Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

TIAGO DORINI DE OLIVEIRA CARVALHO ROSSI

HOMEBASE: DESENVOLVENDO UMA APLICAÇÃO DE GERENCIAMENTO DE CONDOMÍNIOS

Projeto de pesquisa apresentado ao Curso de Análise e Desenvolvimento de Sistemas do Instituto Municipal de Ensino Superior de Assis – IMESA e à Fundação Educacional do Município de Assis – FEMA, como requisito parcial à obtenção do Certificado de Conclusão.

Orientando(a): Tiago Dorini de Oliveira Carvalho Rossi
Orientador(a): Me. Guilherme de Cleva Farto

Assis/SP
2024

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

TIAGO DORINI DE OLIVEIRA CARVALHO ROSSI

HOMEBASE: DESENVOLVENDO UMA APLICAÇÃO DE GERENCIAMENTO DE CONDOMÍNIOS

Projeto de pesquisa apresentado ao Curso de Análise e Desenvolvimento de Sistemas do Instituto Municipal de Ensino Superior de Assis – IMESA e à Fundação Educacional do Município de Assis – FEMA, como requisito parcial à obtenção do Certificado de Conclusão.

Orientando(a): Tiago Dorini de Oliveira Carvalho Rossi
Orientador(a): Me. Guilherme de Cleva Farto

Assis/SP
2024

SUMÁRIO

1. INTRODUÇÃO.....9
1.1. OBJETIVO GERAL.....9
1.2. PÚBLICO ALVO.....10
1.3. JUSTIFICATIVAS.....10
2. MÉTODO DE DESENVOLVIMENTO.....10
3. ANÁLISE E ESPECIFICAÇÃO DO SISTEMA.....12
3.1. LEVANTAMENTO DE REQUISITOS.....12
3.2. MAPA MENTAL.....13
3.3. DIAGRAMA DE CLASSE.....13
3.4. DIAGRAMAS FLUXO.....16
4. DESENVOLVIMENTO DO PROJETO.....20
4.1. FERRAMENTAS UTILIZADAS.....20
4.1.1. VISUAL STUDIO CODE.....20
4.1.2. REACT.JS E VITE.....20
4.1.3. NEST.JS e TYPEORM.....21
4.1.4 POSTGRESQL.....21
5. IMPLEMENTAÇÕES E RESULTADO FINAL.....21
6. CONCLUSÕES.....27
6.1. TRABALHOS FUTUROS.....28
REFERÊNCIAS.....29

LISTA DE ILUSTRAÇÕES

Figura 1: Mapa Mental.....13
Figura 2: Diagrama de classe de Usuário, Morador e Apartamento.....14
Figura 3: Diagrama de Classe Usuário e Tipo de Usuário.....14
Figura 4: Diagrama de Classe de Apartamento com possíveis eventos do sistema. 15
Figura 5: Diagrama de Classe Entre Reserva de Área Comum e Área Comum.....16
Figura 6: Fluxo de Login.....16
Figura 7: Registrar Reclamação.....17
Figura 8: Reservar Área Comum.....18
Figura 9: Registrar Entrega.....19
Figura 10: Tela de Login.....22
Figura 11: Morador - Tela Home.....22
Figura 12: Portaria - Tela Home.....23
Figura 13: Tela Registrar Reclamação.....23
Figura 14: Tela Registrar Visitantes.....24
Figura 15: Tela Registrar Entrega.....24
Figura 16: Tela Reserva Espaço Comum.....25
Figura 17: Tela Registrar Aviso.....25
Figura 18: Tela Visualizar Apartamentos.....26
Figura 19: Tela Visualizar Moradores.....27

LISTA DE TABELAS

Tabela 1: Efetuar Login.....16
Tabela 2: Registrar Reclamações.....17
Tabela 3: Fluxo da Reserva de Área Comum.....18
Tabela 4: Registrar Entrega.....18

RESUMO

O presente trabalho de conclusão de curso tem como objetivo desenvolver uma aplicação web para o gerenciamento de condomínios, visando otimizar a convivência entre moradores e a administração do espaço. A solução propõe utilizar as tecnologias React e Nest.js, permitindo a implementação de funcionalidades como chat integrado, agendamento de áreas comuns e registro de reclamações. A importância da comunicação eficaz no ambiente condominial é ressaltada, destacando como o software pode facilitar a interação entre residentes e síndicos, promovendo um ambiente mais harmonioso. Espera-se que a implementação deste sistema contribua para a melhoria da qualidade de vida dos moradores, proporcionando maior transparência e eficiência nas operações condominiais.

Palavras-chave: Gerenciamento de condomínios; Comunicação; React; Nest.js; Qualidade de vida.

ABSTRACT

This graduation thesis aims to develop a web application for condominium management, intending to optimize the coexistence between residents and the administration of the space. The proposed solution utilizes React and Nest.js technologies, allowing the implementation of functionalities such as integrated chat, scheduling of common areas, and complaint registration. The importance of effective communication in the condominium environment is emphasized, highlighting how the software can facilitate interaction between residents and managers, promoting a more harmonious environment. It is expected that the implementation of this system will contribute to improving the quality of life for residents, providing greater transparency and efficiency in condominium operations.

Keywords: Condominium management; Communication; React; Nest.js; Quality of life.

1. INTRODUÇÃO

Gerenciar um condomínio residencial pode ser desafiador, especialmente em estruturas de maior porte, devido às diversas atividades que precisam ser organizadas e controladas, como agendamento de reuniões, comunicação com moradores, controle de acesso de visitantes e até mesmo reconhecimento facial para liberar acessos. Um software que atenda essas demandas pode ser ajustado conforme os recursos implementados e os objetivos do projeto, otimizando a gestão condominial

Para os condôminos, a segurança, conforto e uma boa convivência dentro do condomínio são de extrema importância. Além disso, esses são fatores que podem influenciar diretamente no valor comercial dos imóveis.

Dito isso, este projeto propõe criar um software que pretende auxiliar, de forma geral, os residentes e funcionários que habitam e frequentam a propriedade. Em um primeiro momento, será desenvolvido um sistema Web composto, principalmente, por React e Nest.js junto de outras ferramentas que irão oferecer auxílio.

O software contará com recursos para facilitar a comunicação entre condôminos, funcionários e síndicos, além da visualização fácil de informações relacionadas ao dia a dia dos moradores. Também estarão presentes recursos como a possibilidade de agendar áreas comuns como quiosques e churrasqueiras, além de um dashboard com uma lista de avisos importantes para os usuários como manutenção de uma área determinada dentro do condomínio.

O projeto tem como intenção realizar o desenvolvimento de um software que possui recursos que podem auxiliar no dia a dia dos condôminos, funcionários, síndicos, visitantes e qualquer indivíduo que frequente um imóvel.

1.1. OBJETIVO GERAL

Este projeto tem como objetivo geral desenvolver um software web que facilite a vivência daqueles que frequentam um condomínio, sendo seu foco inicial condomínios de apartamentos não comerciais:

• Realizar a implementação do sistema usando as bibliotecas React.js e Nest.js, utilizando Typescript em ambas;

• Realizar análise e modelagem aplicativo proposto;

• Desenvolver o software;

1.2. PÚBLICO ALVO

O presente trabalho tem como público alvo moradores e colaboradores de condomínios residenciais. O foco é ajudá-los a ter uma melhor qualidade de vida e facilitar a jornada de trabalho.

1.3. JUSTIFICATIVAS

A convivência dentro de um ambiente de condomínio pode ser difícil e estressante. Por esse motivo, este software visa melhorar a qualidade de vida dos moradores e facilitar o trabalho dos síndicos e possíveis funcionários.

Entender o que está acontecendo dentro do condomínio pode trazer um conforto maior para aqueles que residem, além de evitar e amenizar possíveis transtornos.

Facilitar a comunicação em prol de uma melhor convivência pode melhorar a qualidade do ambiente, assim como obter uma maior transparência com eventos, sejam eles de manutenção ou de confraternização, o que deve ser um impacto positivo na satisfação dos residentes.

2. MÉTODO DE DESENVOLVIMENTO

No intuito de realizar o processo de desenvolvimento deste projeto, foram utilizados vários recursos como documentações, cursos e artigos de sites sobre desenvolvimento de software e metodologias de desenvolvimento.

No processo que antecede a programação, foi essencial identificar os requisitos do sistema e realizar suas respectivas análises. Após esta etapa, foram criados diagramas para visualizar de forma gráfica o fluxo das funcionalidades, juntamente de diagramas de classe para validar seus relacionamentos.

O processo de desenvolvimento teve inspiração nos princípios apontados pelo livro *Clean Code* (Código Limpo), publicado em 2009 e escrito por Robert C. Martin, também conhecido como *Uncle Bob* (Tio Bob). Como principal exemplo de princípios de programação apresentado neste livro está o SOLID, que, traduzido do inglês, pode ser dividido nos seguintes pontos:

- S: Princípio da responsabilidade única;

- O: Princípio Aberto/Fechado;

- L: Princípio da substituição de Liskov;

- I: Princípio da Segregação da Interface;

- D: Princípio da inversão da dependência.

O “Princípio da Responsabilidade Única” define que cada classe deve ter uma responsabilidade única dentro do código, facilitando assim o entendimento do código e sua manutenção. O segundo princípio, sendo esse o “Princípio Aberto/Fechado”, afirma que as entidades do software, como classes e módulos, devem estar abertas para serem estendidas, no entanto, fechadas para serem modificadas, ou seja, as entidades devem permitir a adição de novas funcionalidades, porém, sem que afetem o código existente.

O “Princípio da Substituição” de Liskov, por sua vez, define que os objetos de uma classe derivada de outra devem ser capazes de substituir objetos da classe da qual foram estendidos, sem alterar as propriedades desejadas software. Tal princípio possibilita de forma segura que a herança não comprometa o comportamento da aplicação. Em seguida, o “Princípio da Segregação de Interfaces” sugere que é melhor ter várias interfaces específicas do que uma única interface geral, assim evitando que classes sejam forçadas a ter comportamentos que não utilizam.

Por fim, o ”Princípio da Inversão de Dependência” define que módulos de alto nível não devem depender de módulos de baixo nível; ambos devem depender de abstrações. Além disso, as abstrações não devem depender de detalhes, os detalhes devem depender de abstrações.

3. ANÁLISE E ESPECIFICAÇÃO DO SISTEMA

Para melhor entendimento do sistema foram criados Diagramas de Classes e de Fluxos. Em adição, narrativas descrevendo os recursos mais importantes do sistema estão neste documento juntamente com o mapa mental representando o levantamento de requisitos.

3.1. LEVANTAMENTO DE REQUISITOS

No intuito melhorar o entendimento do projeto, foram modelados Diagramas UML (*Unified Modeling Language*) assim como um mapa mental na plataforma *Lucidchart,* demonstrando as funcionalidades da aplicação.

A aplicação contém as seguintes funcionalidades:

- Registro de Reclamações;

- Registro de Avisos;

- Registro de Eventos;

- Registro de Entregas;

- Registro de Visitante;

- Reserva de Áreas Comuns;

- Visualização de Moradores;

- Visualização de Apartamentos.

3.2. MAPA MENTAL

A Figura 1 demonstra quais funcionalidades foram identificadas. A maioria destas ferramentas são possíveis de serem realizadas tanto pelos moradores quanto pela portaria.

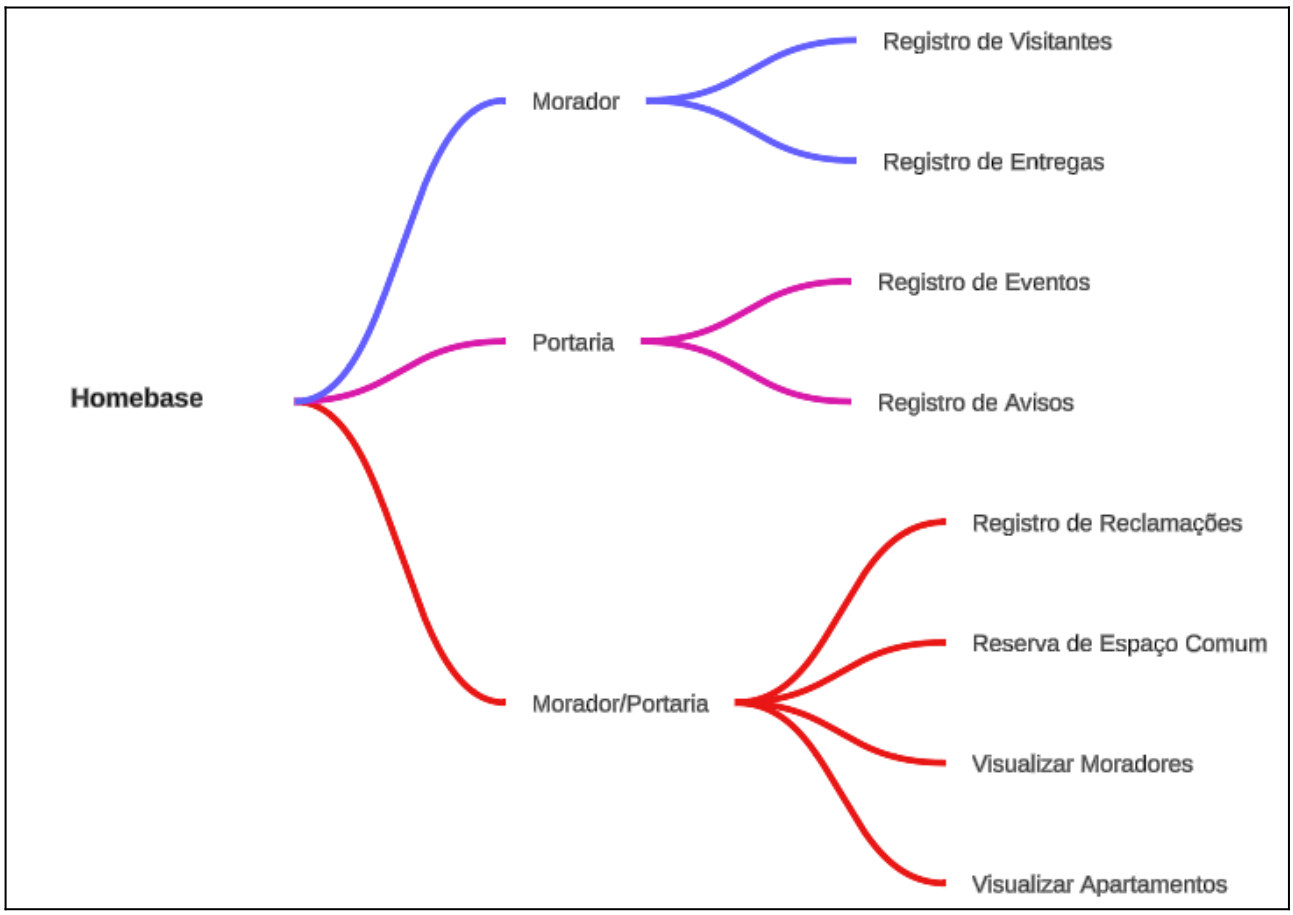

Figura 1: Mapa Mental

3.3. DIAGRAMA DE CLASSE

Os Diagrama de Classe apresentados a seguir contém informações das entidades conjuntamente com seus relacionamentos.

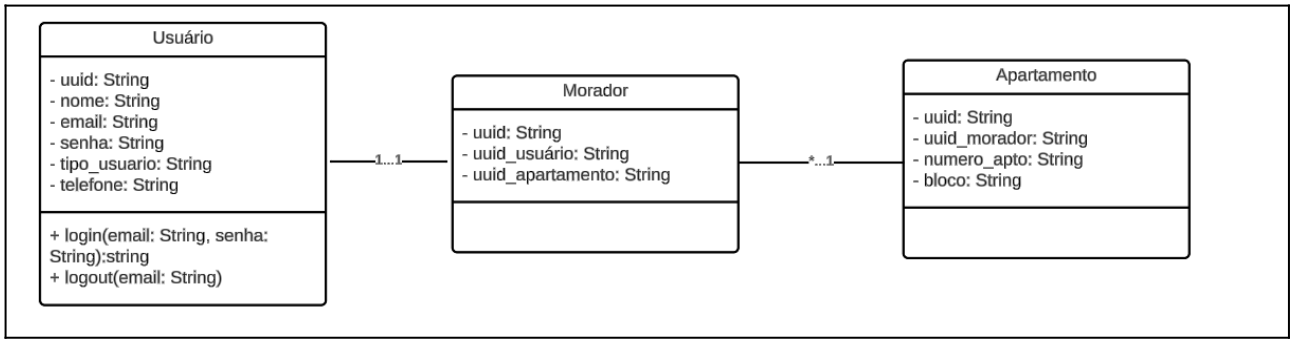

Figura 2: Diagrama de classe de Usuário, Morador e Apartamento

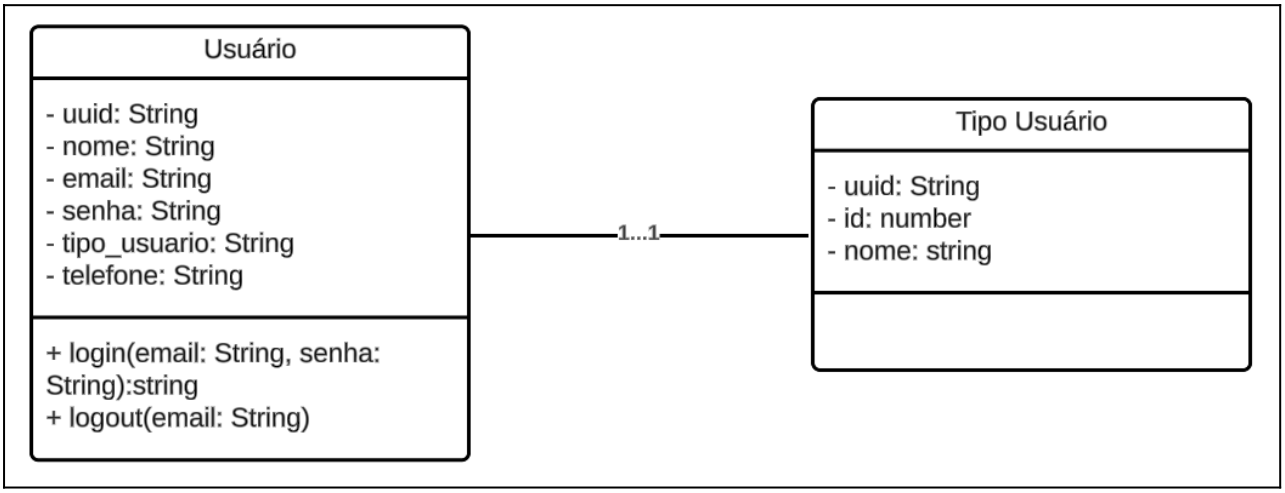

Figura 3: Diagrama de Classe Usuário e Tipo de Usuário

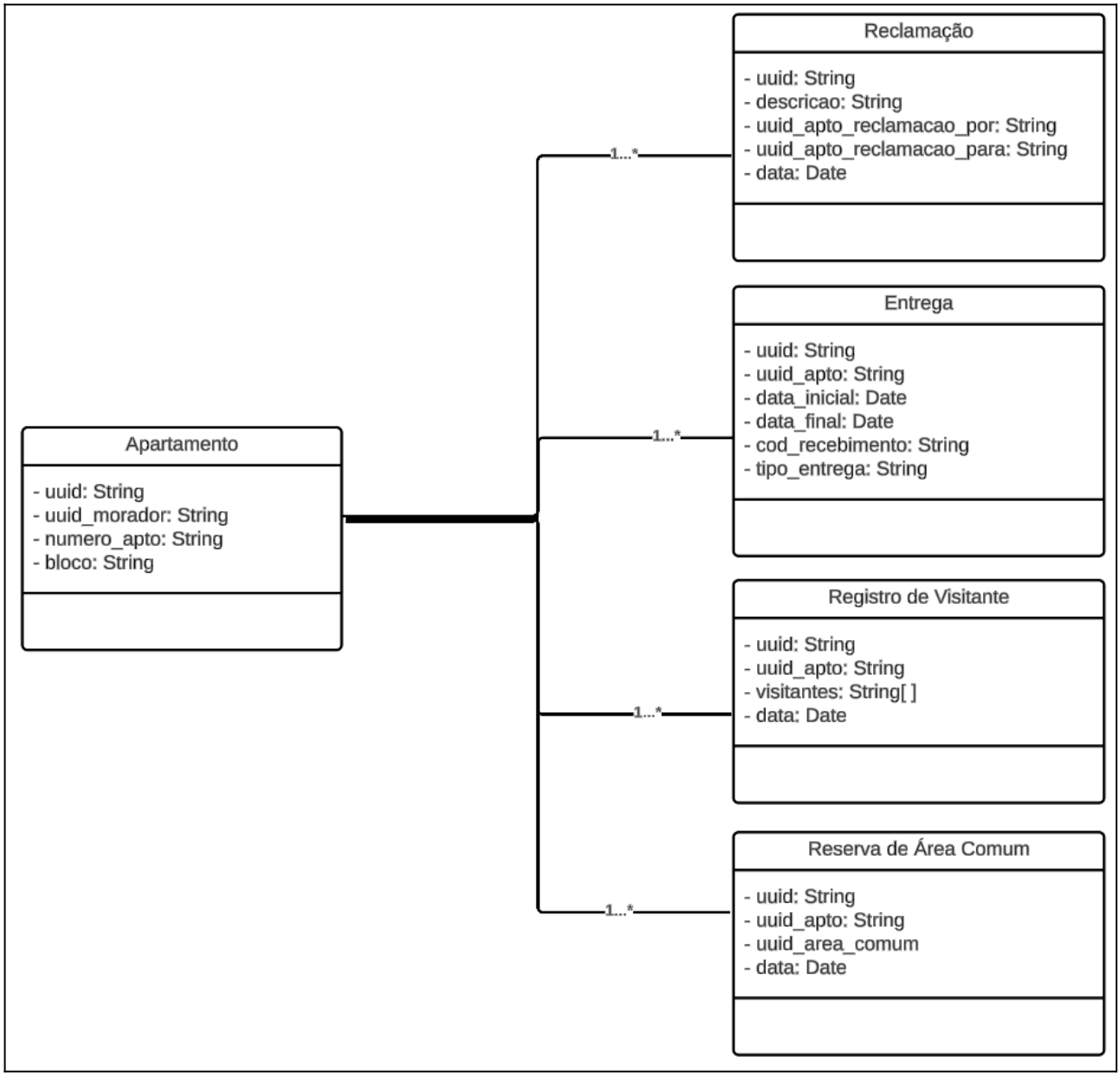

Figura 4: Diagrama de Classe de Apartamento com possíveis eventos do sistema

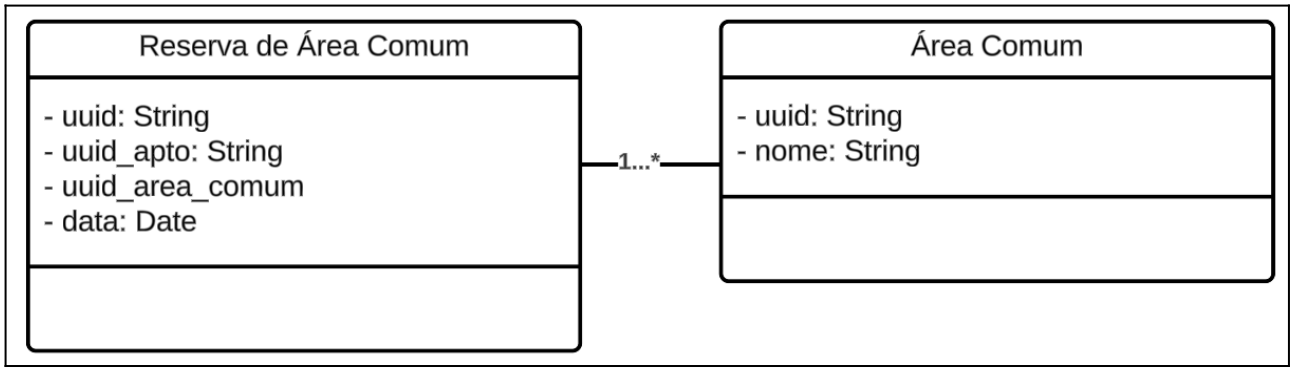

Figura 5: Diagrama de Classe Entre Reserva de Área Comum e Área Comum

3.4. DIAGRAMAS FLUXO

Este capítulo contém os diagramas de fluxo do projeto em questão, com seus eventos e atores.

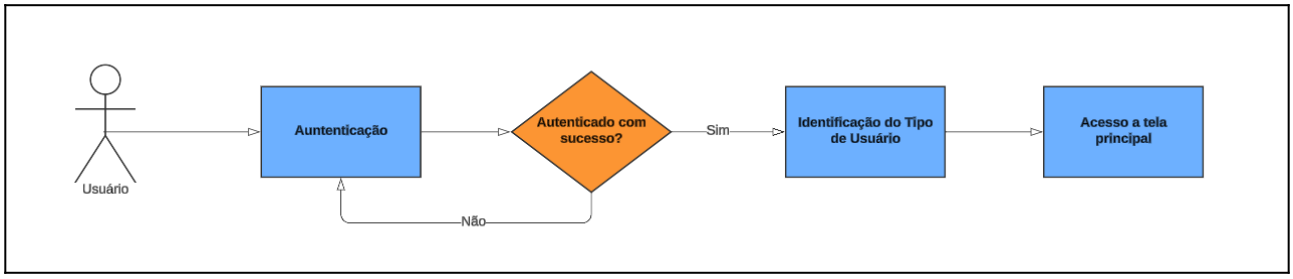

Figura 6: Fluxo de Login

Nome do Fluxo	Efetuar Login
Atores	Usuário, Morador e Portaria
Cenário Principal	1. A aplicação mostra a tela de autenticação para o Usuário. 2. O Usuário executa o login no sistema inserindo suas credenciais. 3. O Sistema verifica e valida os dados retornando informações sobre o Usuário que serão armazenadas no navegador. 4. Usuário se conecta.
Cenário Alternativo	1. Usuário não possui cadastro, neste caso, deve ser solicitado a adição do usuário ao sistema.
Casos de Teste	1. Caso as credenciais estiverem corretas, executa o Login. 2. Caso as credenciais estiverem incorretos o Login é cancelado.

Tabela 1: Efetuar Login

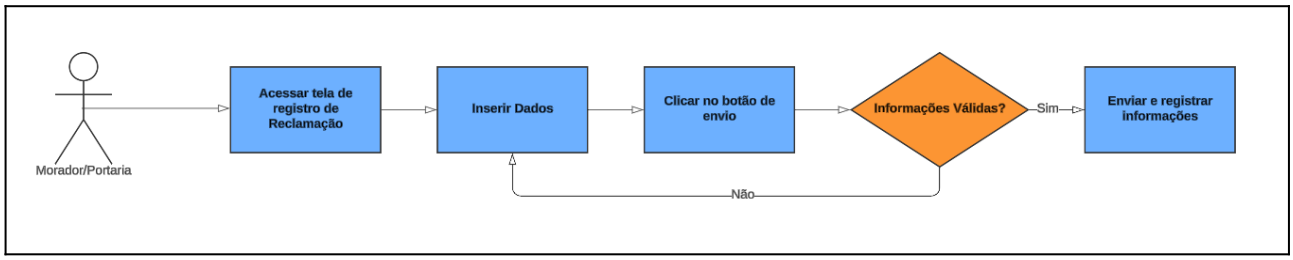

Figura 7: Registrar Reclamação

Nome do Fluxo	Registrar Reclamação
Atores	Morador / Portaria
Pré-condições	Estar Autenticado
Cenário Principal	Registrar uma reclamação de algum morador: 1. Apartamento da Denuncia (apenas caso Portaria); 2. Descrição; 3. Bloco; 4. Andar.
Cenário Alternativo	Não tem.
Casos de Teste	1. Validação das informações 2. Caso o usuário clique em cancelar, os campos devem ter seus valores padrões vazios.

Tabela 2: Registrar Reclamações

Os fluxos relacionados aos demais registros seguem a mesma lógica, no entanto, sendo apresentados em suas respectivas telas.

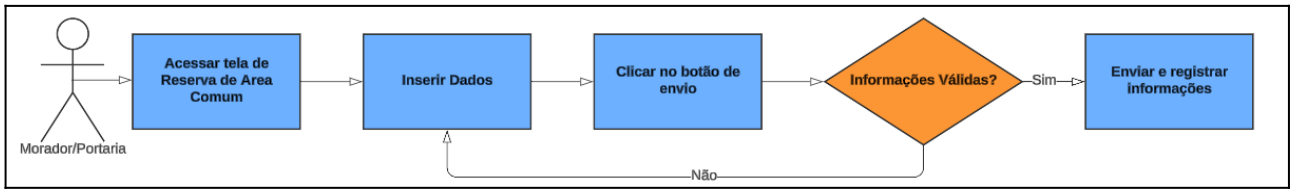

Figura 8: Reservar Área Comum

Nome do Fluxo	Reserva Área Comum
Atores	Morador / Portaria
Pré-condições	Estar Autenticado
Cenário Principal	Registrar uma reserva de área comum: 1. Apartamento da reserva (apenas caso Portaria); 2. Selecionar Área Comum desejada 3. Selecionar Data
Cenário Alternativo	Não Tem.
Casos de Teste	Caso Área Comum já esteja reservada para o mesmo dia, resultar em erro para o usuário alertando sobre tal.

Tabela 3: Fluxo da Reserva de Área Comum

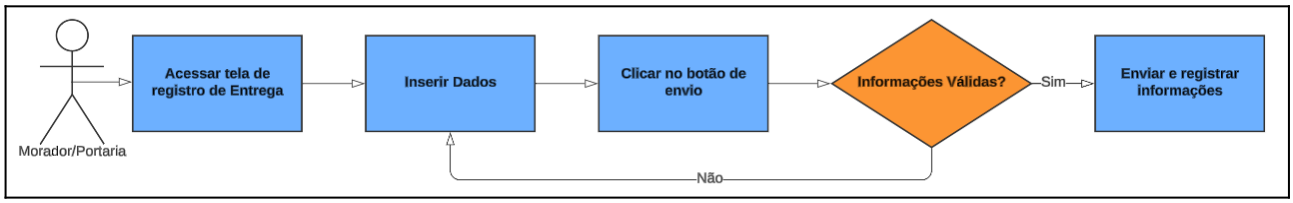

Figura 9: Registrar Entrega

Nome do Fluxo	Registrar Entrega
Atores	Morador
Pré-condições	Estar Autenticado
Cenário Principal	Registrar uma entrega designada para um apartamento: 1. Selecionar Tipo de Entrega entre Delivery ou Encomenda 2. Intervalo de Data previsto 3. Código de recebimento (opcional)
Cenário Alternativo	Não Tem.

Tabela 4: Registrar Entrega

4. DESENVOLVIMENTO DO PROJETO

No processo de desenvolvimento do trabalho, alguns tópicos se apresentaram redundantes como, por exemplo, a criação de eventos. Por conta disso, foi necessário reformular os diagramas de classes, assim resultando no que foi apresentado aqui. Juntamente, devido a dificuldade da implementação de algumas ferramentas, as funcionalidades de chat e envio de e-mail foram retiradas desta etapa.

4.1. FERRAMENTAS UTILIZADAS

Nos próximos capítulos serão descritos as ferramentas utilizadas no processo de criação deste software.

4.1.1. VISUAL STUDIO CODE

O principal editor de código utilizado foi o “Visual Studio Code”, escolhido por sua capacidade de lidar facilmente com TypeScript. Sua possibilidade de trabalhar tanto com React quanto com Nest.js ajudou no desenvolvimento, principalmente, devido suas ferramentas integradas.

4.1.2. REACT.JS E VITE

Sendo uma das principais ferramentas de desenvolvimento web atualmente, o React.js é uma biblioteca JavaScript desenvolvida pelo Meta, antigo Facebook, utilizada para a construção de interfaces de usuário dinâmicas e interativas. Com seu lançamento em 2013, sua principal característica é a criação de componentes reutilizáveis que facilitam a manutenção e a escalabilidade de aplicações web.

Utilizando um conceito chamado "Virtual DOM", o React.js otimiza o desempenho através da atualização de apenas das partes da interface que sofreu alterações, em vez de atualizar toda a página. Juntamente a essa ferramente foi necessário utilizar a Vite, que se trata de uma ferramenta responsável pelo gerenciamento do processo de build dos pacotes do React.js. Além disso, ela também auxilia durante o processo de desenvolvimento através do recarregamento rápido após cada alteração.

4.1.3. NEST.JS e TYPEORM

O Nest.js atua como um framework para o desenvolvimento de aplicações backend que utilizam o Node.js. Tal ferramenta se inspira na arquitetura modular do Angular, fazendo o uso de módulos, controladores e serviços, facilitando assim a organização do código. Utilizando o TypeScript de forma padrão, a ferramenta Nest.js proporciona uma tipagem estática. Sua integração com o TypeORM, uma biblioteca de mapeamento objeto-relacional, permite o gerenciamento do bancos de dados utilizando conceitos como entidades e repositórios.

4.1.4 POSTGRESQL

O PostgreSQL é o sistema de banco de dados qual foi utilizado para realizar o desenvolvimento deste projeto.

5. IMPLEMENTAÇÕES E RESULTADO FINAL

Nesta seção do projeto, será abordado a implementação no intuito de demonstrar o resultado atingido.

Segue a tela de Login juntamente das demais telas do sistema.

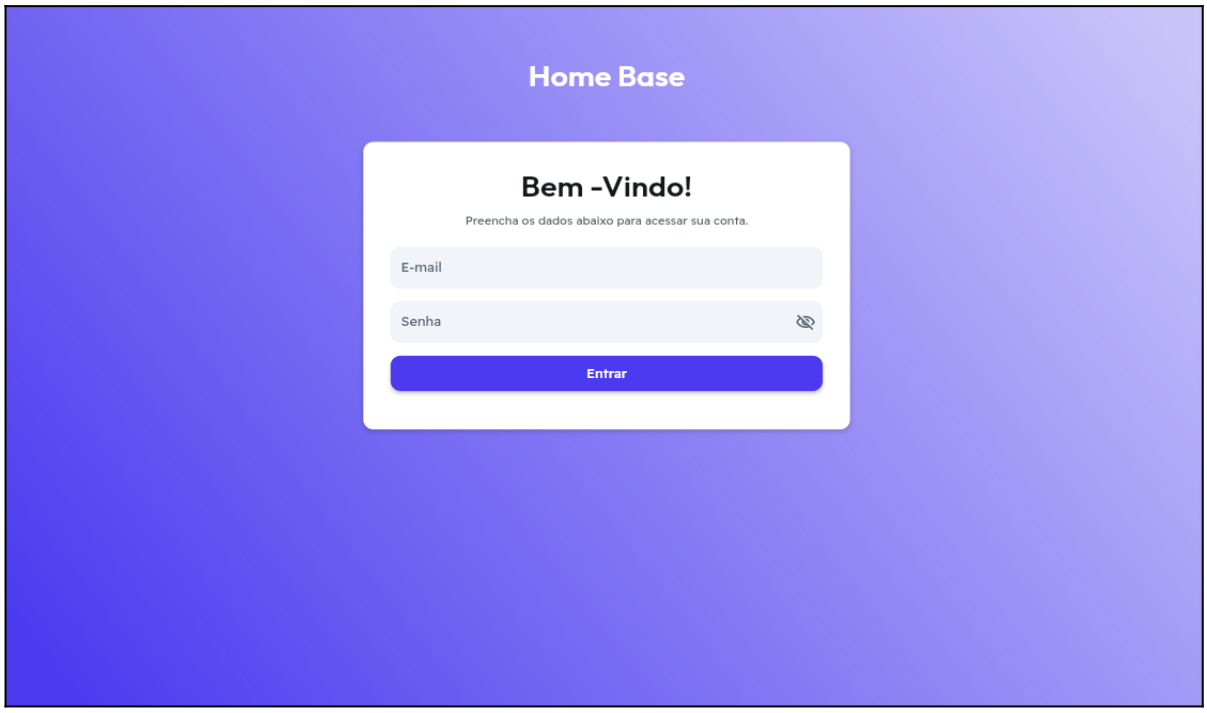

Figura 10: Tela de Login

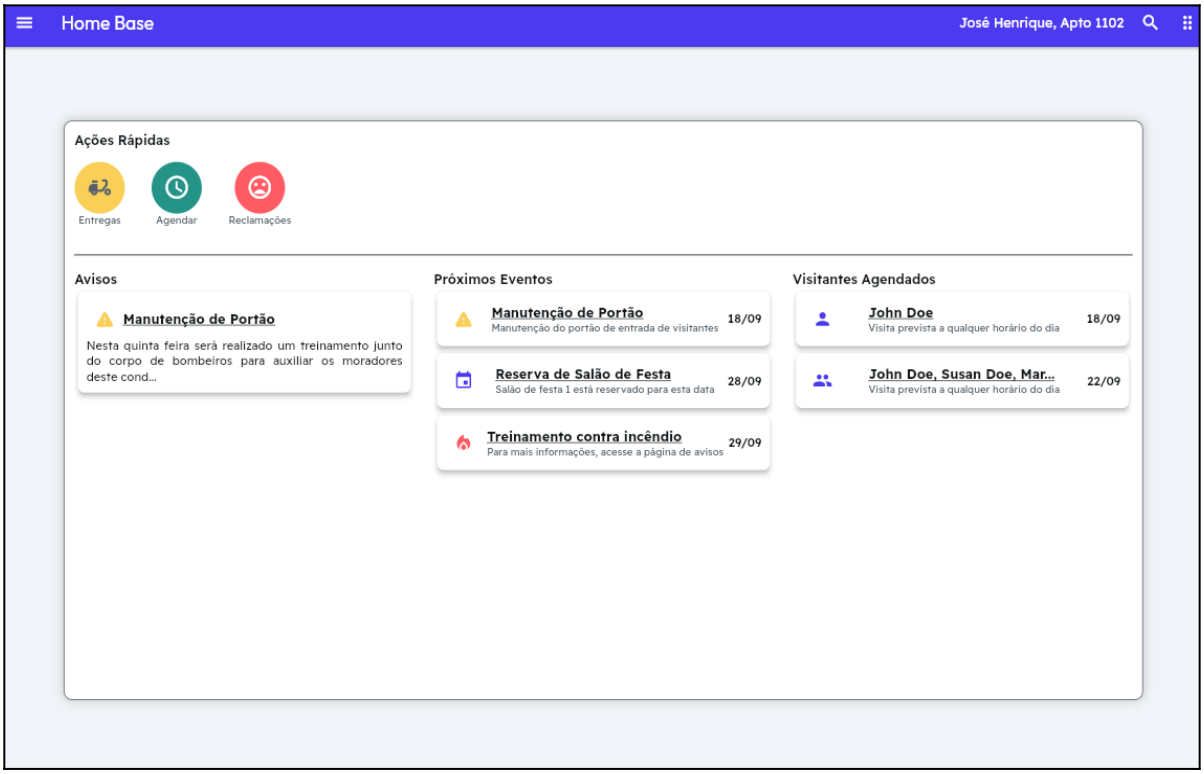

Figura 11: Morador - Tela Home

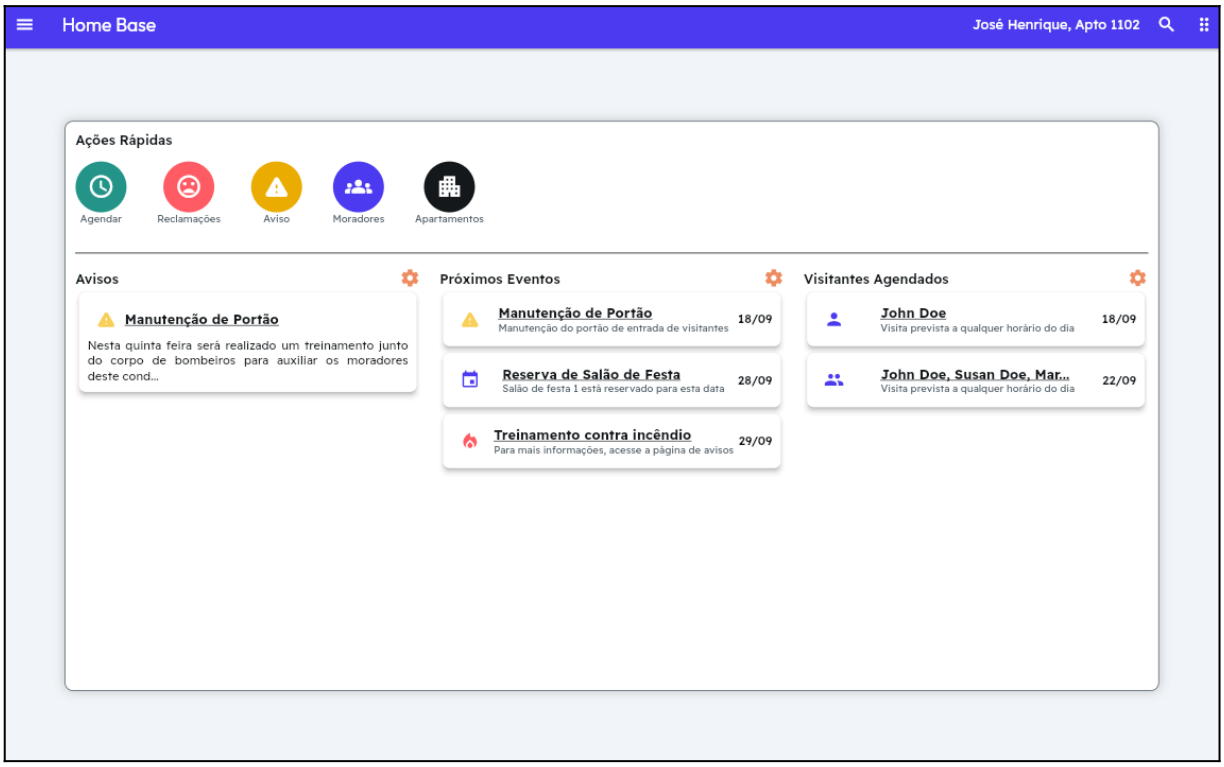

Figura 12: Portaria - Tela Home

Figura 13: Tela Registrar Reclamação

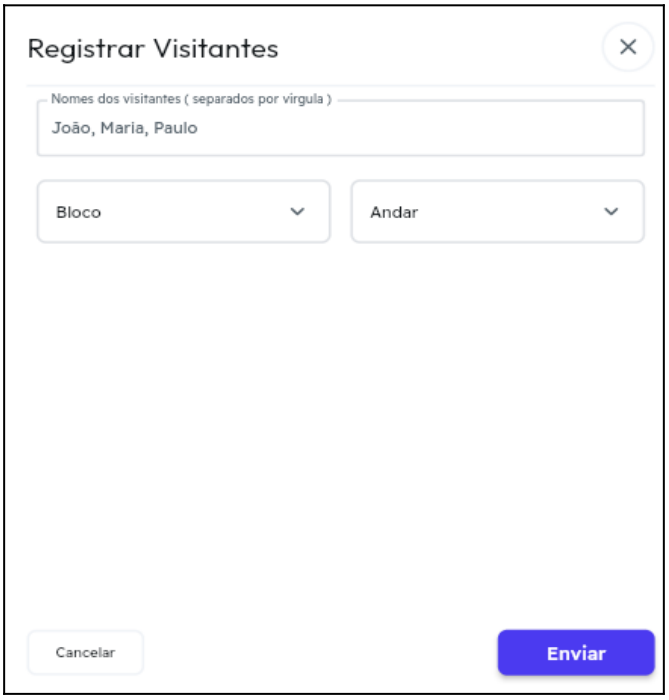

Figura 14: Tela Registrar Visitantes

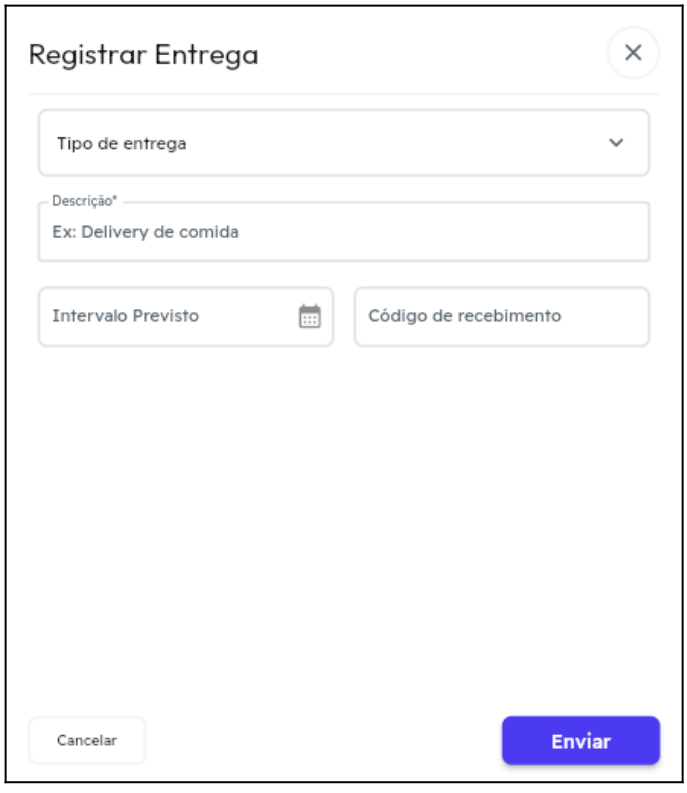

Figura 15: Tela Registrar Entrega

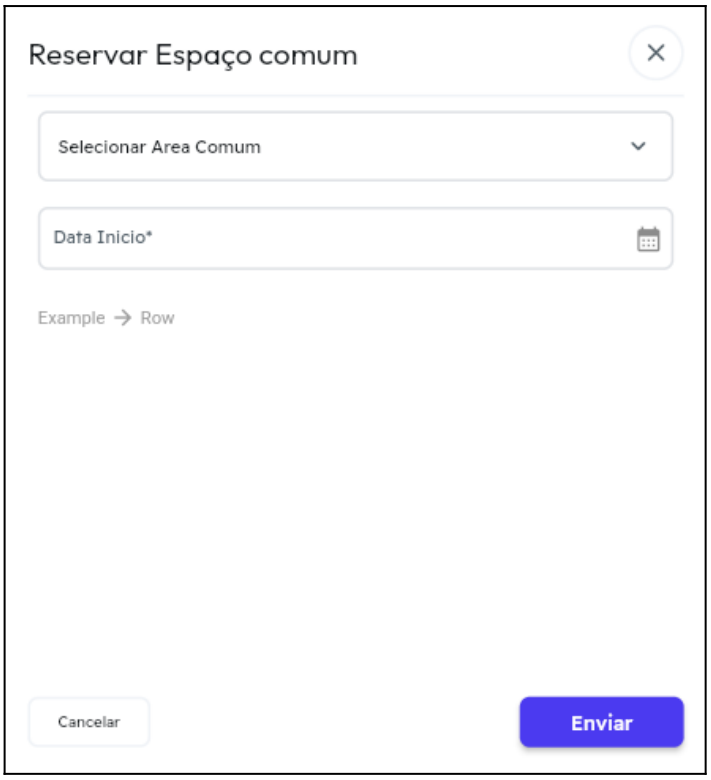

Figura 16: Tela Reserva Espaço Comum

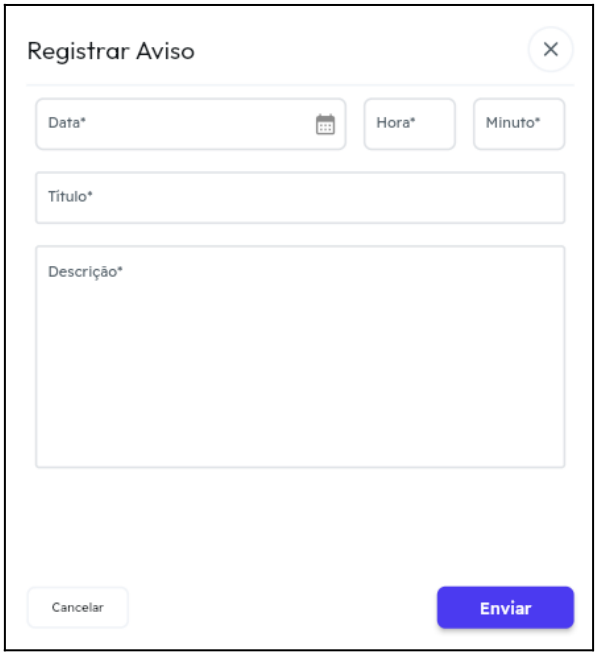

Figura 17: Tela Registrar Aviso

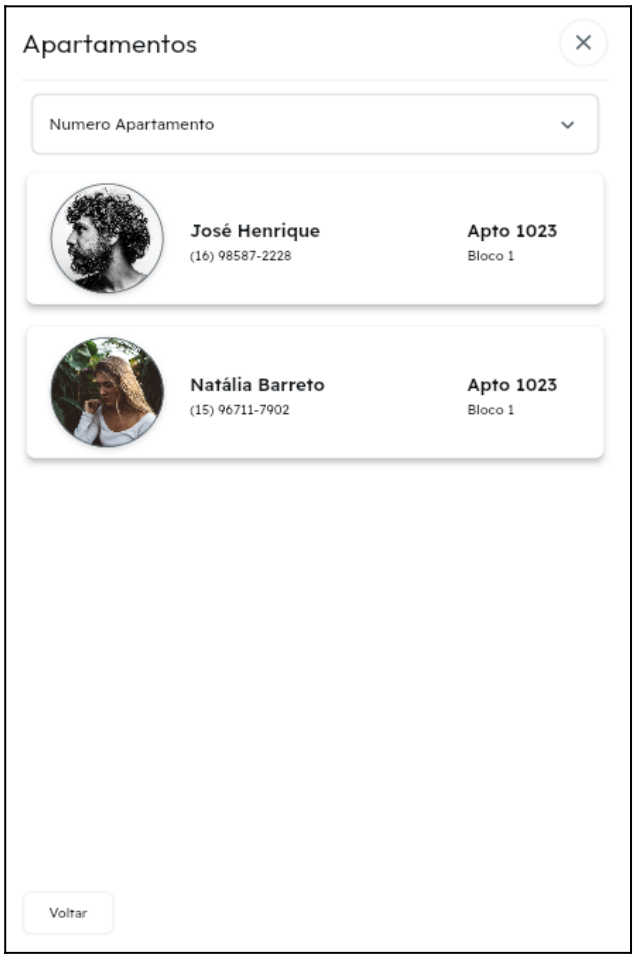

Figura 18: Tela Visualizar Apartamentos

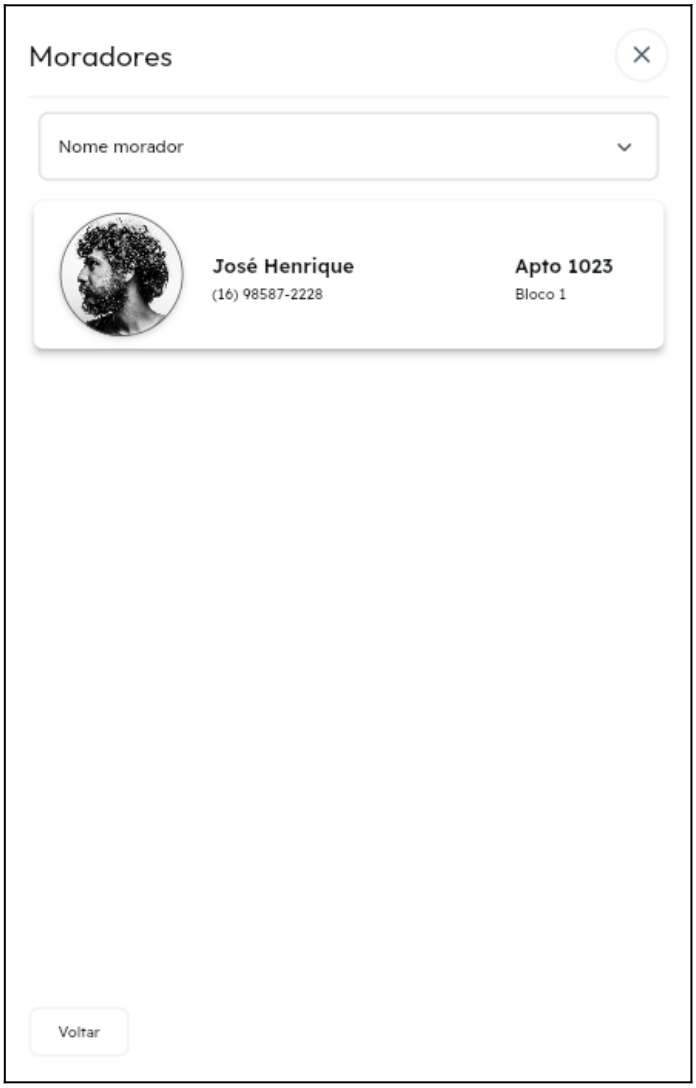

Figura 19: Tela Visualizar Moradores

6. CONCLUSÕES

O desenvolvimento da aplicação web para o gerenciamento de condomínios, proposta neste trabalho de conclusão de curso, ressalta a importância de soluções tecnológicas na melhoria da convivência e na eficiência administrativa dos espaços habitacionais. O projeto, que tem como fundamento principal as tecnologias React.js e Nest.js, resultou em um sistema que visa atender as necessidades diárias dos moradores e oferecer uma plataforma integrada que facilite a comunicação e a interação dos usuários.

Uma das principais contribuições desta aplicação é a promoção de uma comunicação transparente, algo essencial para a construção de um ambiente harmonioso. Além disso, o sistema de agendamento de áreas comuns e o registro de reclamações tem como objetivo organizar e gerenciar as atividades e eventos cotidianos, minimizando conflitos e promovendo um uso mais eficiente da área do condomínio.

Em resumo, as funcionalidades implementadas visam não apenas a organização administrativa, mas também a promoção de um ambiente de convivência mais saudável. Para futuras pesquisas e desenvolvimentos, recomenda-se a exploração de novos recursos como a integração de sistemas financeiros, expandindo ainda mais as capacidades da aplicação para atender futuras demandas.

6.1. TRABALHOS FUTUROS

No intuito de aprimorar a aplicação, melhorias como a implementação de um chat integrado e um sistema de disparo de e-mails serão trabalhadas. A ferramenta Twilio já disponibiliza esses recursos para serem integrados com aplicações React.js em um plano gratuito.

Futuras pesquisas podem trabalhar a adaptação deste trabalho para um dispositivo móvel, afim de trazer um melhor conforto e facilidade em sua utilização.

REFERÊNCIAS

NestJS - A progressive Node.js framework. Disponível em: <https://nestjs.com/>. Acesso em 12/04/2024

META OPEN SOURCE. React. Disponível em: <https://react.dev/>. Acesso em 15/04/2024

MARTIN, R. C. Clean code: A handbook of agile software craftsmanship. Filadélfia, PA, USA: Prentice Hall, 2009.

Communication APIs for SMS, voice, video & authentication. Disponível em: <https://www.twilio.com/en-us>. Acesso em: 14/04/2024.

Figma: The Collaborative Interface Design Tool. Disponível em: <https://www.figma.com/>. Acesso em: 20/04/2024.

ALURA. O que é JSON Web Tokens. Disponível em: <https://www.alura.com.br/artigos/o-que-e-json-web-tokens?srsltid=AfmBOoqfqv_7OPdO5gAAQqYvk8dyulEWzP849XLYEK8rFo5QEKZhCkQb>. Acesso em: 01/08/2024.

IETF. RFC 7519: JSON Web Token (JWT). Disponível em: <https://datatracker.ietf.org/doc/html/rfc7519>. Acesso em: 04/08/2024.

LUCIDCHART. Lucidchart: a visual workspace for diagramming. Disponível em: <https://www.lucidchart.com/pages/landing?utm_source=google&utm_medium=cpc&utm_campaign=_chart_en_tier3_mixed_search_brand_exact_&km_CPC_CampaignId=1484560207&km_CPC_AdGroupID=60168114191&km_CPC_Keyword=lucidchart&km_CPC_MatchType=e&km_CPC_ExtensionID=&km_CPC_Network=g&km_CPC_AdPosition=&km_CPC_Creative=442433234360&km_CPC_TargetID=kwd-33511936169&km_CPC_Country=9100772&km_CPC_Device=c&km_CPC_placement=&km_CPC_target=&gad_source=1&gclid=CjwKCAjwvKi4BhABEiwAH2gcw47qpMtCwFO15xoLfC3FDxHlsvOARn5_ZPv6C-Qzx15EMa9FfTOd3xoCXawQAvD_BwE>. Acesso em: 29/09/2024.

TYPEORM. TypeORM. Disponível em: https://typeorm.io/. Acesso em: 01/10/2024.

POSTGRESQL. About PostgreSQL. Disponível em: https://www.postgresql.org/about/. Acesso em: 14 out. 2024.

REACT. Reference | React. Disponível em: https://react.dev/reference/react. Acesso em: 01/10/2024.