



**UNIVERSIDADE DE FORTALEZA
VICE REITORIA DE GRADUAÇÃO
CENTRO DE CIÊNCIAS TECNOLÓGICAS
TECNÓLOGO EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

Weyne Gabriel
Luanna Campos
Davi Sales Vila Nova
Joao Pedro Lopes
Rhyan
Ricardo Junior
Joao Victor Guimarães

**DOCUMENTAÇÃO TÉCNICA
ASSISCONNECT 2.0**

FORTALEZA
2025

DOCUMENTAÇÃO TÉCNICA

Sistema XPTO de Gestão de Pedras Ornamentais

Este documento contém a documentação técnica do sistema **AssisConnect 2.0**, desenvolvido na componente curricular **N392 - Projeto Aplicado Plataformas Web**, como requisito para a obtenção de nota.

O sistema foi elaborado com o objetivo de modernizar a gestão do **Lar Francisco de Assis**, centralizando informações sobre idosos, atividades, usuários e rotinas diárias, promovendo maior organização, rastreabilidade e eficiência nas operações da instituição.

Supervisor: Prof. Bruno Lopes, Me

FORTALEZA

2025

SUMÁRIO

Sumário

1. INTRODUÇÃO	5
1.1. CONTEXTO E JUSTIFICATIVA.....	5
1.2. OBJETIVOS.....	5
6.1. ESCOPO E DELIMITAÇÃO.....	6
7. ENGENHARIA DE REQUISITOS	7
7.1. REQUISITOS FUNCIONAIS (RFs)	7
7.2. REQUISITOS NÃO FUNCIONAIS (RNFs)	7
8. PROJETO E ARQUITETURA DO SOFTWARE	9
8.1. ARQUITETURA GERAL.....	9
8.2. PROJETO DO BANCO DE DADOS	10
1. TECNOLOGIAS E FERRAMENTAS	15
2. Frontend (Camada de Apresentação):	15
3. React 19 — Biblioteca JavaScript para construção da interface do usuário.....	15
4. Vite 7 — Ferramenta de build moderna, utilizada para otimizar o desenvolvimento e o carregamento da aplicação.....	15
5. Axios 1.11 — Biblioteca para realizar requisições HTTP entre frontend e backend.	15
6. Bootstrap Icons — Biblioteca de ícones vetoriais usada na interface.	15
7. CSS3 Modularizado — Organização visual e responsiva, garantindo melhor experiência em múltiplos dispositivos.	15
8. Backend (Camada de Aplicação):	15
9. Java 21 — Linguagem orientada a objetos robusta e multiplataforma.....	15
10. Spring Boot 3.5.5 — Framework que simplifica o desenvolvimento de APIs RESTful.....	15
11. Spring Security — Responsável pela autenticação e autorização via tokens JWT.	15
12. Spring Data JPA — Gerenciamento de entidades e persistência de dados.	16

13.	JJWT 0.11.5 — Geração e validação dos tokens JWT para autenticação.....	16
14.	MySQL Connector/J — Driver de integração com o banco de dados.....	16
15.	Banco de Dados (Camada de Dados):	16
16.	MySQL 8.0 — Sistema de gerenciamento de banco de dados relacional, escolhido por sua confiabilidade e compatibilidade com o Spring Boot.	16
17.	IMPLEMENTAÇÃO E RESULTADOS	17
17.1.	TELAS DO SISTEMA	17
18.	AMBIENTE E GUIA DE IMPLANTAÇÃO	22
	PROCESSO DE IMPLANTAÇÃO.....	22
7.	CONCLUSÃO	24
7.1.	TRABALHOS FUTUROS.....	24
7.2.	LIÇÕES APRENDIDAS	24

1. INTRODUÇÃO

1.1. CONTEXTO E JUSTIFICATIVA

O cuidado com idosos em instituições de longa permanência exige organização e atenção constante, especialmente na gestão de informações sobre saúde, atividades e comunicação com familiares. No entanto, muitas dessas instituições ainda dependem de planilhas, registros manuais e trocas de mensagens informais, o que torna o acompanhamento diário suscetível a erros, perdas de dados e falhas de comunicação entre cuidadores, coordenadores e familiares.

O **Lar Francisco de Assis**, instituição escolhida como estudo de caso, enfrentava justamente esse cenário: a ausência de um sistema unificado dificultava o controle de rotinas, o registro de atividades e a atualização das informações de cada residente.

O sistema **AssisConnect 2.0** foi desenvolvido para resolver esses desafios, oferecendo uma plataforma web centralizada e intuitiva que automatiza processos, organiza cadastros de idosos, permite o agendamento e acompanhamento de atividades, além de gerar relatórios e indicadores úteis para a gestão. Essa digitalização proporciona mais eficiência, transparência e segurança na administração das rotinas, contribuindo para um cuidado mais humanizado e estruturado dentro da instituição.

1.2. OBJETIVOS

Desenvolver um sistema web denominado **AssisConnect 2.0**, voltado para o gerenciamento de rotinas e atividades em instituições de acolhimento de idosos, com o propósito de centralizar informações, otimizar a comunicação entre cuidadores e familiares e aprimorar a organização das atividades diárias do lar.

Objetivos Específicos:

2. Implementar um módulo de **cadastro e gerenciamento de idosos**, contendo dados pessoais, histórico de saúde e responsável associado.
3. Desenvolver um **sistema de gestão de atividades**, permitindo o registro, edição, exclusão e associação de idosos a cada atividade.
4. Criar um **mecanismo de autenticação e controle de acesso**, com perfis diferenciados (Administrador, Funcionário e Familiar), garantindo segurança e integridade das informações.
5. Disponibilizar **relatórios e métricas rápidas**, como o número total de idosos cadastrados e a listagem de aniversariantes do mês.
6. Proporcionar uma **interface moderna e responsiva**, que facilite o uso por diferentes perfis de usuários e dispositivos.

6.1. ESCOPO E DELIMITAÇÃO

O sistema AssisConnect 2.0 abrange o gerenciamento completo das informações relacionadas aos idosos residentes no Lar Francisco de Assis, suas atividades e os usuários responsáveis pelo cuidado e administração da instituição. O sistema possibilita:

- Cadastro, edição e exclusão de idosos, com informações pessoais e dados de saúde.
- Gestão de usuários, permitindo o controle de acessos e permissões conforme o papel de cada perfil (Administrador, Funcionário ou Familiar).
- Cadastro e gerenciamento de atividades, com definição de horários, responsáveis e alocação de idosos.
- Visualização de relatórios e indicadores, como contagem total de idosos e aniversariantes do mês.
- Autenticação e autorização via JWT, assegurando o acesso protegido às funcionalidades do sistema.
- Interface web responsiva e intuitiva, permitindo o uso em diferentes dispositivos.

Delimitação (fora do escopo):

- O sistema não contempla módulos financeiros, como controle de contas, folha de pagamento ou gestão de despesas.
- Não há integração com sistemas externos de saúde, prontuários eletrônicos ou aplicativos móveis nesta versão.
- O módulo de cardápio e controle alimentar ainda não foi implementado, sendo planejado para futuras versões.
- O sistema não realiza comunicação direta via chat entre familiares e cuidadores, limitando-se ao controle e registro das informações institucionais.
- O processo de backup e restauração de dados deverá ser feito manualmente via banco de dados, sem automação integrada.

7. ENGENHARIA DE REQUISITOS

7.1. REQUISITOS FUNCIONAIS (RFs)

ID	Nome do Requisito	Descrição
RF01	Cadastrar Idoso	O sistema deve permitir que um administrador cadastre um novo idoso, informando nome, data de nascimento, estado de saúde, sexo e responsável.
RF02	Gerenciar Usuários	O sistema deve permitir que o administrador cadastre, edite e exclua usuários, definindo o papel de acesso (Administrador, Funcionário ou Familiar).
RF03	Autenticar Usuário	O sistema deve permitir o login seguro de usuários utilizando e-mail e senha válidos, com autenticação baseada em JWT.
RF04	Cadastrar Atividade	O sistema deve permitir o registro de novas atividades, informando nome, data, horário e responsável.
RF05	Alocar Idoso em Atividade	O sistema deve permitir associar um ou mais idosos a uma atividade cadastrada, possibilitando a organização da rotina diária.
RF06	Desalocar Idoso de Atividade	O sistema deve permitir a remoção de um idoso de uma atividade específica.
RF07	Listar Atividades por Idoso	O sistema deve permitir visualizar todas as atividades nas quais um idoso está cadastrado.
RF08	Gerar Relatórios e Métricas	O sistema deve apresentar informações consolidadas, como número de idosos cadastrados e aniversariantes do mês.
RF09	Exibir Dashboard	O sistema deve apresentar um painel de controle com indicadores gerais e atalhos para as principais funções administrativas.

7.2. REQUISITOS NÃO FUNCIONAIS (RNFs)

Usabilidade

- *RNF01: A interface deve ser intuitiva e responsiva, adaptando-se a diferentes dispositivos (desktop, tablet e mobile).*
- *RNF02: As telas devem seguir um padrão visual consistente, facilitando o aprendizado dos usuários.*

Desempenho

- *RNF03: As páginas principais devem carregar em até 3 segundos em ambiente local.*

- *RNF04: As respostas da API devem ocorrer em menos de 500 milissegundos para consultas simples.*

Segurança

- *RNF05: As senhas devem ser criptografadas com algoritmo seguro (BCrypt).*
- *RNF06: A comunicação entre frontend e backend deve ocorrer apenas via HTTPS.*
- *RNF07: O token JWT deve ser armazenado localmente e validado a cada requisição.*

Compatibilidade

- *RNF08: O sistema deve funcionar corretamente nas últimas versões dos navegadores Chrome, Firefox e Edge.*

Manutenibilidade

- *RNF09: O código deve ser modularizado seguindo o padrão MVC no backend e componentes reutilizáveis no frontend.*
- *RNF10: O sistema deve permitir fácil integração com futuras versões mobile.*

8. PROJETO E ARQUITETURA DO SOFTWARE

8.1. ARQUITETURA GERAL

O sistema AssisConnect 2.0 foi projetado utilizando uma Arquitetura em 3 Camadas, composta por:

1. *Camada de Apresentação (Frontend):*

Desenvolvida como uma Single Page Application (SPA) em React com o Vite como ferramenta de build.

É responsável pela interface gráfica e pela experiência de uso do sistema, permitindo navegação fluida e responsiva entre os módulos de idosos, atividades e usuários.

A escolha do React se deu por sua eficiência na criação de interfaces dinâmicas, sua comunidade ativa e a facilidade de integração com APIs REST.

2. *Camada de Aplicação (Backend API):*

Implementada em Java 21 utilizando o framework Spring Boot 3.5.5.

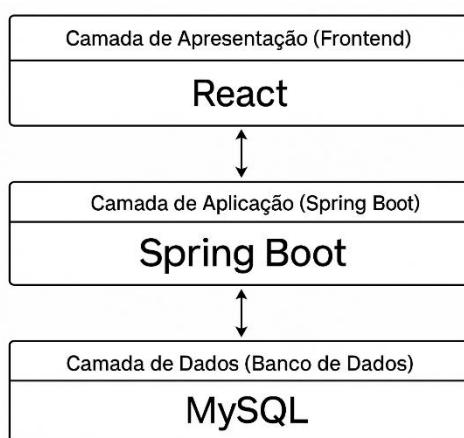
Essa camada contém toda a lógica de negócio, realiza a comunicação entre o frontend e o banco de dados e aplica as regras de autenticação e autorização baseadas em JWT (JSON Web Tokens).

O uso do Spring Boot foi escolhido por sua robustez, modularidade e integração simplificada com o Spring Data JPA, o que facilita o gerenciamento de entidades e a persistência de dados.

3. *Camada de Dados (Banco de Dados):*

O sistema utiliza um banco de dados relacional MySQL, responsável pela persistência e integridade das informações dos usuários, idosos e atividades.

O MySQL foi escolhido por sua estabilidade, ampla documentação e fácil integração com o Spring Boot, garantindo confiabilidade nas transações e escalabilidade para futuras expansões do sistema.



8.2. PROJETO DO BANCO DE DADOS

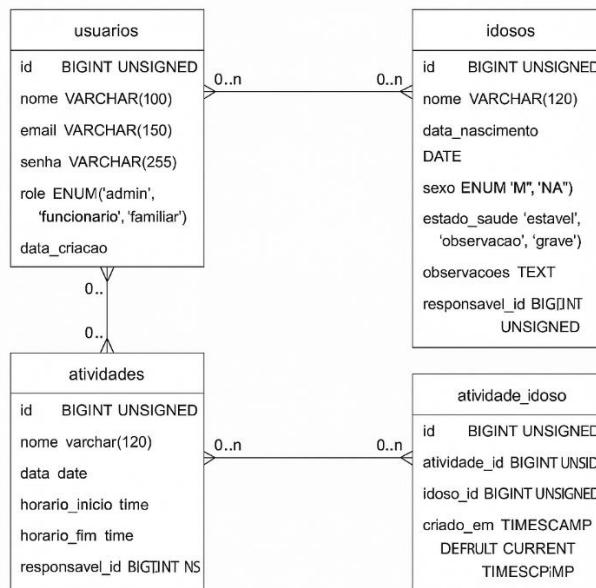
Entidades: *usuarios*, *idosos*, *atividades*, *atividade_idoso* (tabela de associação).

Relacionamentos:

- *usuarios* (1) —< (N) *idosos* via *idosos.responsavel_id*
- *usuarios* (1) —< (N) *atividades* via *atividades.responsavel_id*
- *atividades* (1) —< (N) *atividade_idoso* — (N) *idosos*

Restrições relevantes:

- *atividade_idoso* possui unicidade em (*atividade_id*, *idoso_id*).
- *atividades* possui CHECK garantindo *horario_fim > horario_inicio*.
- Índices para busca: *idx_idosos_nome*, *idx_ativ_data_inicio*, *idx_ativ_responsavel*.
- **ON DELETE/UPDATE: FKs com RESTRICT (em usuários) e CASCADE (na tabela de associação).**



Dicionário de Dados — usuarios

Nome do campo	Tipo de dados	Chave (PK/FK)	Nulo ?	Descrição
id	BIGINT UNSIGNED (AI)	PK	Não	Identificador único do usuário.
nome	VARCHAR(100)	—	Não	Nome completo do usuário.
email	VARCHAR(150) UNIQUE	—	Não	E-mail único para autenticação .
senha	VARCHAR(255)	—	Não	Hash da senha (ex.: BCrypt).
role	ENUM('admin','funcionario','familia r')	—	Não	Papel de acesso do usuário no sistema.
data_criacao	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	—	Sim	Data/hora de criação do registro.

Dicionário de Dados — idosos

Nome do campo	Tipo de dados	Chave (PK/FK)	Nulo ?	Descrição
id	BIGINT UNSIGNED (AI)	PK	Não	Identificador único do idoso.
nome	VARCHAR(120)	—	Não	Nome completo do idoso.
data_nascimento	DATE	—	Não	Data de nascimento.
sexo	ENUM('M','F','NA')	—	Não	Sexo biológico ou não informado.
estado_saude	ENUM('estavel','observacao','grave') DEFAULT 'estavel'	—	Não	Estado de saúde atual.
observacoes	TEXT	—	Sim	Observações clínicas/gerais.
responsavel_id	BIGINT UNSIGNED	FK → usuarios(id)	Não	Usuário responsável pelo cadastro/gestão.
criado_em	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	—	Sim	Data/hora de criação.
atualizado_em	TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	—	Sim	Última atualização do registro.

Índices: idx_idosos_responsavel (responsavel_id), idx_idosos_nome (nome).

Dicionário de Dados — atividades

Nome do campo	Tipo de dados	Chave (PK/FK)	Nulo?	Descrição
id	BIGINT UNSIGNED (AI)	PK	Não	Identificador único da atividade.
nome	VARCHAR(120)	—	Não	Nome/título da atividade.
data	DATE	—	Não	Data da realização.
horario_inicio	TIME	—	Não	Horário de início.
horario_fim	TIME	—	Não	Horário de término (deve ser > início).
responsavel_id	BIGINT UNSIGNED	FK → usuarios(id)	Não	Usuário responsável pela atividade.
observacoes	TEXT	—	Sim	Observações gerais.
criado_em	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	—	Sim	Data/hora de criação.
atualizado_em	TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP	—	Sim	Última atualização do registro.

Índices: idx_ativ_data_inicio (data, horario_inicio), idx_ativ_responsavel (responsavel_id)

Regra de integridade: CHECK (horario_fim > horario_inicio).

Dicionário de Dados — atividade_idoso (associação)

Nome do campo	Tipo de dados	Chave (PK/FK)	Nulo?	Descrição
id	BIGINT UNSIGNED (AI)	PK	Não	Identificador único da associação.
atividade_id	BIGINT UNSIGNED	FK → atividades(id)	Não	Atividade vinculada.
idoso_id	BIGINT UNSIGNED	FK → idosos(id)	Não	Idoso vinculado à atividade.
criado_em	TIMESTAMP DEFAULT CURRENT_TIMESTAMP	—	Sim	Data/hora de criação do vínculo.

Regras:

- **UNIQUE** uk_ai (atividade_id, idoso_id) — impede vínculo duplicado.
- **FKs** com ON DELETE CASCADE — ao remover a atividade ou o idoso, o vínculo é removido automaticamente.

Notas de implementação

- **Charset/Collation:** utf8mb4 / utf8mb4_0900_ai_ci (suporta acentos e emojis).
- **Integridade referencial:** RESTRICT em entidades mestre (usuarios), CASCADE na tabela de associação.
- **Performance:** índices criados para as consultas mais frequentes (por nome de idoso, por data/horário de atividade e por responsável).

1. TECNOLOGIAS E FERRAMENTAS

2. **Frontend (Camada de Apresentação):**
3. **React 19** — Biblioteca JavaScript para construção da interface do usuário.
4. **Vite 7** — Ferramenta de build moderna, utilizada para otimizar o desenvolvimento e o carregamento da aplicação.
5. **Axios 1.11** — Biblioteca para realizar requisições HTTP entre frontend e backend.
6. **Bootstrap Icons** — Biblioteca de ícones vetoriais usada na interface.
7. **CSS3 Modularizado** — Organização visual e responsiva, garantindo melhor experiência em múltiplos dispositivos.
8. **Backend (Camada de Aplicação):**
9. **Java 21** — Linguagem orientada a objetos robusta e multiplataforma.
10. **Spring Boot 3.5.5** — Framework que simplifica o desenvolvimento de APIs RESTful.
11. **Spring Security** — Responsável pela autenticação e autorização via tokens JWT.

12. **Spring Data JPA** — Gerenciamento de entidades e persistência de dados.
13. **JJWT 0.11.5** — Geração e validação dos tokens JWT para autenticação.
14. **MySQL Connector/J** — Driver de integração com o banco de dados.
15. **Banco de Dados (Camada de Dados):**
16. **MySQL 8.0** — Sistema de gerenciamento de banco de dados relacional, escolhido por sua confiabilidade e compatibilidade com o Spring Boot.

Ferramenta	Finalidade
Visual Studio Code	Edição do frontend (React/Vite).
IntelliJ IDEA	IDE para desenvolvimento Java e execução do backend.
MySQL Workbench	Modelagem e consultas SQL.
Postman / Insomnia	Testes de rotas e validação da API REST.
Git & GitHub	Controle de versão e colaboração em equipe.
Trello	Planejamento das tarefas e acompanhamento das sprints.

Justificativa das Tecnologias

A escolha de **React** e **Spring Boot** foi motivada pela ampla adoção no mercado, documentação completa e facilidade de integração.

Essa combinação permite criar uma aplicação **SPA (Single Page Application)** performática, com **API RESTful segura e escalável**.

O uso de MySQL garante estabilidade e eficiência no armazenamento de dados relacionais.

17. IMPLEMENTAÇÃO E RESULTADOS

17.1. TELAS DO SISTEMA

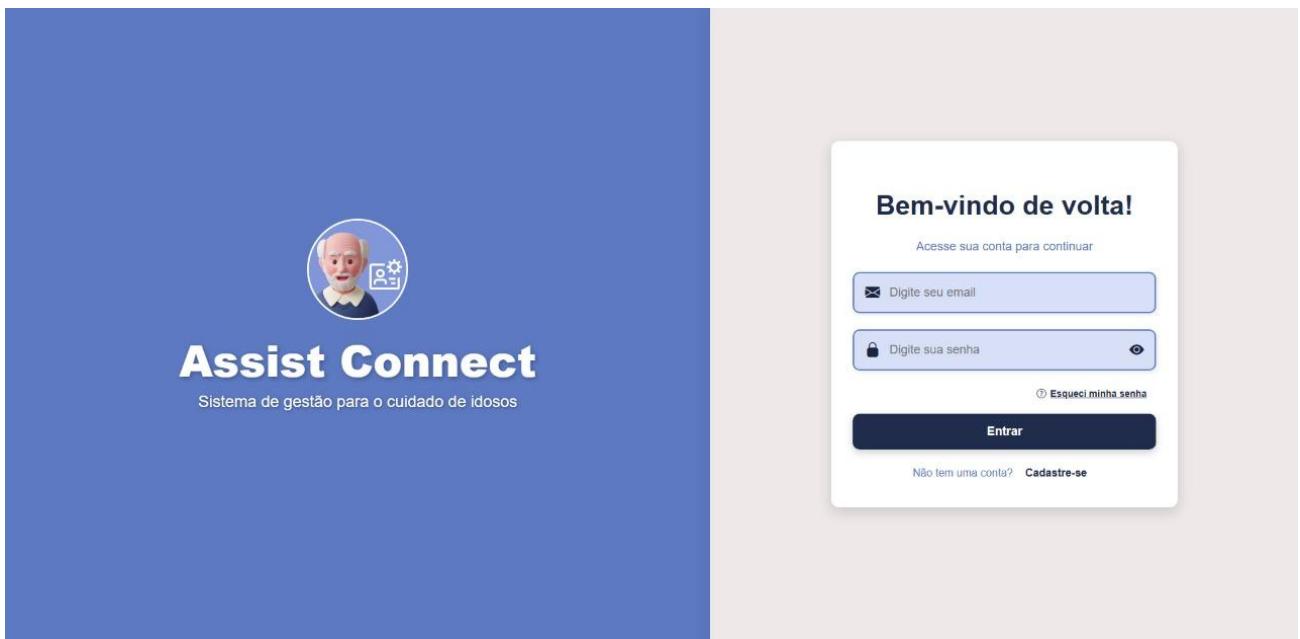


Figura 1: Tela de Login: Interface inicial do sistema AssisConnect 2.0, com campos de e-mail e senha para autenticação do usuário.

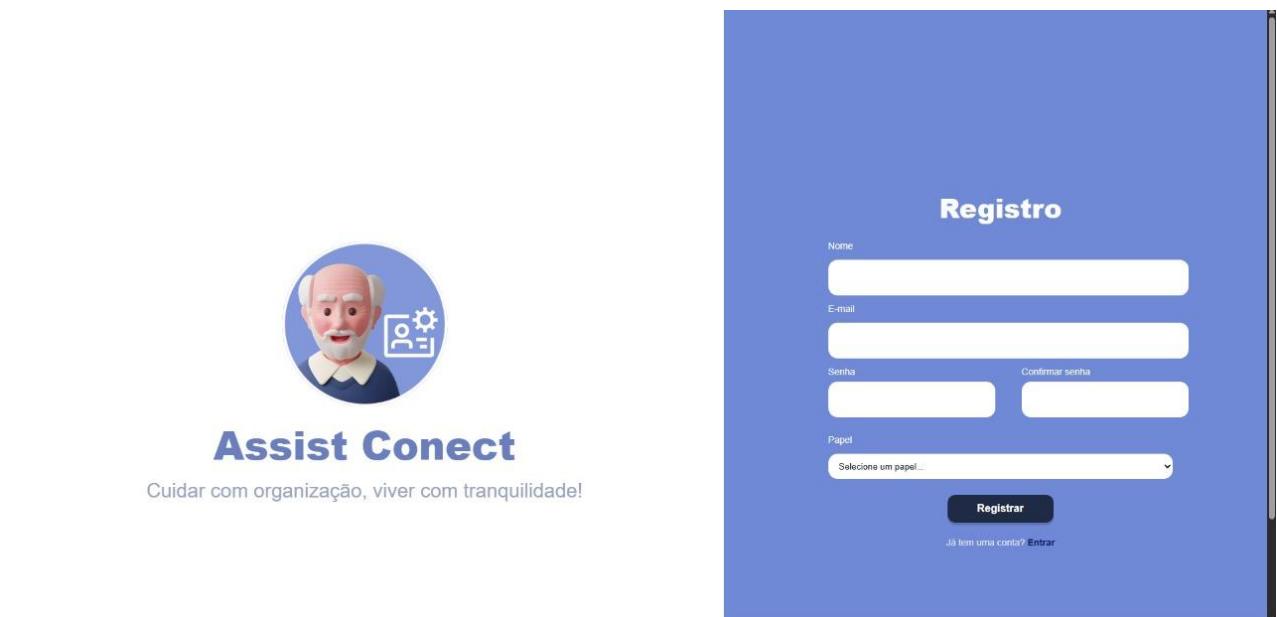


Figura 2 – Tela de Registro: Interface de cadastro do sistema AssisConnect 2.0, onde o usuário informa seus dados e define o papel de acesso (Administrador, Funcionário ou Familiar).

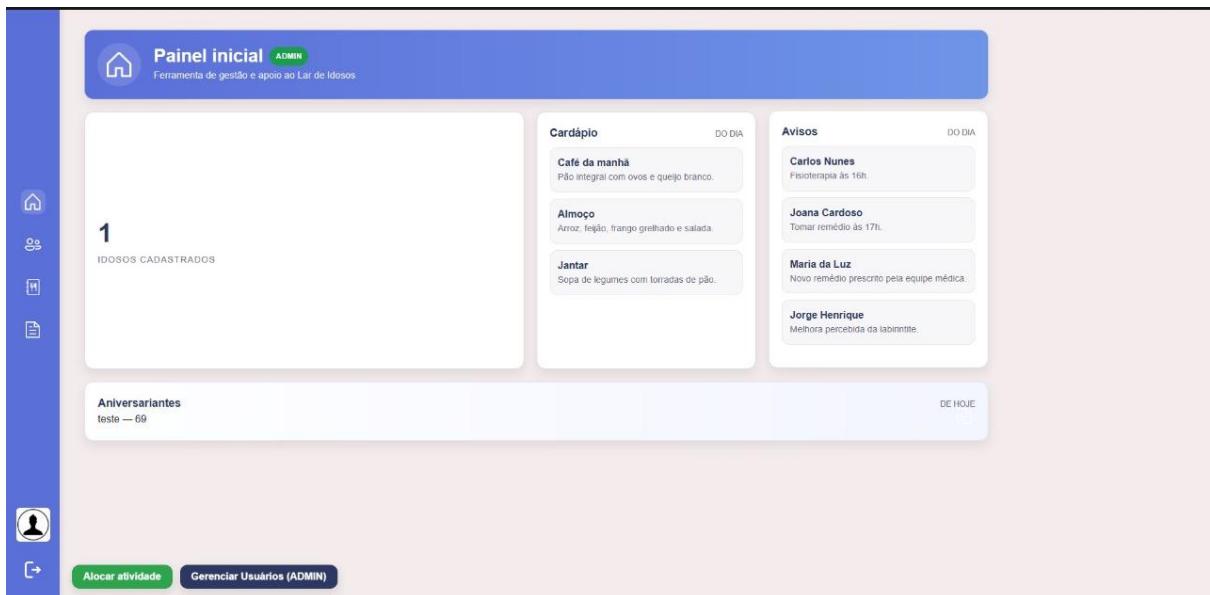


Figura 3 – Tela de Dashboard: Painel inicial do sistema AssisConnect 2.0, exibindo informações gerais como número de idosos cadastrados, cardápio do dia, avisos e aniversariantes.

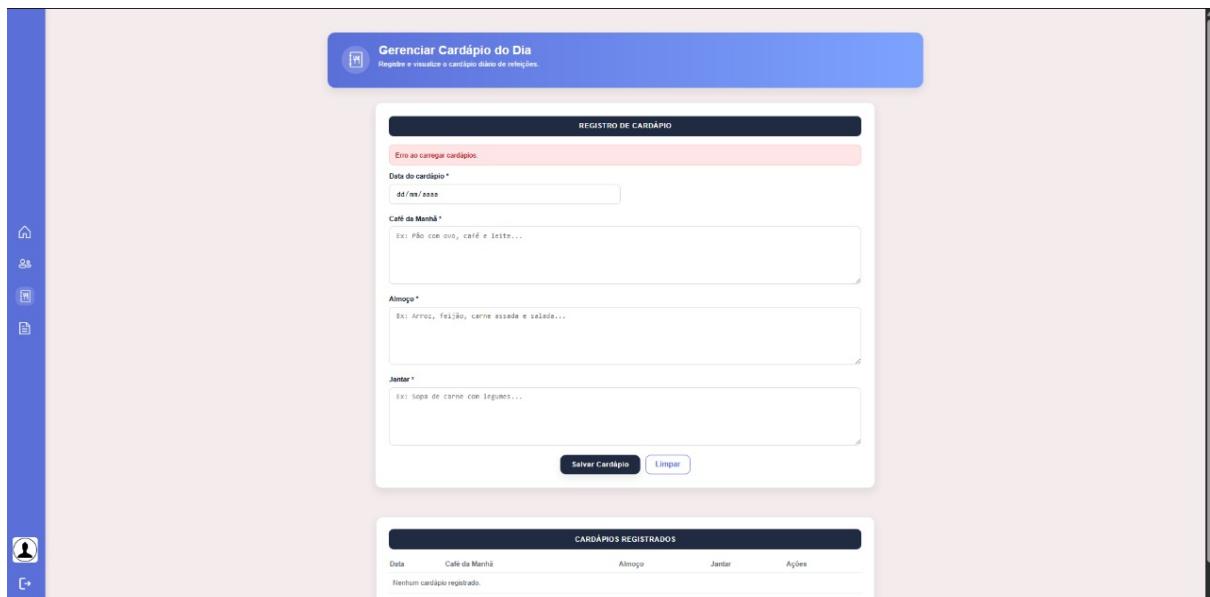


Figura 4 – Tela de Gerenciamento de Cardápio: Tela utilizada para registrar e visualizar o cardápio diário de refeições, permitindo cadastrar os itens de café da manhã, almoço e jantar.

Gerenciar Atividades
Crie, edite e visualize as atividades recreativas.

CADASTRO DE ATIVIDADES

Nome da atividade *	Ex: Bingo Musical
Data *	dd/mm/aaaa
Horário de inicio *	dd:mmaaaa
Horário de término *	dd:mmaaaa
Responsável (funcionario) *	Selecionar
Observações adicionais	Ex.: Materiais necessários, restrições de saúde, observações...

TODAS AS ATIVIDADES REGISTRADAS.

Nome	Data	Inicio	Termino	Responsável
teste	07/11/2025	10:00	11:00	Rhyan
aaaaa	06/12/2025	11:00	12:00	Rhyan

Figura 5 – Tela de Gerenciamento de Atividades: Interface destinada ao cadastro e visualização de atividades recreativas, permitindo registrar nome, data, horários, responsável e observações adicionais.

Gerenciar Idosos
Cadastre, edite ou remova idosos cadastrados.

CADASTRO DE IDOSO

Nome Completo *	Ex: Maria da Luz	Data de Nascimento *	dd/mm/aaaa
Sexo *	Selecionar	Estado de Saúde *	Selecionar
Observações	Alergias, medicações, observações clínicas...		
Responsável *	Selecionar		

IDOSOS CADASTRADOS

teste	Responsável: Admin	M • ESTAVEL	Detalhar	Ver atividades
-------	--------------------	-------------	----------	----------------

Figura 6 – Tela de Gerenciamento de Idosos: Tela destinada ao cadastro e visualização dos idosos do lar, com campos para informações pessoais, estado de saúde e responsável vinculado.

The screenshot shows a user management interface titled "Gerenciamento de Usuários". At the top, there are search fields for "Buscar por nome" and "Buscar por email", a dropdown for "Todas as roles" with "10 por página" selected, and buttons for "Aplicar" and "Limpar". To the right, a blue box shows "Total 3" and a "Atualizar" button. Below this is a table with three rows of user data:

	Nome	Email	Role	Ações
R	Rhyan	rhyandarocha2006@gmail.com	FUNCIONÁRIO	Editar Excluir
A	Admin	admin@gmail.com	FAMILIAR	Editar Excluir
R	Rhyan	rhyan@gmail.com	ADMIN	Editar Excluir

At the bottom, there are navigation links "← Anterior" and "Próxima →". On the right side, there is a sidebar titled "Ações rápidas" with a "Recarregar lista" button. The sidebar also includes a legend for role colors: red for Admin, blue for Funcionário, and green for Familiar.

Figura 7 – Tela de Gerenciamento de Usuários: Interface administrativa que permite visualizar, editar e excluir usuários, além de filtrar por nome, e-mail e tipo de acesso (Admin, Funcionário ou Familiar).

The screenshot shows a screen titled "Atividades do Idoso #2". On the left, a vertical sidebar has icons for Home, User, and Log Out. The main area starts with a section for adding activities:

Adicionar atividade: Vincular

teste
Data: 07/11/2025
Horário: 10:00 - 11:00

Below this, there is a list of activities:

- teste
- teste
- teste

Figura 8 – Tela de Atividades do Idoso: Exibe as atividades vinculadas a um idoso específico, permitindo adicionar ou remover participações de forma simples e rápida.

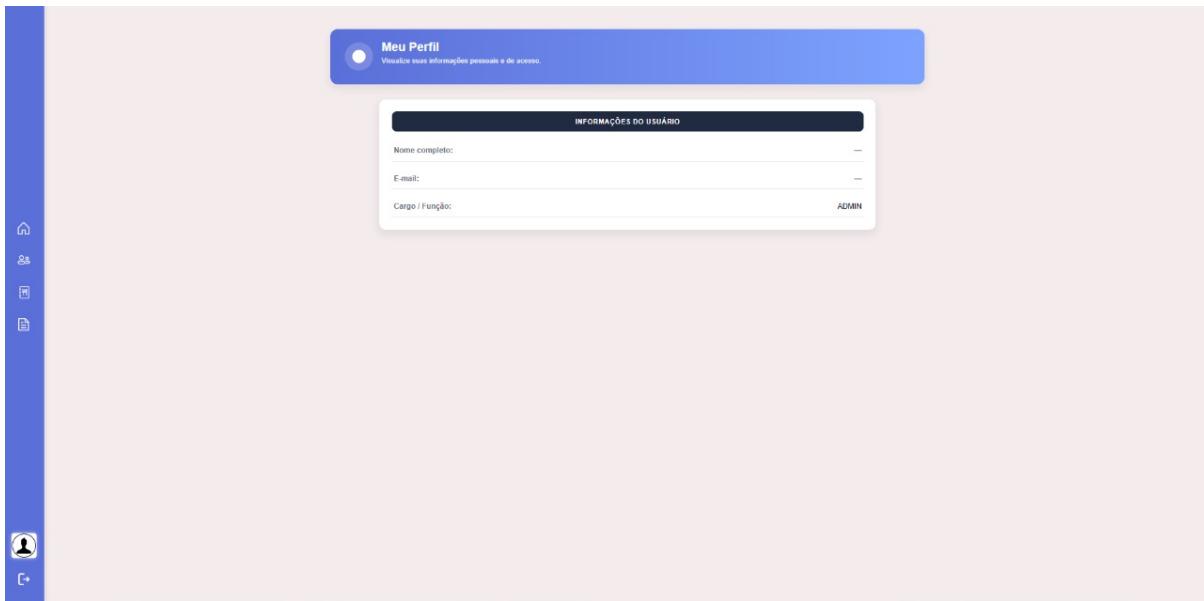


Figura 9 – Tela de Perfil do Usuário: Exibe as informações pessoais e o papel do usuário logado, permitindo a visualização de dados como nome, e-mail e função no sistema.

18. AMBIENTE E GUIA DE IMPLANTAÇÃO

Para execução do sistema **AssisConnect 2.0**, é necessário um ambiente configurado com os seguintes componentes:

Componente	Versão Mínima	Finalidade
Sistema Operacional	Windows 10 / Ubuntu 22.04	Execução local do sistema e do banco de dados.
JDK (Java Development Kit)	21	Execução da API Spring Boot.
Maven	3.9+	Gerenciamento de dependências e build do backend.
MySQL	8.0+	Armazenamento dos dados do sistema.
Node.js	18+	Execução do frontend React.
npm	10+	Instalação de dependências do frontend.
Git	2.34+	Controle de versão e integração com o repositório.

PROCESSO DE IMPLANTAÇÃO

1. Clonar o repositório

`git clone https://github.com/WeyneG/AssisConect2.0.git`

2. Configurar o banco de dados

- Criar um banco no MySQL com o nome *asilo*.
- Atualizar as credenciais no arquivo:

`backend/src/main/resources/application.properties`

Exemplo:

```
spring.datasource.url=jdbc:mysql://localhost:3306/asilo  
spring.datasource.username=root  
spring.datasource.password=suasenha  
spring.jpa.hibernate.ddl-auto=update
```

3.0 Executar o Backend

```
cd backend/backend  
mvn spring-boot:run
```

A API será iniciada em <http://localhost:8080>.

5.0 Executar o Frontend

```
cd Frontend  
npm install  
npm run dev
```

Acesso ao Sistema

Usuário padrão: admin@assisconnect.com

Senha: admin123

Perfis disponíveis: Administrador, Funcionário e Familiar.

Observações Importantes

Certifique-se de que o backend esteja ativo antes de iniciar o frontend.

Para ambientes de produção, recomenda-se:

Alterar as variáveis sensíveis (usuário do banco e JWT secret) para variáveis de ambiente.

Configurar servidor HTTPS.

Implementar políticas de CORS restritivas.

7. CONCLUSÃO

7.1. TRABALHOS FUTUROS

Embora o sistema AssisConnect 2.0 atenda aos objetivos propostos de automatizar o gerenciamento de idosos, atividades e usuários, há várias possibilidades de aprimoramento e expansão da plataforma. Entre as principais evoluções planejadas, destacam-se:

- *Implementação de um módulo de relatórios avançados, com gráficos interativos e filtros personalizados.*
- *Criação de um aplicativo mobile em Flutter, voltado para cuidadores e familiares, permitindo acesso rápido às informações e notificações.*
- *Desenvolvimento de um módulo de cardápio e controle alimentar, integrando rotinas nutricionais e planos de refeição.*
- *Integração com sistemas de prontuário eletrônico, permitindo histórico médico mais completo.*
- *Implementação de notificações em tempo real via WebSockets, melhorando a comunicação entre administradores e cuidadores.*
- *Implantação de containers Docker e CI/CD, para facilitar o deploy e a manutenção em ambientes produtivos.*

Esses aprimoramentos visam tornar o sistema ainda mais robusto, completo e escalável, consolidando o AssisConnect como uma ferramenta essencial para a gestão de instituições de acolhimento de idosos.

7.2. LIÇÕES APRENDIDAS

8. O desenvolvimento do **AssisConnect 2.0** proporcionou uma experiência prática de integração entre tecnologias modernas de frontend e backend, reforçando conceitos de engenharia de software, modelagem de banco de dados e controle de versão colaborativo.
9. O principal desafio técnico enfrentado foi a implementação da **autenticação JWT** e do **controle de permissões por perfis de usuário**, exigindo um entendimento aprofundado das camadas de segurança do Spring Boot e da manipulação de tokens no React.
10. Além disso, o trabalho em equipe evidenciou a importância da **comunicação constante**, da **organização das tarefas com metodologias ágeis (Kanban)** e do **uso eficiente de ferramentas colaborativas**.
11. Como resultado, a equipe consolidou competências em desenvolvimento full stack, boas práticas de arquitetura, versionamento e implantação, compreendendo a relevância de planejar cuidadosamente o ciclo de vida do software desde a concepção até a entrega final.