



**UNINASSAU**

**CENTRO ACADÊMICO MAURÍCIO DE NASSAU**

**CURSO DE CIÊNCIA DA COMPUTAÇÃO**

**BIBLIOTECA API**

**André Luis Cavalcanti  
Rafael Hilario Dias Barbosa  
Lucas José Leite Marinho  
Daniel Lins Aretakis  
Wenny Santana de Andrade**

**Dezembro – 2024  
RECIFE – PE**

# SUMÁRIO

<b>INTRODUÇÃO .....</b>	<b>2</b>
<b>ESTRUTURA ANALÍTICA .....</b>	<b>3</b>
<b>ATRIBUIÇÃO DE TAREFAS .....</b>	<b>4</b>
<b>PLANILHA DE CUSTOS .....</b>	<b>5</b>
<b>CRONOGRAMA GANTT .....</b>	<b>6</b>
<b>RESUMO TRELLO .....</b>	<b>7</b>
<b>FLUXO NO REPOSITÓRIO .....</b>	<b>8</b>
<b>CASOS DE USO .....</b>	<b>9</b>
<b>PROTÓTIPOS .....</b>	<b>10</b>
<b>EXTRATOS DO CÓDIGO .....</b>	<b>11</b>
<b>DEMONSTRAÇÃO DE USO .....</b>	<b>12</b>
<b>LIÇÕES APRENDIDAS .....</b>	<b>13</b>
<b>AGRADECIMENTOS .....</b>	<b>14</b>
<b>REFERÊNCIAS .....</b>	<b>15</b>

# INTRODUÇÃO

Este projeto tem como objetivo fornecer uma solução completa para o gerenciamento de uma biblioteca, atendendo a requisitos funcionais essenciais. O sistema permite o controle eficiente de livros, usuários e empréstimos, integrando as seguintes funcionalidades:

O gerenciamento de livros é realizado por meio de um CRUD, que permite cadastrar, visualizar, editar e excluir informações, com campos como título, autor, gênero e ano de publicação. Também inclui o gerenciamento de usuários, possibilitando a criação e manutenção de dados como nome, endereço, e-mail e telefone.

Além disso, o sistema oferece controle completo de empréstimos e devoluções de livros, com a limitação do número máximo de empréstimos por usuário, gestão de prazos de devolução e sinalização de pendências. Para aprimorar o acompanhamento, são gerados relatórios básicos, como a listagem dos livros mais emprestados e dos usuários com devoluções pendentes.

O desenvolvimento utilizou tecnologias modernas e robustas. A lógica e funcionalidade do sistema foram implementadas em **JavaScript**, enquanto a interface de usuário foi desenvolvida com **React**, garantindo uma experiência dinâmica e interativa. O armazenamento de dados e a autenticação de usuários foram integrados ao **Firebase**, proporcionando segurança e escalabilidade.

Combinando praticidade e inovação, este projeto se destaca como uma ferramenta eficiente para bibliotecas, facilitando o gerenciamento diário e a tomada de decisões.

# ESTRUTURA ANALÍTICA



## ATRIBUIÇÃO DE TAREFAS

### **Tarefas do Scrum Master - Lucas José Leite Marinho:**

O projeto foi organizado com uma estrutura analítica clara, apresentada em uma página A4. O cronograma foi detalhado em um Gráfico de Gantt para acompanhar as atividades planejadas. A gestão de tarefas foi realizada por meio do Trello, utilizando colunas específicas (TODO, PLANNED, DOING, etc.) e integração ao Google Calendar para monitorar prazos e responsabilidades. Uma planilha de custos foi elaborada e exportada em PDF, e todo o material final foi consolidado em uma apresentação em PDF, garantindo uma execução eficiente e uma entrega bem estruturada.

### **Tarefas do Gerente De Configuração – Wenny Santana:**

O Gerente de Configuração criou o repositório *LibraryApi* no GitHub, com arquivos base como `.gitignore`, `LICENSE`, `README.md` (com imagens e seções), e `CONTRIBUTING.md`. A estrutura de pastas foi organizada em `libraryapi`, `librarydocs` (documentação) e `libraryweb`, usando `.gitkeep` quando necessário. O fluxo de trabalho foi definido com branches para desenvolvimento e merges para a branch principal, documentado em um fluxograma no `README.md`. Tutoriais sobre Git e ferramentas foram exportados em PDF e referenciados nos arquivos de documentação, garantindo um gerenciamento eficiente e colaborativo.

### **Tarefas do Documentador – André Luis Cavalcanti:**

O Documentador elaborou casos de uso em formato de tabela, com seções como ID, título, ator principal, objetivos e fluxos, acompanhados de diagramas simples e exportados em PDF. Diagramas UML e, opcionalmente, diagramas ER foram criados e exportados em PNG ou PDF, sendo armazenados no repositório Git para organização do projeto.

### **Tarefas Desenvolvedores – Rafael Hilario Dias Barbosa / Daniel Lins Aretakis:**

O desenvolvedor é responsável por codificar a solução do projeto, transformando requisitos em código funcional e eficiente. Isso inclui implementar features, lógica de negócios, integrar componentes e corrigir bugs, garantindo código limpo e de qualidade. Também realiza testes para verificar a funcionalidade e colabora com a equipe para alinhar a implementação aos objetivos do projeto.

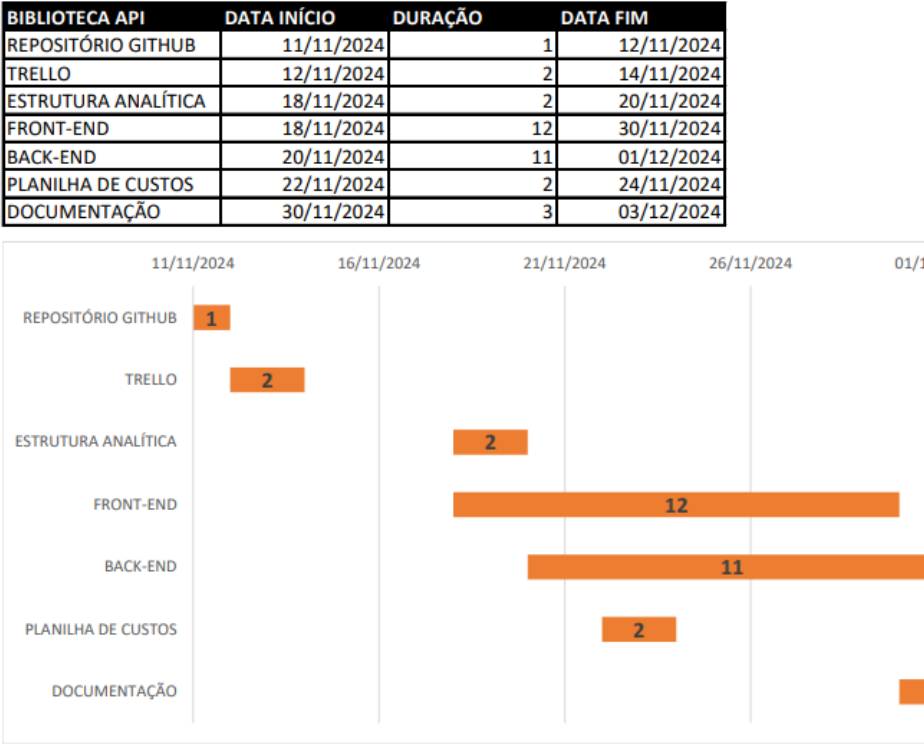
## PLANILHA DE CUSTOS

### PLANILHA DE CUSTOS

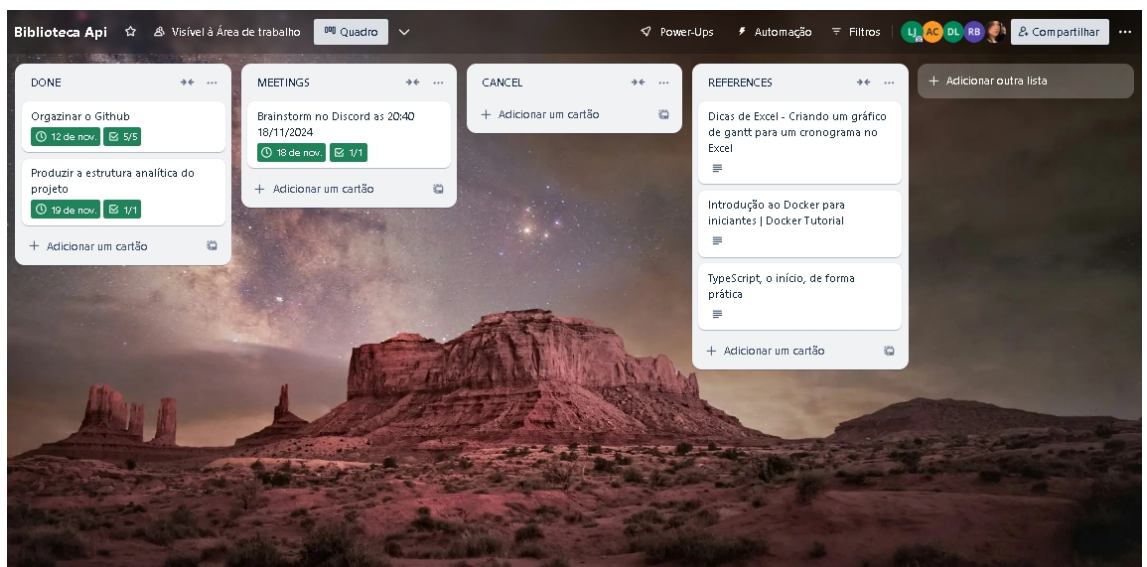
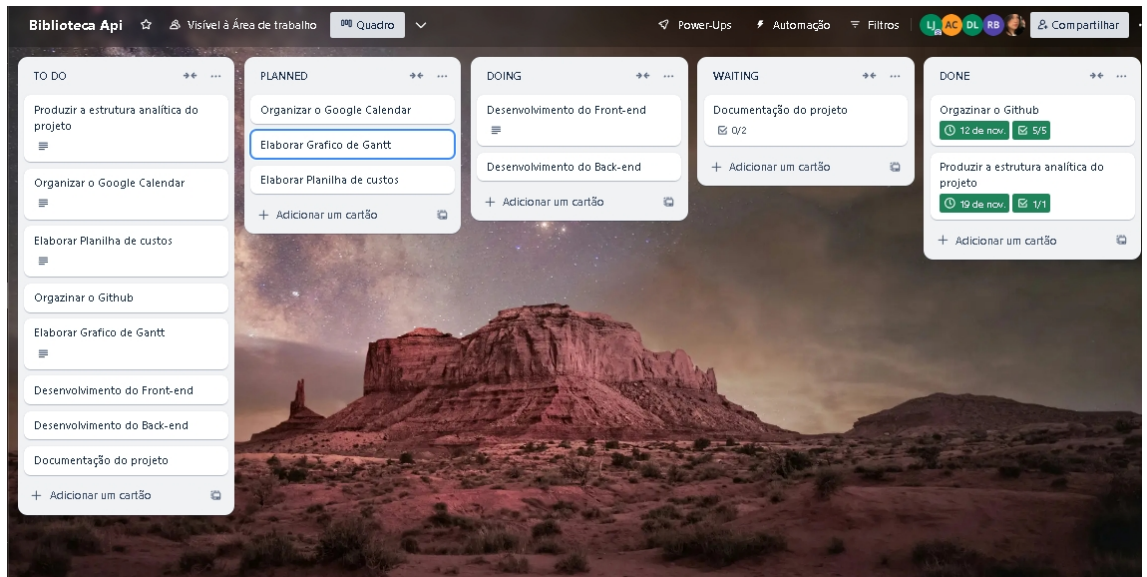
Integrantes	Desktop	Periféricos	Internet	Total
Lucas José	R\$ 2.000,00	R\$ 850,00	R\$ 100,00	R\$ 2.950,00
Daniel Lins	R\$ 2.200,00	R\$ 1.200,00	R\$ 100,00	R\$ 3.500,00
Wenny Santana	R\$ 3.300,00	R\$ 1.150,00	R\$ 100,00	R\$ 4.550,00
Rafael Hilario	R\$ 2.500,00	R\$ 900,00	R\$ 100,00	R\$ 3.500,00
André Luis	R\$ 3.000,00	R\$ 950,00	R\$ 100,00	R\$ 4.050,00

<b>Total Geral</b>	<b>R\$ 18.550,00</b>
--------------------	----------------------

# GRÁFICO DE GANTT

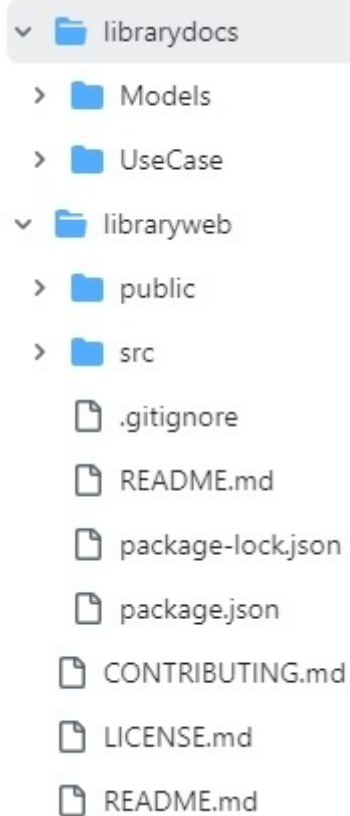


# TRELLO





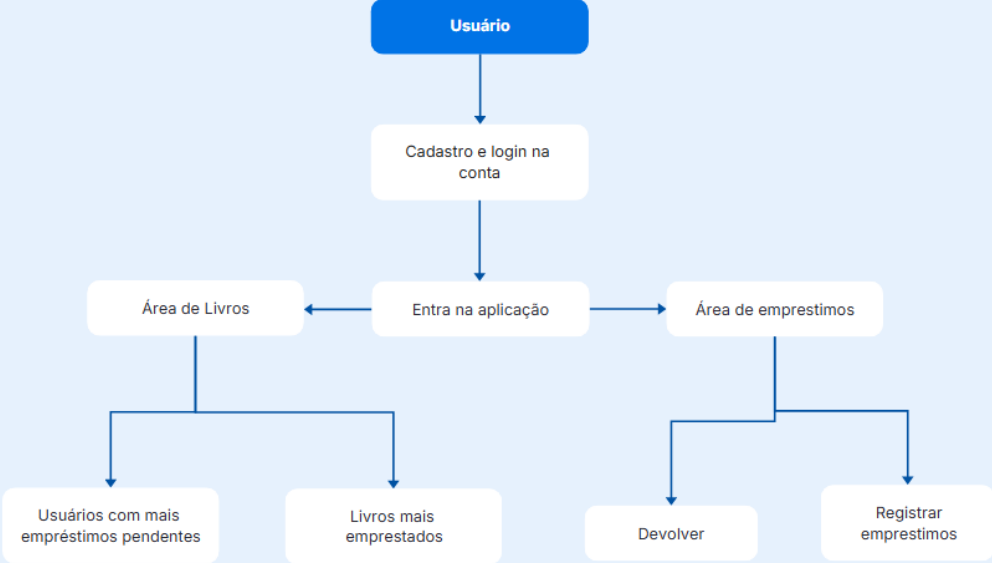
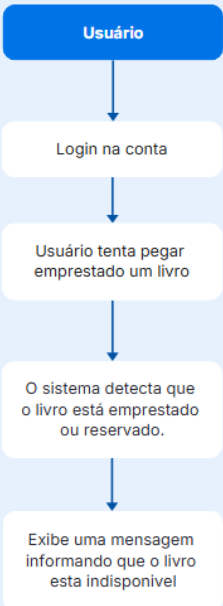
## FLUXO NO REPOSITÓRIO



## CASOS DE USO

ID:	UC001
-----	-------

Título:	gerenciar livros em uma biblioteca
Ator principal:	Usuário
Objetivo:	Permitir o gerenciamento eficiente de uma biblioteca, oferecendo funcionalidades para cadastro e consulta de livros, controle de usuários (leitores) e gestão de empréstimos e devoluções.
Pré-condição:	O usuário deve ter um cadastro prévio no sistema. O usuário deve estar autenticado no sistema.
Pós-condições:	As informações de livros, usuários ou empréstimos foram atualizadas no sistema, de acordo com a ação realizada. O sistema atualiza e mantém todas as operações realizadas de forma consistente. O usuário está autenticado no sistema e pode acessar as funções disponíveis para seu perfil.

Fluxo principal:	<div data-bbox="475 203 1536 994"> <h3>Fluxo Principal</h3>  <pre> graph TD     U[Usuário] --&gt; CL[Cadastro e login na conta]     CL --&gt; EA[Entra na aplicação]     EA --&gt; AL[Área de Livros]     EA --&gt; AE[Área de empréstimos]     AL --&gt; UMP[Usuários com mais empréstimos pendentes]     AL --&gt; LME[Livros mais emprestados]     AE --&gt; D[Devolver]     AE --&gt; RE[Registrar empréstimos] </pre> </div>
Fluxo alternativo:	<div data-bbox="475 1088 1536 1890"> <h3>Fluxo Alternativo</h3>  <pre> graph TD     U[Usuário] --&gt; L[Login na conta]     L --&gt; U1[Usuário tenta pegar emprestado um livro]     U1 --&gt; S[O sistema detecta que o livro está emprestado ou reservado.]     S --&gt; M[Exibe uma mensagem informando que o livro está indisponível] </pre> </div>
Tratamento de exceções:	<p><b>Causa:</b> Falha ao conectar-se ao banco de dados durante o registro do empréstimo.</p> <p><b>Tratamento:</b> O sistema deve exibir uma mensagem de erro genérica</p>

	<p><b>Causa:</b> O usuário tenta solicitar o empréstimo de um livro inexistente (por exemplo, ID inválido).</p> <p>Tratamento: O sistema retorna um código de erro 404 (Not Found) com a mensagem: <i>"Livro não encontrado."</i></p> <p><b>Causa:</b> O usuário já atingiu o número máximo permitido de livros emprestados.</p> <p><b>Tratamento:</b> O sistema retorna um código de erro 403 (Forbidden) com a mensagem: <i>"Limite de empréstimos atingido. Devolva livros antes de solicitar novos."</i></p>

## PROTÓTIPOS

<p>+ Iniciar coleção</p> <p>livros &gt;</p>	<p>+ Adicionar documento</p> <p>Não há documentos nesta coleção</p>	<p>+ Iniciar coleção</p> <p>+ Adicionar campo</p> <p>Este documento não existe e não vai aparecer em consultas ou instantâneos. <a href="#">Saiba mais</a></p>
---	---	--

+ Iniciar coleção	+ Adicionar documento	+ Iniciar coleção
livros >	Ig1v8yriJ3qDjdZurMnD >	+ Adicionar campo
		anoPublicacao: "" autor: "sdf" genero: "" titulo: "sdff"

## EXTRATOS DO CÓDIGO

### CSS DA HOME DO PROJETO:

```

1  @import url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap');
2
3  body {
4    font-family: 'Poppins', sans-serif;
5    background: linear-gradient(135deg, #f47fa, #c3cfe2);
6    margin: 0;
7    padding: 0;
8    box-sizing: border-box;
9  }
10
11  .home-container {
12    display: flex;
13    flex-direction: column;
14    align-items: center;
15    justify-content: center;
16    min-height: 100vh;
17    padding: 20px;
18    background: linear-gradient(135deg, #f47fa, #c3cfe2);
19  }
20
21  h1 {
22    font-size: 2.5rem;
23    color: #5bcbde;
24    margin-bottom: 20px;
25    font-weight: 600;
26    text-align: center;
27  }
28
29  p {
30    font-size: 1.2rem;
31    color: #6c757d;
32    margin-bottom: 30px;
33    text-align: center;
34  }
35
36  .links-container {
37    display: grid;
38    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
39    gap: 15px;
40    width: 80%;
41    max-width: 500px;
42  }
43
44  .home-link {
45    display: flex;
46    align-items: center;
47    justify-content: center;
48    padding: 15px;
49  }

```

**FUNÇÕES DO USUÁRIO :**

```

23
24 // Buscar usuários
25 const fetchUsuarios = async () => {
26   const data = await getDocs(usuariosRef);
27   setUsuarios(data.docs.map((doc) => ({ ...doc.data(), id: doc.id })));
28 };
29
30 // Adicionar usuário
31 const adicionarUsuario = async () => {
32   if (!novoUsuario.nome || !novoUsuario.email) {
33     alert("Nome e e-mail são obrigatórios!");
34     return;
35   }
36   await addDoc(usuariosRef, novoUsuario);
37   fetchUsuarios();
38   setNovoUsuario({ nome: "", endereco: "", email: "", telefone: "" });
39 };
40
41 // Excluir usuário
42 const deletarUsuario = async (id) => {
43   await deleteDoc(doc(db, "usuarios", id));
44   fetchUsuarios();
45 };
46
47 // Modo de edição
48 const iniciarEdicao = (usuario) => {
49   setEditando(usuario.id);
50   setUsuarioEditado(usuario);
51 };
52
53 // Salvar alterações
54 const salvarEdicao = async (id) => {
55   const usuarioRef = doc(db, "usuarios", id);
56   await updateDoc(usuarioRef, usuarioEditado);
57   fetchUsuarios();
58   setEditando(null);
59   setUsuarioEditado({ nome: "", endereco: "", email: "", telefone: "" });
60 };
61
62 useEffect(() => {
63   fetchUsuarios();
64 }, []);
65
66 return (
67   <div className="usuarios-container">

```

**FUNÇÕES DO USUÁRIO :**

```

13
14
15 useEffect(() => {
16   const fetchData = async () => {
17     try {
18       // Buscar empréstimos
19       const empréstimosSnapshot = await getDocs(collection(db, "emprestimos"));
20       const empréstimosData = empréstimosSnapshot.docs.map((doc) => {
21         const data = doc.data();
22         return {
23           id: doc.id,
24           ...data,
25           dataEmprestimo: new Date(data.dataEmprestimo),
26           dataDevolucao: new Date(data.dataDevolucao),
27         };
28       });
29
30       // Buscar usuários
31       const usuariosSnapshot = await getDocs(collection(db, "usuarios"));
32       const usuariosMap = usuariosSnapshot.docs.reduce((map, doc) => {
33         const data = doc.data();
34         map[doc.id] = data.nome; // Presumindo que o campo "nome" contém o nome do usuário
35         return map;
36       }, {});
37
38       // Buscar livros
39       const livrosSnapshot = await getDocs(collection(db, "livros"));
40       const livrosMap = livrosSnapshot.docs.reduce((map, doc) => {
41         const data = doc.data();
42         map[doc.id] = data.titulo; // Presumindo que o campo "titulo" contém o título do livro
43         return map;
44       }, {});
45
46       setEmprestimos(emprestimosData);
47       setUsuariosMap(usuariosMap);
48       setLivrosMap(livrosMap);
49     } catch (error) {
50       console.error("Erro ao carregar dados:", error);
51     }
52   }
53   fetchData();
54 }, []);
55

```

## FUNÇÕES DO LIVRO :



```

22  const adicionarLivro = async () => {
23    if (!novolivro.titulo || !novolivro.autor) {
24      alert("Por favor, preencha os campos obrigatórios!");
25      return;
26    }
27    await addDoc(livrosRef, novolivro);
28    fetchLivros();
29    setNovoLivro({ titulo: "", autor: "", genero: "", anoPublicacao: "" });
30  };
31
32  const deletarLivro = async (id) => {
33    await deleteDoc(doc(db, "livros", id));
34    fetchLivros();
35  };
36
37  useEffect(() => {
38    fetchLivros();
39  }, [fetchLivros]);
40
41  return (
42    <div className="livros-container">
43      <h2>Controle de Livros</h2>
44
45      <form
46        className="add-livro-form"
47        onSubmit={(e) => {
48          e.preventDefault();
49          adicionarLivro();
50        }}
51      >
52        <input
53          type="text"
54          placeholder="Título *"
55          value={novolivro.titulo}
56          onChange={(e) =>
57            setNovoLivro({ ...novolivro, titulo: e.target.value })
58          }
59        />
60        <input
61          type="text"
62          placeholder="Autor *"
63          value={novolivro.autor}

```

**FUNÇÕES DO EMPRESTIMO :**

```

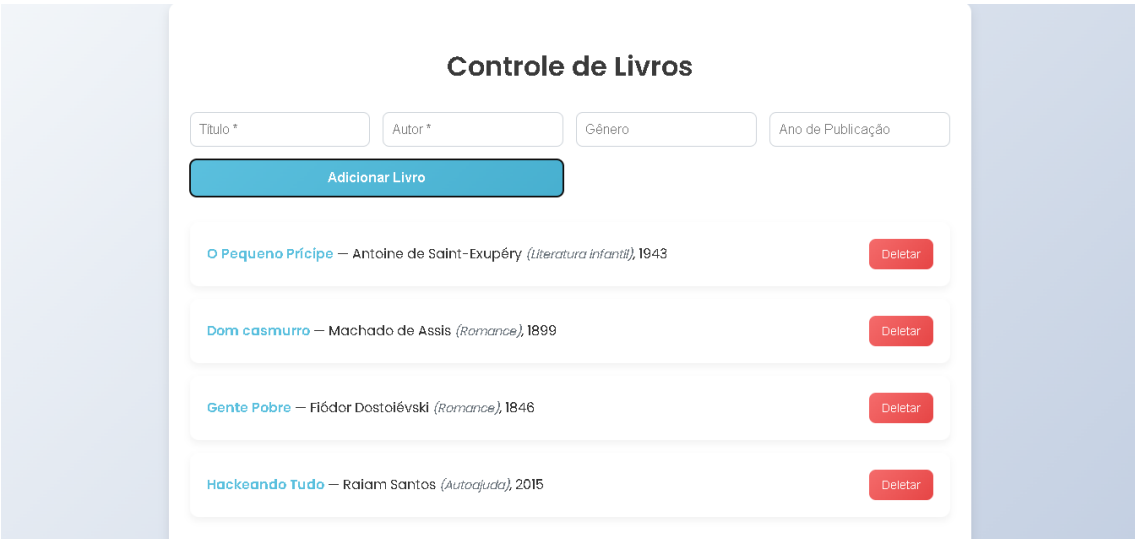
13  const Emprestimos = () => {
14    const [emprestimos, setEmprestimos] = useState([]);
15    const [usuarios, setUsuarios] = useState([]);
16    const [livros, setLivros] = useState([]);
17    const [novoEmprestimo, setNovoEmprestimo] = useState({
18      usuarioId: "",
19      livroId: "",
20      dataEmprestimo: "",
21      dataDevolucao: "",
22    });
23    const [loading, setLoading] = useState(false);
24
25    useEffect(() => {
26      const fetchData = async () => {
27        setLoading(true);
28        try {
29          const emprestimosSnapshot = await getDocs(collection(db, "emprestimos"));
30          const usuariosSnapshot = await getDocs(collection(db, "usuarios"));
31          const livrosSnapshot = await getDocs(collection(db, "livros"));
32
33          setEmprestimos(
34            emprestimosSnapshot.docs.map((doc) => {
35              const data = doc.data();
36              return {
37                id: doc.id,
38                ...data,
39                dataDevolucao: data.dataDevolucao instanceof Timestamp
40                  ? data.dataDevolucao.toDate()
41                  : new Date(data.dataDevolucao),
42              };
43            })
44          );
45
46          setUsuarios(
47            usuariosSnapshot.docs.map((doc) => ({ id: doc.id, ...doc.data() }))
48          );
49          setLivros(
50            livrosSnapshot.docs.map((doc) => ({ id: doc.id, ...doc.data() }))
51          );
52        } catch (error) {
53          console.error("Erro ao carregar dados:", error);
54        }
55        setLoading(false);
56      };
57
58      fetchData();

```

HOME DO PROJETO :



TELA DE CONTROLE DE LIVROS :



GERENCIAMENTO DE USUÁRIOS :

### Gerenciar Usuários

Adicionar Usuário

Lucas José Leite Marinho - lucasjosebr@gmail.com

Editar

Deletar

## GERENCIAMENTO DE EMPRÉSTIMOS :

### Gerenciamento de Empréstimos

#### Registrar novo empréstimo

Selecione um usuário

Selecione um livro

dd/mm/aaaa

dd/mm/aaaa

Adicionar empréstimo

#### Empréstimos Ativos

Usuário:

Livro:

Data de Devolução: 25/03/2024

Marcar como Devolvido

# LIÇÕES APRENDIDAS

A realização deste projeto foi uma experiência rica em aprendizado e crescimento, tanto no aspecto técnico quanto no organizacional. Entre os principais aprendizados, destaco a capacidade de elaborar a **Estrutura Analítica do Projeto (EAP)**, ferramenta essencial para dividir e organizar as tarefas em entregas gerenciáveis. Isso contribuiu diretamente para a compreensão das etapas necessárias e o acompanhamento eficaz do progresso.

Outro ponto importante foi o desenvolvimento e gerenciamento do cronograma por meio do **Gráfico de Gantt**. Essa prática trouxe clareza na visualização de prazos e dependências entre tarefas, permitindo melhor controle sobre as entregas e ajustes ao longo do caminho. A utilização do **Trello** como ferramenta colaborativa também foi fundamental para centralizar informações, distribuir tarefas e monitorar a evolução do projeto de maneira integrada com a equipe.

Além disso, a criação de uma planilha de custos foi um aprendizado significativo, proporcionando uma visão mais estruturada e estratégica sobre a gestão de recursos financeiros. No aspecto técnico, houve um grande aprimoramento no uso de tecnologias modernas como **JavaScript, React e Firebase**, que desempenharam papéis centrais no desenvolvimento do projeto. A integração entre essas ferramentas permitiu entregar um produto final robusto e alinhado aos requisitos.

Esses aprendizados não apenas agregaram valor ao projeto, mas também contribuíram para nossa evolução como profissionais, fortalecendo habilidades que serão valiosas em desafios futuros.

## AGRADECIMENTOS

Gostaríamos de expressar nossa sincera gratidão a todos que contribuíram de forma direta ou indireta para a realização deste projeto.

Primeiramente, agradecemos ao nosso professor João Ferreira, por todo o suporte técnico, acadêmico e pelas orientações valiosas que foram fundamentais para a execução e o sucesso deste trabalho.

Estendemos nosso agradecimento à Uninassau, por disponibilizar os recursos necessários e por proporcionar um ambiente que favorece o aprendizado e o desenvolvimento de habilidades práticas.

Aos membros da equipe, nosso reconhecimento pelo comprometimento, pela troca de ideias e pelo espírito colaborativo demonstrado ao longo de todo o projeto. O esforço conjunto foi essencial para superar os desafios e alcançar os objetivos propostos.

A todos, o nosso muito obrigado!

## REFERÊNCIAS

Dicas de Excel - Criando um gráfico de Gantt para um cronograma no Excel sem utilizar fórmulas. Disponível em: <https://www.youtube.com/watch?v=s6Bfo1ihHwA&t=5s>. Acesso em: 1 dez. 2024.

Como Fazer Planilha no Excel - Passo a Passo para Criar Planilhas do Zero no Nível Iniciante. Disponível em: <https://www.youtube.com/watch?v=9QbIulZMVUw>. Acesso em: 20 nov. 2024.

Como fazer a EAP ESTRUTURA ANALÍTICA DO PROJETO na prática. Disponível em: <https://www.youtube.com/watch?v=Q6U2H-od5eA>. Acesso em: 21 dez. 2024.

Tutorial completo de Trello 2024 - Gestão de tarefas pessoais e da equipe. [video] Disponível em: <https://www.youtube.com/watch?v=yEjI1KLIYQs&t=703s>. Acesso em: 18 dez. 2024.