



# Chronos

Sistema de Agendamento  
para Barbearias

**Integrantes:**

Daniel Navarro Porto  
Lucas Silva de Oliveira

**Orientador:**

Prof. Me. Robson Ferreira Lopes

# Conteúdo da Apresentação

---

**01**

Motivações

**02**

Objetivos

**03**

Visão Geral do Sistema

**04**

Modelo de Caso de Uso

**05**

Modelo de DER e de Classe

**06**

Tecnologias Utilizadas

**07**

Diagrama de Sequência

**08**

Dificuldades

**09**

Cronograma

**10**

Conclusões Parciais

# Motivações

---

01

# Motivações de Aprendizado

---

## Aprofundamento Técnico

O projeto Chronos serviu como um laboratório prático para aplicar e consolidar conhecimentos de diversas disciplinas cruciais para a formação em desenvolvimento de software.

## Integração de Conhecimento

Proporcionou uma visão do ciclo de vida de um projeto real, desde a sua concepção até o deploy, unindo os conceitos teóricos e práticos.

## Desafios

Permitir explorar e solucionar problemas complexos em um ambiente simulado de produção.

# Abrangência de Disciplinas

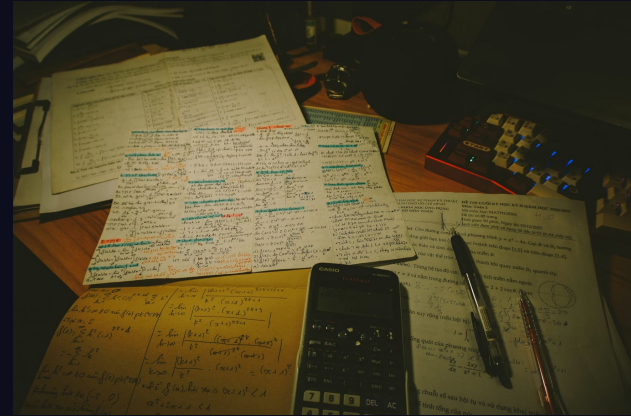
Desenvolvimento & Arquitetura	Engenharia de Dados	Infraestrutura & Gestão
Análise e Programação Orientada a Objetos	Engenharia de Software	Segurança da Informação
Desenvolvimento com Frameworks	Banco de Dados	DevOps
APIs e Microsserviços	Interação Humano-Computador (UI / UX)	Gestão de Projetos
Desenvolvimento Front-end		

# Motivações de Mercado

---

## Problema

A maioria das barbearias de pequeno porte utiliza métodos de gestão manuais (cadernos, ligações, WhatsApp)



## Ineficiência Operacional

Isso gera erros, duplicidade de horários, perda de dados e gargalos de atendimento, impactando a produtividade e a satisfação do cliente



# Vamos mostrar alguns dados

---

## Nossa solução

Uma aplicação web acessível para modernizar a gestão e otimizar o tempo do cliente e do profissional

### Conveniência e Eficiência

64%

dos brasileiros optam por interações online para maior praticidade e economia de tempo

### Oportunidade de Mercado

66%

das micro e pequenas empresas estão em níveis iniciais de maturidade digital

# Objetivos

---

02



# Principais Objetivos

---

Desenvolver um protótipo funcional de aplicação web que automatize e otimize o processo de agendamento das barbearias, solucionando o problema da gestão manual



## Banco de Dados

Mitigar erros de agendamentos através de um banco de dados relacional robusto



## Back-end

Construir uma API RESTful para centralizar toda a lógica de negócio



## Segurança

Implementar autenticação e autorização baseada em perfis (Administrador, Profissional e Cliente)



## Front-end

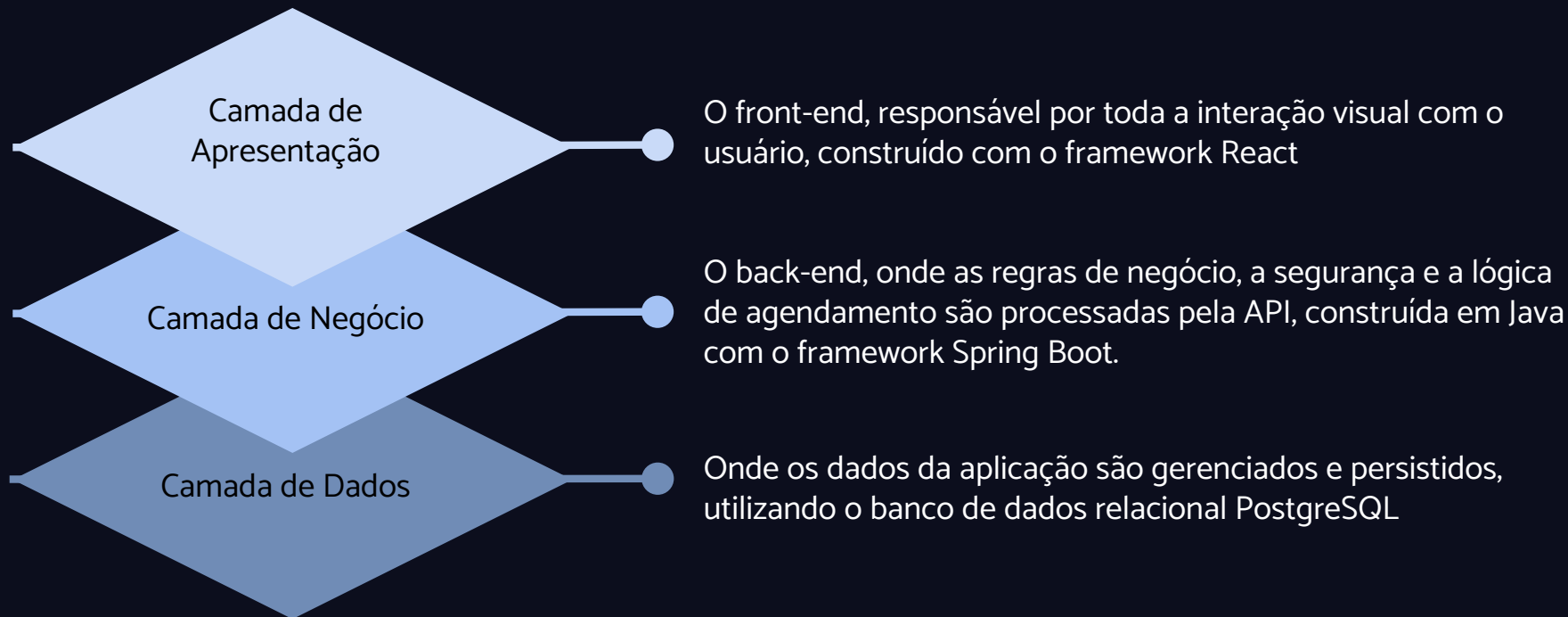
Construir uma Interface de usuário intuitiva (SPA) focada na usabilidade

# Visão Geral do Sistema

---

03

## ESTRUTURA EM TRÊS CAMADAS



# Benefícios da Arquitetura

---

## **Separação de Responsabilidade**

Cada camada é responsável por uma função específica (UI, Negócios, Dados)

## **Flexibilidade e Escalabilidade**

Permite escalar ou atualizar uma camada independente das outras.

## **Segurança**

### **Aprimorada**

A camada de dados é protegida da interface do usuário tendo a camada de negócio como intermediária

## **Testabilidade**

### **Simplificada**

As camadas pode ser testadas isoladamente

## **Reutilização de**

### **Código**

A lógica de negócio pode ser reutilizada por diferentes interfaces (Web, Mobiles, APIs)

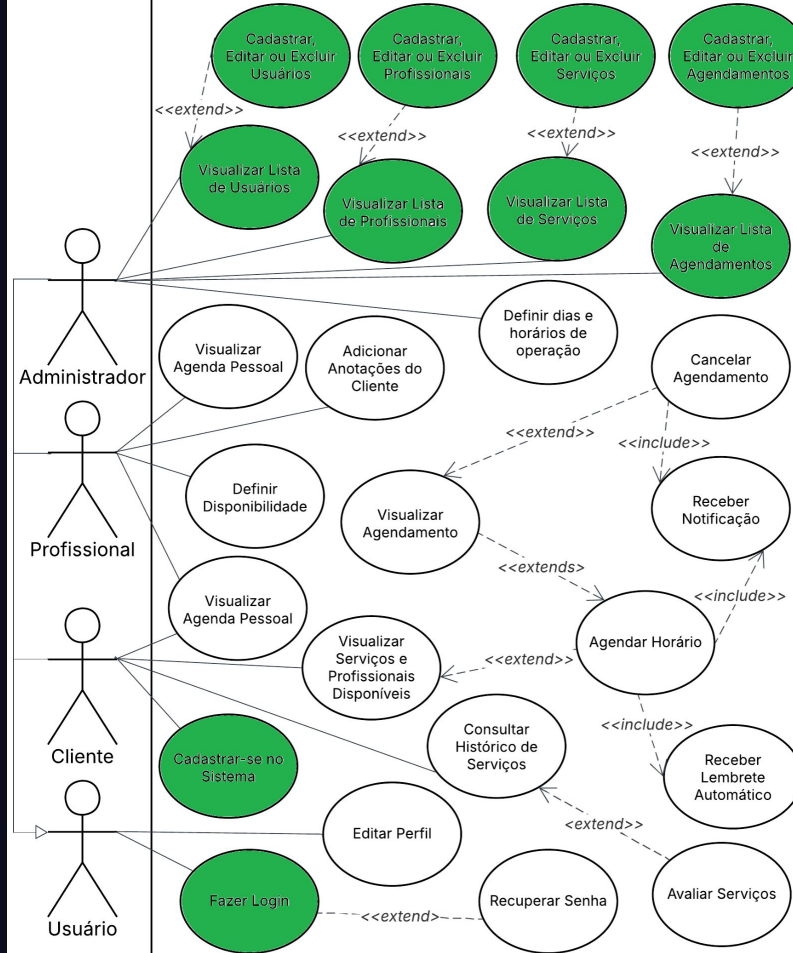
# Modelo de Caso de Uso

---

04



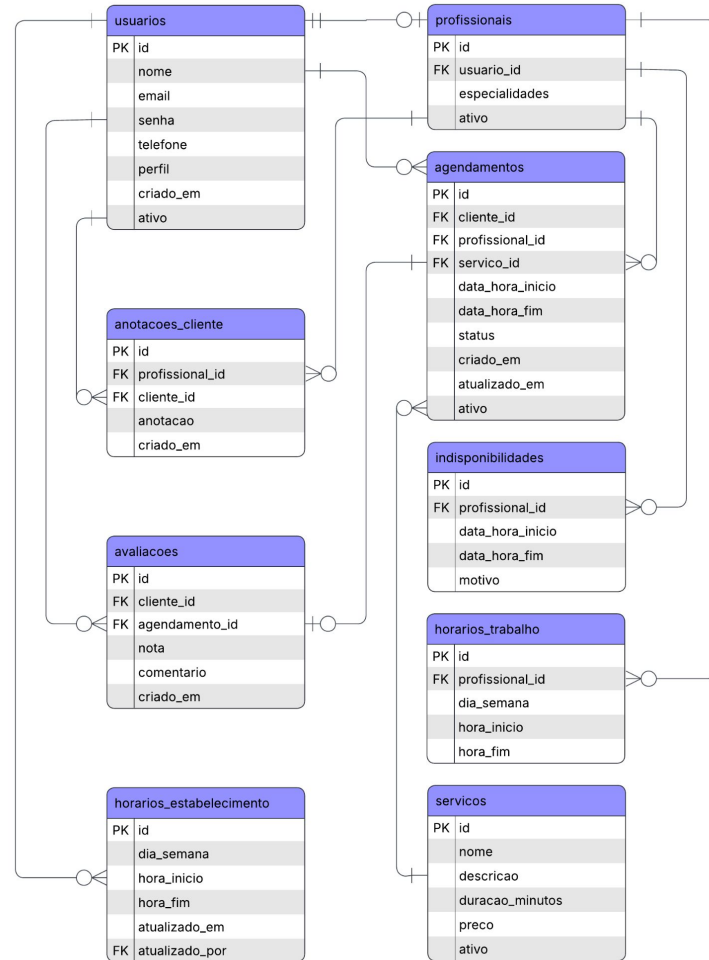
## Chronos - Sistema de Agendamentos para Barbearia



# Modelo DER e de Classe

---

05





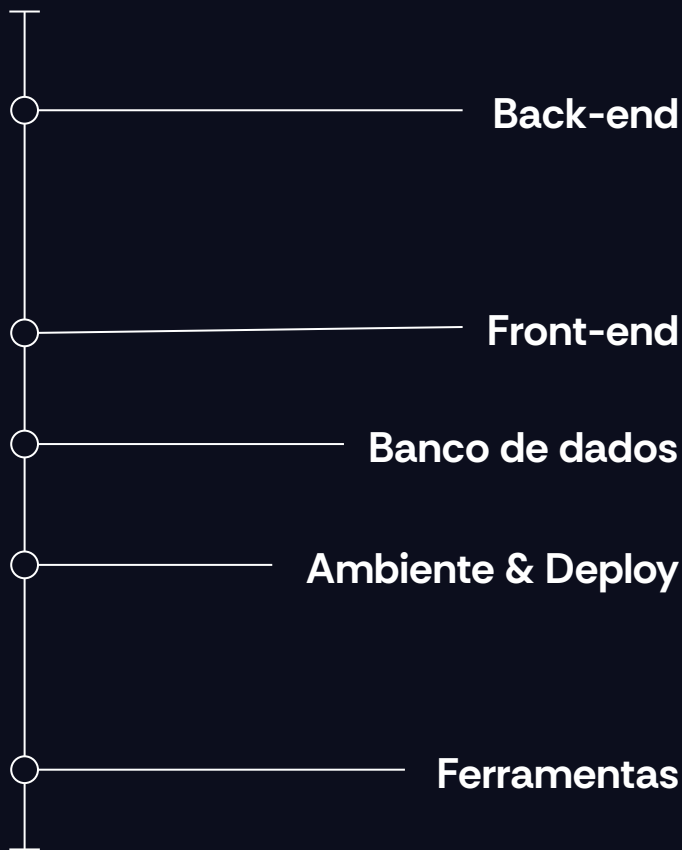
# Tecnologías Utilizadas

---

06



# Tecnologías Utilizadas



- Java 21
- Spring Boot 3
- Spring Data JPA (Hibernate)
- Spring Security (JWT)
- Maven

- React (Vite + SWC)
- React Router DOM
- Axios

- PostgreSQL 18

- Docker / Docker Compose
- Render (Hospedagem Back-end / DB)
- Vercel (Hospedagem Front-end)

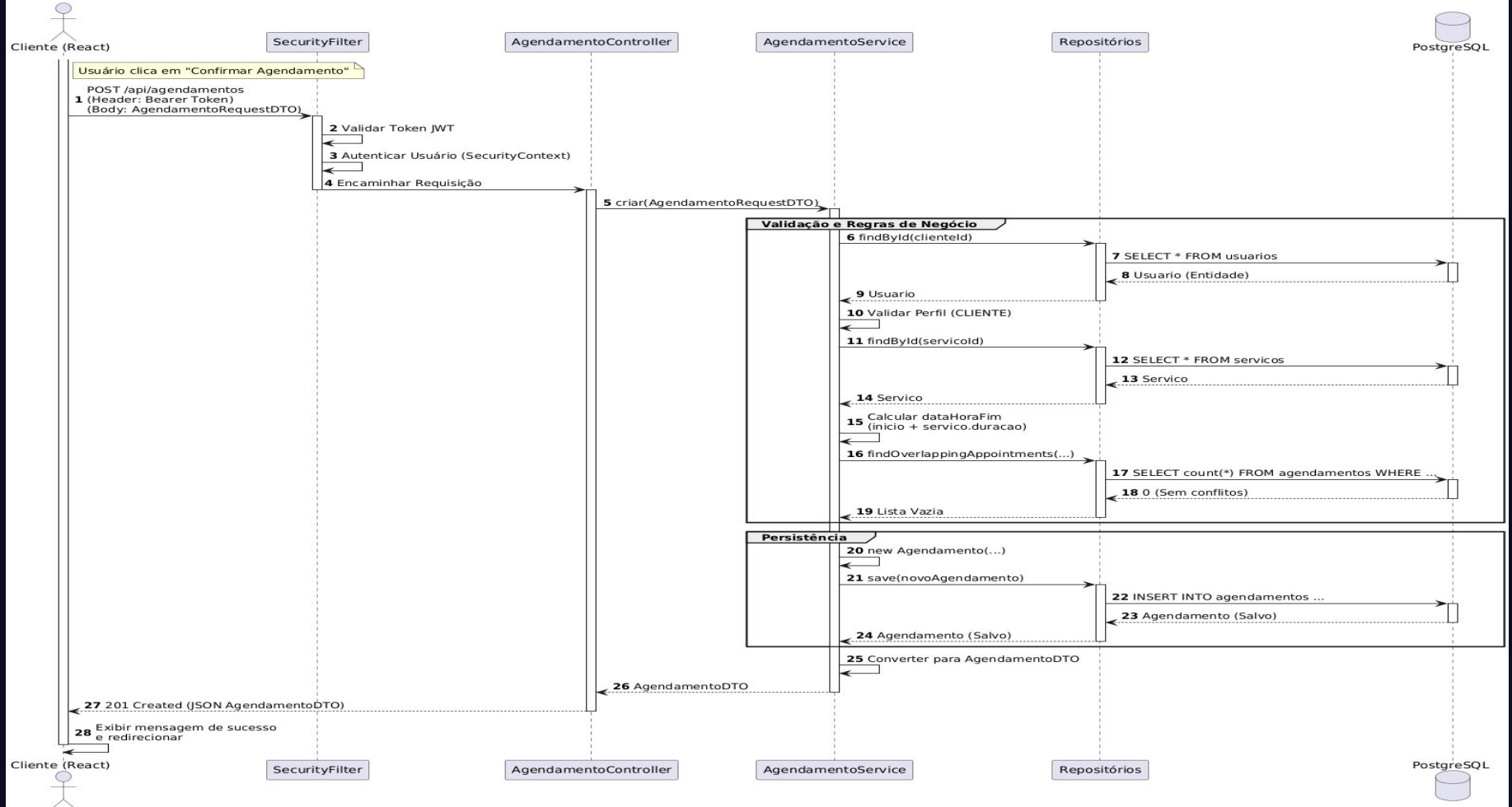
- Git / Github
- Postman
- Dbeaver
- VS Code / Eclipse

# Diagrama de Sequência

---

07

Diagrama de Sequência: Realizar Agendamento (RF10)



# Dificuldades

---

08

## Curva de Aprendizado e Novas Tecnologias

### Dificuldades:

- A stack tecnológica definida (Java 21, Spring Boot 3, React) constituía de novos conhecimentos para a equipe, exigindo um esforço significativo de adaptação.

### Soluções:

- Realização de extensa pesquisa técnica e estudo autodidata para dominar as ferramentas e aplicar as melhores práticas de desenvolvimento.



## Gerenciamento de Ambiente e Compatibilidade

### Dificuldade:

- Conflitos de quebras de linhas (LF vs. CRLF) entre sistemas.
- Variações do encoder dos textos,
- Variações de versões do Node.js na equipe.

### Soluções:

- Implementação do *.gitattributes* para padronização de quebra de linhas
- Implementação do *.editorconfig* para padronização nos editores de texto
- Implementação do *.nvmrc* (para gerenciamento de versão do Node.js) para garantir consistência.

## Serialização de Dados e Segurança da API

### Dificuldade:

- **Lazy Loading:** Exceções `LazyInitializationException` ao desativar `spring.jpa.open-in-view=true` para otimização, devido ao fechamento da sessão do Hibernate antes da serialização.
- **Exposição de Dados:** Retorno de entidades JPA diretamente na API, expondo dados sensíveis e acoplando a API à estrutura do banco.

### Soluções:

- Refatoração da API para utilizar DTOs na entrada e saída de dados. A camada de *Service* agora busca as entidades, acessa dados LAZY dentro do contexto transacional e os converte em DTOs limpos, resolvendo ambos os problemas de serialização e segurança.

## Conectividade entre Front-end e Back-end

### Dificuldade:

- **Servidor:** Requisições do *front-end* (Vercel) eram bloqueadas pelo *back-end* (Render), devido a política de mesma origem.
- **Local:** Requisições retornando erro “403 Forbidden”.

### Soluções:

- **Servidor:** Configuração explícita da política de CORS no Spring Boot (WebConfig) para permitir requisições de origem específicas, e permissão para requisições OPTIONS no SecurityConfig (para *preflight* requests).
- **Local:** Reordenação e refinamento das regras de autorização no SecurityConfig para garantir que os privilégios de cada perfil estejam corretamente mapeadas.

# Cronograma

---

09

Descrição das Atividades	Código	Atribuição	Meses									
			SET	OUT	NOV	DEZ	JAN	FEV	MAR	ABR	MAI	JUN
<b>Fase 1 - Planejamento</b>	<b>1</b>											
Levantamento e Refinamento de Requisitos	1a	Ambos	X	X								
Definição da Arquitetura, Tecnologias e Escopo do MVP	1b	Ambos		X								
Elaboração do Capítulo 1 (Introdução) do relatório	1c	Ambos		X								
<b>Fase 2 - Modelagem e Documentação</b>	<b>2</b>											
Elaboração dos Diagramas UML (Casos de Uso, Classes, DER)	2a	Daniel		X								
Elaboração do Capítulo 2 (Metodologia) do relatório	2b	Lucas		X								
Elaboração do Capítulo 4 e 5 (Conclusões / Cronograma)	2c	Ambos		X								
<b>Fase 3 - Desenvolvimento MVP - Core</b>	<b>3</b>											
Configuração do Ambiente de Desenvolvimento e Repositório Git	3a	Lucas			X							
Configurar Projeto Spring Boot, BD PostgreSQL e Entidades JPA	3b	Lucas			X							
Configurar Projeto React, Roteamento e Componentes de UI	3c	Daniel			X							
Back-end: Implementar Segurança (Spring Security - Cadastro/Login)	3d	Lucas			X							
Back-end: Implementar CRUD Usuários (Serviços, Endpoints)	3e	Lucas			X							
Front-end: Telas de Login/Cadastro e Integração com API	3f	Daniel			X							

Atrasado
Finalizado
Pendente (Ambos)
Pendente (Daniel)
Pendente (Lucas)



Descrição das Atividades	Código	Atribuição	Meses									
			SET	OUT	NOV	DEZ	JAN	FEV	MAR	ABR	MAI	JUN
<b>Fase 4 - Desenvolvimento MVP - Funcional</b>	<b>4</b>											
Back-end: CRUD Serviços e Profissionais	4a	Lucas			X							
Back-end: Lógica de Agendamento (API - Básico)	4b	Lucas			X							
Front-end: Tela de Agendamento (Cliente) e Integração com API	4c	Daniel			X							
Front-end: Telas Admin (Gerenciar Serviços/Profissionais)	4d	Daniel				X	X	X				
<b>Fase 5 - Desenvolvimento Full Scope - Barbeiro</b>	<b>5</b>											
Back-end: Endpoints de Agenda Pessoal e Bloqueio de Horários	5a	Lucas						X	X			
Front-end: Telas do Perfil Barbeiro (Agenda, Bloqueios)	5b	Daniel						X	X			
<b>Fase 6 - Desenvolvimento Full Scope - Cliente</b>	<b>6</b>											
Back-end: Endpoints (Histórico, Avaliações, Anotações)	6a	Lucas							X	X		
Front-end: Telas (Histórico, Avaliação)	6b	Daniel							X	X		
Front-end: Telas (Recuperar Senha, Editar Perfil)	6c	Daniel							X	X		

Atrasado

Finalizado

Pendente (Ambos)

Pendente (Daniel)

Pendente (Lucas)

Descrição das Atividades	Código	Atribuição	Meses									
			SET	OUT	NOV	DEZ	JAN	FEV	MAR	ABR	MAI	JUN
<b>Fase 7 - Desenvolvimento Full Scope - Admin/Notif.</b>	<b>7</b>											
Back-end: Implementar Serviço de Notificações/Lembretes	7a	Lucas								X	X	
Front-end: Tela Admin (Gerenciar Agenda Geral)	7b	Daniel								X	X	
<b>Fase 8 - Testes e Refinamento</b>	<b>8</b>											
Testes de Integração (Todos os fluxos e perfis)	8a	Ambos									X	
Correção de Bugs e refinamento de Usabilidades	8b	Ambos									X	
<b>Fase 9 - Entrega Final</b>	<b>9</b>											
Finalização da Redação do Relatório Técnico Completo	9a	Ambos										X
Preparação da Apresentação Final e Deploy da Aplicação	9b	Ambos										X

- Atrasado
- Finalizado
- Pendente (Ambos)
- Pendente (Daniel)
- Pendente (Lucas)

# Conclusões Parciais

---

10

# Conclusões Parciais

---

## Planejamento e Modelagem

- As fases iniciais de planejamentos foram concluídas sem grandes problemas.
- Definimos 20 Requisitos Funcionais e 5 Requisitos Não Funcionais.
- Conforme o avanço do projeto, novos requisitos podem vir a ser inclusos.

## Prototipo MVP Funcional Desenvolvido

- O desenvolvimento do MVP Core (Backend) foi finalizado.
- CRUDs completos (Usuário, Serviço, Profissional)
- Sistema de Autenticação JWT e autorização de perfil
- Lógica principal de agendamento com detecção de conflitos
- Páginas para testes dos CRUDs

# Conclusões Parciais

---

## Arquitetura Validada

- A escolha da arquitetura desacoplada (SPA com React + API Java) provou-se eficiente, permitindo o desenvolvimento independente do front-end e back-end com integração fluída.

## Infraestrutura

- Ambiente de produção automatizado (CI/CD). O sistema já está hospedado e pronto para receber as próximas funcionalidades.
- Avaliação de migração de hospedagem para o Backend e Banco de Dados, devido às limitações do plano gratuito.



# Obrigado!

---

Alguma pergunta?

A aplicação pode ser acessada em:

*[chronos-agendamentos.vercel.app](https://chronos-agendamentos.vercel.app)*

**CREDITS:** This presentation template was created by **Slidesgo**,  
and includes icons, infographics & images by **Freepik**



## Chronos

Sistema de Agendamento  
para Barbearias