem Structured Query Language (SQL). Os códigos SQL de criação, inserção e buscas do projeto estão dispostos como apêndice do documento.

5.2 Planilhas de testes

Cumprindo um dos processos do CMMI descritos anteriormente e visando a garantia da qualidade do software como propõe a ISO 9126, serão feitos processos de verificação, para garantir que o software foi construído da maneira correta, e validação, para conferir se ele cumpre seu propósito sendo útil aos usuários finais. A seguir, serão apresentados testes para verificar a implementação, seguidos de testes do banco de dados, utilizando uma carga inicial fictícia.

5.2.1 Verificação do código por meio de Casos de Teste.

Casos de Teste servem para verificar a qualidade interna do software, utilizando requisitos funcionais ou outros requisitos necessários como referência para o teste. A especificação dos casos de teste contém simulações da sequência de ações que o algoritmo a ser implementado deve seguir.

Quadro 10 - Casos de teste para verificação

Caso de teste	Referência	Especificação
CT01	RF01, RF01.1, RF01.2	CT01 – Acessos do sistema desktop: 1. Inserir dados de login. 2. Se dados forem inválidos, mostrar mensagem e pedir inserção dos dados novamente. 3. Se dados forem válidos, prosseguir para tela correspondente. 4. Tentar acessar controle de preços. 5. Se usuário for gerente geral, permitir acesso. 6. Se não, impedir acesso.
CT02	RF02, RF02.1, RF03, RF03.1	CT02 – Realização de nova reserva: 1. Preencher dados. 2. Registrar nova reserva. 3. Sistema testa cadastro do hóspede

		<p>4. Se não for cadastrado, sistema efetua cadastro do hóspede.</p> <p>5. Se já for cadastrado, sistema atualiza os dados.</p> <p>6. Sistema registra a nova reserva.</p> <p>7. Ir para mapa de reservas e verificar se a nova reserva aparece na lista.</p>
CT03	RF04	<p>CT03 – Manipulação de reserva:</p> <p>1. Acessar mapa de reservas.</p> <p>2. Acessar uma reserva específica.</p> <p>3. Realizar manipulação (cancelamento, check-in, check-out).</p> <p>4. Verificar se os dados da reserva foram atualizados corretamente.</p>
CT04	RF05, RF05.1	<p>CT04 – Novo pagamento:</p> <p>1. Acessar mapa de reservas.</p> <p>2. Acessar uma reserva específica.</p> <p>3. Inserir dados do pagamento.</p> <p>4. Sistema valida se novo valor somado aos já existentes não ultrapassa total das despesas da reserva.</p> <p>5. Se for inválido, exibir mensagem.</p> <p>6. Se for válido, adicionar o pagamento.</p> <p>7. Verificar se os dados foram atualizados corretamente.</p>
CT05	RF06	<p>CT05 – Gerência dos quartos:</p> <p>1. Acessar visualização dos quartos.</p> <p>2. Alterar status de um quarto.</p> <p>3. Verificar se a alteração ocorreu corretamente.</p>
CT06	RF07	<p>CT06 – Adição de consumo do hóspede:</p> <p>1. Acessar tela de consumos.</p> <p>2. Preencher dados de novo consumo.</p> <p>3. Adicionar novo consumo.</p> <p>4. Acessar tela de produtos.</p>

		<p>5. Verificar se quantidade em estoque do produto diminuiu.</p> <p>6. Acessar reserva relacionada no mapa de reservas.</p> <p>7. Verificar se total das despesas foi alterado devido ao consumo do hóspede.</p>
CT07	RF08	<p>CT07 – Adição de produto no estoque:</p> <p>1. Acessar tela de produtos</p> <p>2. Cadastrar novo produto.</p> <p>3. Acessar tela de consumos.</p> <p>4. Verificar se novo produto está disponível para seleção em um novo consumo.</p>
CT08	RF09	<p>CT08 – Relatórios da gerência:</p> <p>1. Acessar tela de relatórios.</p> <p>2. Verificar se relatórios exibidos estão com informações corretas e atualizadas.</p>
CT09	RF02, RF02.1	<p>CT09 – Manipulação de dados pessoais:</p> <p>1. Acessar tela de hóspedes ou funcionários.</p> <p>2. Pesquisar por um registro.</p> <p>3. Inserir novos dados.</p> <p>4. Salvar alterações.</p> <p>5. Verificar se dados foram atualizados.</p>
CT10	RF10, RF10.1, RF10.2	<p>CT10 – Alteração de preços de hospedagem:</p> <p>1. Acessar controle de preços.</p> <p>2. Editar preços.</p> <p>3. Salvar alterações.</p> <p>4. Acessar nova reserva.</p> <p>5. Verificar se novos preços são usados.</p>
CT11	RF11	<p>CT11 – Apresentação de informações:</p> <p>1. Acessar página inicial do sistema web.</p> <p>2. Verificar se informações estão corretas.</p>
CT12	RF12	<p>CT12- Acesso à conta de usuário:</p> <p>1. Acessar página web de login.</p>

		2. Inserir nome de usuário e senha. 3. Sistema valida dados de login. 4. Se forem incorretos, exibir mensagem. 5. Se forem corretos, acessar conta de usuário.
CT13	RF13, RF13.1, RF13.2	CT13 – Registro de reserva: 1. Acessar página de reservas. 2. Inserir dados e pesquisar reserva. 3. Realizar reserva dentre os resultados da pesquisa. 4. Se não houver usuário logado, pedir login. 5. Acessar conta do usuário e mostrar mensagem de registro.

Fonte: Autor, 2021.

5.2.2 Testes do banco de dados

As principais buscas por dados a serem utilizadas no sistema foram utilizadas para os testes a seguir.

Quadro 11 - Casos de teste do sistema

Teste	Especificação e Resultado																																			
	Objetivo: Obter dados do funcionário em ordem alfabética.																																			
01	<table border="1"> <thead> <tr> <th></th> <th>FUNCIONARIO</th> <th>NOME</th> <th>CARGO</th> <th>SALARIO</th> <th></th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Bruno Tavante</td> <td>Gerente</td> <td>4000.00</td> <td></td> <td></td> </tr> <tr> <td>3</td> <td>Diego Rocha</td> <td>Recepção</td> <td>2500.00</td> <td></td> <td></td> </tr> <tr> <td>2</td> <td>Fabiana Souza</td> <td>Governança</td> <td>3000.00</td> <td></td> <td></td> </tr> </tbody> </table>							FUNCIONARIO	NOME	CARGO	SALARIO		1	Bruno Tavante	Gerente	4000.00			3	Diego Rocha	Recepção	2500.00			2	Fabiana Souza	Governança	3000.00								
	FUNCIONARIO	NOME	CARGO	SALARIO																																
1	Bruno Tavante	Gerente	4000.00																																	
3	Diego Rocha	Recepção	2500.00																																	
2	Fabiana Souza	Governança	3000.00																																	
	Objetivo: Obter reservas agendadas ou iniciadas.																																			
02	<table border="1"> <thead> <tr> <th>ID</th> <th>HOSPEDE</th> <th>CHECKIN</th> <th>CHECKOUT</th> <th>DESPEZAS</th> <th>QUARTO</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Peter Parker</td> <td>2021-11-25 00:00:00.000</td> <td>2021-11-26 00:00:00.000</td> <td>70.00</td> <td>13</td> </tr> <tr> <td>2</td> <td>Harry Thiago Potter</td> <td>2021-11-23 19:02:54.000</td> <td>2021-11-23 19:03:35.900</td> <td>40.00</td> <td>1</td> </tr> <tr> <td>3</td> <td>Harry Thiago Potter</td> <td>2021-11-23 19:05:52.000</td> <td>2021-11-23 19:06:50.593</td> <td>55.00</td> <td>2</td> </tr> <tr> <td>4</td> <td>Mayara Oliveira</td> <td>2021-12-10 00:00:00.000</td> <td>2021-12-11 00:00:00.000</td> <td>40.00</td> <td>1</td> </tr> </tbody> </table>						ID	HOSPEDE	CHECKIN	CHECKOUT	DESPEZAS	QUARTO	1	Peter Parker	2021-11-25 00:00:00.000	2021-11-26 00:00:00.000	70.00	13	2	Harry Thiago Potter	2021-11-23 19:02:54.000	2021-11-23 19:03:35.900	40.00	1	3	Harry Thiago Potter	2021-11-23 19:05:52.000	2021-11-23 19:06:50.593	55.00	2	4	Mayara Oliveira	2021-12-10 00:00:00.000	2021-12-11 00:00:00.000	40.00	1
ID	HOSPEDE	CHECKIN	CHECKOUT	DESPEZAS	QUARTO																															
1	Peter Parker	2021-11-25 00:00:00.000	2021-11-26 00:00:00.000	70.00	13																															
2	Harry Thiago Potter	2021-11-23 19:02:54.000	2021-11-23 19:03:35.900	40.00	1																															
3	Harry Thiago Potter	2021-11-23 19:05:52.000	2021-11-23 19:06:50.593	55.00	2																															
4	Mayara Oliveira	2021-12-10 00:00:00.000	2021-12-11 00:00:00.000	40.00	1																															
	Objetivo: Consumo com produto de reservas iniciadas.																																			
03	<table border="1"> <thead> <tr> <th>RESERVA</th> <th>HOSPEDE</th> <th>PRODUTO</th> <th>MOMENTO</th> <th>TOTAL</th> </tr> </thead> <tbody> <tr> <td>3</td> <td>Harry Thiago Potter</td> <td>Água mineral</td> <td>2021-11-23 19:06:26.743</td> <td>15.00</td> </tr> </tbody> </table>						RESERVA	HOSPEDE	PRODUTO	MOMENTO	TOTAL	3	Harry Thiago Potter	Água mineral	2021-11-23 19:06:26.743	15.00																				
RESERVA	HOSPEDE	PRODUTO	MOMENTO	TOTAL																																
3	Harry Thiago Potter	Água mineral	2021-11-23 19:06:26.743	15.00																																
04	Objetivo: Pagamentos de determinada reserva.																																			

	RESERVA	HOSPEDE	DESPESAS	VALOR	FORMA	MOMENTO	
	2	Harry Thiago Potter	40.00	20.00	DINHEIRO	2021-11-23 19:03:13.630	
	2	Harry Thiago Potter	40.00	15.00	DINHEIRO	2021-11-23 19:03:27.260	
	2	Harry Thiago Potter	40.00	5.00	CRÉDITO	2021-11-23 19:03:34.250	
05		Objetivo: Reservas agendadas e iniciadas.					
		RESERVA	HOSPEDE	QUARTO	CHECKIN	STATUSRES	
		4	Mayara Oliveira	1	2021-12-10 00:00:00.000	RESERVADA	
06		Objetivo: Histórico de reservas do hóspede.					
		NOME	RESERVA	ENTRADA	SAIDA	QUARTO	NUMADULTOS
		Harry Thiago Potter	2	2021-11-23	2021-11-23	1	1
		Harry Thiago Potter	3	2021-11-23	2021-11-23	2	1
		Mayara Oliveira	4	2021-12-10	2021-12-11	1	1
		Peter Parker	1	2021-11-25	2021-11-26	13	2

Fonte: Autor, 2021.

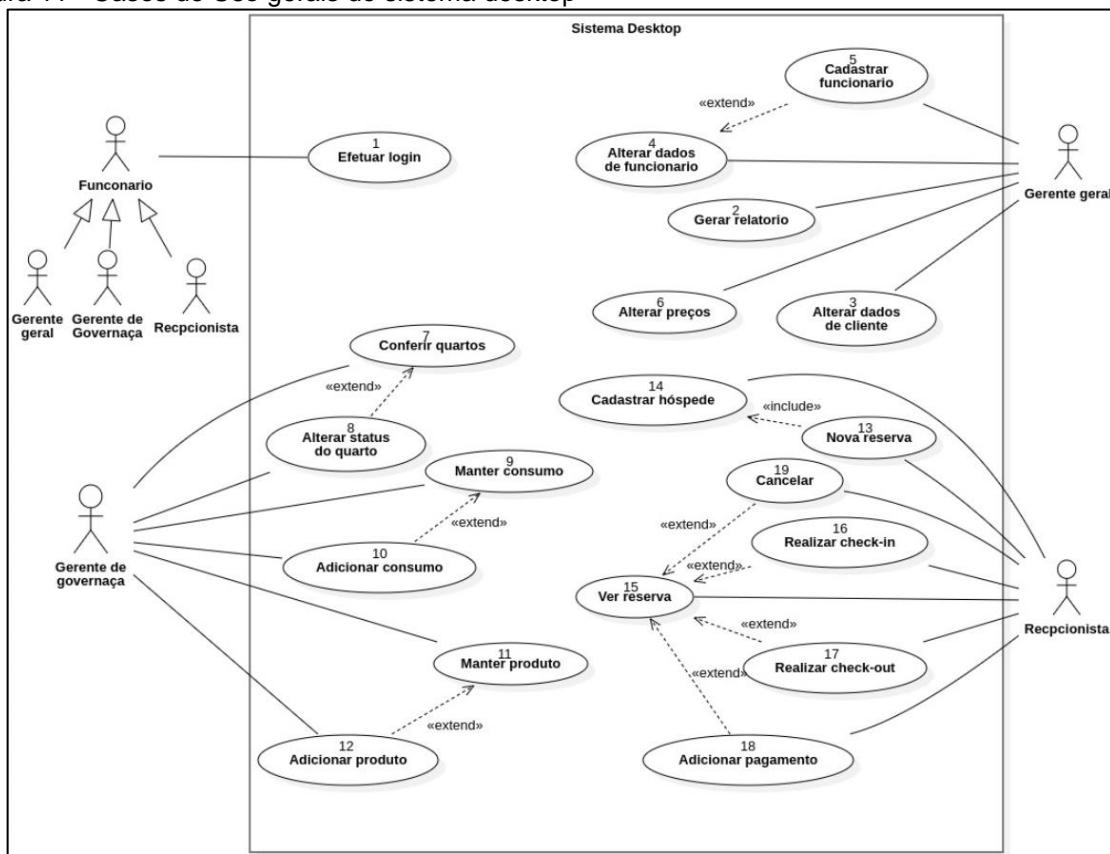
6 MODELAGEM DO SISTEMA

Neste tópico, são utilizados alguns dos diagramas da Unified Modeling Language (UML), ou Linguagem de Modelagem Unificada, para representação de como o software irá adaptar o negócio do cliente, como serão suas funcionalidades e como os atores do sistema irão utilizá-lo. Steve Mellor e Martin Fowler (2005, p. 25) propõem que a UML possui três modos de uso: esboço, projeto e linguagem de programação, sendo esboço o mais comum. Também observam que esboços podem ser utilizados tanto no desenvolvimento, para criação de novos softwares, como na engenharia reversa, a partir de um código pronto para seu melhor entendimento. Naturalmente, neste projeto os diagramas UML serão utilizados no desenvolvimento. “No desenvolvimento, desenha-se um diagrama UML antes de se escrever o código[...].” (FOWLER, 2005, p. 25).

6.1 Diagramas de Casos de Uso.

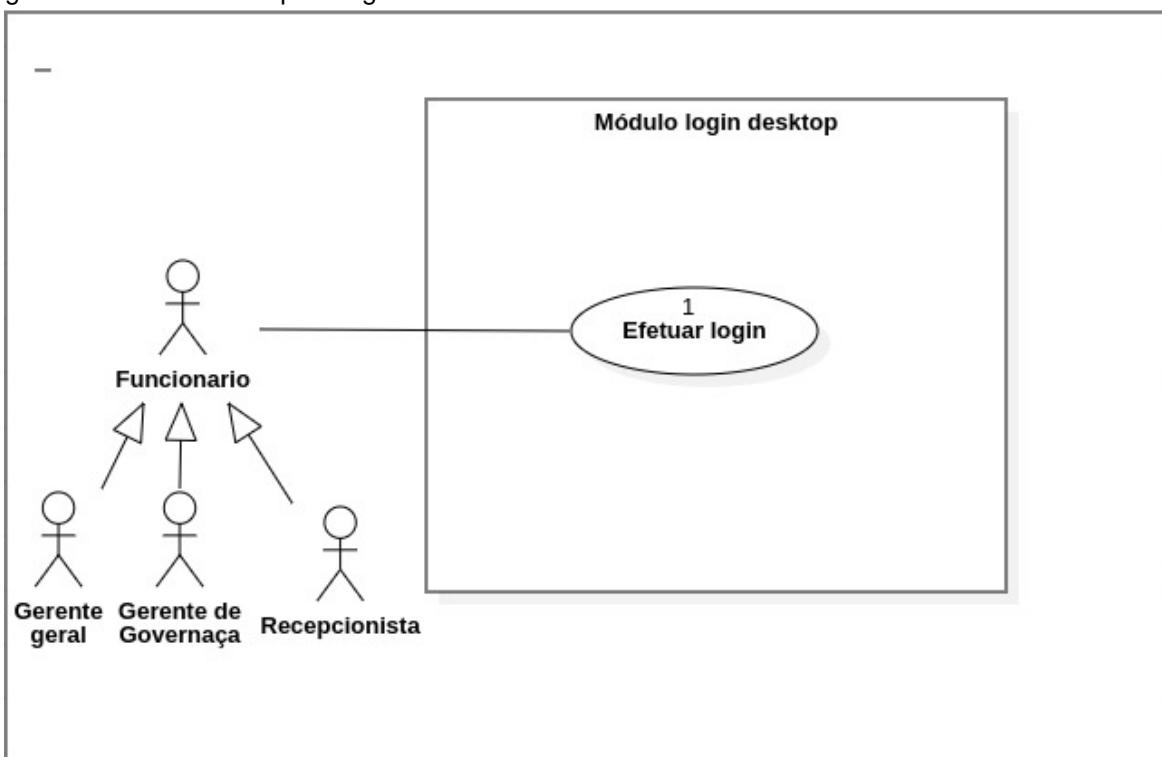
Os diagramas de caso de uso servem para identificar atores (usuários do sistema) e atividades a serem executadas por eles, de modo a demonstrar as restrições de cada ator, e como certas atividades interagem entre si. Os casos de uso a seguir buscam representar os casos em que poderão se utilizar os sistemas desktop e web.

Figura 11 - Casos de Uso gerais do sistema desktop



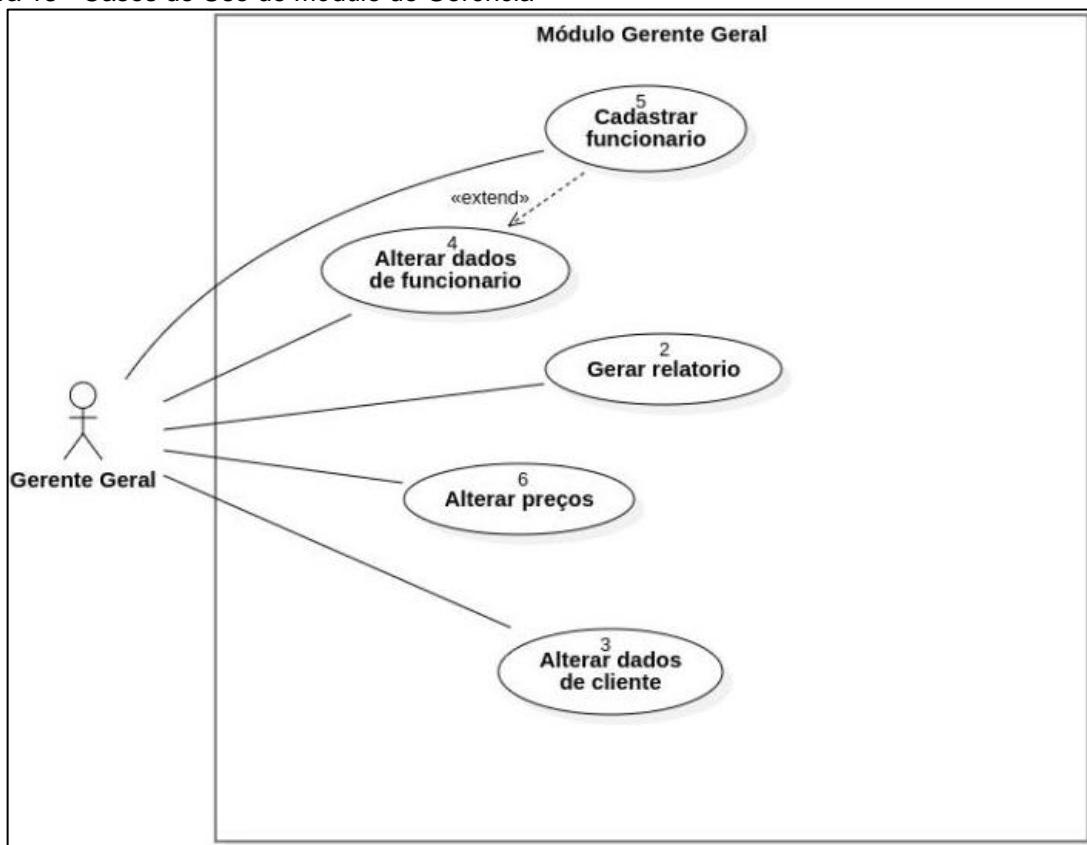
Fonte: Autor, 2021.

Figura 12 - Caso de Uso para login no sistema



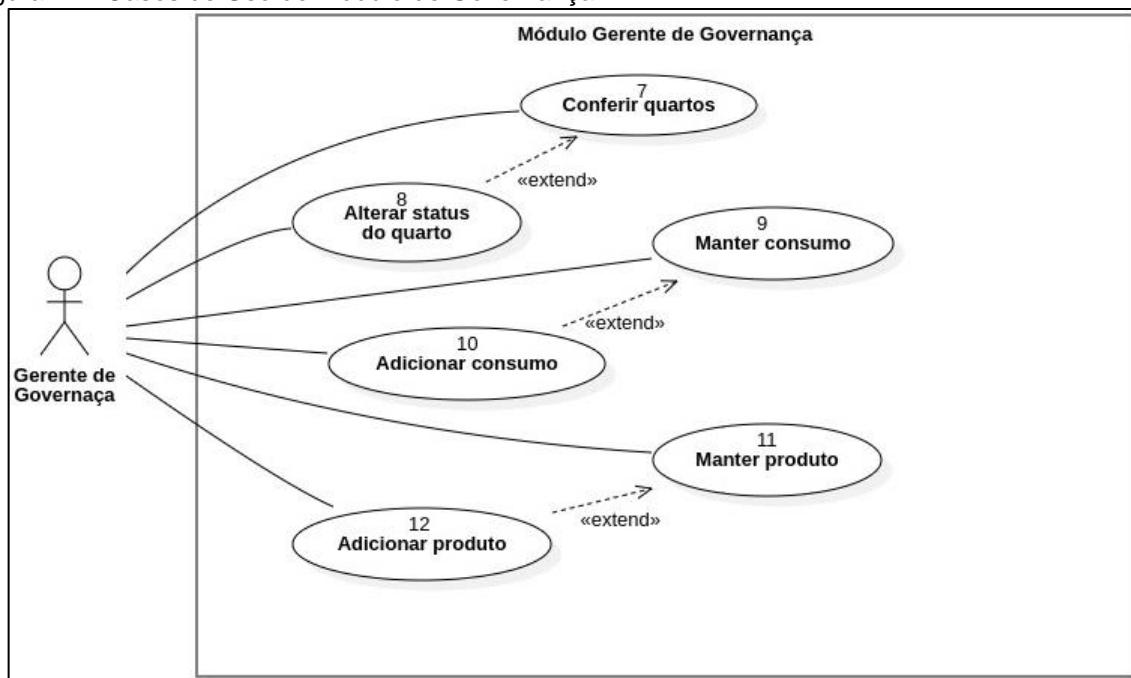
Fonte: Autor, 2021.

Figura 13 - Casos de Uso do módulo de Gerência



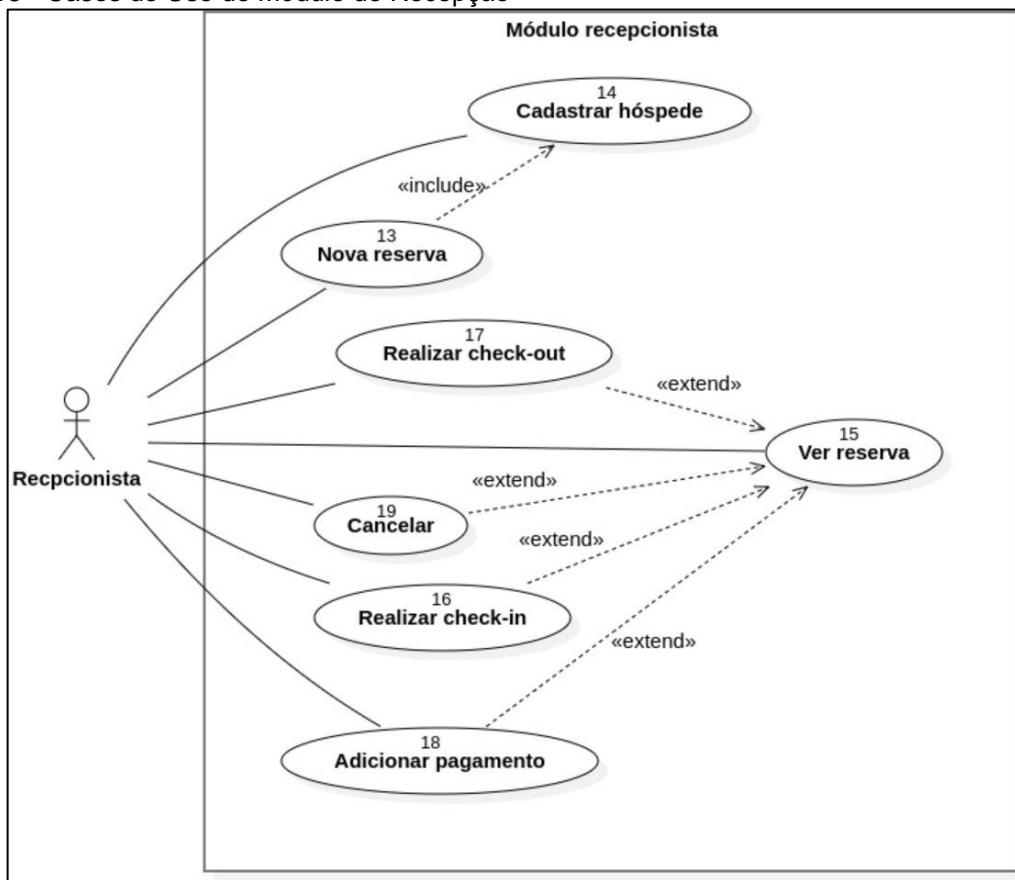
Fonte: Autor, 2021.

Figura 14 - Casos de Uso do módulo de Governança



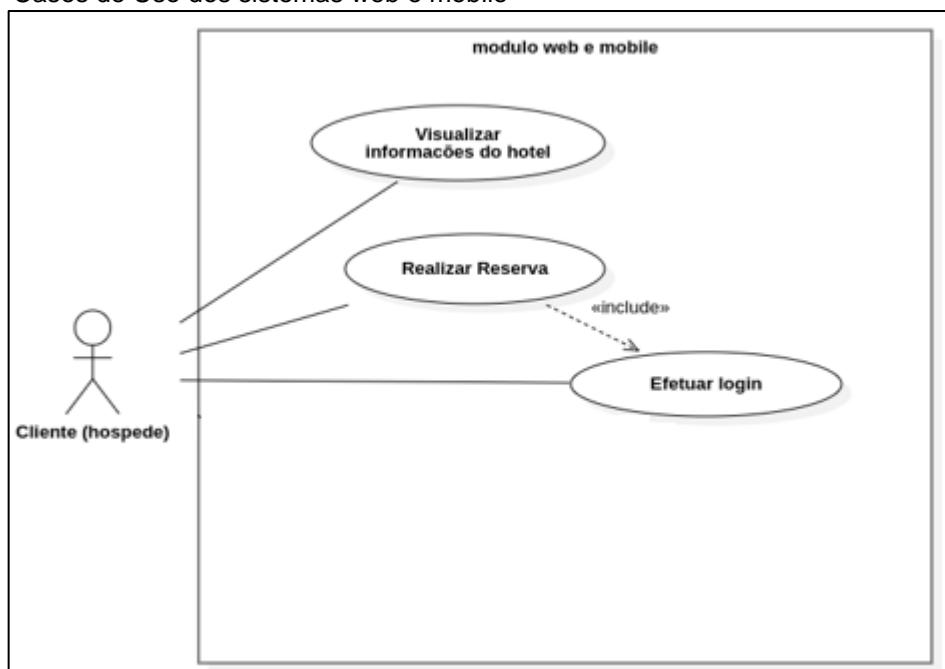
Fonte: Autor, 2021.

Figura 15 - Casos de Uso do módulo de Recepção



Fonte: Autor, 2021.

Figura 16 - Casos de Uso dos sistemas web e mobile



Fonte: Autor, 2021.

6.1.1 Descrição dos Casos de Uso.

As descrições a seguir buscam detalhar como cada caso de uso é de fato utilizado, em uma sequência de ações. É tomada uma determinada ação ou caso de uso para representar um fluxo principal, enquanto outras ações possíveis são dispostas em fluxos alternativos.

Quadro 12 - Descrição de caso de uso de login

Caso de Uso 01	Efetuar login
Atores	Funcionários
Pré-condições	1 Líder da gerencia libera aos funcionários (Gerente e Repcionista) o acesso ao sistema
Pós-condições	2 Logado no sistema
Fluxo Principal de Evento	
1.1 O Funcionário informa login e senha	
1.2 O sistema verifica se o login e a senha são válidos (verifica-se se o login e senha pertencem a uma conta)	
1.3 O sistema registra o início de uma sessão	
Fluxo Alternativo de Eventos	
2 Se o login ou a senha forem inválidos, o sistema exibe uma mensagem (inválido)	

Fonte: Autor, 2021.

Quadro 13 - Descrição de caso de uso relatório

Caso de Uso 02	Gerar relatório
Atores	Gerente geral
Pré-condições	1 O usuário que vai executar tal atividade deve estar logado e ter permissão no sistema para realizá-lo. 2 Haver dados registrados para os relatórios
Pós-condições	não se aplica
Fluxo Principal de Eventos	
1. Gerente geral acessar o sistema	
2. O sistema valida as informações do gerente geral e verifica se tem permissão para visualizar relatórios.	
3. O sistema mostra o tipo de relatório que pode ser emitido.	
4. Sistema gera relatório de hospedagem e finanças gerais.	

Fluxo Alternativo de Eventos
5. Gerente vai especificar o tipo de relatório.
6. O sistema gera o relatório.
7. O sistema exibe o relatório.

Fonte: Autor, 2021.

Quadro 14 - Descrição de caso de uso alterar dados hóspede

Caso de Uso 03	Alterar dados de clientes
Atores	Gerente geral
Pré-condições	1. Haver dados registrados
Pós-condições	Dados atualizados
Fluxo Principal de Eventos	
1 O Gerente geral informa, no caso de “Alteração”. 1.2 Sistema exibe os dados cadastrados e os habilita para edição. 1.2 Gerente altera dados do cliente. 1.3 Cliente confirma a operação realizada. 1.4 O sistema atualiza os dados do cliente.	
Fluxo Alternativo de Eventos	

Fonte: Autor, 2021.

Quadro 15 - Descrição de caso de uso alterar dados funcionário

Caso de Uso 04	Alterar dados de funcionário
Atores	Gerente geral
Pré-condições	Haver dados cadastrados
Pós-condições	Dados atualizados
Extension Points	Cadastrar funcionário
Fluxo Principal de Evento	
1. Gerente geral acessa dados pessoais no módulo gerência. 2. Sistema exibe área de dados e lista dos funcionários. 3. Gerente geral poderá alterar dados de um funcionário específico. 4. Extension point : Gerente geral pode cadastrar funcionário. 5. Gerente geral finaliza as ações e salva.	

Fluxo Alternativo de Eventos
2.1 Sistema exibe a função alterada dos dados de acordo com a opção.
5.1 Dados inseridos são inválidos e alterações não são salvas.

Fonte: Autor, 2021.

Quadro 16 - Descrição de caso de uso cadastrar funcionário

Caso de Uso 05	Cadastrar funcionário
Extends	Alterar funcionário
Extension point	Cadastrar funcionário.
Atores	Repcionista
Pré-condições	não se aplica
Pós-condições	não se aplica
Regras de Negócio	RN01
Fluxo Principal de Evento	
1.	Gerente geral não seleciona um funcionário, mas insere dados nos campos.
2.	O sistema da opção cadastrar funcionário.
3.	Gerente geral solicita dados do funcionário.
4.	Funcionário informa os dados.
5.	Sistema registra os dados do funcionário.
Fluxo Alternativo de Eventos	
5.1	Dados são inválidos e cadastro não ocorre.

Fonte: Autor, 2021.

Quadro 17 - Descrição de caso de uso alterar preços

Caso de Uso 06	Alterar preços
Atores	Gerente geral
Pré-condições	O usuário que vai executar tal atividade deve estar logado e ter permissão no sistema para realizá-lo
Pós-condições	Preços atualizados.
Fluxo Principal de Eventos	
1.	Gerente geral acessar o sistema.
2.	O sistema valida as informações do gerente geral e verifica se tem permissão para Visualizar os Preços.
3.	O sistema mostra os Preços.

4. Gerente geral altera os Preços.
5. O sistema valida e salva os Preços.
Fluxo Alternativo de Eventos
5.1 Preços inseridos são inválidos e não são salvos.

Fonte: Autor, 2021.

Quadro 18 - Descrição de caso de uso conferir quartos

Caso de Uso 07	Conferir quartos
Atores	Gerente de Governança
Pré-condições	Quartos cadastrados no banco de dados.
Pós-condições	Não se aplica.
Regras de Negócio	RN1, RN12
Fluxo Principal de Eventos	
1. Sistema exibe todos os quartos.	
2. Extension Point : Gerente de governança pode alterar status dos quartos.	
Fluxo Alternativo de Eventos	
1.1 Gerente de governança informa os registros dos quartos com tipo e status, filtrando a visualização.	

Fonte: Autor, 2021.

Quadro 19 - Descrição de caso de uso alterar quartos

Caso de Uso 08	Alterar status dos quartos
Extends	Conferir quartos
Extension Point	Alterar status
Atores	Gerente de Governança
Pré-condições	Dados de cada quarto estarem registrados no banco de dados.
Pós-condições	Quartos terão status diferentes.
Fluxo Principal de Eventos	
1. Sistema exibe dados dos quartos	
2. Gerente de governança seleciona “alterar status do quarto”	
4. É escolhido o status (disponível, ocupado, para limpeza ou em reparo)	
5. Gerente de governança seleciona novo status.	
6. Sistema atualiza status.	
Fluxo Alternativo de Eventos	

--

Fonte: Autor, 2021.

Quadro 20 - Descrição de caso de uso manter consumo

Caso de Uso 09	Manter consumo
Atores	Gerente de Governança
Pré-condições	não se aplica
Pós-condições	não se aplica
Regras de Negocio	RN08
Fluxo Principal de Eventos	
1. O gerente acessa a tabela de consumos. 2. Extension point : Sistema da opção de registrar consumo.	
Fluxo Alternativo de Eventos	

Fonte: Autor, 2021.

Quadro 21 - Descrição de caso de uso adicionar consumo

Caso de Uso 10	Adicionar consumo
Atores	Gerente de Governança
Extends	Manter consumo
Extension point	Registrar consumo
Pré-condições	Ter produto no estoque
Pós-condições	Diminui quantidade do produto no estoque
Regras de Negocio	RN08
Fluxo Principal de Eventos	
1. O sistema da opção de registrar consumo. 2. O gerente informa os dados. 3. Sistema registra consumo do cliente.	
Fluxo Alternativo de Eventos	

Fonte: Autor, 2021.

Quadro 22 - Descrição de caso de uso manter produtos

Caso de Uso 11	Manter produtos
Atores	Gerente de Governança

Pré-condições	não se aplica
Pós-condições	não se aplica
Regras de Negocio	RN08
Fluxo Principal de Eventos	
1. O gerente acessa a tabela de produtos. 2. Extension point: Sistema da opção de registrar produto.	
Fluxo Alternativo de Eventos	

Fonte: Autor, 2021.

Quadro 23 - Descrição de caso de uso adicionar produto

Caso de Uso 12	Adicionar produto
Atores	Gerente de Governança
Extends	Manter produtos
Extension point	Registrar produto
Pré-condições	Não se aplica
Pós-condições	Novo produto registrado
Regras de Negocio	RN08
Fluxo Principal de Eventos	
1. O sistema da opção de registrar produto. 2. Gerente informa dados do produto. 3. Sistema cadastrá o produto e gera (Produto cadastrado com sucesso).	
Fluxo Alternativo de Eventos	

Fonte: Autor, 2021.

Quadro 24 - Descrição de caso de uso nova reserva

Caso de Uso 13	Nova reserva
Atores	Repcionista
Pré-condições	1. Ter dados do cliente registrados.
Pós-condições	1. Nova reserva registrada no sistema.
Regra de Negócio	RN02, RN03, RN04
Fluxo Principal de Eventos	
1 O recepcionista seleciona a opção de “nova reserva”	

2 O sistema solicita os dados necessários para o cadastro do cliente.
3 Sistema verifica se cliente já está cadastrado.
3 O recepcionista verifica se digitou os dados corretamente.
4 O recepcionista solicita o preenchimento das informações referentes à alugamento
5 O recepcionista insere a data de início da estadia e do seu fim
6 O recepcionista insere a quantidade de hóspedes (Adultos e crianças)
7 Os dados da reserva são armazenados no sistema
Fluxo Alternativo de Eventos
3.1 Se o hóspede não estiver cadastrado
3.2 <<include>> Caso de Uso 14 – Cadastrar hóspede.
4.1 O recepcionista limpa os dados de nova reserva.

Fonte: Autor, 2021.

Quadro 25 - Descrição de caso de uso ver reservas

Caso de Uso 15	Visualizar reservas do hotel
Atores	Recepcionista
Pré-condições	Haver reservas registradas
Pós-condições	não se aplica
Extension Points	Check-in, Check-out
Fluxo Principal de Evento	
<ol style="list-style-type: none"> 1. Recepcionista acessa reservas do hotel. 2. Sistema exibe reservas por ordem de ocorrência. 3. Recepcionista acessa uma reserva específica. 4. Extension point: Recepcionista pode realizar check-in. 5. Extension point: Recepcionista pode realizar check-out. 6. Extension point: Recepcionista pode realizar pagamento. 7. Extension point: Recepcionista pode cancelar a reserva. 8. Recepcionista finaliza as ações. 	
Fluxo Alternativo de Eventos	
<ol style="list-style-type: none"> 2.1 Sistema exibe reservas de acordo com a busca. 	

Fonte: Autor, 2021.

Quadro 26 - Descrição de caso de uso check-in

Caso de Uso 16	Check-in
Extends	Visualizar reservas
Extension point	Realizar check-in
Atores	Recepção
Pré-condições	1. Ter reserva referente no status “reservada”.
Pós-condições	2. Reserva referente terá status “ativa”.
Regras de Negócio	RN09
Fluxo Principal de Evento	
<ol style="list-style-type: none"> 1. Recepção acessa determinada reserva. 2. O recepcionista seleciona a opção de “Realizar Check-in”. 3. Os dados da reserva são atualizados. 	
Fluxo Alternativo de Eventos	

Fonte: Autor, 2021.

Quadro 27 - Descrição de caso de uso check-out

Caso de Uso 17	Realizar o check-out
Extends	Visualizar reservas
Extension point	Realizar check-out
Atores	Recepção
Pré-condições	1. Ter reserva referente com status “ativa”.
Pós-condições	1. Reserva referente terá status “finalizada”.
Regras de Negócio	RN03, RN08, RN10, RN11
Fluxo Principal de Evento	
<ol style="list-style-type: none"> 1. Recepção acessa determinada reserva. 2. recepcionista seleciona a opção de “Realizar Check-out”, no sistema 3. Os dados de check-out são armazenados no sistema 	
Fluxo Alternativo de Eventos	

Fonte: Autor, 2021.

Quadro 28 - Descrição de caso de uso pagamento

Caso de Uso 18	Pagamento
Extends	Visualizar reservas

Extension point	Registrar pagamento
Atores	Recepção
Pré-condição	não se aplica
Pós-condição	não se aplica
Fluxo Principal de Evento	
1 Recepção solicita valor a pagar para estadia. 2. Sistema exibe valor a pagar e a forma de pagamento. 3. Cliente efetua pagamento. 4. Sistema atualiza estadia como paga.	
Fluxo Alternativo de Eventos	
1. Em qualquer momento o cliente pode pedir para cancelar a execução	

Fonte: Autor, 2021.

Quadro 29 - Descrição de caso de uso cancelar reserva

Caso de Uso 19	Cancelar
Extends	Visualizar reservas
Extension point	Cancelar reserva
Atores	Recepção
Pré-condição	Ter reserva não iniciada.
Pós-condição	Reserva é mantida agora com o status “cancelada”.
Fluxo Principal de Evento	
1. Recepção acessa reserva. 2. Seleciona a opção “cancelar” 3. Sistema requisita confirmação da ação. 4. Recepção confirma e ação é realizada.	
Fluxo Alternativo de Eventos	

Fonte: Autor, 2021.

Quadro 30 - Descrição de caso de uso ver informações

Caso de Uso 20	Visualizar informações do hotel
Atores	Cliente (hospede)
Pré-condições	não se aplica
Pós-condições	não se aplica

Fluxo Principal de Eventos
1 Cliente acessa a informações sobre reservas, sobre o hotel.
1.2 Sistema exibe informações sobre reserva e hotel.
Fluxo Alternativo de Eventos

Fonte: Autor, 2021.

Quadro 31 - Descrição de caso de uso login hóspede

Caso de Uso 21	Efetuar login
Atores	Cliente (hospede)
Pré-condições	não se aplica
Pós-condições	Cliente logado
Fluxo Principal de Eventos	
1.1 O Cliente informa login e senha	
1.2 O sistema verifica se o login e a senha são válidos	
1.3 O sistema registra o início de uma sessão	
Fluxo Alternativo de Eventos	
2 Se o login ou a senha forem inválidos, o sistema exibe uma mensagem (invalido)	

Fonte: Autor, 2021.

Quadro 32 - Descrição de caso de uso reserva web

Caso de Uso 22	Realizar reserva
Atores	Cliente (hospede)
Pré-condições	Cliente deve estar logado no site.
Pós-condições	Haverá uma reserva desse hóspede registrada.
Fluxo Principal de Eventos	
1 O Cliente fornece os dados para realizar as reservas	
Fluxo Alternativo de Eventos	
1 Se o cliente não estiver logado no sistema	
1.1 <<include>> Fazer Login	

Fonte: Autor, 2021.

6.1.2 Matriz de Rastreabilidade.

Os requisitos, agora definidos, servirão de base para os demais artefatos de análise do software, principalmente os requisitos funcionais, pois abrangem todas as

funcionalidades com uma descrição mais objetiva. De acordo com Pressman (2016, p. 134): “Uma matriz de rastreabilidade permite a um engenheiro de requisitos representar a relação entre requisitos e outros artefatos da engenharia de software”. A matriz de rastreabilidade a seguir segue esse modelo, representando os casos de uso, artefatos mais genéricos da documentação, nas linhas, e os requisitos funcionais nas colunas. A marcação das células indica a correspondência entre os elementos. “Muitas vezes, as matrizes de rastreabilidade podem ser usadas para garantir que os artefatos de engenharia consideraram todos os requisitos.” (Pressman, 2016, p. 134).

Quadro 33 - Matriz de Rastreabilidade

Requisito Artefato \	01	02	03	04	05	06	07	08	09	10	11	12	13
Caso de uso 01	X												
Caso de uso 02	X								X				
Caso de uso 03	X	X											
Caso de uso 04	X	X											
Caso de uso 05	X	X											
Caso de uso 06	X									X			
Caso de uso 07	X					X							
Caso de uso 08	X					X							
Caso de uso 09	X						X						
Caso de uso 10	X						X						
Caso de uso 11	X							X					
Caso de uso 12	X							X					
Caso de uso 13	X	X	X										
Caso de uso 14	X	X											
Caso de uso 15	X		X	X									
Caso de uso 16	X		X	X									
Caso de uso 17	X		X	X	X								
Caso de uso 18	X		X										
Caso de uso 19	X		X	X									
Caso de uso 20	X												X
Caso de uso 21										X			
Caso de uso 22	X										X	X	

Fonte: Autor, 2021.

6.2 Diagramas de Atividades.

O diagrama de atividades busca apresentar as possíveis atividades de um cenário específico, usando para tal figuras que representem ações, desvios de condição e escolha, e início e fim das atividades. Através das descrições dos casos de uso, é possível obter as atividades relacionadas ao sistema a ser desenvolvido. Essas atividades estão representadas nos cenários e diagramas a seguir:

DA01 – Cenário da recepção:

Fluxo principal:

1. Repcionista faz login corretamente
2. Acessa mapa de reservas
3. Fim da atividade

Fluxo alternativo 1:

- 2.1 Repcionista acessa uma reserva específica
- 2.2 Realiza manipulação (check-in, check-out, cancelar)
3. Fim da atividade.

Fluxo alternativo 2:

- 2.1 Repcionista acessa uma reserva específica
- 2.4 Manipula seus pagamentos relacionados (adiciona ou deleta)
3. Fim da atividade.

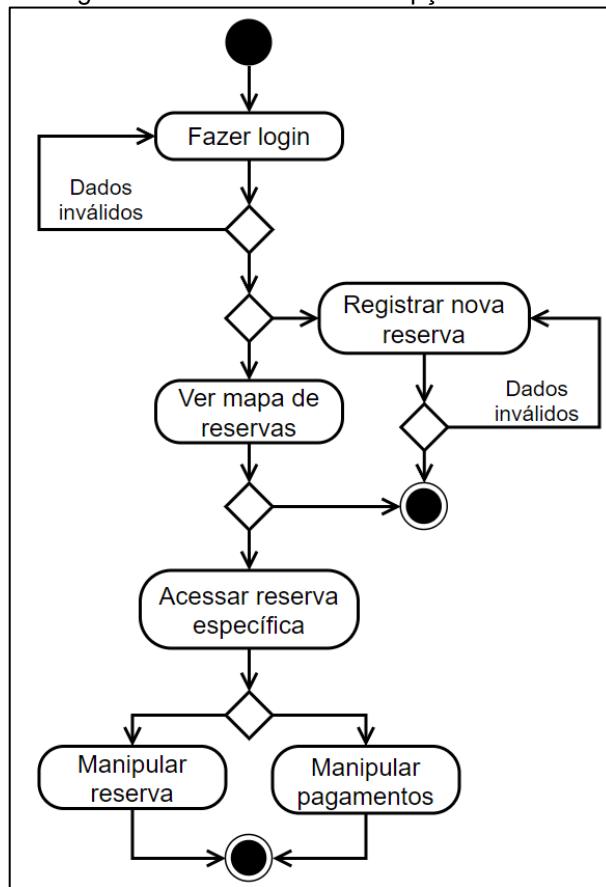
Fluxo alternativo 3:

2. Repcionista acessa tela de nova reserva
- 2.1 Preenche dados e registra nova reserva
- 2.2 Sistema valida a nova reserva
- 2.3 Reserva é válida.
3. Fim da atividade.

Fluxo alternativo 4:

- 2.4 A reserva é inválida e retorna para tela de nova reserva
- 2.5 Usuário pode tentar novamente ou finalizar a atividade.

Figura 17 - Atividades de recepção do hotel



Fonte: Autor, 2021.

DA02 – Cenário da governança:

Fluxo principal:

1. Gerente de governança faz login corretamente
2. Acessa tela dos quartos
3. Confere situação dos quartos
4. Fim da atividade

Fluxo alternativo 1:

- 2.1 Usuário acessa tela dos quartos.
- 2.2 Altera o status do quarto.
- 2.3 Delega tarefas à funcionários.
3. Fim da atividade.

Fluxo alternativo 2:

- 2.1 Usuário acessa tela dos quartos.
- 2.2 Adiciona novo consumo relacionado a algum quarto.
3. Fim da atividade.

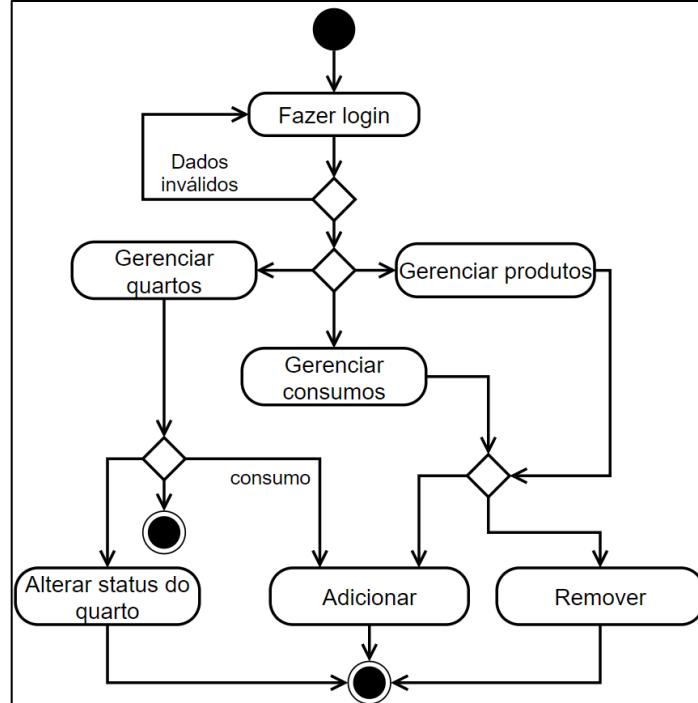
Fluxo alternativo 3:

- 2.1 Usuário acessa tela de consumos
- 2.2 Adiciona ou remove consumo
3. Fim da atividade

Fluxo alternativo 4:

- 2.1 Usuário acessa tela de produtos
- 2.2 Adiciona ou remove produto
3. Fim da atividade

Figura 18 - Atividades de governança do hotel



Fonte: Autor, 2021.

DA03 – Cenário da recepção:

Fluxo principal:

1. Gerente geral faz login
2. Acessa tela de relatórios.
3. Fim da atividade.

Fluxo alternativo 1:

- 2.1 Usuário acessa funções de recepção ou governança.
3. Fim da atividade.

Fluxo alternativo 2:

- 2.1 Usuário acessa tela preços
- 2.2 Confere ou altera preços
3. Fim da atividade.

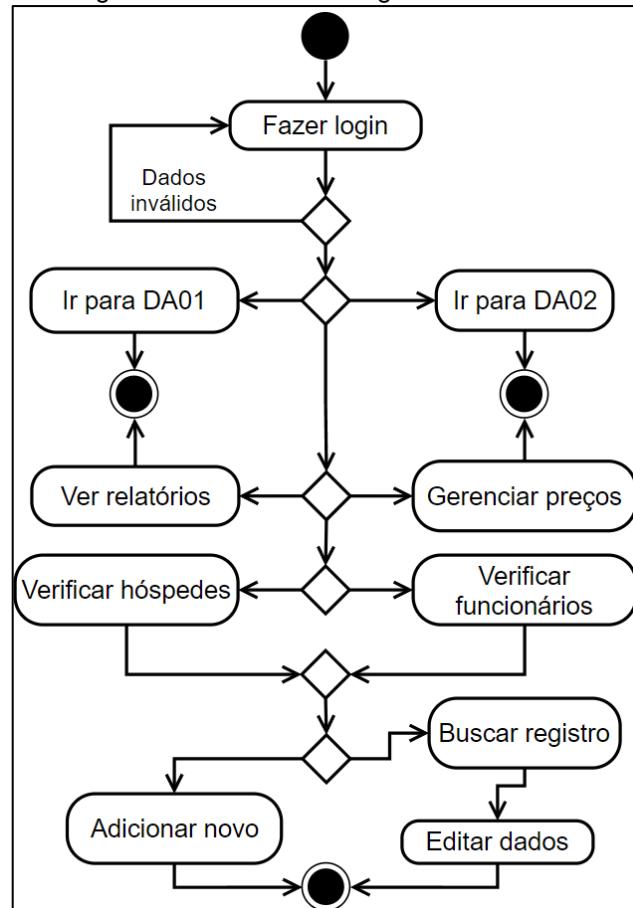
Fluxo alternativo 3:

- 2.1 Usuário acessa tela de hóspedes ou funcionário.
- 2.2 Registra nova pessoa no sistema (hóspede ou funcionário)
3. Fim da atividade.

Fluxo alternativo 4:

- 2.1 Usuário acessa tela de hóspedes ou funcionário.
- 2.2 Busca por registro cadastrado (hóspede ou funcionário)
- 2.3 Editar os dados e salva alterações.
3. Fim da atividade.

Figura 19 - Atividades de gerência do hotel



Fonte: Autor, 2021.

DA04 – Cenário do hóspede:

Fluxo principal:

1. Hóspede acessa o site para obter informações
2. Fim da atividade.

Fluxo alternativo 1:

- 1.1 Hóspede acessa tela de login
- 1.2 Hóspede cria nova conta
3. Fim da atividade

Fluxo alternativo 2:

- 2.1 Usuário acessa tela de editar seus dados
- 2.3 Usuário edita os dados e salva alterações
3. Fim da atividade

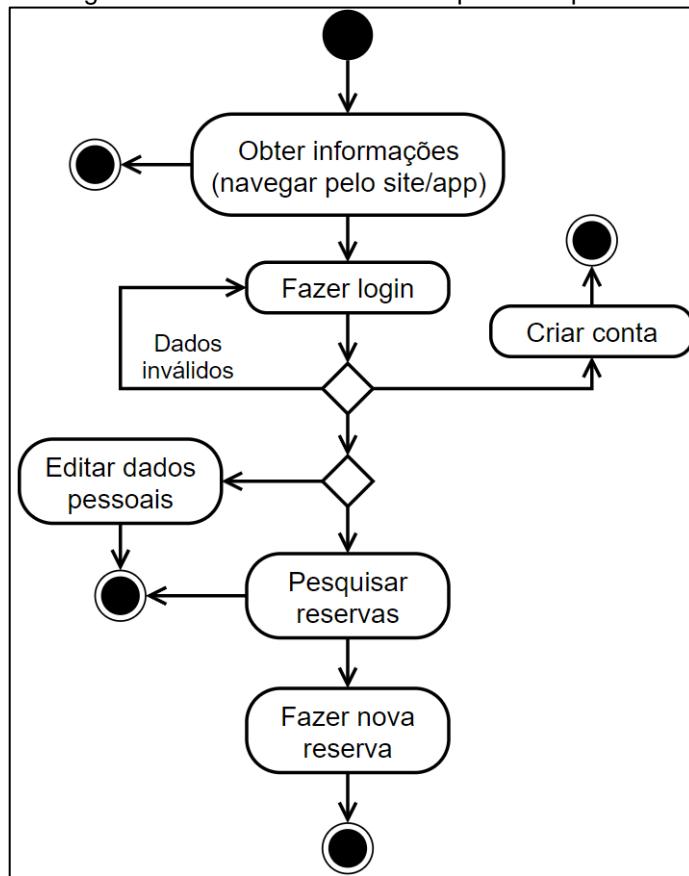
Fluxo alternativo 3:

2. Usuário pesquisa por reservas de hospedagem
3. Fim da atividade

Fluxo alternativo 4:

3. Usuário realiza reserva de hospedagem
4. Fim da atividade

Figura 20 - Atividades realizadas pelos hóspedes

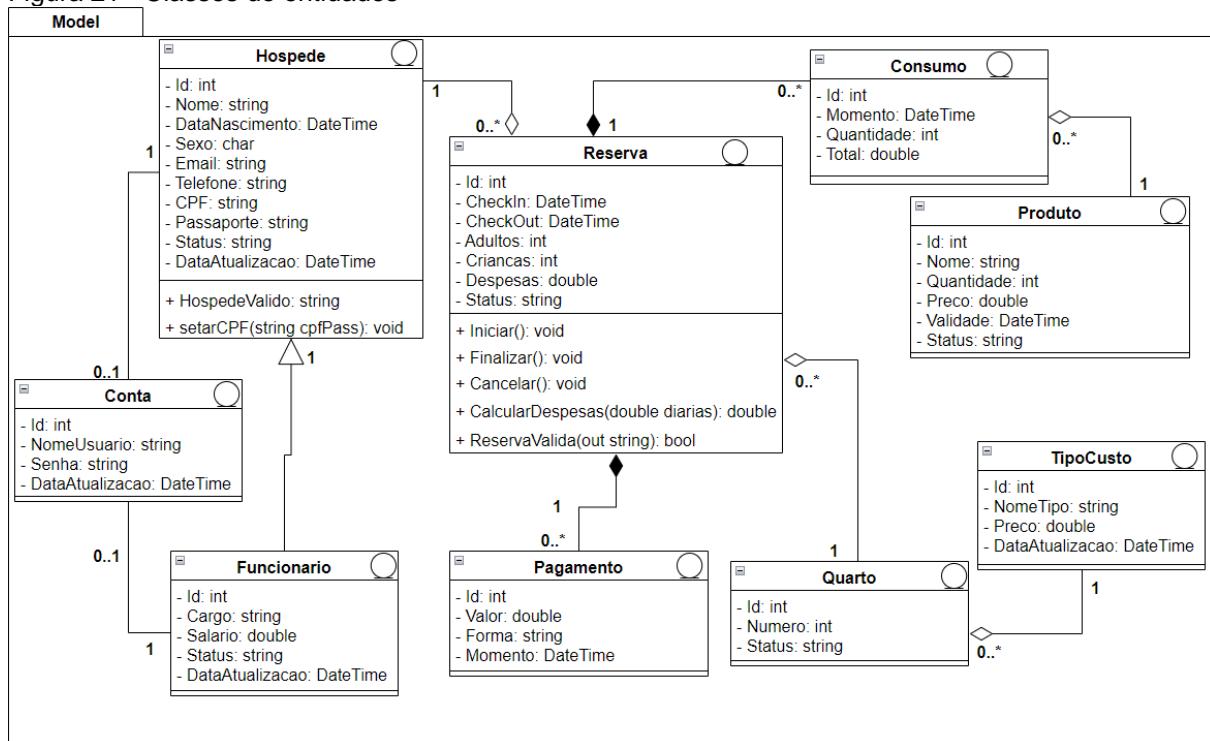


Fonte: Autor, 2021.

6.3 Diagrama de Classes

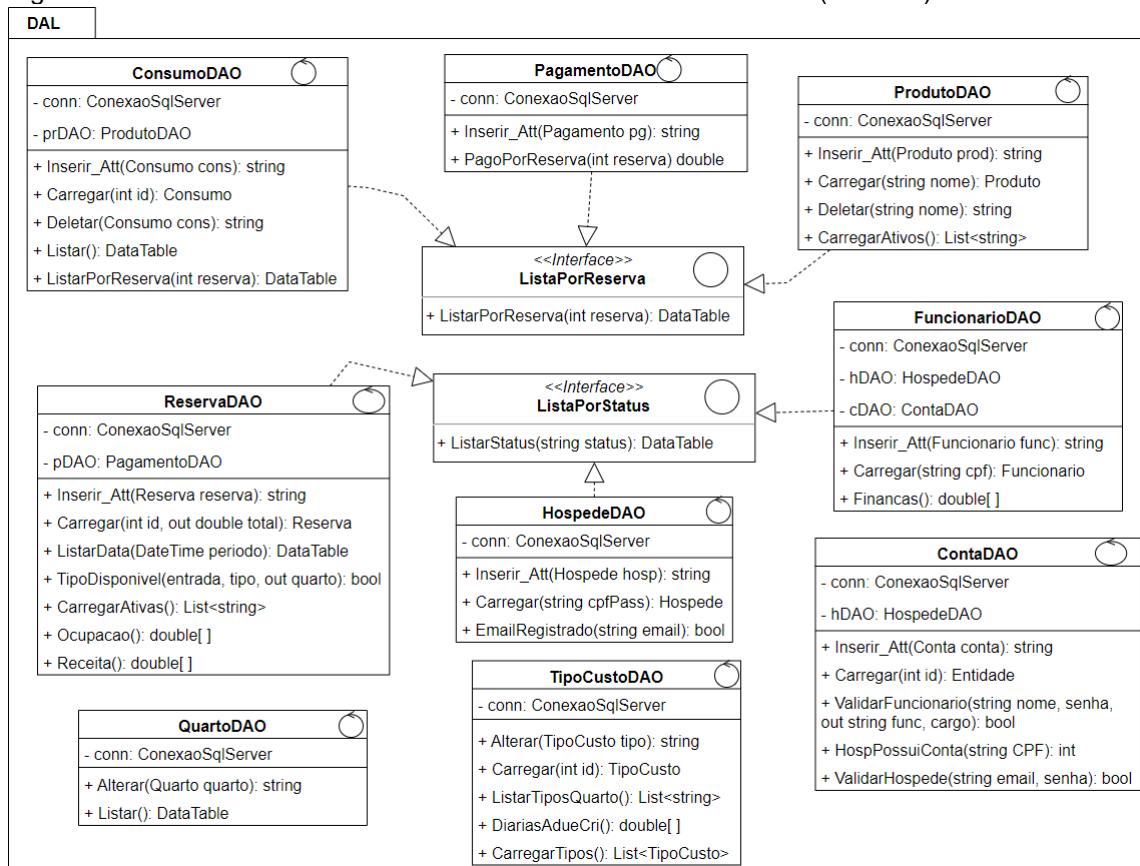
Após identificar as atividades e como cada ator, ou usuário, poderão e irão efetuá-las, é o momento de pensar em como o software realizará isso. O diagrama de classes busca criar uma estrutura sólida que cumpra as atividades necessárias, dividindo atributos e métodos do software nas classes corretas. Isso serve para garantir um sistema pensado na orientação a objetos, onde uma das características é uma estrutura do sistema bem definida, possibilitando melhor entendimento e eventuais manutenções. Os diagramas a seguir representam as classes a serem implementadas:

Figura 21 - Classes de entidades



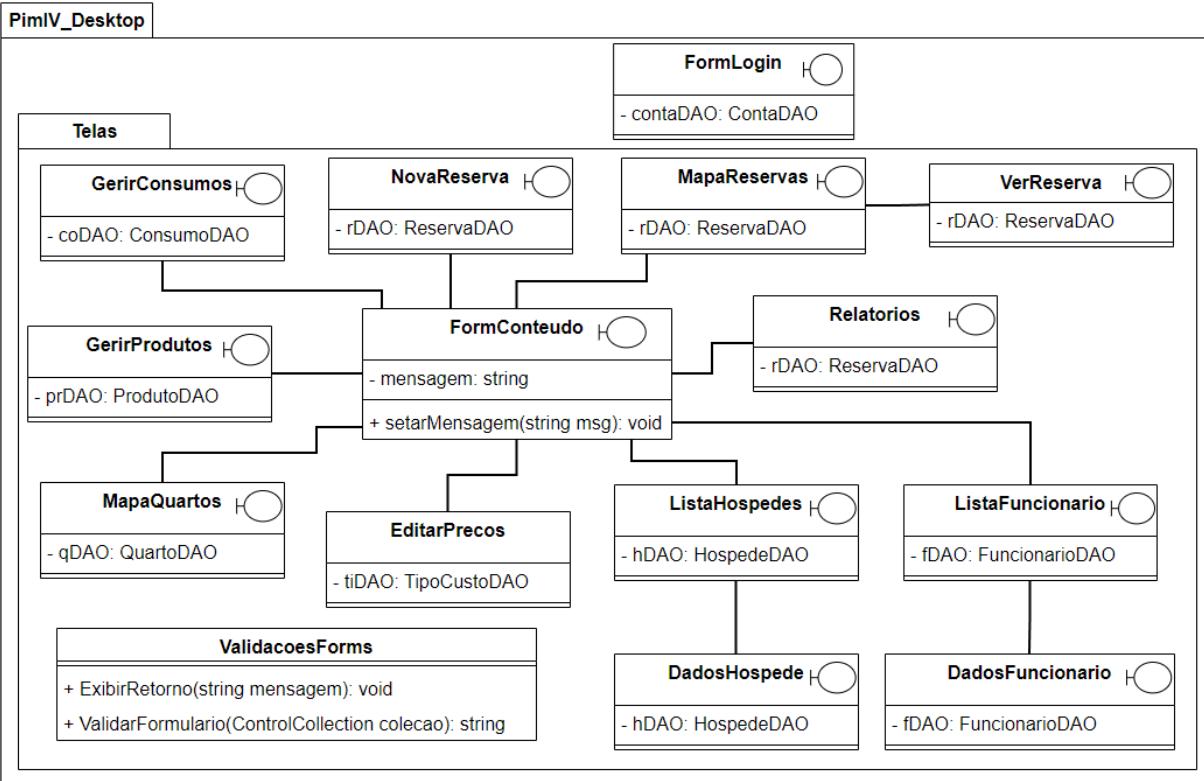
Fonte: Autor, 2021.

Figura 22 - Classes de acesso ao banco de dados e suas interfaces (contrato)



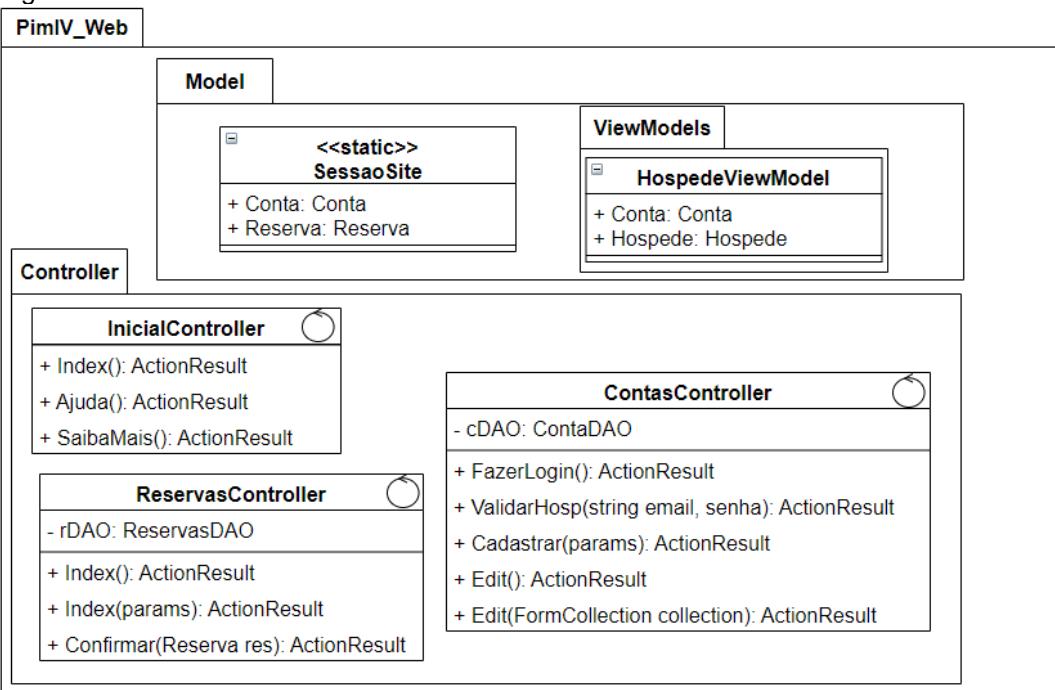
Fonte: Autor, 2021.

Figura 23 - Classes da interface Desktop



Fonte: Autor, 2021.

Figura 24 - Classes da interface web

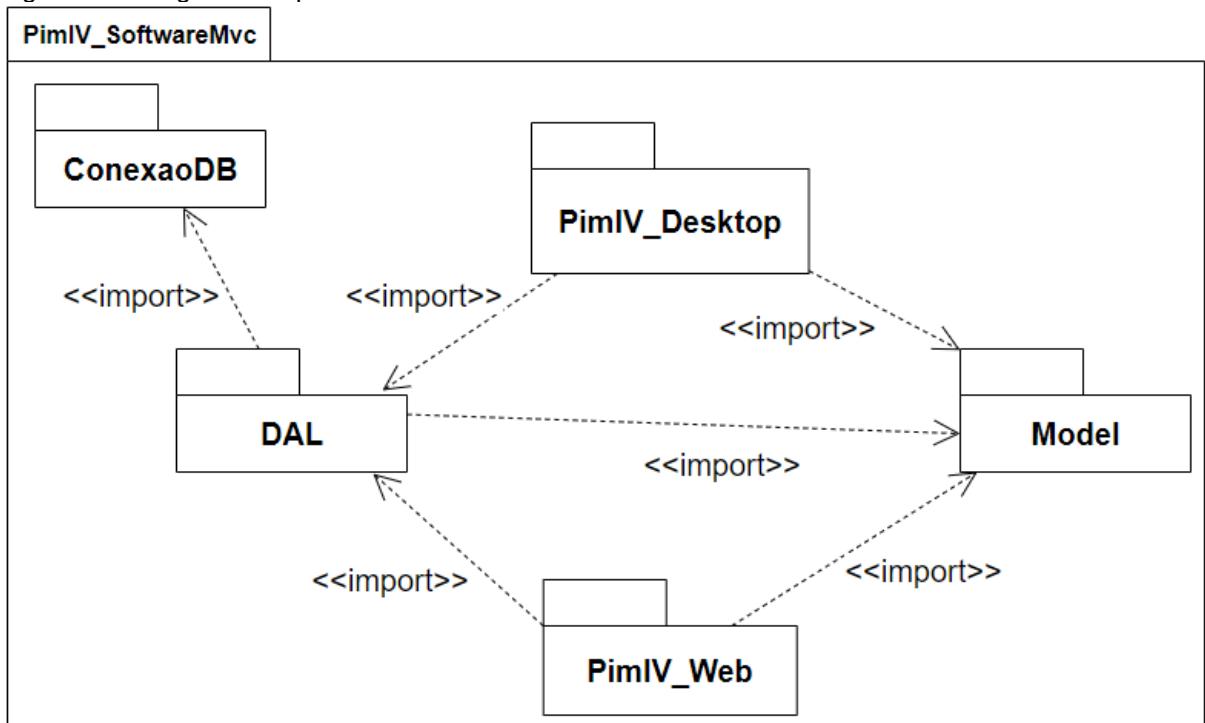


Fonte: Autor, 2021.

6.4 Diagrama de pacotes.

O diagrama de pacotes ilustra o agrupamento de itens em seções lógicas. O padrão arquitetural utilizado neste projeto foi o MVC, que significa Model, View e Controller, junto da estrutura em camadas, que permite melhor reutilização de artefatos em diferentes projetos, e garantia de importação somente do necessário, visto que é preciso adicionar referências entre projetos antes da importação. Nessa arquitetura, os arquivos e classes que geram interface com o usuário ficam na camada que representa a View, nos pacotes PimIV/Desktop e PimIV/Web, representando sistemas desktop e web respectivamente. As definições das classes das entidades e implementação de regras de negócio ficam na camada Model, e os artefatos que obtém dados do banco de dados ficam na camada DAL, que significa Data Access Layer (Camada de Acesso a Dados), onde há classes DAO, que significa Data Access Object (Objeto de Acesso à Dados) já apresentados no diagrama de classes anterior. A camada ConexaoDB é utilizada para obter e abrir conexão com o banco de dados SQL Server.

Figura 25 - Diagrama de pacotes



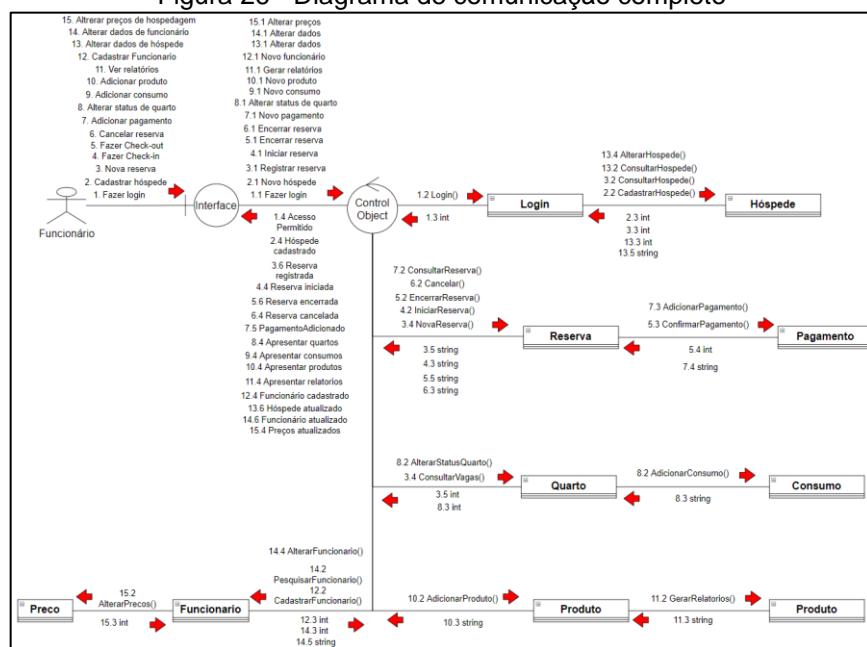
Fonte: Autor, 2021.

No pacote web a pasta Controller contém as classes responsáveis por interligar as diferentes páginas web, cujas interfaces gráficas em HTML ficam na pasta View.

6.4 Diagramas de comunicação.

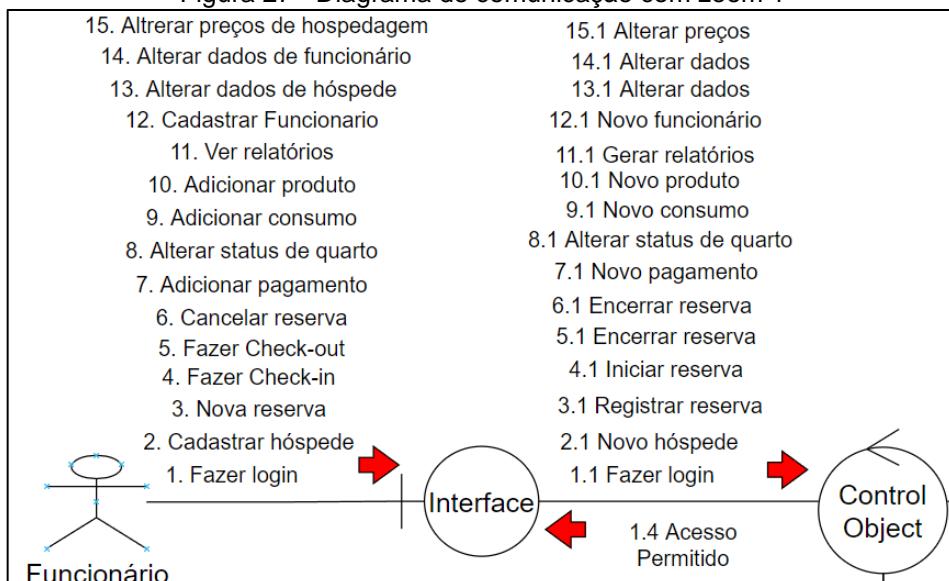
Os sistemas deste projeto utilizam o paradigma orientado a objetos. Isso significa que durante a execução do programa, os objetos das classes irão se comunicar, de um modo chamado troca de mensagens. O diagrama de comunicação busca representar essa troca de informações de modo resumido, sem se atentar à ciclo de tempo dos objetos. Seu foco é mostrar a interação entre ator e sistema, o que cada objeto responsável retorna para a interface do usuário.

Figura 26 - Diagrama de comunicação completo



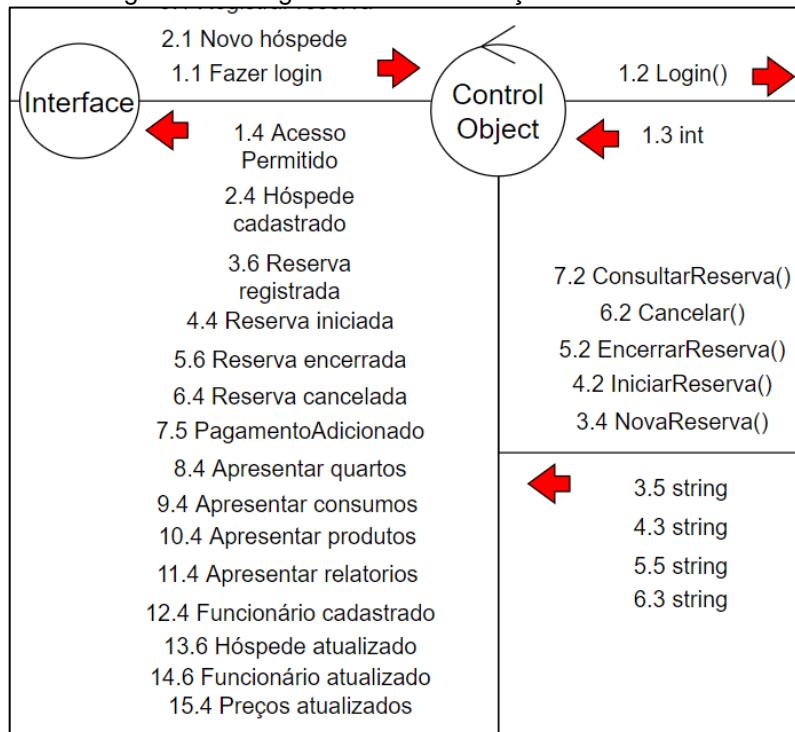
Fonte: Autor, 2021.

Figura 27 - Diagrama de comunicação com zoom 1



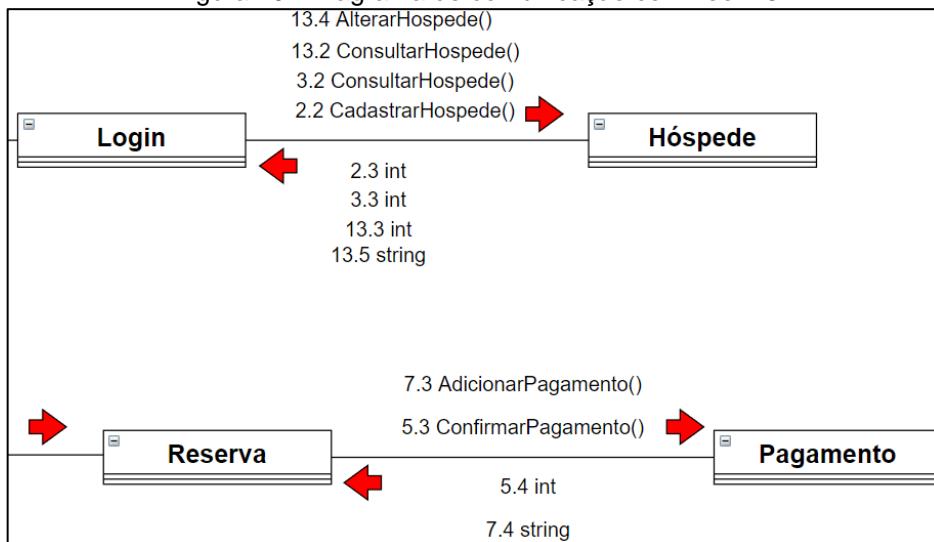
Fonte: Autor, 2021.

Figura 28 - Diagrama de comunicação com zoom 2



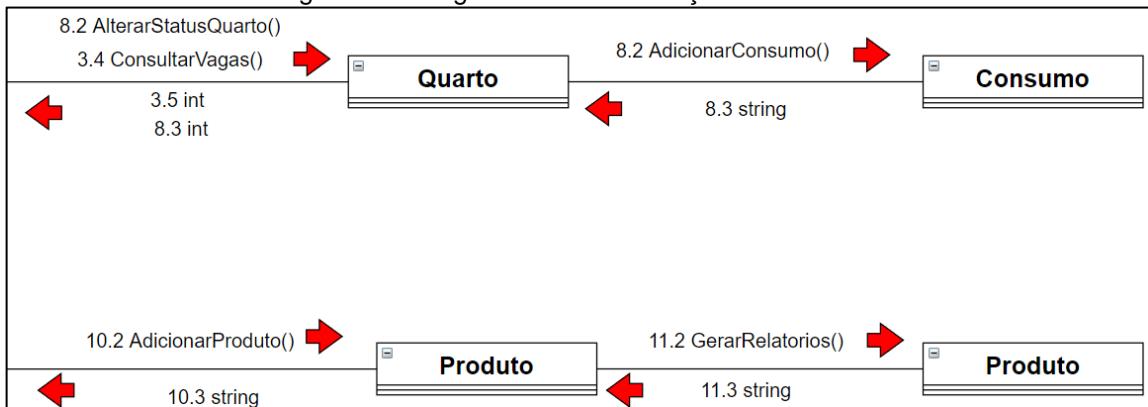
Fonte: Autor, 2021.

Figura 29 - Diagrama de comunicação com zoom 3



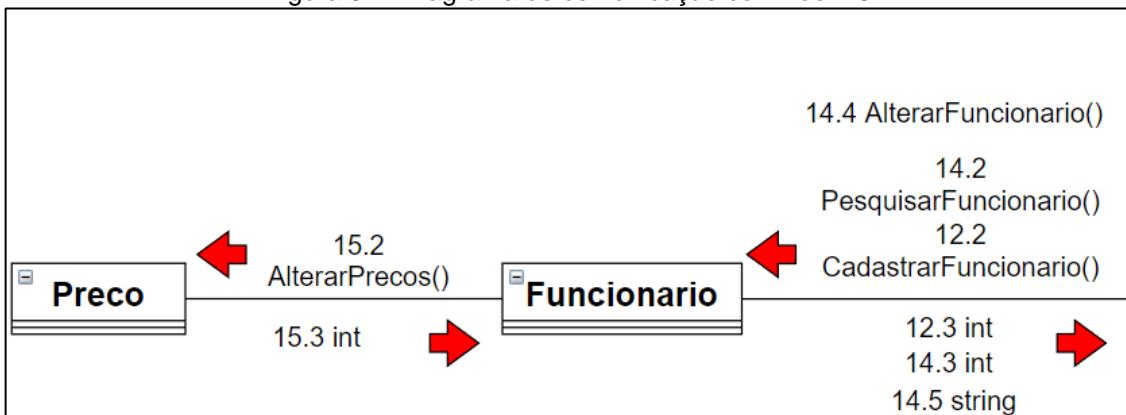
Fonte: Autor, 2021.

Figura 30 - Diagrama de comunicação com zoom 4



Fonte: Autor, 2021.

Figura 31 - Diagrama de comunicação com zoom 5

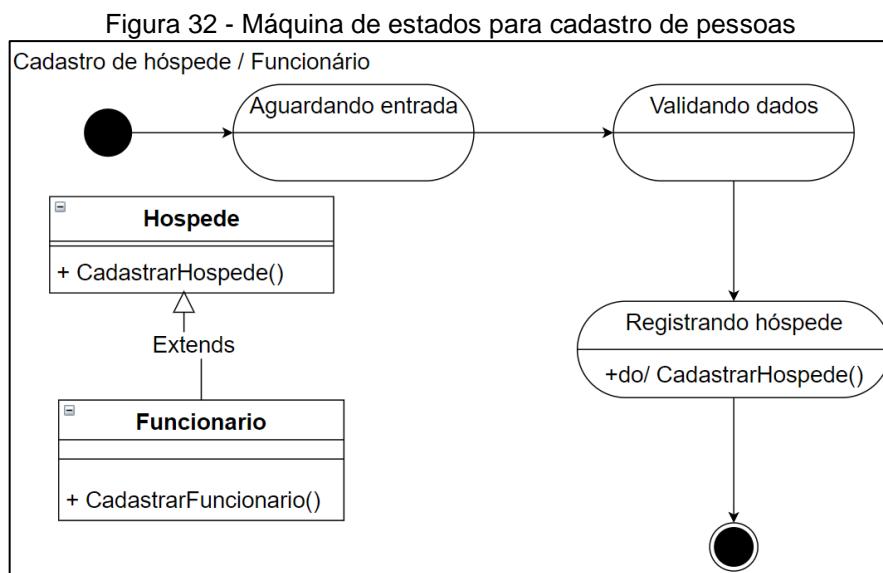


Fonte: Autor, 2021.

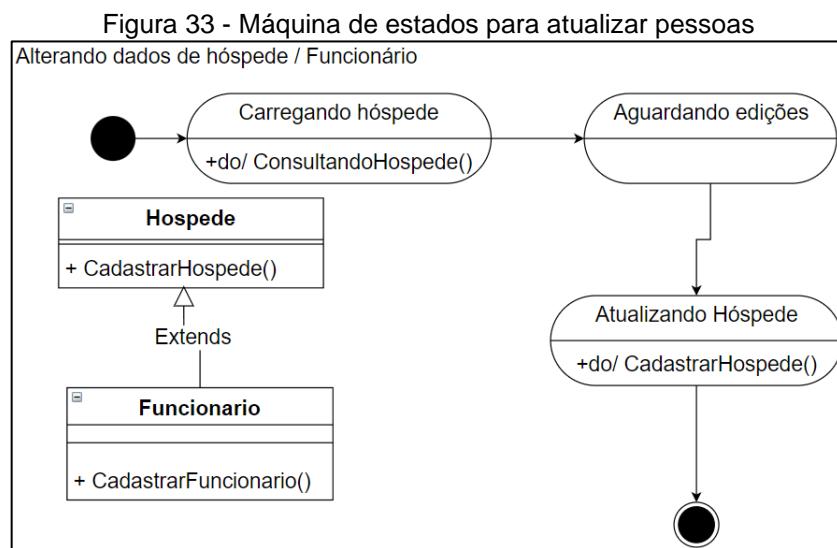
Foi feito diagrama de comunicação apenas para o sistema Desktop, visto que o sistema web será apenas para teste da completa implementação futura.

6.5 Diagramas de máquina de estados.

Como dito anteriormente, os objetos durante a execução trocam mensagens entre si. Isso significa que cada objeto é usado apenas no momento que é necessário. O Diagrama de máquina de estados busca ilustrar a lógica do sistema enquanto aguarda a interação do usuário, para então utilizar as classes necessárias. Foram feitos tais diagramas apenas para funcionalidades consideradas necessárias.

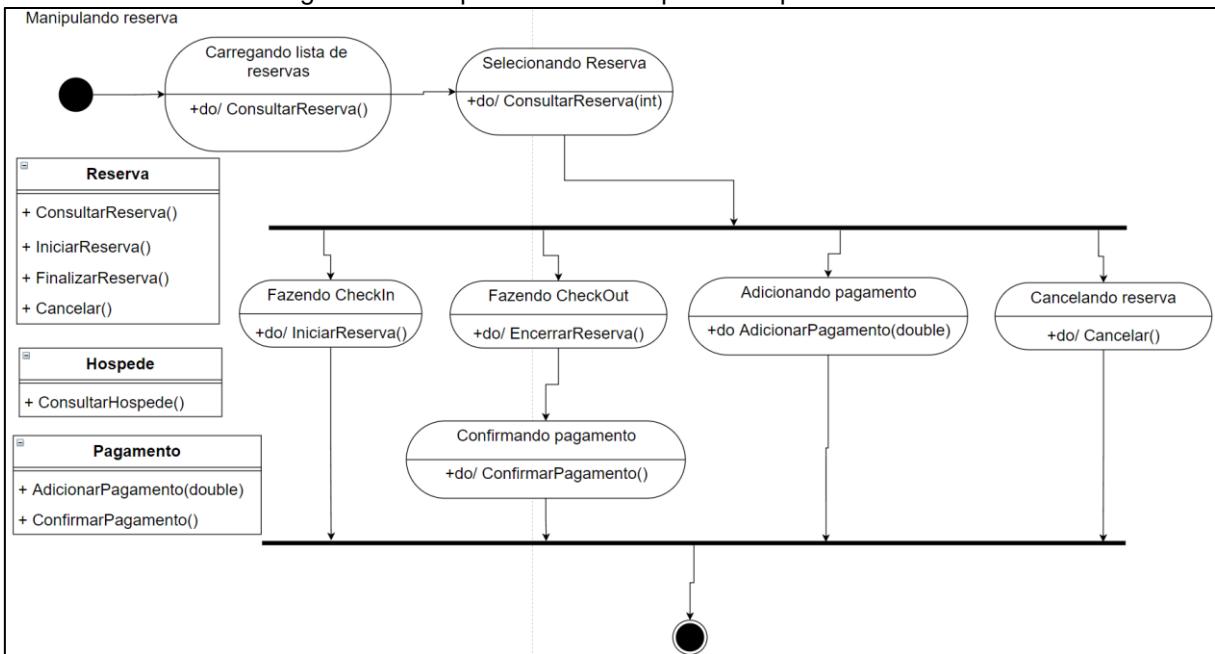


Fonte: Autor, 2021.



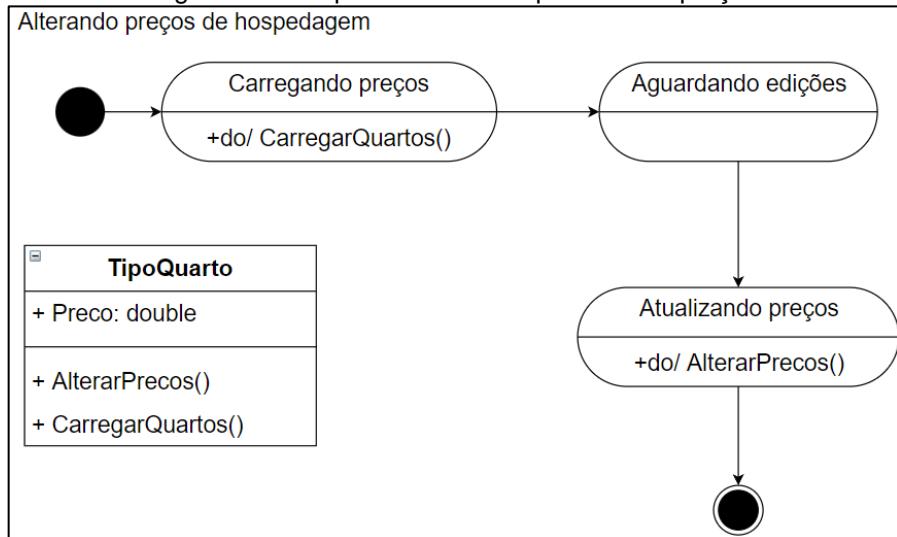
Fonte: Autor, 2021.

Figura 34 - Máquina de estados para manipular reserva



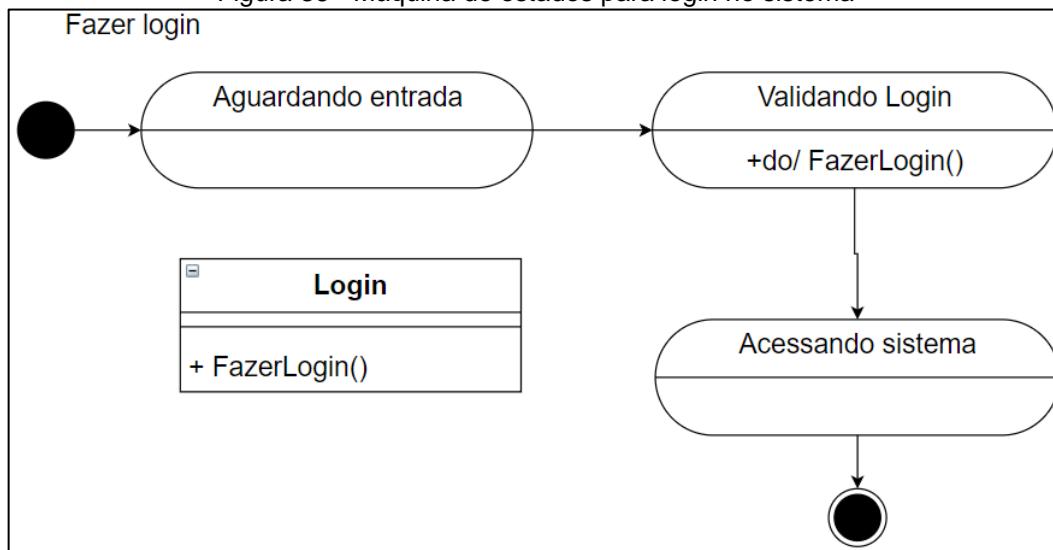
Fonte: Autor, 2021.

Figura 35 - Máquina de estados para alterar preços



Fonte: Autor, 2021.

Figura 36 - Máquina de estados para login no sistema

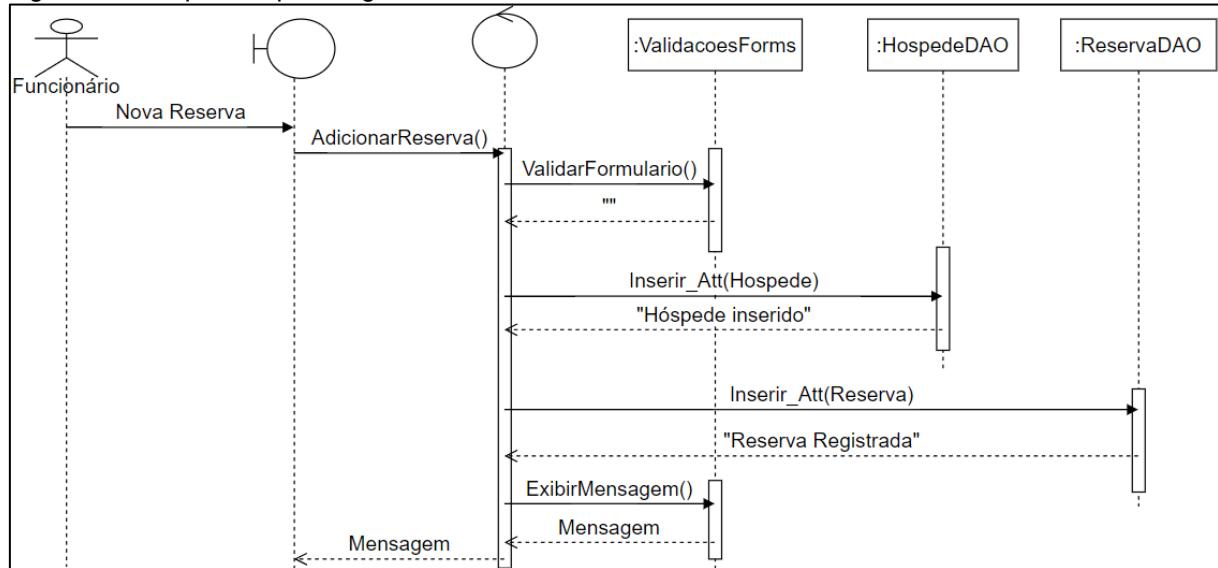


Fonte: Autor, 2021.

6.4 Diagramas de Sequência

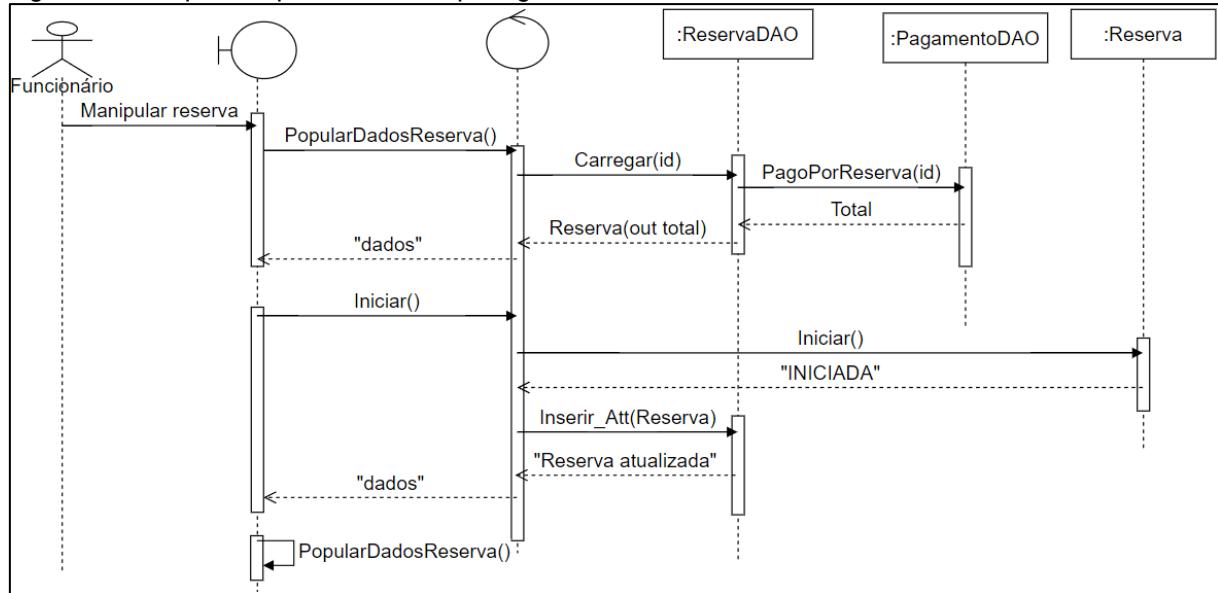
Com as classes definidas, é preciso também pensar na sequência que seus relacionamentos ocorrerão. Para sistemas orientados a objetos, é definido que a comunicação entre classes ocorre através de troca de mensagens. O diagrama de sequência busca demonstrar a sequência em que ocorre essa troca de mensagens entre as classes. Segundo Fowler (2005, p.74): “Os diagramas de sequência são bons para mostrar as colaborações entre os objetos, mas não são tão bons para uma definição precisa do comportamento”. Ainda assim, o objetivo atual é obter uma visualização dessa comunicação para análise e desenvolvimento inicial de protótipos, portanto eles são adequados. A seguir, os diagramas de sequência desenvolvidos.

Figura 37 - Sequência para registro de reserva



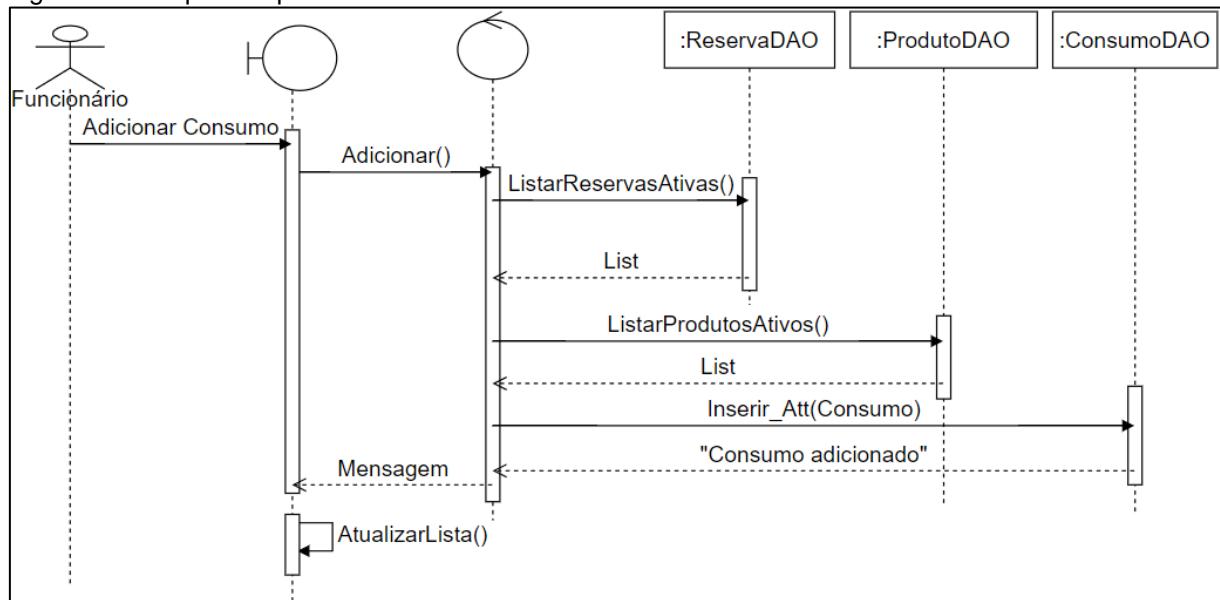
Fonte: Autor, 2021.

Figura 38 - Sequência para iniciar hospedagem



Fonte: Autor, 2021.

Figura 39 - Sequência para adicionar consumo

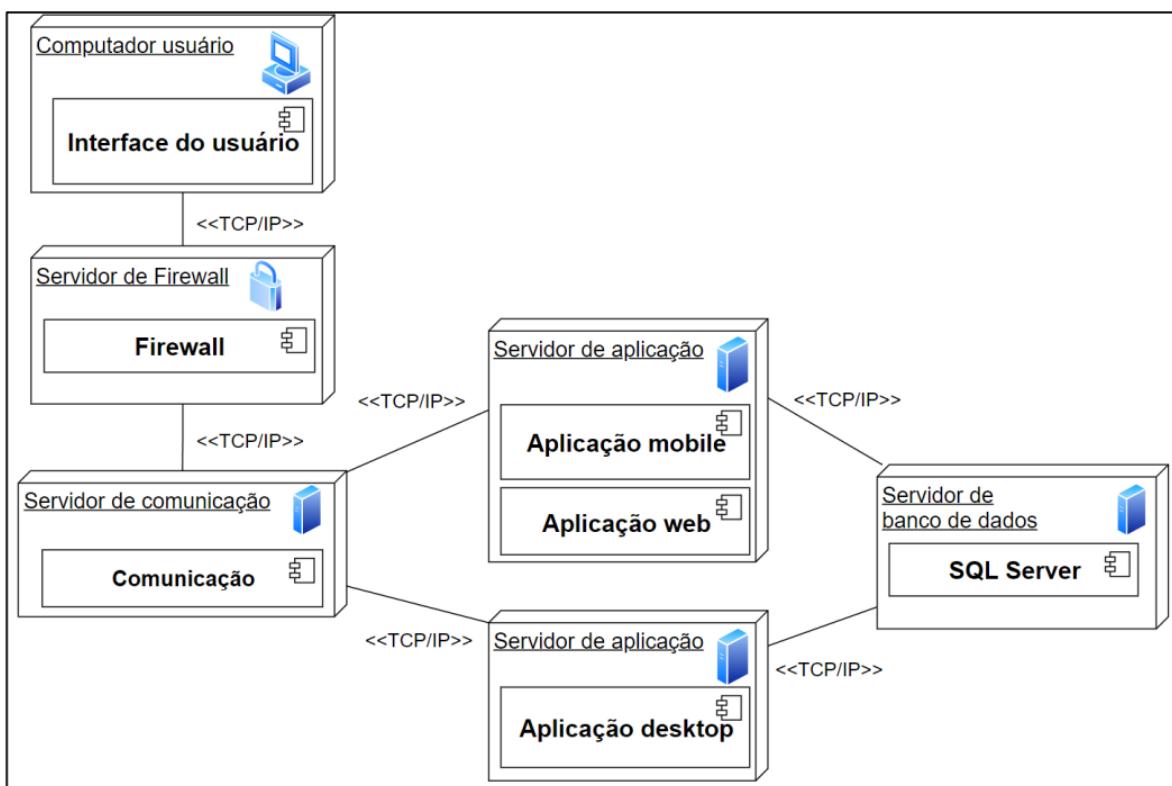


Fonte: Autor, 2021.

6.5 Diagrama de Implantação

Após a implementação ser devidamente validada, é o momento de implantar, ou seja, entregar o software pronto e funcional para que o cliente o utilize. Segundo Medeiros (2004, p. 230): “Um diagrama de implantação modela o inter-relacionamento entre recursos de infra-estrutura, de rede ou artefatos de sistemas. Normalmente representamos servidores neste diagrama”. Portanto, o diagrama de implantação a seguir aborda três camadas, e representa a conexão entre servidores que armazenam os dados ou aplicações, com os dispositivos que os utilizam.

Diagrama 1 - Diagrama de Implantação do projeto



Fonte: Autor, 2021

7 ESTUDO DO USUÁRIO E ESTUDO DE INTERFACE

Um aspecto importantíssimo e determinante na qualidade e sucesso de um software é a interface. O meio pelo qual um sistema será utilizado deve ser além de funcional, adequado, caso contrário o uso tende a ser problemático. Portanto, foi utilizada a abordagem do desenvolvimento centrado no humano, o que significa que a interface para uso do software contará com estudo e análise do usuário final, proporcionando maior produtividade, satisfação potencialmente maior, entre outros

benefícios. Benyon (2011, p. 33), reconhece que desenvolver com design de interface centrado no humano é caro, mas ressalta também que a abordagem é vantajosa por diversas razões, entre as quais são citadas retorno do investimento, segurança e ética. O estudo do usuário contou com a elaboração de documento de visão, perfis de usuário, requisitos de usabilidade e personas.

7.1 Estudo do usuário.

7.1.1 Documento de visão.

Este documento consiste em uma análise geral do usuário final. São consideradas suas principais características, ambiente de trabalho, contexto do negócio, entre outros.

O registro produzido para identificar as características-chave que a solução deve fornecer e a perspectiva do cliente ou usuário de alto nível sobre o sistema a ser desenvolvido nessa etapa do processo é o documento de visão[...]. É um documento valioso, mas vale lembrar que o problema sofre mudanças enquanto é solucionado e esse documento não deve se tornar um obstáculo para a inovação. A tentativa de definir previamente todas as especificidades pode custar caro e limitar sensivelmente o aparecimento de novas ideias. (KERR, 2015, p. 31).

Início do Documento de Visão.

1. Introdução

Neste documento consta a análise inicial do projeto de software para desktop SGQT (Sistema de Gestão Quiet Time), que visa proporcionar aos colaboradores do hotel uma melhoria na gestão do negócio.

Consta também a análise inicial do projeto de software direcionado aos clientes, que comporta versões web e mobile, cujo objetivo é possibilitar acesso às informações e maior contato com o hotel diretamente pelos clientes.

1.1 Escopo

Este documento tem ênfase na usabilidade do sistema, relacionando as atividades já realizadas pelos colaboradores de forma manual e como elas serão adaptadas e possivelmente melhoradas pelo sistema de software.

2. Contextualização

A seguir consta resumo do porquê o sistema ter sido desenvolvido.

2.1 Motivação do projeto

Após a renovação da liderança no Hotel Quiet Time, foi decidida a integração de tecnologia através de um Sistema de Gestão Hoteleira visando melhorias nas atividades do hotel, uma vez que esta prática tem se tornado comum e fator decisivo para uma melhor competitividade no setor hoteleiro.

Portanto, a versão inicial deste software conta com as principais atividades do hotel, que se resumem ao controle de reservas de hospedagem e governança dos quartos. Juntamente, visando o contato com clientes, um software auxiliar que os permita realizar suas reservas.

2.2 Descrição das atividades

As atividades já realizadas pelos colaboradores que o sistema visa aprimorar são:

- Informar clientes (hóspedes) sobre o hotel e suas regras.
- Realizar e controlar reservas de hospedagem.
- Gerir a governança dos quartos do hotel.
- Gerir consumos dos hóspedes durante estadia.
- Gerir estoque de produtos de consumo.
- Controlar dados pessoais (hóspedes e funcionários).
- Analisar dados para estudo de desempenho.
- Controlar os preços de hospedagem devido suas variações.

2.3 Software próprio x Software comercial

A decisão por desenvolver um software exclusivo ao invés de efetuar a compra de um Sistema de Gestão Hoteleira comercial está relacionada à tradição do hotel, na preferência de ter suas atividades adaptadas a um sistema, contra ter de mudar suas atividades de acordo como um sistema comercial funciona.

3. Usuários do Sistema

Mesmo que um indivíduo não realize os cliques do mouse ou digite no teclado, ainda assim pode ser considerado um tipo de usuário de um sistema. Abaixo consta essa definição.

3.1 Usuários primários.

Usuários que irão interagir com as interfaces são considerados primários.

Quadro 34 - Usuários primários do sistema

Identificação	Descrição	Envolvimento
Recepção	Responsável pelo contato com cliente.	Informa, realiza reservas e auxilia durante hospedagem.
Gerente de Governança	Administra a qualidade da estadia.	Gere os quartos do hotel. Registra os consumos e controla o estoque.
Gerente Geral	Administra o hotel, supervisionando departamentos.	Controla dados pessoais, preços de hospedagem e analisa desempenho.
Cliente (hóspede)	Pessoa física que utiliza serviços do hotel.	Pesquisa e realiza reservas e controla seus dados.

Fonte: Autor, 2021

3.2 Usuários secundários.

Indivíduos cujas atividades sejam influenciadas pelos usuários primários, são considerados usuários secundários.

Quadro 35 - Usuários secundários do sistema

Identificação	Descrição	Envolvimento
Camareira	Responsável pela limpeza e inspeção de quartos.	Será instruída pelo Gerente de Governança.
Assistente de Manutenção	Realiza pequenos reparos.	É instruído pelo Gerente de Governança.
Administrador financeiro	Controla as finanças do hotel.	Utiliza relatórios do Gerente Geral para análises.

Fonte: Autor, 2021

4. Ambiente de Utilização do sistema

Os **colaboradores do hotel** utilizarão o sistema em computadores de mesa (desktop).

Os **clientes** poderão utilizar seu software auxiliar tanto em sistemas desktop (computadores de mesa, notebooks) quanto em sistemas mobile (celulares, tablets).

5. Considerações técnicas e detalhes.

O sistema utilizará arquitetura cliente servidor, visto que os sistemas web e mobile devem ser operacionais 24 horas por dia, e o banco de dados servirá tanto para o software desktop quanto para os demais. Detalhes relevantes:

- Os sistemas contarão com os modos de interação Manipulação e Navegação, Exploração e Pesquisa.
- Sistema web deve ser responsivo.

6. Prioridades das principais características

A implantação de um software causa mudanças em uma organização, abaixo constam principais tipos de influência que as interfaces devem considerar.

Quadro 36 - Prioridades da interface

Prioridade	Característica	Benefícios
Alta	Objetividade na utilização.	Permite melhor entendimento do sistema e proporciona um modelo mental mais correto ao usuário
Alta	Garantir uma base de dados otimizada	Melhora a eficiência do sistema, diminuindo o tempo de espera dos usuários.
Alta	Utilizar interfaces compactas e não complexas	Permite melhor memorização quanto à sequência de passos para realizar atividades.

Fonte: Autor, 2021

7. Premissas desejáveis dos usuários

Considerando o ambiente no qual o software será implantado, existem premissas, conceitos que podem se esperar dos usuários, são eles:

Quanto aos hóspedes:

- Conhecimento básico de navegação web ou uso de aplicativos mobile para poder utilizar os mecanismos de pesquisa.
- Familiaridade com realização de login e logout, para acesso e controle de seus dados.

Quanto aos colaboradores do hotel:

- Conhecimento básico em informática para construção de um modelo mental do sistema que permita utilização fluida.
- Entendimento dos processos de hotelaria para melhor realização das atividades no sistema desktop.

8. Prioridades das tarefas dos usuários

Considerando que são várias as tarefas adaptadas para o software, é feita uma seleção de prioridade de tarefas que as interfaces devem considerar.

Quadro 37 - Prioridades de tarefas de usuários

Prioridade	Descrição da tarefa	Papéis atuantes
Alta	Registrar e manipular reservas de hospedagem	Recepção / Gerente Geral / Hóspede
Alta	Controlar preços de hospedagem	Gerente Geral
Alta	Administrar consumos do usuário e estoque de produtos	Gerente de Governança / Gerente Geral
Alta	Administrar dados pessoais e potencialmente sensíveis	Hóspede, Gerente Geral
Média	Obter informações do hotel	Hóspede

Média	Gerir quartos do hotel	Gerente de Governança
-------	------------------------	-----------------------

Fonte: Autor, 2021

9. Requisitos de Usabilidade iniciais e semelhantes

Estes requisitos visam garantir uma boa experiência do usuário durante a utilização do software. São eles:

- Interfaces objetivas e simples para garantir maior otimização do software, eficiência no uso e acessibilidade à usuários inexperientes.
- Utilização de fontes com alta pregnância, para facilitar leitura e sequência de ações.
- Disposição de botões facilmente visíveis para localização objetiva de atividades.

Fim do documento de visão.

7.1.2 Requisitos de usabilidade.

Os requisitos de usabilidade a seguir foram definidos através do estudo do usuário, com apresentação e concordância dos stakeholders para com eles. São utilizados como base para avaliar o design das interfaces construídas, visando cumprir o desenvolvimento centrado no humano.

Quadro 38 - Requisitos de Usabilidade

Estilo	Descrição
Desempenho	Usuários inexperientes devem conseguir realizar uma nova reserva em até 10 minutos. Usuários experientes devem realizar em até 6 minutos.
Integridade	O sistema desktop deve limitar o acesso a funções de acordo com o cargo do funcionário. Enquanto o Web deve ter caminhos claros para suas funcionalidades.
Clareza	As funcionalidades do sistema desktop devem estar semelhantes ao estilo já praticado pelos funcionários do hotel.
Praticidade	A sequência de passos para realizar atividades deve ter no máximo 10 passos.

Coloração	Uso de cores neutras no sistema desktop para propor simplicidade e objetividade ao usuário.
Objetividade	Apresentar de forma simples e direta os itens de utilização, para que aprendam rapidamente seus significados.
Retorno	Exibir mensagens de retorno após a execução de atividades (se foi realizada com sucesso ou não).

Fonte: Autor, 2021.

7.1.3 Perfis de usuário e Personas.

Os perfis de usuário buscam gerar um modelo de usuário final, para que os desenvolvedores considerem isso no momento de desenhar as interfaces. O nível de familiaridade com a tecnologia, de habilidade em digitação, são exemplos claros de como o design de uma interface pode gerar diferentes experiências para cada tipo de usuário. A seguir, as análises de perfil de usuário para os colaboradores do hotel e os hóspedes (clientes).

Quadro 39 - Perfil de usuário para Gerente Geral

INFORMAÇÕES PROFISSIONAIS
<ul style="list-style-type: none"> • Atua há mais de seis anos no ramo hoteleiro, dois desses no Quiet Time. • Iniciou na governança até chegar à gerência geral. • Maior salário do hotel, pois tem função de garantir seu funcionamento. • Graduação em Turismo e Processos Gerenciais. Inglês fluente. • Cumpre turno de oito horas. Deve estar totalmente acessível aos demais colaboradores. • Analisa relatórios de gestão individualmente e em reuniões gerenciais. • Avaliado de acordo o desempenho do hotel.
COMPORTAMENTO COM RELAÇÃO A TECNOLOGIA EM GERAL
<ul style="list-style-type: none"> • Aprecia a praticidade da tecnologia, é apto e tem facilidade em seu uso. • Tem interesse em novidades tecnológicas, razão de seus estudos sobre a área.
EXPERIÊNCIA COM COMPUTADORES
<ul style="list-style-type: none"> • Utiliza diariamente por aproximadamente três horas para estudo e lazer. • Utiliza navegadores web, softwares de escritório e outros.

- Utiliza computadores desde quando pôde adquirir um, em sua época de estudos.
- Digitação correta e veloz.

Fonte: Autor, 2021.

Quadro 40 - Perfil de usuário para Gerente de Governança

INFORMAÇÕES PROFISSIONAIS
<ul style="list-style-type: none"> • Atua há mais de três anos no ramo hoteleiro, um no Quiet Time. • Esteve desde o início no setor de Governança. • Salário maior que de recepcionistas, por zelar da hospitalidade. • Graduação em Hotelaria. Proativo e bom administrador. • Cumpre turno de oito horas. Supervisiona e auxilia funcionários do setor. • Gere informações individualmente e realiza as atividades coletivamente com funcionários do setor. • Supervisionado pelo Gerente Geral.
COMPORTAMENTO COM RELAÇÃO A TECNOLOGIA EM GERAL
<ul style="list-style-type: none"> • Tende a resistir o uso de tecnologia e leva algum tempo para aprender.
EXPERIÊNCIA COM COMPUTADORES
<ul style="list-style-type: none"> • Utiliza celular diariamente quando necessário. • Utiliza computadores apenas quando o celular se mostra insuficiente. • Digitação lenta.

Fonte: Autor, 2021.

Quadro 41 - Perfil de usuário para Recepcionista

INFORMAÇÕES PROFISSIONAIS
<ul style="list-style-type: none"> • Tanto experientes quanto iniciantes. • Em aprendizagem sobre hotelaria... • Salário médio dentre os funcionários. Idiomas aumentam remuneração. • Cursando Turismo ou curso relacionado. Facilidade em lidar com pessoas. • Turno de aproximadamente oito horas. Tarefas bem definidas. • Atende hóspedes, realiza e controla reservas de hospedagem. • Supervisionado pelo Gerente de Recepção e Gerente Geral.

COMPORTAMENTO COM RELAÇÃO A TECNOLOGIA EM GERAL
<ul style="list-style-type: none"> • Indiferente quanto a novidades, porém tem facilidade em aprender. • Busca utilizar apenas tecnologias realmente necessárias.
EXPERIÊNCIA COM COMPUTADORES
<ul style="list-style-type: none"> • Utiliza diariamente por aproximadamente quatro horas para estudo e lazer. • Utiliza navegadores web, softwares de escritório e outros. • Utiliza apenas dispositivo móvel, mas comprehende sistemas desktop. • Digitação aceitável e razoavelmente rápida.

Fonte: Autor, 2021.

Quadro 42 - Perfil de usuário para Hóspedes

INFORMAÇÕES PROFISSIONAIS
<ul style="list-style-type: none"> • É preciso considerar pessoas com todo tipo de profissão, tanto as que envolvem quanto as que não envolvem uso de tecnologia. • Também não é possível mensurar o nível de escolaridade, assim abrangendo todos. • É de se esperar que busquem simplicidade no contato com o hotel, e facilidade em realizar sua hospedagem.
COMPORTAMENTO COM RELAÇÃO A TECNOLOGIA EM GERAL
<ul style="list-style-type: none"> • Deve-se considerar todos os níveis, desde indiferentes à profissionais da área.
EXPERIÊNCIA COM COMPUTADORES
<ul style="list-style-type: none"> • Deve-se considerar todos os níveis, desde iniciantes na informática à profissionais da área.

Fonte: Autor, 2021.

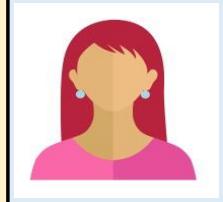
Personas são úteis para proporcionar uma situação fictícia concreta. Com estes personagens, é possível imaginá-lo utilizando o sistema, e através de sua perspectiva obter erros ou dificuldades no uso do software que podem ser corrigidos antes mesmo da primeira implementação.

Quadro 43 - Persona de Gerente Geral

	Laercio Rossi Alves, 31 anos. Brasileiro, Casado. Possui formação em Processos Gerenciais e experiência com hotelaria. Profissão atual: Gerente geral de hotel. É uma pessoa calma, com fácil aprendizagem, comprometido, responsável .
OBJETIVO: Gerenciar o hotel de forma mais prática e com mais eficiência. DIFICULDADES : Conhecimento de estratégias para divulgação online; Tempo para desenvolver e gerenciar o hotel com mais praticidade.	COMO PODEMOS AJUDAR : <ul style="list-style-type: none"> • Sistema prático para gestão do hotel; • Sistema integrado com as principais tarefas; • Proporcionar aos hóspedes uma maneira fácil, rápida e prática de fazerem suas reservas; • Para os funcionários do hotel, um sistema prático com as tarefas necessárias do dia a dia de cada um facilitando o trabalho. OJBECÕES: Não ter de treinar os funcionários a utilizar o sistema, por isso deve ser prático

Fonte: Autor, 2021.

Quadro 44 - Persona de Hóspede

	Anna Ribeiro de Queiros, 25 anos. Brasileira, Solteira. Possui formação em Pedagogia. Profissão atual: Professora de Pré-escola. Uma pessoa extrovertida, porém, calma, responsável, comprometida, gosta de aprender coisas novas e ensinar.
OBJETIVOS: Não perder tempo e trabalhar da melhor maneira possível. Aprender cada vez mais com novas experiências. DIFICULDADES : Tirar um tempo para descansar.	COMO PODEMOS AJUDAR : Anna está a procura de um hotel para passar uma semana e descansar um pouco, já que está de férias. Mas ao mesmo tempo que deseja um lugar calmo para descansar, ela quer um local que caso seja preciso, também possa ter tranquilidade para montar suas aulas, para assim que retornar a escola já ter material pronto para os alunos.

Fonte: Autor, 2021.

Particularmente, o uso de personas pode ajudar a focar determinadas questões. Por exemplo, uma mulher idosa com artrite pode ser uma das personas, ressaltando, com isso, questões de acesso e a interação dos deficientes físicos com a tecnologia. Por fim, em se tratando de cenários, é importante fornecer um contexto muito rico. Os princípios que orientam a criação de cenários são pessoas, atividades, contextos e tecnologias. (BENYON, 2011, p. 66).

7.2 Protótipos de interface.

Estando finalizado o estudo do usuário, os dados coletados serão utilizados para gerar os protótipos de interface. Esses protótipos são pensados já com caminhos de execução para realização de tarefas, e serão submetidos a Testes de Caixa Preta, um método onde o usuário final tenta realizar suas tarefas utilizando os protótipos de interface, preferencialmente sem nenhuma ajuda dos desenvolvedores assistindo. Como método para validar e inspecionar estas interfaces, serão utilizadas reuniões Walkthrough, onde nas reuniões de revisão o Gerente do Projeto atuará como Autor do Produto, o Gerente de Sistema como Revisor, o Analista de Sistema como Líder da sessão e o Time de Desenvolvimento como Secretários, auxiliando nos caminhos e possibilidades para os testes. A seguir, constam os protótipos de interface desenvolvidos para cada sistema.

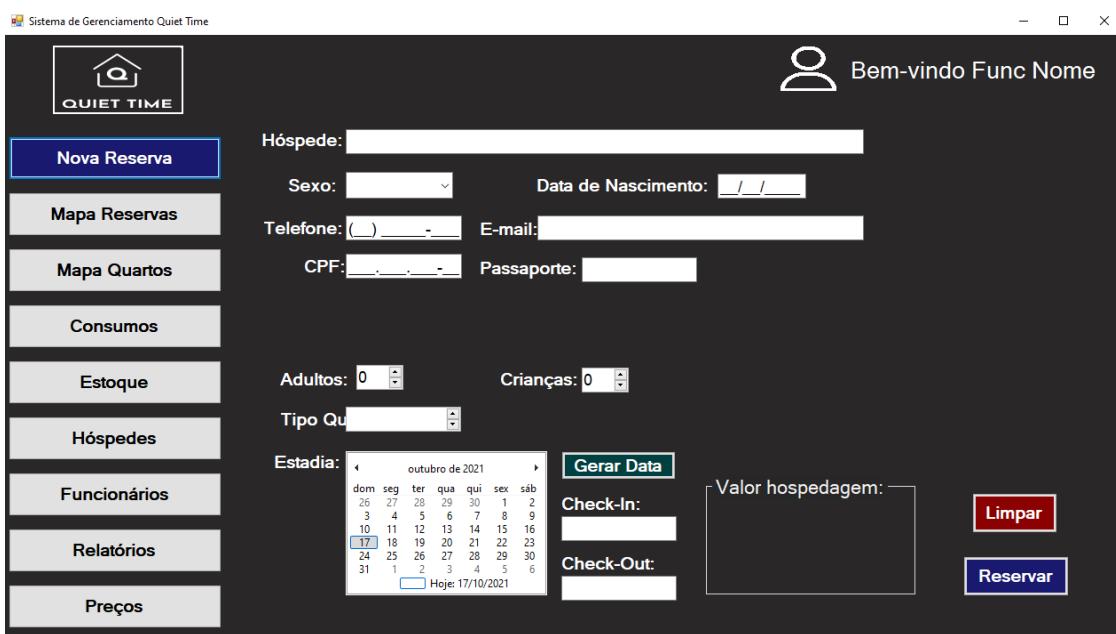
7.2.1 Interfaces dos sistemas Desktop, Web e Mobile.

Abaixo constam apenas algumas interfaces de cada sistema para apresentação do estilo de cores e organização de botões. As interfaces em sua totalidade podem ser conferidas no Apêndice B – Manual de Uso dos Softwares.

Figura 40 – Interface nova reserva em tela cheia (desktop responsivo).

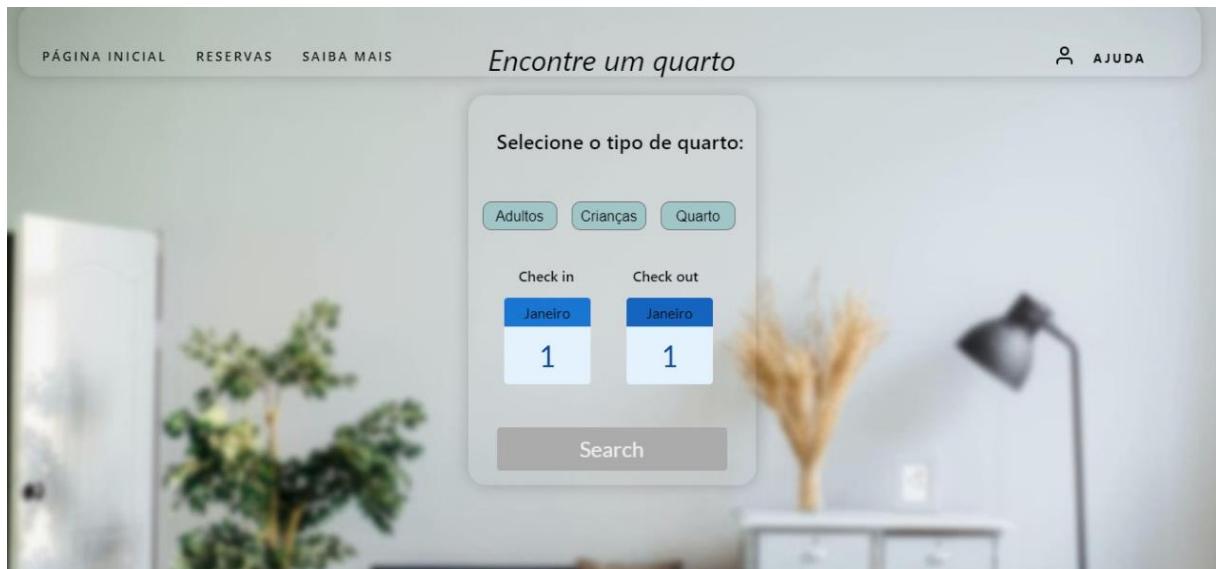
Fonte: Autor, 2021.

Figura 41 - Interface para registrar nova reserva.



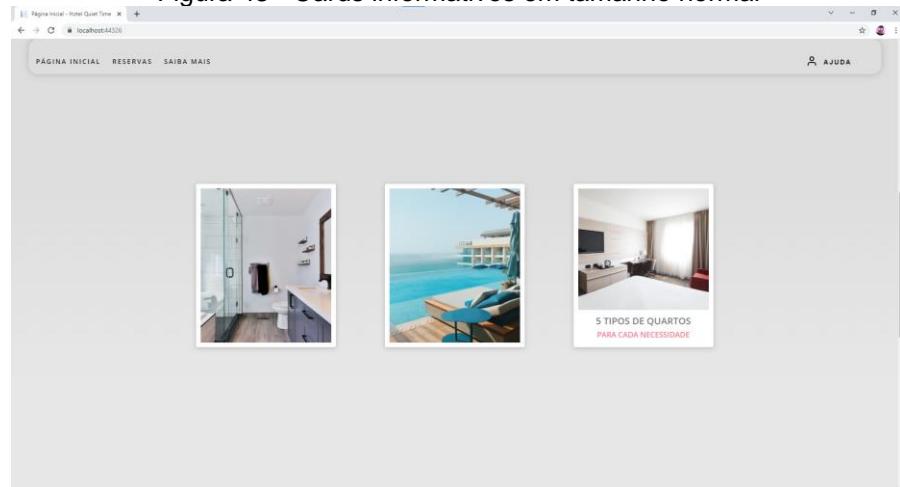
Fonte: Autor, 2021.

Figura 42 - Página inicial web seção de pesquisar reserva



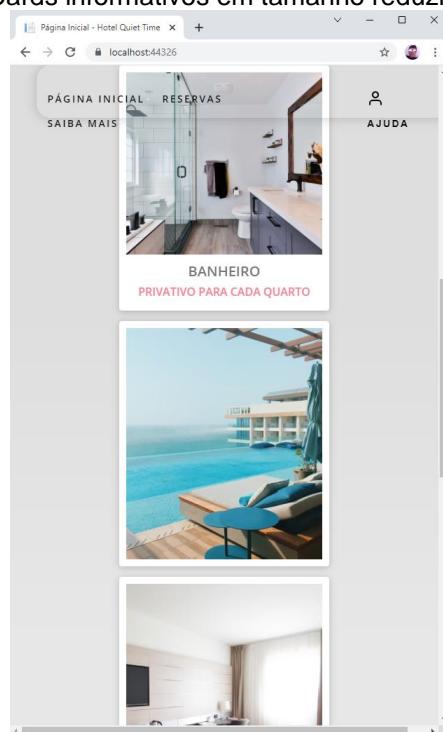
Fonte: Autor, 2021.

Figura 43 - Cards informativos em tamanho normal



Fonte: Autor, 2021.

Figura 44 - Cards informativos em tamanho reduzido responsivo



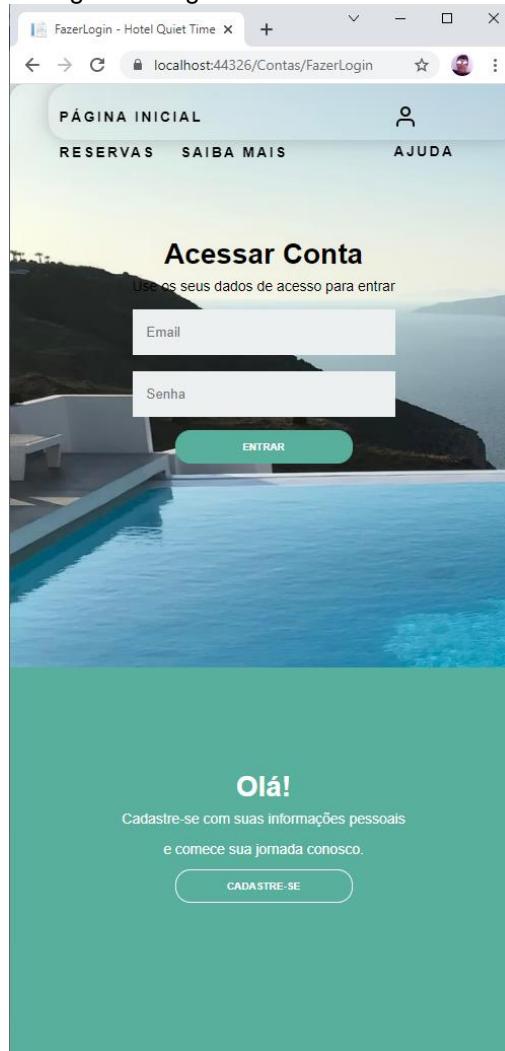
Fonte: Autor, 2021.

Figura 45 - Página de login em tamanho normal



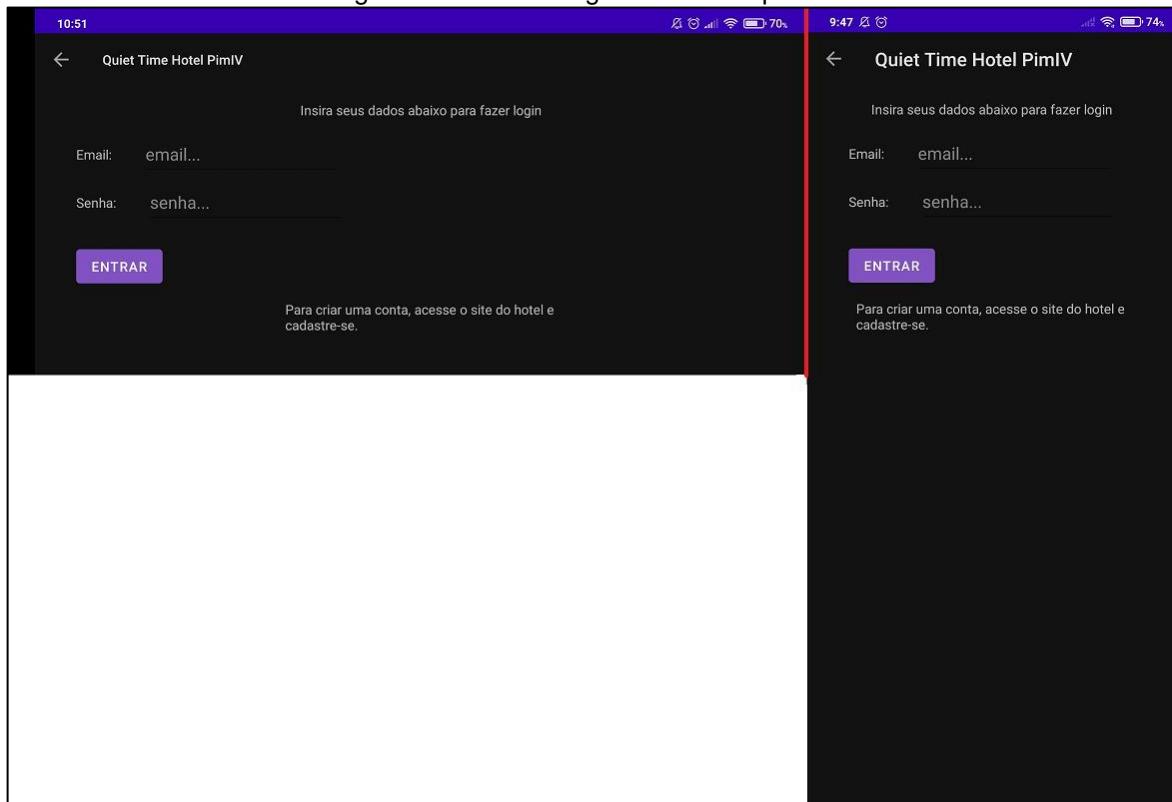
Fonte: Autor, 2021.

Figura 46 - Página de login em tamanho reduzido responsivo



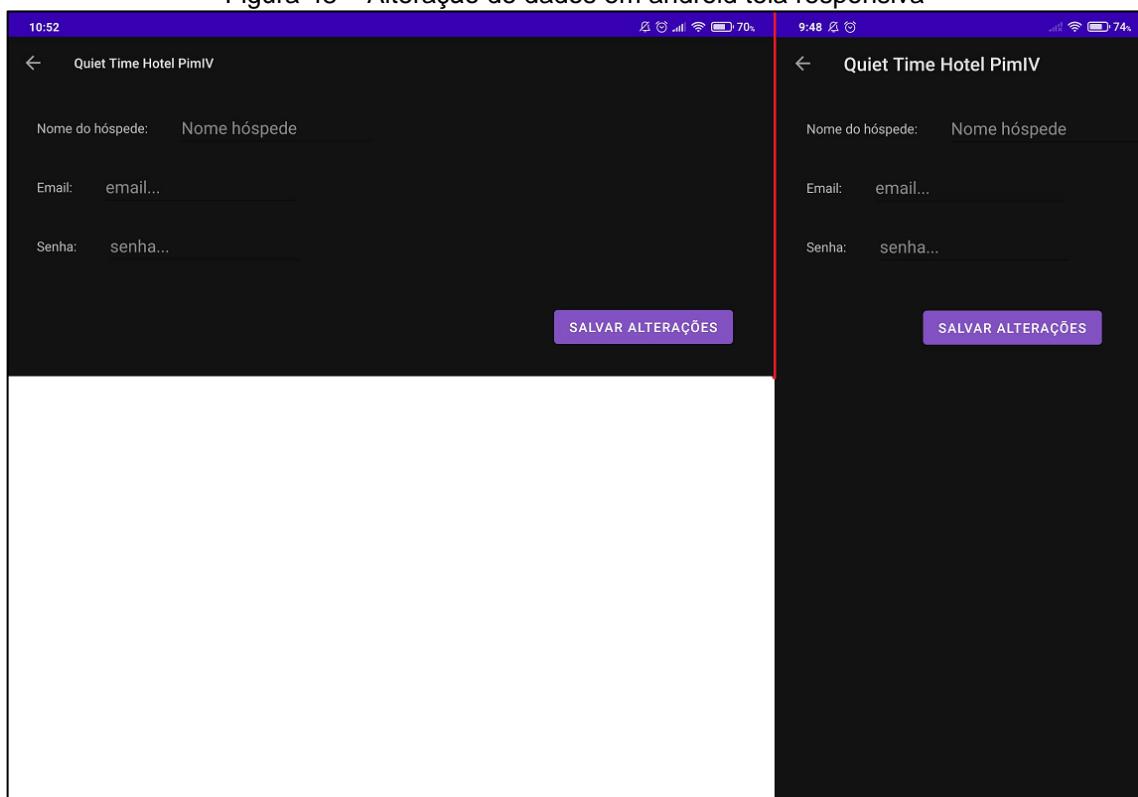
Fonte: Autor, 2021.

Figura 47 - Tela de login android responsiva



Fonte: Autor, 2021.

Figura 48 – Alteração de dados em android tela responsiva



Fonte: Autor, 2021.

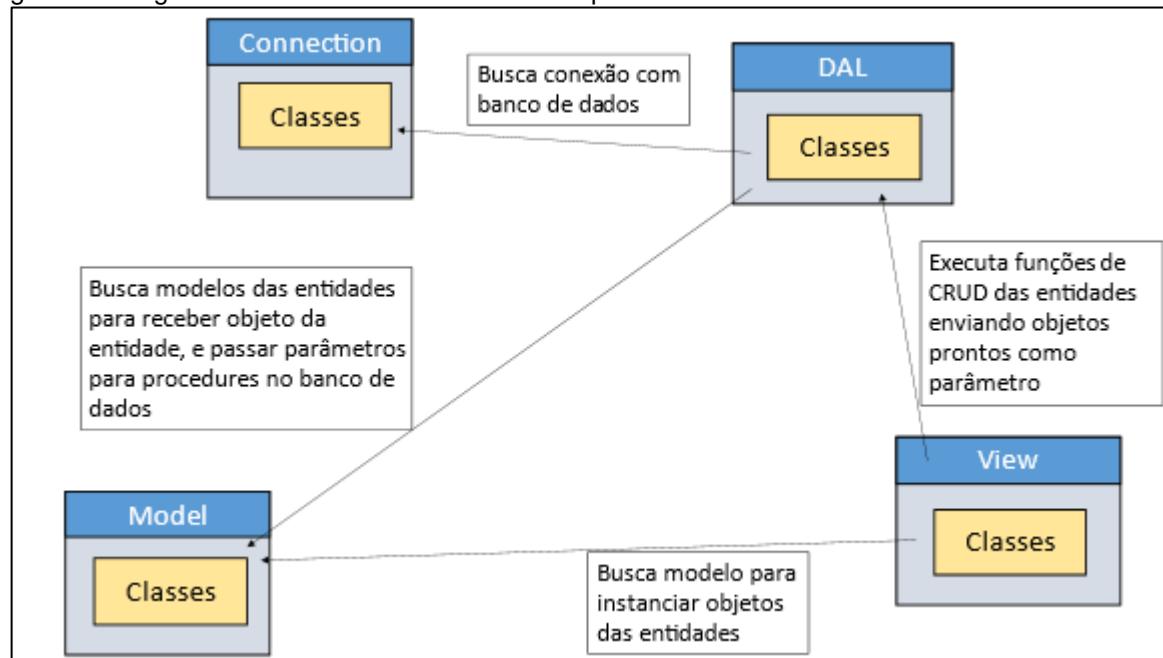
8 CODIFICAÇÃO DOS SISTEMAS

A seguir constam comentários sobre o desenvolvimento dos sistemas.

8.1 Aplicação Desktop

O projeto criado no ambiente de desenvolvimento Visual Studio foi do tipo Aplicativo Windows Forms com .NET Framework. Foi dividido em cinco camadas, sendo duas com, e três sem interface (DLLs). O padrão arquitetural utilizado, MVC, resultou na seguinte lógica de utilização de camadas:

Figura 49 - Lógica das camadas do sistema desktop e web



Fonte: Autor, 2021.

No projeto representado por **Connection**, constará classe de acesso ao banco de dados. Ela contém um método que retorna uma conexão já aberta com o SQL Server, banco selecionado no projeto.

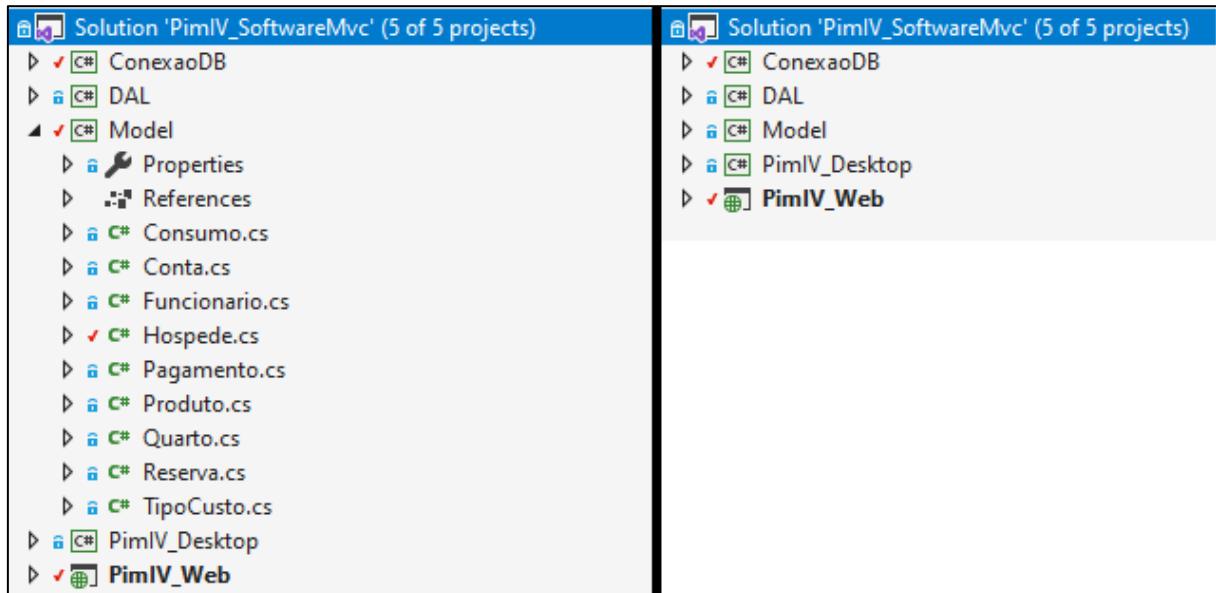
No projeto **Model**, constarão as classes que representam as entidades e seus comportamentos.

No projeto **DAL**, constarão as classes DAO de cada entidade, com os métodos de CRUD (Create, Read, Update, Delete), que representam ações de Inserção, Leitura, Atualização e Exclusão no banco de dados. Cada método recebe a conexão já aberta da classe de conexão, conforme a figura, executa suas instruções e fecha a conexão.

No projeto **View**, constarão as interfaces de usuário. Foram utilizados classes Form e User Control na interface desktop, para organização de botões e telas para cada atividade.

Na figura a seguir é possível identificar os cinco projetos na solução do Visual Studio, de modo que os códigos escritos nas DLLs, podem ser usados por ambas as interfaces Desktop e Web, pois ambas usam a linguagem c#.

Figura 50 - Gerenciador de solução com os projetos



Fonte: Autor, 2021.

Todos os métodos de classes DAO utilizam estrutura “try catch”, com o catch exibindo a mensagem do erro, conforme no exemplo da Figura 51.

Figura 51 - Código com tratamento de erro

```
try
{
    DataTable tabela = new DataTable();
    using(SqlConnection conn = _conexao.AbrirConexao())
    {
        SqlCommand procedure = new SqlCommand("HOSP_ListarStatus", conn);
        procedure.CommandType = CommandType.StoredProcedure;

        procedure.Parameters.Add("@Status", SqlDbType.VarChar).Value = st
    }

    SqlDataAdapter da = new SqlDataAdapter(procedure);
    da.Fill(tabela);
}

return tabela;
}
catch(Exception err)
{
    throw new Exception(err.Message);
}
```

Fonte: Autor, 2021.

A classe que conecta ao banco utiliza o meio por “string” de conexão. Para devido funcionamento, é preciso que o nome do servidor, em “Data Source”, seja correto com o computador executando o programa via Visual Studio. O catálogo inicial também deve estar correto, mas o script de criação do banco no Apêndice D já está alinhado com tal.

Figura 52 - string de conexão com banco de dados



```

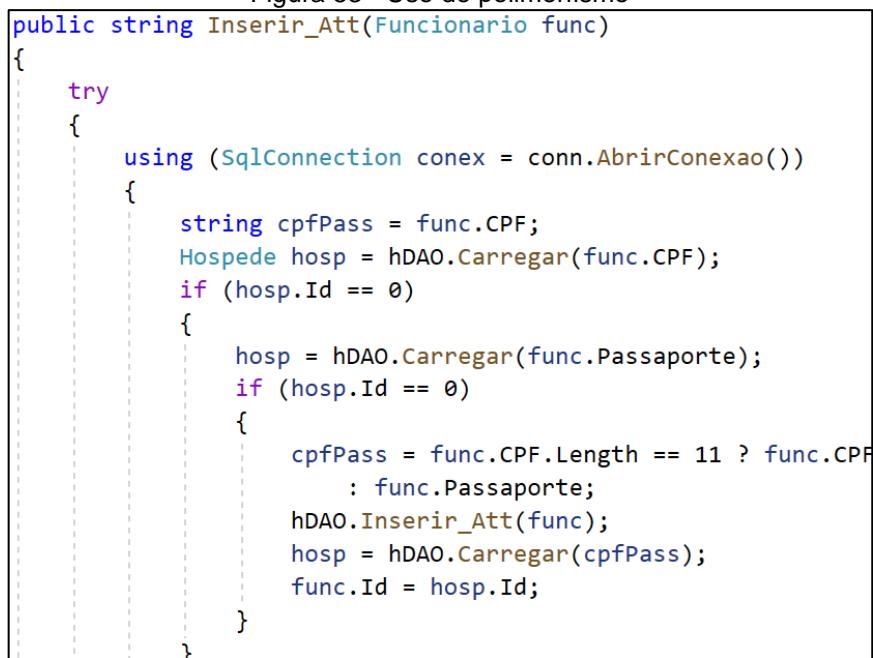
public class ConexaoSQLServer
{
    public SqlConnection AbrirConexao()
    {
        try
        {
            string connString = "Data Source=DESKTOP-6FNB3PC\\SQLEXPRESS";
            SqlConnection conexao = new SqlConnection(connString);
            conexao.Open();
            return conexao;
        }
        catch (Exception)
        {
            return null;
        }
    }
}

```

Fonte: Autor, 2021.

Um grande e poderoso diferencial da orientação a objetos é o polimorfismo, que pode ser entendido como “uma classe sendo tratada com outra, quando há relação de herança entre si”. A classe Funcionário herda de Hóspede neste projeto. Para verificar se um funcionário já está registrado, sendo então feita a conexão com seus dados no banco, foi utilizado o método “Carregar”, que recebe um objeto do tipo Hóspede, passando um objeto do tipo Funcionário. A lógica funcionou corretamente graças ao polimorfismo da linguagem c#, conforme a Figura 53.

Figura 53 - Uso de polimorfismo



```

public string Inserir_Att(Funcionario func)
{
    try
    {
        using (SqlConnection conex = conn.AbrirConexao())
        {
            string cpfPass = func.CPF;
            Hospede hosp = hDAO.Carregar(func.CPF);
            if (hosp.Id == 0)
            {
                hosp = hDAO.Carregar(func.Passaporte);
                if (hosp.Id == 0)
                {
                    cpfPass = func.CPF.Length == 11 ? func.CPF
                        : func.Passaporte;
                    hDAO.Inserir_Att(func);
                    hosp = hDAO.Carregar(cpfPass);
                    func.Id = hosp.Id;
                }
            }
        }
    }
}

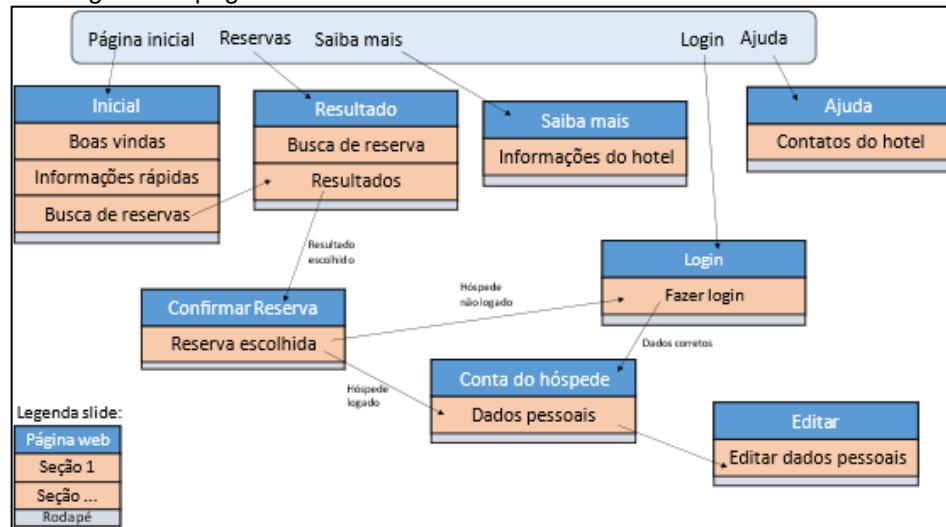
```

Fonte: Autor, 2021.

8.2 Aplicação Web.

O tipo do projeto para a interface web foi Aplicativo Web ASP.NET com .NET Framework. O projeto já carregava por padrão uma arquitetura em MVC própria, com classes “controladoras” pré-preparadas para tratar as requisições e respostas. A lógica para as páginas web foi a seguinte:

Figura 54 - Lógica das páginas web

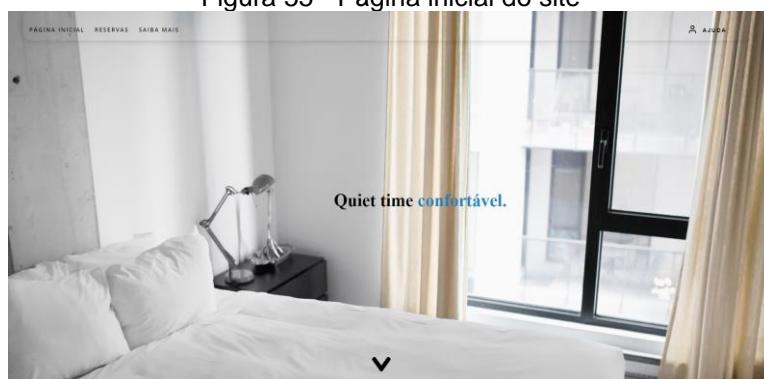


Fonte: Autor, 2021.

Em caso de **teste da interface web**, é recomendado realizar os procedimentos “clean solution” (Limpar solução) e logo após “rebuild solution” (Recompilar solução) do Visual Studio, para evitar erros detectados durante desenvolvimento, que foram resolvidos da mesma forma.

Foram desenvolvidas páginas em HTML e CSS. Algumas utilizam um pouco de Javascript para implementação de detalhes estéticos, como na página inicial, que contém uma animação em Javascript que troca a palavra após “Quiet Time”, mostrando outros adjetivos e com cores diferentes para cada palavra, conforme a Figura 55.

Figura 55 - Página inicial do site



Fonte: Autor, 2021.

Na pasta Views ficam armazenadas os arquivos HTML. Este tipo de projeto com o modelo MVC citado, traz suporte aos arquivos de extensão “cshtml”, o que significa que é possível utilizar linguagem C# para codificar algumas lógicas das páginas. Conforme a Figura 56, foi utilizado esse mecanismo para alterar o comportamento de um link. Se o usuário estiver na página inicial, a página apenas desce para a seção de busca. Se estiver em qualquer outra página, é feito redirecionamento para a página de reservas.

Figura 56 - Uso de C Sharp junto de HTML

```
@if (ViewBag.Title == "Página Inicial")
{
    <li class="active"><a href="#Reservas"> Reservas</a> </li>
}
else
{
    <li class="active">@Html.ActionLink("Reservas", "Index", "Reservas")</li>
}
```

Fonte: Autor, 2021.

Já na Figura 57 , costa o carregamento de uma lista com os tipos de quartos vinda do banco de dados, e o uso de C# para inserir os nomes desses tipos em um “select” HTML, nomes estes que são usados como parâmetros em buscas por reserva, e por isso devem estar corretos.

Figura 57 - Conexão com banco em página HTML

```
@using DAL;
@{
    ViewBag.Title = "Pesquisar Reservas";
    TipoCustoDAO tDAO = new TipoCustoDAO();
    List<string> tipos = tDAO.ListarTiposQuarto();
}

<select class="text_input w--whole up1" name="tQuarto">
    <option label="Camas"></option>
    <option value="@tipos[0].ToString()">@tipos[0].ToString()</option>
    <option value="@tipos[1].ToString()">@tipos[1].ToString()</option>
    <option value="@tipos[2].ToString()">@tipos[2].ToString()</option>
    <option value="@tipos[3].ToString()">@tipos[3].ToString()</option>
    <option value="@tipos[4].ToString()">@tipos[4].ToString()</option>
</select>
```

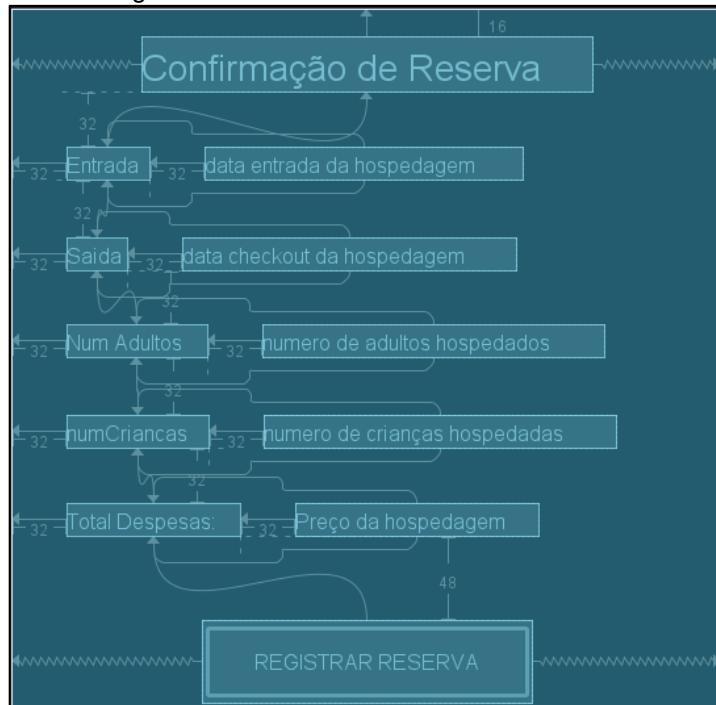
Fonte: Autor, 2021.

8.3 Aplicação Android.

Durante a execução do projeto, foi decidida a priorização nas interfaces desktop e web. Portanto, apenas protótipos de tela foram feitos para a aplicação android.

O design das interfaces e a organização de botões foram feitos por meio de constraints, garantindo que os elementos fiquem “amarrados” da mesma forma mesmo com o dispositivo em modo paisagem (responsividade).

Figura 58 - Constraints de interface android



Fonte: Autor, 2021.

Foram utilizadas strings xml para a adição de textos nas interfaces, evitando que o texto de botões e elementos semelhantes sejam “hardcoded”.

Figura 59 - strings para uso em android

```
<string name="dataEntrada">Entrada</string>
<string name="dataSaida">Saida</string>
<string name="qtAdultos">Num Adultos</string>
<string name="qtCrianças">numCrianças</string>
<string name="valorDespesas">Total Despesas:</string>
<string name="botaoPesquisar">Pesquisar</string>
```

Fonte: Autor, 2021.

CONCLUSÃO

A tecnologia tem grande impacto sobre a definição de qualidade de uma empresa, portanto possuir canais eficientes de atendimento e autoatendimento tornam-se essenciais.

Durante o desenvolvimento do presente trabalho foi possível identificar quão necessário se faz a criação de um software para gestão, ainda mais quando **se trata de prestação de serviço direta aos clientes**. Com o desenvolvimento do sistema de gerência, foi possível identificar que o tempo, considerado algo precioso para as empresas, quando bem administrado torna-se um **item** de destaque em meio ao mercado.

Com o uso de um sistema para administrar o funcionamento do hotel o tempo de serviço é reduzido e isso pode ser algo decisivo na decisão da contratação do serviço pelo cliente. O cliente ganha tempo, podendo realizar as reservas de forma autônoma ou rapidamente no próprio hotel, ***uma vez que o tempo gasto preenchendo papéis e aguardando o atendimento não existirá (ou será reduzido)***. Outra vantagem considerável é sem dúvidas o fato de que o cliente, ao terminar sua hospedagem, poderá facilmente finalizar sua reserva e o hotel com o sistema de gerência, poderá visualizar os itens consumidos de forma prática e organizada, mantendo o controle de estoque que será atualizado constantemente. As atividades rotineiras do hotel serão facilitadas uma vez que será possível a obtenção dos relatórios gerenciais, auxiliando em uma administração correta e atualizada com constância.

Existem diversas qualidades em realizar a implantação de um sistema conforme mencionado anteriormente, mas de nada adiantaria um sistema complexo demais ou funcionários e clientes que não estejam empenhados em contribuir com o seu funcionamento. Portanto entendemos que deve existir um equilíbrio para que as coisas funcionem corretamente e para que, no caso atual, o hotel consiga obter os resultados esperados, sendo necessário a criação de um sistema que aborda os menores detalhes, como por exemplo, ícones de fácil compreensão nos sistemas de autoatendimento, uma aplicação desktop com todas as funcionalidades necessárias nas atividades realizadas diariamente pelos funcionários e que ainda assim, seja prático. A junção desses itens resultará um retorno positivo ao hotel, garantindo que o seu serviço tenha uma melhora considerável.

O resultado que realmente marca esse projeto, é sem dúvida o aprendizado, a experiência. As experiências adquiridas pelos integrantes da equipe foram de grande valia para projetos futuros, pontos como o trabalho em equipe, a experiência na parte de análise e projeto de sistemas voltados para desktop e web e banco de dados. Portanto, o principal resultado obtido durante esse projeto foi o crescimento profissional de todos.

REFERÊNCIAS

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de Software**: uma abordagem profissional. 8^a edição. Porto Alegre: AMGH, 2016.

MACHADO, Felipe Nery Rodrigues. **Banco de dados**: projeto e implementação. 4^a edição. São Paulo: Érica, 2020.

MOLINARI, L. **Testes de software**: produzindo sistemas melhores e mais confiáveis. São Paulo: Érica, 2003.

MEDEIROS, Ernani Sales de. **Desenvolvendo software com UML 2.0**: definitivo. São Paulo: Pearson Makron Books, 2004.

FOWLER, Martin. **UML essencial**: um breve guia para a linguagem-padrão de modelagem de objetos. 3^a edição. Porto Alegre: Bookman, 2005.

BENYON, David. **Interação humano-computador**. 2^a edição. São Paulo: Pearson Prentice Hall, 2011.

KERR, organizador Eduardo Santos. **Gerenciamento de requisitos**. São Paulo: Pearson Education do Brasil, 2015.

Lei Geral de Proteção de Dados. Disponível em: <https://www.lgpdbrasil.com.br/>. Acesso em 02/05/2021.

LGPD NA HOTELARIA: quais as mudanças para seu hotel? Disponível em: <https://hotelariaweb.com/lgpd-na-hotelaria-quais-as-mudancas-para-seu-hotel/>. Acesso em 02/05/2021.

Sistemas para hotelaria: tudo que você precisa saber. Blog Hospedin, 2021. Disponível em: <http://blog.hospedin.com/sistemas-para-hotelaria/> Acesso em: 26/03/2021.

GLOSSÁRIO

Algoritmo:	Sequência lógica de passos ou ações para um determinado objetivo.
Arcabouço:	Esqueleto ou estrutura que envolve os componentes de algo.
Artefato:	Em software, significa subproduto produzido durante o desenvolvimento.
CI:	Continuous Integration. Melhoria contínua do produto de software.
CD:	Continuous Delivery. Entrega contínua do produto de software.
Desktop:	Área de trabalho. Softwares funcionais em computadores pessoais.
DevOps:	Metodologia que integra Desenvolvimento e Operação.
Feedback:	Reação a um estímulo. Retorno avaliativo à alguma informação.
Implantação:	Fase onde o software pronto é entregue para utilização do cliente.
Implementação:	Fase onde é escrito e testado o código do software.
Kanban:	“Cartão” em japonês. Técnica utilizada para organizar tarefas.
Login:	Processo de acesso por meio de autenticação ou identificação.
Mobile:	Móvel. Softwares funcionais em dispositivos móveis.
Product Owner:	Cargo da metodologia Scrum, responsável por organizar os projetos.
Software:	Programa de computador. Sistema lógico utilizado por meio de interfaces.
Scrum:	Metodologia para desenvolvimento de software de forma ágil.
Scrum Master:	Cargo da metodologia Scrum, responsável por organizar a equipe.
Sprint:	Período determinado de tempo para foco em alguma tarefa.
Stakeholder:	Envolvido, parte interessada no projeto.
Walkthrough:	Tipo de reunião entre envolvidos no desenvolvimento para fim de testes.
Web:	Teia ou rede. Representa softwares executados na internet.

APÊNDICE A – DICIONÁRIO DE DADOS

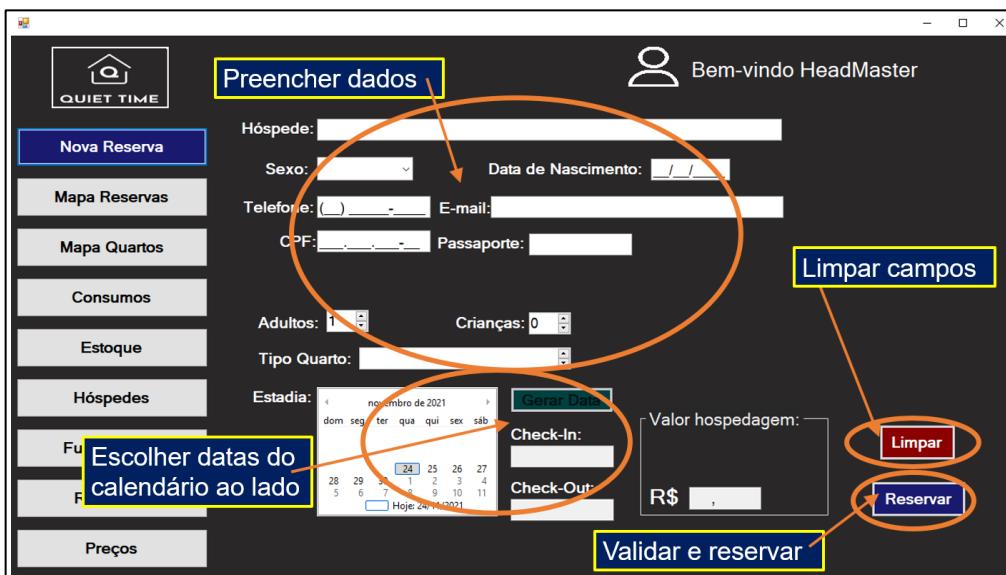
Tabela	Nome Coluna	Tipo de Dados	Tamanho	Restrições	Descrição
HOSPEDE	IDHOSPEDE	Inteiro		PK	Número de identificação do Hóspede.
	NOME	Caracteres	50	NOT NULL	Nome do Hóspede.
	EMAIL	Caracteres	40	NOT NULL	E-mail do Hóspede.
	TELEFONE	Caracteres	15	NOT NULL	Telefone do Hóspede.
	SEXO	Caractere	1	NOT NULL	Gênero do Hóspede.
	PASSAPORT E	Caracteres	10	NULL	Passaporte do Hóspede.
	CPF	Caracteres	11	NULL	CPF do Hóspede.
	DATANASC	Data		NOT NULL	Data de nascimento do Hóspede.
	HOSP_STAT US	Inteiro	1	NOT NULL	Status do Hóspede.
	DATAATT	Data		NOT NULL	Última atualização.
FUNCIONARI O	IDFUNCIONA RIO	Inteiro		PK	Número de identificação do Funcionário.
	ID_HOSPEDE	Inteiro		FK, NOT NULL	Número de identificação do Hóspede.
	ID_DEPARTAMENTO	Inteiro		FK, NOT NULL	Número de identificação do Departamento .
	SALÁRIO	Float			Valor do salário do Funcionário.

	FUNC_STATUS	Inteiro	1	NOT NULL	Status do Funcionário.
	DATAATT	Data		NOT NULL	Última atualização.
CONTALOGIN	IDLOGIN	Inteiro		PK	Número utilizado para Login no sistema.
	ID_FUNCIONARIO	Inteiro		FK	Número de identificação do Funcionário.
	ID_HOSPEDE	Inteiro		FK	Número de Identificação do Hóspede.
	USUARIO	Caracteres	30	NOT NULL	Nome do usuário à logar no Sistema.
	SENHA	Caracteres	50	NOT NULL	Senha única do usuário para logar no Sistema.
	DATAATT	Data		NOT NULL	Última atualização.
QUARTO	IDQUARTO	Inteiro		PK	Número de identificação do Quarto.
	TIPO	Inteiro		NOT NULL	Tipo do Quarto.
	QUARTO_STATUS	Caractere	1	NOT NULL	Status do Quarto.
	NUMERO	Inteiro		NOT NULL	Número do Quarto.
RESERVA	IDRESERVA	Inteiro		PK	Número de identificação da Reserva.
	ID_HOSPEDE	Inteiro		FK, NOT NULL	Número de identificação do Hóspede.

	ID_QUARTO	Inteiro		FK, NOT NULL	Número de identificação do Quarto.
	DATACHECKIN	Data		NOT NULL	Data de Check-In.
	DATACHECKOUT	Data		NOT NULL	Data de Check-Out.
	DESPESAS	Float		NOT NULL	Dado reservado para adicionar possíveis despesas.
	ADULTOS	Inteiro		NOT NULL	Quantidade de pessoas Adultas na Reserva.
	CRIANCAS	Inteiro		NOT NULL	Quantidade de Crianças na Reserva.
	RESERVA_STATUS	Inteiro		NOT NULL	Status da Reserva.
PAGAMENTO	IDPAGAMENTO	Inteiro		PK	Número de identificação do Pagamento.
	ID_RESERVA	Inteiro		FK, NOT NULL	Número de identificação da Reserva.
	HORARIO	Data		NOT NULL	Horário da realização do Pagamento.
	VALOR	Float		NOT NULL	Valor do Pagamento.
	FORMA_PAGAMENTO	Caracteres	20	NOT NULL	Forma da realização do Pagamento.
PRODUTO	IDPRODUTO	Inteiro		PK	Número de Identificação do Produto.

	NOME	Caracteres	20	NOT NULL	Nome do Produto.
	QUANTIDAD E	Inteiro		NOT NULL	Quantidade do Produto.
	DATAVALIDA DE	Data		NOT NULL	Data de Validade do Produto.
	VALOR	Decimal	(5,2)	NOT NULL	Preço
	STATUSPRO D	Caracteres	7	NOT NULL	Ativo ou não.
CONSUMO	IDCONSUMO	Inteiro		PK	Número de identificação do Consumo.
	ID_RESERVA	Inteiro		FK, NOT NULL	Número de identificação da Reserva.
	ID_PRODUT O	Inteiro		FK, NOT NULL	Número de identificação do Produto.
	QUANTIDAD E	Inteiro		NOT NULL	Quantidade do Consumo.
	Momento	Data		NOT NULL	Data do consumo
	Total	Decimal	(5,2)	NOT NULL	Preço total
PRECO	IDPRECO	Inteiro		PK	Número de identificação do Preço.
	NOME_ITEM	Caracteres	30	NOT NULL	Nome do Item.
	PRECO	Float		NOT NULL	Valor do Item.
	DATAATT	Data		NOT NULL	Última atualização.

APÊNDICE B – MANUAL DE USO DOS SOFTWARES



Bem-vindo HeadMaster

Buscar reservas por:

RESERVA	CHECK-IN	CHECK-OUT	HOSPEDE	QUARTO
31	22/11/2021 19:41	22/11/2021 19:41	Steve Rogers Am...	8

Status: INICIADA
Periodo: Estava ativa em: 24/11/2021 Buscar

Duplo clique para acessar a reserva selecionada

Escolher modo de listagem e status para buscar

Escolher modo de listagem e data para buscar

Bem-vindo HeadMaster

Clique em um quarto para alterar seu status

Quarto	Reserva	Status	Tipo	Hospede
1		DISPONIVEL	1 solteiro	
2		DISPONIVEL	1 solteiro	
3		DISPONIVEL	1 solteiro	
6		DISPONIVEL	1 solteiro	
7		DISPONIVEL	2 solteiro	
8	31	OCCUPADO	2 solteiro	Steve Rogers Am...
9		DISPONIVEL	2 solteiro	

Duplo clique para selecionar o quarto a ser alterado

Escolher status e clicar em alterar

Bem-vindo HeadMaster

Dados do consumo

Reserva:	Produto:	Quantidade:	Total:	Salvar	Deletar
31	Agua mineral	2	R\$ 20		

Informar reserva ativa e produto ativo

Inserir dados do consumo

Duplo clique para carregar um consumo e editá-lo

Salvar novo consumo ou deletar existente

Bem-vindo HeadMaster

Dados do produto:	Nome do produto:	Valor:	Quantidade:	Validade:	Status:
	R\$ [campo]	0	24/11/2021	<input type="button" value="Salvar"/>	<input type="button" value="Excluir"/>

Inserir dados do produto

Id	Nome	Estoque	Valor	Validade	Status
1	Água mineral	33	10,00	07/06/2022	ATIVO
3	refrigerante ze...	94	9,50	21/12/2022	ATIVO
6	sorvete	195	30,00	21/02/2022	ATIVO
7	Chocolate barra	39	6,50	21/07/2022	ATIVO

Duplo clique para carregar um produto existente e editá-lo

Salvar novo produto ou deletar existente

Bem-vindo HeadMaster

Busca por hóspede			
CPF:	Status:	Buscar	<input type="button" value="Novo hóspede"/>
Bruno Tavante	M	14/09/1995	34534534522
Fabiana Souza	F	12/02/1995	12312312345
Diego Rocha	M	18/05/1995	19823423499
Matheus Pires	M	11/01/1995	89999922245
Mayara Oliveira	F	19/03/1995	99883334944
Pão de forma	F	19/12/1999	39458588599
Tomioka	M	24/10/1977	

Buscar lista pelo status ou carregar dados de hóspede pelo CPF

Copiar CPF e colar no campo CPF na área de busca

Acessar formulário com campos vazios

Bem-vindo HeadMaster

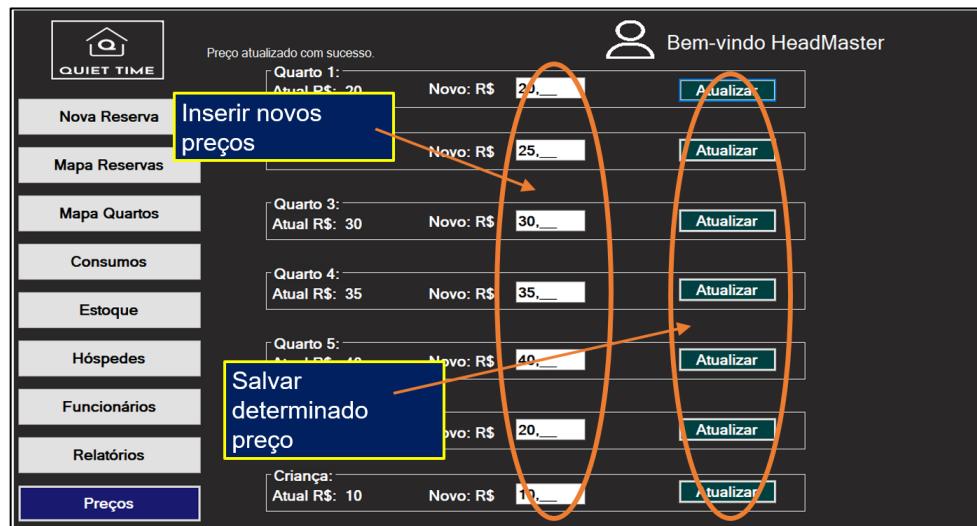
Nova Reserva

Cancelar edição e voltar para listar

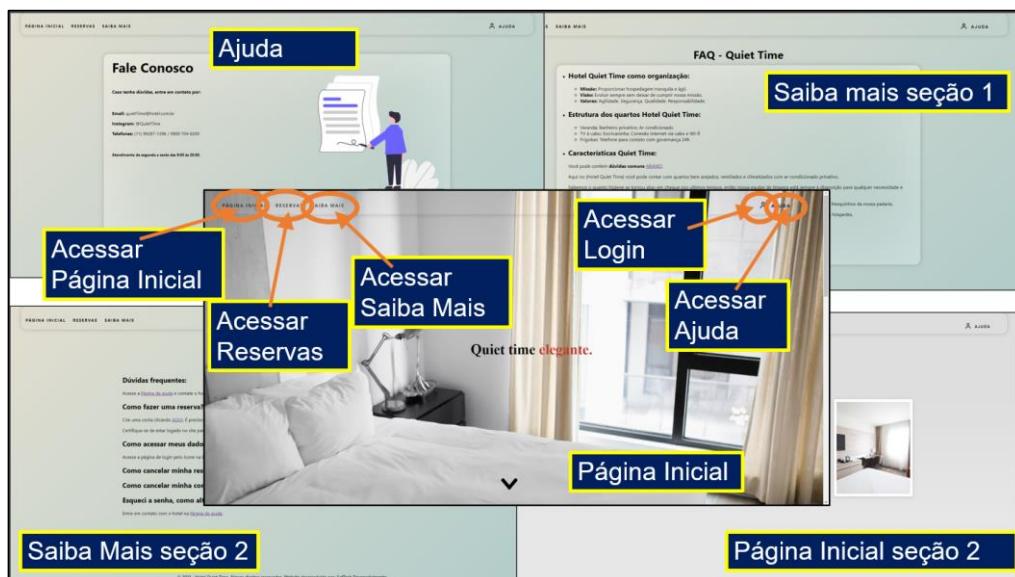
Preencher os dados

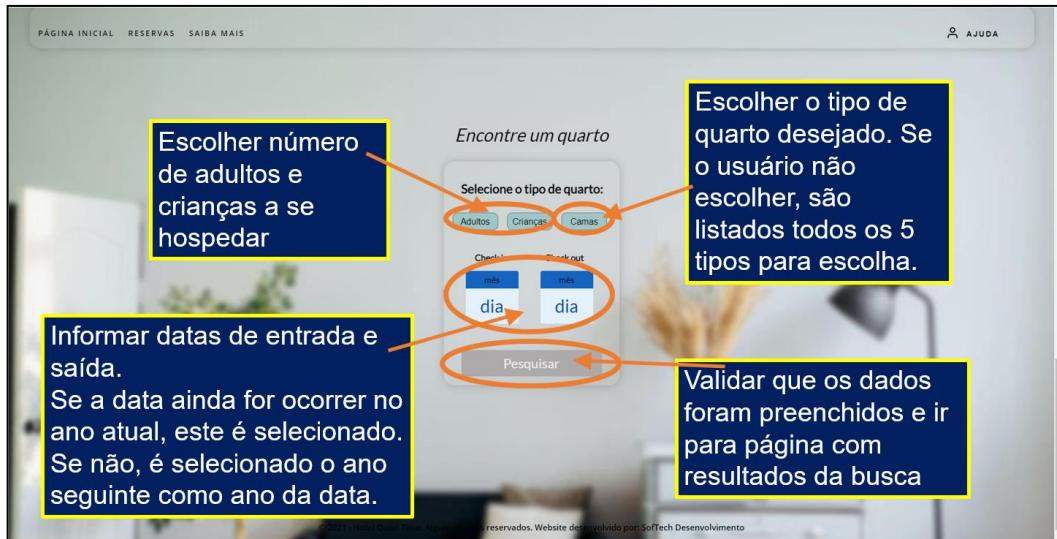
Alterar dados de login da pessoa

Limpar campos ou confirmar dados



A figura abaixo comporta 5 interfaces web “estáticas”, o que significa que seu conteúdo é apenas textual. A figura então demonstra o comportamento dos links da barra de navegação fixa em todas as páginas.





Após o clique em pesquisa, é carregada esta lista de reservas possíveis. Se o usuário tiver escolhido o tipo anteriormente, apenas tal tipo é apresentado

Clicando em reservar é verificado login. Se o usuário não estiver logado, é direcionado para página de login. Se estiver, é direcionado para página de confirmação.

Reservar

Reservar

Clicando em confirmar a reserva é registrada

Mudar tela para fazer login

Preencher dados para login

Acessar Conta

Olá!

Bem-Vind@!

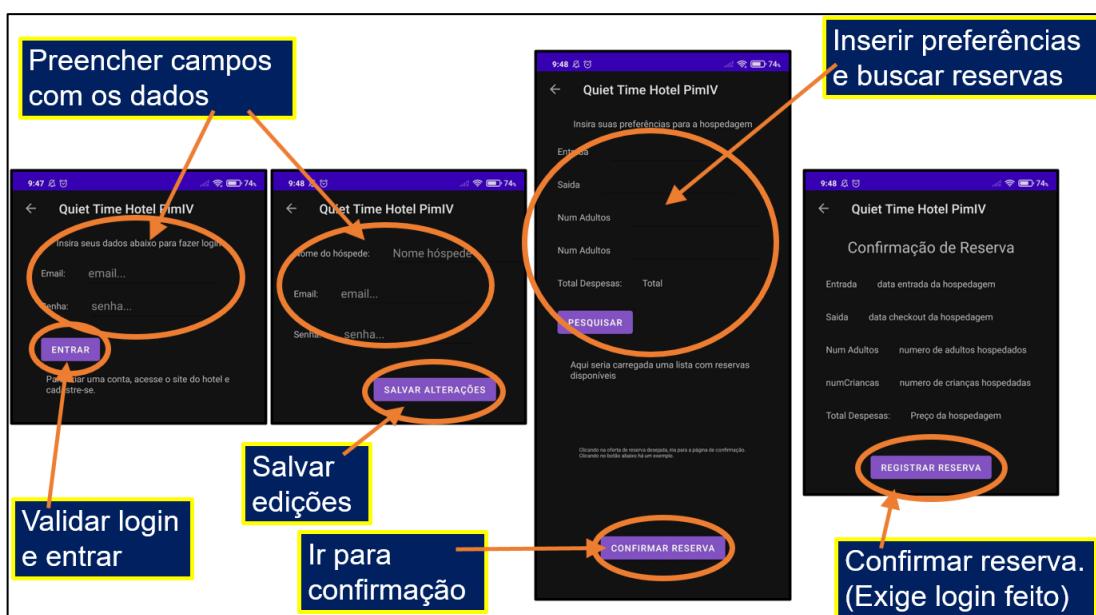
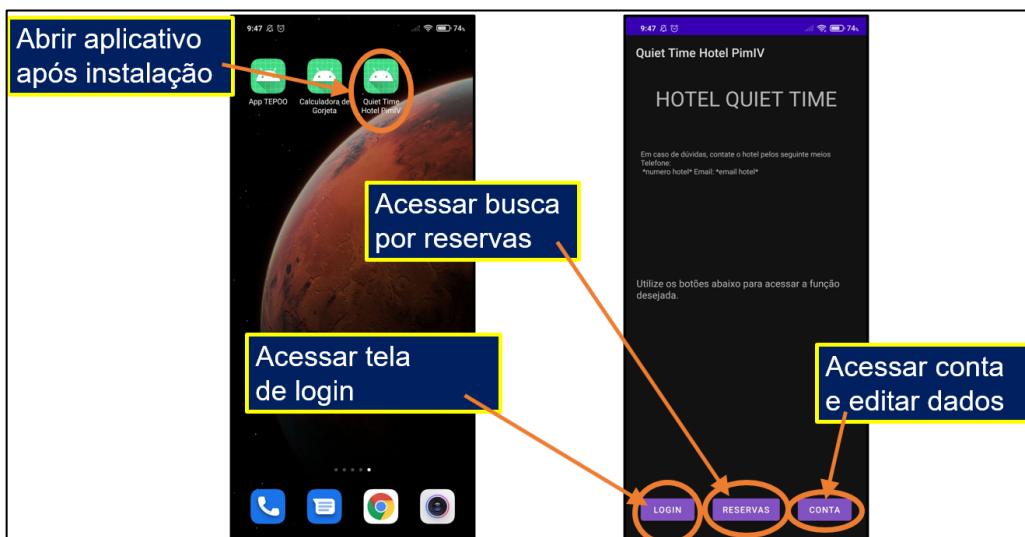
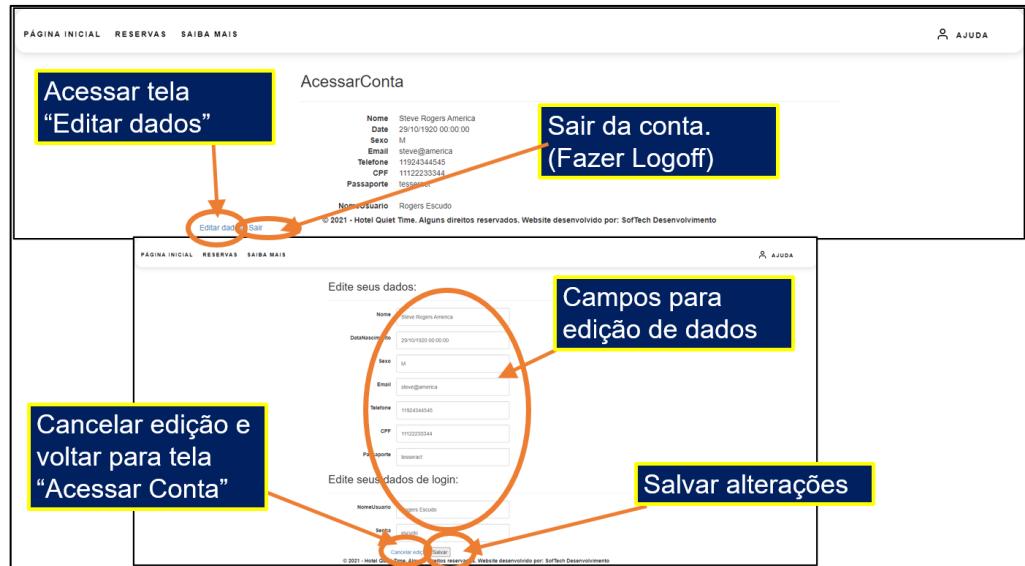
Para garantir sua reserva por favor, logue com seu e-mail preenchido.

Entrar

Mudar tela para cadastro de novo hóspede.

Preencher dados para novo cadastro

Criar Uma Conta



APÊNDICE C – LINKS PARA RELEASES COM CÓDIGOS E PROJETOS

A partir dos links abaixo é possível acessar os projetos completos, no mesmo estado em que foram apresentados neste trabalho.

Link para release do projeto Desktop e Web (Visual Studio, C#):

(O repositório diz que o projeto tem em maioria código Javascript, porém isso ocorre devido aos scripts automaticamente carregados no projeto web, devido ao modelo MVC utilizado. Os códigos escritos pela equipe foram em sua grande maioria, em C#.

https://github.com/BRDiego/PimIV_SoftwareMvc/releases/tag/v1.0.0

Link para release do projeto Mobile Android (Android Studio, Java):

<https://github.com/BRDiego/PimIVprojAndroid/releases/tag/v1.0.0>

APÊNDICE D – SCRIPTS DE CRIAÇÃO DO BANCO DE DADOS

```

Create database PimIVDev;
Use PimIVDev
Create table Hospede(
    Id INTEGER IDENTITY,
    Nome VARCHAR(50) NOT NULL,
    DataNasc DATE NOT NULL,
    Sexo CHAR(1) NOT NULL,
    Email VARCHAR(50) NOT NULL,
    Telefone VARCHAR(12) NOT NULL,
    CPF CHAR(11) NULL,
    Passaporte VARCHAR(12) NULL,
    StatusHosp VARCHAR(7) NOT NULL DEFAULT 'ATIVO',
    DataAtualizacao DATETIME NOT NULL DEFAULT GETDATE(),
    CONSTRAINT PK_Hospede PRIMARY KEY (Id)
);
Create table Funcionario(
    Id INTEGER IDENTITY,
    Cargo VARCHAR(20) NULL,
    Salario DECIMAL(8,2) NULL,
    StatusFunc VARCHAR(7) NOT NULL DEFAULT 'ATIVO',
    DataAtualizacao DATETIME NOT NULL DEFAULT GETDATE(),
    IdHospede INTEGER NOT NULL,
    CONSTRAINT PK_Funcionario PRIMARY KEY (Id),
    CONSTRAINT FK_FuncIdHospede FOREIGN KEY (IdHospede) REFERENCES
HOSPEDE(Id)
);
Create table ContaLogin(
    Id INTEGER IDENTITY,
    NomeUsuario VARCHAR(20) NOT NULL,
    Senha VARCHAR(20) NOT NULL,
    DataAtualizacao DATETIME NOT NULL,
    IdHospede INTEGER NULL,
    IdFuncionario INTEGER NULL,
    CONSTRAINT PK_ContaLogin PRIMARY KEY (Id),
    CONSTRAINT FK_ContalIdHospede FOREIGN KEY (IdHospede) REFERENCES
Hospede(Id),
    CONSTRAINT FK_ContalIdFuncionario FOREIGN KEY (IdFuncionario) REFERENCES
Funcionario(Id)
);
Create table TipoCusto(
    Id INTEGER IDENTITY,
    NomeTipo VARCHAR(30) NOT NULL,
    Preco DECIMAL (8,2) NOT NULL,
    DataAtualizacao DATETIME NOT NULL,
    CONSTRAINT PK_TipoCusto PRIMARY KEY (Id)
);
Create table Quarto(
    Id INTEGER IDENTITY,
    Numero INTEGER NOT NULL,
    StatusQuarto VARCHAR(10) NOT NULL DEFAULT 'DISPONIVEL',
    IdTipo INTEGER NOT NULL,
    CONSTRAINT PK_Quarto PRIMARY KEY (Id),
    CONSTRAINT FK_QuartoTipo FOREIGN KEY (IdTipo) REFERENCES TipoCusto(Id)
);
Create table Reserva(
    Id INTEGER IDENTITY,
    CheckIn DATETIME NOT NULL,
    CheckOut DATETIME NOT NULL,
    StatusRes VARCHAR(10) NOT NULL DEFAULT 'RESERVADA',

```

```

NumAdultos INTEGER NOT NULL,
NumCriancas INTEGER NOT NULL DEFAULT 0,
Despesas DECIMAL(8,2) NOT NULL,
IdHospede INTEGER NOT NULL,
IdQuarto INTEGER NOT NULL,
CONSTRAINT PK_Reserva PRIMARY KEY (Id),
CONSTRAINT FK_ResIdHospede FOREIGN KEY (IdHospede) REFERENCES Hospede (Id),
CONSTRAINT FK_ResIdQuarto FOREIGN KEY (IdQuarto) REFERENCES Quarto (Id)
);
Create table Pagamento(
Id INTEGER IDENTITY,
Valor DECIMAL(8,2) NOT NULL,
Forma VARCHAR(15) NOT NULL,
Momento DATETIME NOT NULL,
IdReserva INTEGER NOT NULL,
CONSTRAINT PK_Pagamento PRIMARY KEY (Id),
CONSTRAINT FK_PagIdReserva FOREIGN KEY (IdReserva) REFERENCES Reserva (Id)
);
Create table Produto(
Id INTEGER IDENTITY,
Nome VARCHAR(30) NOT NULL,
EstoqueQtde INTEGER NOT NULL,
Valor DECIMAL(5,2) NOT NULL,
Validade DATE NULL,
StatusProd VARCHAR(7) NOT NULL,
CONSTRAINT PK_Produto PRIMARY KEY (Id)
);
Create table Consumo(
Id INTEGER IDENTITY,
Quantidade INTEGER NOT NULL,
Momento DATETIME NOT NULL,
Total DECIMAL(5,2) NOT NULL,
IdProduto INTEGER NOT NULL,
IdReserva INTEGER NOT NULL,
CONSTRAINT PK_Consumo PRIMARY KEY (Id),
CONSTRAINT FK_ConIdProduto FOREIGN KEY (IdProduto) REFERENCES PRODUTO
(Id),
CONSTRAINT FK_ConIdReserva FOREIGN KEY (IdReserva) REFERENCES Reserva (Id)
);
INSERT INTO TipoCusto VALUES
('1 solteiro',20.00, GETDATE()),
('2 solteiro',25.00, GETDATE()),
('1 casal',30.00, GETDATE()),
('1 casal 1 solteiro',35.00, GETDATE()),
('1 casal 2 solteiro',40.00, GETDATE()),
('Diaria Adulto',20.00, GETDATE()),
('Diaria Crianca',10.00, GETDATE());
GO
INSERT INTO QUARTO VALUES
(1,'DISPONIVEL',1),
(2,'DISPONIVEL',1),
(3,'DISPONIVEL',1),
(4,'DISPONIVEL',1),
(5,'DISPONIVEL',1),
(6,'DISPONIVEL',1),
(7,'DISPONIVEL',2),
(8,'DISPONIVEL',2),
(9,'DISPONIVEL',2),
(10,'DISPONIVEL',2),
(11,'DISPONIVEL',2),

```

```

(12,'DISPONIVEL',2),
(13,'DISPONIVEL',3),
(14,'DISPONIVEL',3),
(15,'DISPONIVEL',3),
(16,'DISPONIVEL',3),
(17,'DISPONIVEL',3),
(18,'DISPONIVEL',3),
(19,'DISPONIVEL',4),
(20,'DISPONIVEL',4),
(21,'DISPONIVEL',4),
(22,'DISPONIVEL',4),
(23,'DISPONIVEL',4),
(24,'DISPONIVEL',4),
(25,'DISPONIVEL',5),
(26,'DISPONIVEL',5),
(27,'DISPONIVEL',5),
(28,'DISPONIVEL',5),
(29,'DISPONIVEL',5),
(30,'DISPONIVEL',5);
GO
INSERT INTO HOSPEDE VALUES
('Bruno Tavante','14-09-1995','M','gerente@email.nenhum','33-
01020304','33253453544',null,'ATIVO',GETDATE()),
('Fabiana Souza','12-02-1995','F','governante@email.nenhum','22-
01020304','98798798755',null,'ATIVO',GETDATE()),
('Diego Rocha','18-05-1995','M','receptionista@email.nenhum','77-
01020304','54656477644',null,'ATIVO',GETDATE()),
('Matheus Pires','11-07-1995','F','hospede@email.nenhum','55-
01020304','34534534522',null,'ATIVO',GETDATE()),
('Mayara Oliveira','19-03-1995','I','novoemail@email.com','88-
01020304','12312312345',null,'ATIVO',GETDATE());
GO
INSERT INTO FUNCIONARIO VALUES
('Gerente',4000.00,'ATIVO',GETDATE(),(SELECT ID FROM HOSPEDE WHERE Email LIKE
'%gerente%')),
('Governanca',3000.00,'ATIVO',GETDATE(),(SELECT ID FROM HOSPEDE WHERE Email LIKE
'%govern%')),
('Receptionista',2500.00,'ATIVO',GETDATE(),(SELECT ID FROM HOSPEDE WHERE Email LIKE
'%recep%'));
GO
INSERT INTO CONTALOGIN VALUES
('Gerente','ADMIN',GETDATE(),NULL,(SELECT ID FROM FUNCIONARIO
WHERE IdHospede = (SELECT ID FROM HOSPEDE WHERE Email LIKE '%gerente%'))),
('Governante','ADMIN',GETDATE(),NULL,(SELECT ID FROM FUNCIONARIO
WHERE IdHospede = (SELECT ID FROM HOSPEDE WHERE Email LIKE '%govern%'))),
('Receptionista','ADMIN',GETDATE(),NULL,(SELECT ID FROM FUNCIONARIO
WHERE IdHospede = (SELECT ID FROM HOSPEDE WHERE Email LIKE '%recep%'))),
('Matheus','ADMIN',GETDATE(),(SELECT ID FROM HOSPEDE WHERE Email LIKE
'%hospede%'),NULL),
('Mayara','ADMIN',GETDATE(),(SELECT ID FROM HOSPEDE WHERE Email LIKE
'%novo%'),NULL);
GO
Create procedure CONS_Inserir
    @Id INT,
    @Quantidade INT,
    @Produto INT,
    @Reserva INT,
    @Valor DECIMAL(5,2),
    @Retorno VARCHAR(100) OUTPUT
AS

```

```

BEGIN
    If (@Id = 0)
        BEGIN
            INSERT INTO Consumo VALUES(
                @Quantidade, GETDATE(), @Valor, @Produto, @Reserva);
            SET @Retorno = 'Consumo adicionado!';
            UPDATE Produto SET EstoqueQtde =
                (EstoqueQtde - @Quantidade)
            where Id = @Produto;
            UPDATE RESERVA SET Despesas = Despesas + @Valor
            where Id = @Reserva;
        END
    else
        BEGIN
            UPDATE Consumo SET Quantitade = @Quantidade,
                Momento = GETDATE(), IdProduto = @Produto,
                Total = @Valor, IdReserva = @Reserva
            where Id = @Id;
            SET @Retorno = 'Consumo atualizado!';
        END
    END
GO
Create procedure CONS_Listar
as
BEGIN
    Select c.Id, p.Nome as Produto, c.Quantitade, r.Id as Reserva, q.Numero as Quarto, c.Total
    as Total
    from Consumo c
    inner join Reserva r on r.Id = c.IdReserva
    inner join Produto p on p.Id = c.IdProduto
    inner join Quarto q on q.Id = r.IdQuarto
    order by c.Id desc
END
GO
Create procedure CONS_ListarRes
    @Id INT
AS
BEGIN
    SELECT P.NOME,C.QUANTITADE,C.TOTAL
    FROM CONSUMO C
    INNER JOIN PRODUTO P ON P.ID = C.IDPRODUTO
    INNER JOIN RESERVA R ON R.ID = C.IDRESERVA
    WHERE R.ID = @Id
END
GO
Create procedure CONT_Inserir
    @NomeUsuario VARCHAR(20),
    @Senha VARCHAR(20),
    @IdHospede INT,
    @IdFunc INT,
    @Retorno VARCHAR(100) OUTPUT
AS
BEGIN
    IF (@IdHospede = 0)
        BEGIN
            if not exists (Select * from ContaLogin
                where IdFuncionario = @IdFunc)
                BEGIN
                    Insert into ContaLogin values(
                        @NomeUsuario,@Senha,GETDATE(),null,@IdFunc);
                END
        END

```

```

        SET @Retorno = 'Conta de funcionário criada!';
    END
else
BEGIN
    UPDATE ContaLogin SET
        NomeUsuario = @NomeUsuario, Senha = @Senha,
        DataAtualizacao = GETDATE(),
        IdFuncionario = @IdFunc
    where IdFuncionario = @IdFunc;
    Set @Retorno = 'Conta de funcionário atualizada!';
END
END
ELSE
BEGIN
    if not exists (Select * from ContaLogin
    where IdHospede = @IdHospede)
    BEGIN
        Insert into ContaLogin values(
            @NomeUsuario, @Senha, GETDATE(), @IdHospede, null);
        SET @Retorno = 'Conta de hóspede criada!';
    END
else
BEGIN
    UPDATE ContaLogin SET
        NomeUsuario = @NomeUsuario, Senha = @Senha,
        DataAtualizacao = GETDATE(),
        IdHospede = @IdHospede
    where IdHospede = @IdHospede;
    Set @Retorno = 'Conta de hóspede atualizada!';
END
END
END
GO
Create procedure CONT_ValidarFunc
    @NomeUsuario VARCHAR(20),
    @Senha VARCHAR(20)
AS
BEGIN
    Select c.NomeUsuario, c.Senha, p.Nome, f.Cargo
    from ContaLogin c
    inner join Funcionario f ON f.Id = c.IdFuncionario
    inner join Hospede p ON p.Id = f.IdHospede
    where NomeUsuario = @NomeUsuario
    and Senha = @Senha and f.StatusFunc = 'ATIVO';
END
GO
Create procedure CONT_ValidarHosp
    @Email VARCHAR(50),
    @Senha VARCHAR(20)
AS
BEGIN
    Select c.Senha,c.id as conid, h.CPF as cpf, h.Passaporte as passaporte
    from ContaLogin c
    inner join Hospede h ON h.Id = c.IdHospede
    where h.Email = @Email
    and Senha = @Senha and h.StatusHosp = 'ATIVO';
END
GO
Create procedure FUNC_Inserir
    @Cargo VARCHAR(20),

```

```

@Salario DECIMAL(8,2),
@StatusFunc VARCHAR(7),
@IdHospede INT,
@Retorno VARCHAR(100) OUTPUT
AS
BEGIN
    IF not exists (Select * from Funcionario
    where IdHospede = @IdHospede)
    BEGIN
        INSERT into Funcionario VALUES(
            @Cargo, @Salario, @StatusFunc, GETDATE(), @IdHospede);
        SET @Retorno = 'Funcionário registrado!';
    END
    ELSE
    BEGIN
        UPDATE Funcionario SET Cargo = @Cargo,
        Salario = @Salario, StatusFunc = @StatusFunc,
        DataAtualizacao = GETDATE()
        where IdHospede = @IdHospede;
        SET @Retorno = 'Funcionário atualizado!';
    END
END
GO
Create procedure FUNC_Carregar
    @cpfPass VARCHAR(12)
AS
BEGIN
    if (@CpfPass is not null) and (LEN(@CpfPass) = 11)
    BEGIN
        SELECT * FROM Hospede h
        inner join Funcionario f on f.IdHospede = h.Id
        where h.CPF = @CpfPass;
    END
    else
    BEGIN
        SELECT * FROM Hospede h
        inner join Funcionario f on f.IdHospede = h.Id
        where h.Passaporte = @CpfPass;
    END
END
GO
Create procedure FUNC_ListarStatus
    @Status VARCHAR(7)
AS
BEGIN
    if(@Status = "")
    SELECT H.NOME,F.CARGO,F.SALARIO, H.CPF
    FROM FUNCIONARIO F
    INNER JOIN HOSPEDE H ON H.Id = F.IdHospede
    ELSE
    SELECT H.NOME,F.CARGO,F.SALARIO, H.CPF
    FROM FUNCIONARIO F
    INNER JOIN HOSPEDE H ON H.Id = F.IdHospede
    WHERE F.StatusFunc = @STATUS
END
GO
Create procedure HOSP_Inserir
    @Nome VARCHAR(50),
    @DataNasc DATE,
    @Sexo CHAR(1),

```

```

@Email VARCHAR(50),
@Telefone VARCHAR(12),
@CPF VARCHAR(11),
@Passaporte VARCHAR(12),
@StatusHosp VARCHAR(7),
@Retorno VARCHAR(100) OUTPUT
AS
BEGIN
    DECLARE @CPFchar CHAR(11), @identificador VARCHAR(12);
    SET @CPFchar = @CPF;
    if(LEN(@CPF) = 11)
        SET @identificador = @CPF;
    else
        SET @identificador = @Passaporte;
    if not exists (Select * from Hospede where CPF = @identificador)
        and not exists (Select * from Hospede where Passaporte = @identificador)
    BEGIN
        if(@CPFchar = @identificador)
            SET @Passaporte = null;
        else
            SET @CPFchar = null;
        INSERT INTO Hospede VALUES (
            @Nome, @DataNasc, @Sexo, @Email, @Telefone,
            @CPFchar, @Passaporte, @StatusHosp, GETDATE());
        SET @Retorno = 'Hóspede registrado!';
    END
    else
    BEGIN
        DECLARE @parametro VARCHAR(12);
        if exists (Select * from hospede where CPF = @CPFchar)
            SET @parametro = @CPFchar;
        else
            SET @parametro = @Passaporte;
        UPDATE HOSPEDE SET
            Nome = @Nome, DataNasc = @DataNasc, Sexo = @Sexo,
            Email = @Email, Telefone = @Telefone, CPF = @CPFchar,
            Passaporte = @Passaporte, StatusHosp = @StatusHosp,
            DataAtualizacao = GETDATE()
            where CPF = @parametro or Passaporte = @parametro;
        SET @Retorno = 'Hóspede atualizado';
    END
END
GO
Create procedure HOSP_Carregar
    @CpfPass CHAR(11)
AS
BEGIN
    if (@CpfPass is not null) and (LEN(@CpfPass) = 11)
    BEGIN
        SELECT * FROM Hospede where CPF = @CpfPass;
    END
    else
    BEGIN
        SELECT * FROM Hospede where Passaporte = @CpfPass;
    END
END
GO
Create procedure HOSP_ListarStatus
    @Status VARCHAR(7)
AS

```

```

BEGIN
    if(@Status = "")
        SELECT NOME, SEXO, DATANASC, CPF
        FROM HOSPEDE
    else
        SELECT NOME, SEXO, DATANASC, CPF
        FROM HOSPEDE
        WHERE StatusHosp = @Status
END
GO
Create procedure PAGA_Inserir
    @Valor Decimal(8,2),
    @Forma VARCHAR(15),
    @IdReserva INT,
    @Retorno VARCHAR(100) OUTPUT
AS
BEGIN
    INSERT into Pagamento values(
        @Valor, @Forma, GETDATE(), @IdReserva);
    SET @Retorno = 'Pagamento registrado';
END
GO
Create procedure PAGA_TotalRes
    @Id INT
AS
BEGIN
    IF not exists (Select Valor from Pagamento where idreserva = @id)
        SELECT 0 AS TOTAL;
    ELSE
        SELECT SUM(VALOR) AS TOTAL FROM PAGAMENTO
        WHERE IDRESERVA = @Id
END
GO
Create procedure PAGA_ListarRes
    @Id INT
AS
BEGIN
    SELECT P.Valor, P.Momento, P.Forma
    from pagamento P
    Inner join Reserva R on R.id = p.idreserva
    where r.id = @Id
END
GO
Create procedure PROD_Inserir
    @Id INT,
    @Nome VARCHAR(30),
    @Qtde INT,
    @Valor DECIMAL(5,2),
    @Validade DATE,
    @Retorno VARCHAR(100) OUTPUT
AS
BEGIN
    if (@Id = 0)
    BEGIN
        Insert into Produto VALUES(
            @Nome, @Qtde, @Valor, @Validade, 'ATIVO');
        SET @Retorno = 'Produto registrado!';
    END
    else
    BEGIN

```

```

        UPDATE PRODUTO SET Nome = @Nome,
        EstoqueQtde = @Qtde, Valor = @Valor,
        Validade = @Validade
        where (Nome = @Nome) or (Id = @Id);
        SET @Retorno = 'Produto atualizado!';
    END
END
GO
Create procedure PROD_Deletar
    @Id INT,
    @Retorno VARCHAR(100) OUTPUT
AS
BEGIN
    Update PRODUTO SET
    EstoqueQtde = 0,
    Valor = 00.0,
    Validade = null,
    StatusProd = 'Inativo'
    where Id = @Id;
    SET @Retorno = 'Produto removido do estoque';
END
GO
Create procedure QUAR_Alterar
    @Numero INT,
    @Status VARCHAR(10),
    @Retorno VARCHAR(100) OUTPUT
AS
BEGIN
    UPDATE Quarto SET
    StatusQuarto = @Status
    where Numero = @Numero
    SET @Retorno = 'Quarto atualizado.';

END
GO
Create procedure QUAR_Listar
AS
BEGIN
    select q.numero as Quarto, r.id as Reserva, q.StatusQuarto as [Status],
    t.nometipo as Tipo, h.nome as Hospede from Quarto q
    left join reserva r on r.IdQuarto = q.Id and r.statusres = 'INICIADA'
    left join Tipocusto t on t.id = q.IdTipo
    left join hospede h on h.id = r.idhospede
    order by q.numero
END
GO
Create procedure RESE_INSERIR
    @Id INT,
    @Chin DATETIME,
    @Chout DATETIME,
    @StatusRes VARCHAR(10),
    @NumAdultos INT,
    @NumCrianças INT,
    @Despesas DECIMAL (8,2),
    @IdHospede INT,
    @NumQuarto INT,
    @Retorno VARCHAR(100) OUTPUT
AS
BEGIN TRY
    if not exists(Select Id from Reserva Where Id = @Id)
    BEGIN

```

```

Insert into Reserva Values(
    @Chin, @Chout, @StatusRes, @NumAdultos, @NumCriancas,
    @Despesas, @IdHospede,
    (SELECT Id From Quarto Where Numero = @NumQuarto))
    SET @Retorno = 'Reserva efetuada!';

END
else
BEGIN
    Update RESERVA SET
        StatusRes = @StatusRes,
        CheckIn = @Chin,
        CheckOut = @Chout
        WHERE Id = @Id;
        if(@StatusRes = 'INICIADA')
            UPDATE Quarto set StatusQuarto = 'OCUPADO'
            where Numero = @NumQuarto;
        if(@StatusRes = 'FINALIZADA')
            UPDATE Quarto set StatusQuarto = 'LIMPEZA'
            where Numero = @NumQuarto;
            SET @Retorno = 'Reserva atualizada!';

    END
END TRY
BEGIN CATCH
    SET @Retorno = 'Erro no procedimento: NovaReserva';
END CATCH
GO
Create procedure RESE_Carregar
    @Id INT
AS
BEGIN
    SELECT R.ID AS RESERVA, R.CHECKIN, R.CHECKOUT, R.StatusRes AS [STATUS],
    R.NUMADULTOS, R.NUMCRIANCAS, R.DESPESAS, Q.NUMERO AS QUARTO,
    T.NOMETIPO AS TIPO,
    H.NOME, H.DATANASC, H.SEXO, H.EMAIL, H.TELEFONE, H.CPF, H.PASSAPORTE
    FROM RESERVA R
    INNER JOIN QUARTO Q ON Q.ID = R.IDQUARTO
    INNER JOIN TIPOCUSTO T ON Q.IDTIPO = T.ID
    INNER JOIN HOSPEDA H ON H.ID = R.IDHOSPEDA
    WHERE R.ID = @Id
END
GO
Create procedure RESE_ListarStatus
    @Status VARCHAR(10)
AS
BEGIN
    Select r.id as [RESERVA], r.checkin as [CHECK-IN],
    r.checkout as [CHECK-OUT], h.nome as [HÓSPEDE],
    q.numero as [QUARTO] from Reserva r
    inner join Hospede h ON r.IdHospede = h.id
    inner join Quarto q ON r.IdQuarto = q.id
    where r.StatusRes = @Status
    order by r.id desc
END
GO
Create procedure RESE_ListarData
    @Periodo DATE
AS
BEGIN
    Select r.id as [RESERVA], r.checkin as [CHECK-IN],
    r.checkout as [CHECK-OUT], h.nome as [HÓSPEDE],

```

```

q.numero as [QUARTO] from Reserva r
inner join Hospede h ON r.IdHospede = h.id
inner join Quarto q ON r.IdQuarto = q.id
where (r.StatusRes <> 'CANCELADA')
and (MONTH(r.CheckIn) = MONTH(@periodo)
and YEAR(r.CheckIn) = YEAR(@periodo))
and (MONTH(r.CheckOut) = MONTH(@periodo)
and YEAR(r.CheckOut) = YEAR(@periodo))
order by r.Id desc
END
GO
Create procedure RESE_TipoDisponivel
    @Entrada DATE,
    @TipoDeQuarto VARCHAR(30)
AS
BEGIN
    DECLARE @Ocupados INT;
    SET @Ocupados = (Select COUNT(r.ID) FROM RESERVA r
    inner join Quarto q ON q.Id = r.IdQuarto
    inner join TipoCusto t ON t.id = q.IdTipo
    where r.CheckOut > @Entrada and r.StatusRes = 'INICIADA'
    and t.NomeTipo = @TipoDeQuarto);
    if (@Ocupados < 6)
        SELECT
            top 1 q.numero as numero, 1 AS RETORNO from Quarto q
            inner join TipoCusto t ON q.IdTipo = t.Id
            where q.id not in (Select qa.id from reserva r
            inner join quarto qa ON qa.id = r.idquarto
            where r.checkout > @Entrada)
            and t.id = (Select Id From TipoCusto where NomeTipo = @TipoDeQuarto);
    else
        SELECT 0 AS RETORNO;
    END
    GO
Create procedure TIPO_Carregar
    @Id INT
AS
BEGIN
    SELECT * from TipoCusto WHERE ID = @Id;
END
GO
Create procedure TIPO_Alterar
    @Id INT,
    @Preco DECIMAL(8,2)
AS
BEGIN
    UPDATE TipoCusto SET
    Preco = @Preco,
    DataAtualizacao = GETDATE()
    where Id = @Id;
END
GO
Create procedure RELAT_Ocupacao
AS
BEGIN
    DECLARE @Porcent INT = 0, @ChinFeito INT = 0, @ChinEsp INT = 0,
    @ChoutFeito INT = 0, @ChoutEsp INT = 0;
    SET @Porcent = (SELECT COUNT(NUMERO) FROM QUARTO
                    WHERE STATUSQUARTO = 'OCUPADO');
    SET @Porcent = (@Porcent * 100) / 30;

```

```

SET @ChinFeito = (SELECT COUNT(ID) FROM RESERVA
                  WHERE STATUSRES = 'INICIADA' AND
                  CONVERT(DATE,CheckIN) = CONVERT(DATE,GETDATE()));
SET @ChinEsp = (SELECT COUNT(ID) FROM RESERVA
                  WHERE CONVERT(DATE,CheckIN) = CONVERT(DATE,GETDATE()));
SET @ChoutFeito = (SELECT COUNT(ID) FROM RESERVA
                  WHERE STATUSRES = 'FINALIZADA' AND
                  CONVERT(DATE,CheckIN) = CONVERT(DATE,GETDATE()));
SET @ChoutEsp = (SELECT COUNT(ID) FROM RESERVA
                  WHERE CONVERT(DATE,CheckIN) = CONVERT(DATE,GETDATE()));
SELECT @Porcent, @ChinFeito, @ChinEsp, @ChoutFeito, @ChoutEsp
END
GO
Create procedure RELAT_Receita
AS
BEGIN
    DECLARE @Hoje FLOAT = 0.0, @EsseMes FLOAT = 0.0, @MesPassado FLOAT = 0.0;
    SET @Hoje = (SELECT SUM(p.Valor) from Pagamento p
                 inner join Reserva R ON R.Id = p.IdReserva
                 where (CONVERT(DATE,p.Momento)) = (CONVERT(DATE,GETDATE())));
    SET @EsseMes = (SELECT SUM(p.Valor) from Pagamento p
                 inner join Reserva R ON R.Id = p.IdReserva
                 where (MONTH(R.CheckOut)) = (MONTH(GETDATE())));
    SET @MesPassado = (SELECT SUM(p.Valor) from Pagamento p
                 inner join Reserva R ON R.Id = p.IdReserva
                 where (MONTH(R.CheckOut)) = (MONTH(GETDATE()) - 1));
    SELECT @HOJE,@EsseMes,@MesPassado;
END
GO
Create procedure RELAT_Financas
AS
BEGIN
    DECLARE @GERENTE FLOAT = 0.0, @GOVERNANCA FLOAT = 0.0,
            @RECEP FLOAT = 0.0, @TOTAL FLOAT = 0.0, @GANHOS FLOAT = 0.0;
    SET @GERENTE = (SELECT SUM(Salario) FROM Funcionario F where
                    StatusFunc = 'ATIVO' and Cargo = 'GERENTE');
    SET @GOVERNANCA = (SELECT SUM(Salario) FROM Funcionario F where
                    StatusFunc = 'ATIVO' and Cargo = 'GOVERNANCA');
    SET @RECEP = (SELECT SUM(Salario) FROM Funcionario F where
                    StatusFunc = 'ATIVO' and Cargo = 'RECEPCIONISTA');
    SET @TOTAL = @GERENTE + @GOVERNANCA + @RECEP;
    SET @GANHOS = (SELECT SUM(p.Valor) from Pagamento p
                   inner join Reserva R ON R.Id = p.IdReserva
                   where (MONTH(R.CheckOut)) = (MONTH(GETDATE())));
    SELECT @GERENTE, @GOVERNANCA, @RECEP, @TOTAL, @GANHOS;
END
GO

```

APÊNDICE E – CÓDIGOS DE PRINCIPAIS ARTEFATOS

Amostras de códigos desenvolvidos em cada interface.

Classes C# para camadas Model e DAL respectivamente:

```

namespace Model
{
    public class Reserva
    {
        private int _Id;
        private DateTime _CheckIn;
        private DateTime _CheckOut;
        private int _Adultos;
        private int _Criancas;
        private double _Despesas;
        private string _Status;
        private Hospede _Hospede;
        private Quarto _Quarto;
        public int Id { get => _Id; set => _Id = value; }
        public DateTime CheckIn { get => _CheckIn; set => _CheckIn = value; }
        public DateTime CheckOut { get => _CheckOut; set => _CheckOut = value; }
        public int Adultos { get => _Adultos; set => _Adultos = value; }
        public int Criancas { get => _Criancas; set => _Criancas = value; }
        public double Despesas { get => _Despesas; set => _Despesas = value; }
        public string Status { get => _Status; set => _Status = value; }
        public Hospede Hospede { get => _Hospede; set => _Hospede = value; }
        public Quarto Quarto { get => _Quarto; set => _Quarto = value; }
        public Reserva()
        {
            Status = "RESERVADA";
            _Hospede = new Hospede();
            _Quarto = new Quarto();
        }
        public void Finalizar()
        {
            CheckOut = DateTime.Now;
            Status = "FINALIZADA";
        }
    }
}

namespace DAL
{
    public class FuncionarioDAO : IListaPorStatus
    {
        ConexaoSQLServer conn = new ConexaoSQLServer();
        HospedeDAO hDAO = new HospedeDAO();
        ContaDAO cDAO = new ContaDAO();
        public DataTable ListarStatus(string status)
        {
            try
            {
                DataTable tabela = new DataTable();
                using (SqlConnection conex = conn.AbrirConexao())
                {
                    SqlCommand procedure = new SqlCommand("FUNC_ListarStatus", conex);
                    procedure.CommandType = CommandType.StoredProcedure;
                    procedure.Parameters.Add("@Status", SqlDbType.VarChar).Value = status;
                    SqlDataAdapter da = new SqlDataAdapter(procedure);
                    da.Fill(tabela);
                }
            }
        }
    }
}

```

```
        }
        return tabela;
    }
    catch (Exception err)
    {
        throw new Exception(err.Message);
    }
}
```

Classe de “controller” para interface web:

```
using System;
using System.Collections.Generic;
using System.Web.Mvc;
using Model;
using DAL;
using PimIV_Web.Models;
namespace PimIV_Web.Controllers
{
    public class ReservasController : Controller
    {
        private ReservaDAO _resDAO = new ReservaDAO();
        static List<Reserva> disponiveis = new List<Reserva>();
        TipoCustoDAO tDAO = new TipoCustoDAO();
        public ActionResult Index()
        {
            return View();
        }
        public ActionResult Confirmar(string tipo)
        {
            Reserva reserva = disponiveis
                .Find(item => item.Quarto.Tipo.NomeTipo == tipo);
            if(SessaoSite.Conta == null)
            {
                SessaoSite.Mensagem = 4;
                SessaoSite.Reserva = reserva;
                return RedirectToAction("FazerLogin", "Contas");
            }
            else
            {
                if(SessaoSite.Reserva != null && SessaoSite.Reserva.CheckIn != null)
                {
                    reserva = SessaoSite.Reserva;
                }
                else
                {
                    SessaoSite.Reserva = reserva;
                }
                reserva.Hospede = SessaoSite.Conta.HospAssociado;
            }
            return View(reserva);
        }
    }
}
```

Classe Java e arquivo XML para a MainActivity:

```
package br.unip.pimivandroid;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
```

```

import android.os.Bundle;
import android.view.View;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    public void buscarReservas(View view) {
        Intent intent = new Intent(this, ReservasActivity.class);
        startActivity(intent);
    }
    public void fazerLogin(View view) {
        Intent intent = new Intent(this, LoginActivity.class);
        startActivity(intent);
    }
    public void acessarConta(View view) {
        Intent intent = new Intent(this, ContaActivity.class);
        startActivity(intent);
    }
}

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="24dp"
        android:text="@string/NomeHotel"
        android:textSize="34sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <TextView
        android:id="@+id/textView2"
        android:layout_width="345dp"
        android:layout_height="152dp"
        android:layout_marginTop="60dp"
        android:text="@string/InfoContato"
        android:textSize="12sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView" />
    <TextView
        android:id="@+id/textView3"
        android:layout_width="359dp"
        android:layout_height="49dp"
        android:layout_marginTop="56dp"
        android:text="@string/InfoBotoes"
        android:textSize="16sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView2" />

```

```
<Button
    android:id="@+id/botaoLogin"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="24dp"
    android:layout_marginLeft="24dp"
    android:layout_marginBottom="24dp"
    android:onClick="fazerLogin"
    android:text="@string/botaoLogin"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent" />
<Button
    android:id="@+id/botaoReservas"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="24dp"
    android:layout_marginLeft="24dp"
    android:layout_marginEnd="24dp"
    android:layout_marginRight="24dp"
    android:layout_marginBottom="24dp"
    android:onClick="buscarReservas"
    android:text="@string/BotaoReservas"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/botaoConta"
    app:layout_constraintStart_toEndOf="@+id/botaoLogin" />
<Button
    android:id="@+id/botaoConta"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginEnd="24dp"
    android:layout_marginRight="24dp"
    android:layout_marginBottom="24dp"
    android:onClick="acessarConta"
    android:text="@string/botaoConta"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```