

GUILHERME H. B. DAMASCENO
DOUGLAS S. DE OLIVEIRA

POD TECH

ReserveAqui

FIAP Tech Challenge

Ver código no Github <https://github.com/Guihenrry/reserve-aqui>

Sumário

| | |
|---|---|
| Introdução | 3 |
| Processo de desenvolvimento | 4 |
| Stack escolhida | 6 |
| Desafios encontrados | 7 |
| Documentação das APIs desenvolvidas | 8 |

Introdução

O software "Reserve Aqui" foi desenvolvido para proporcionar aos clientes uma forma intuitiva e eficiente de reservar espaços para festas, quadras esportivas, e outros tipos de locações. Com uma interface amigável e acessível, os usuários podem verificar a disponibilidade dos espaços em tempo real e selecionar os horários que melhor atendem às suas necessidades. Além disso, em caso de imprevistos, o software permite o cancelamento da reserva de maneira simples e rápida, garantindo flexibilidade e conveniência ao cliente. "Reserve Aqui" surge como uma solução completa e moderna para a gestão de reservas online, otimizando o processo tanto para os clientes quanto para os administradores dos espaços.

Processo de desenvolvimento

O desenvolvimento da nossa aplicação seguiu um processo estruturado conforme descrito a seguir:

1 - Definição de Requisitos:

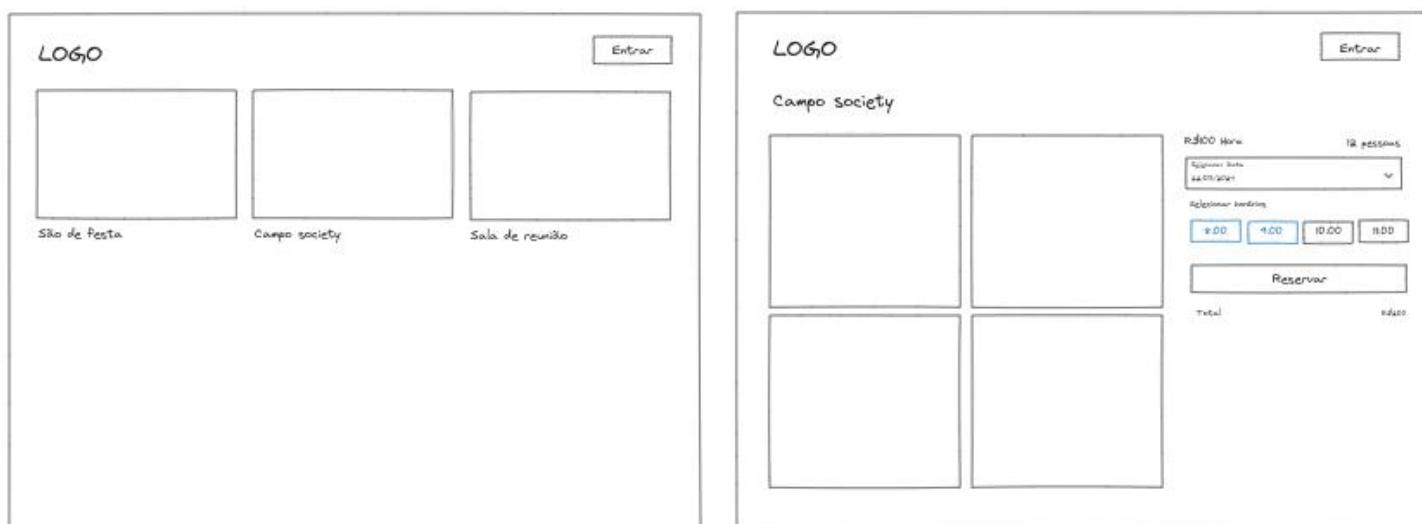
Requisitos Funcionais: Listamos todas as funcionalidades que a aplicação deve oferecer, como a possibilidade de reservar espaços, verificar a disponibilidade em tempo real, e permitir cancelamentos.

Requisitos Não Funcionais: Identificamos aspectos como desempenho, segurança, usabilidade, e escalabilidade que a aplicação precisa atender.

Regras de Negócio: Estabelecemos as regras que governam o funcionamento do sistema, incluindo políticas de cancelamento, limites de reserva, e gestão de horários.

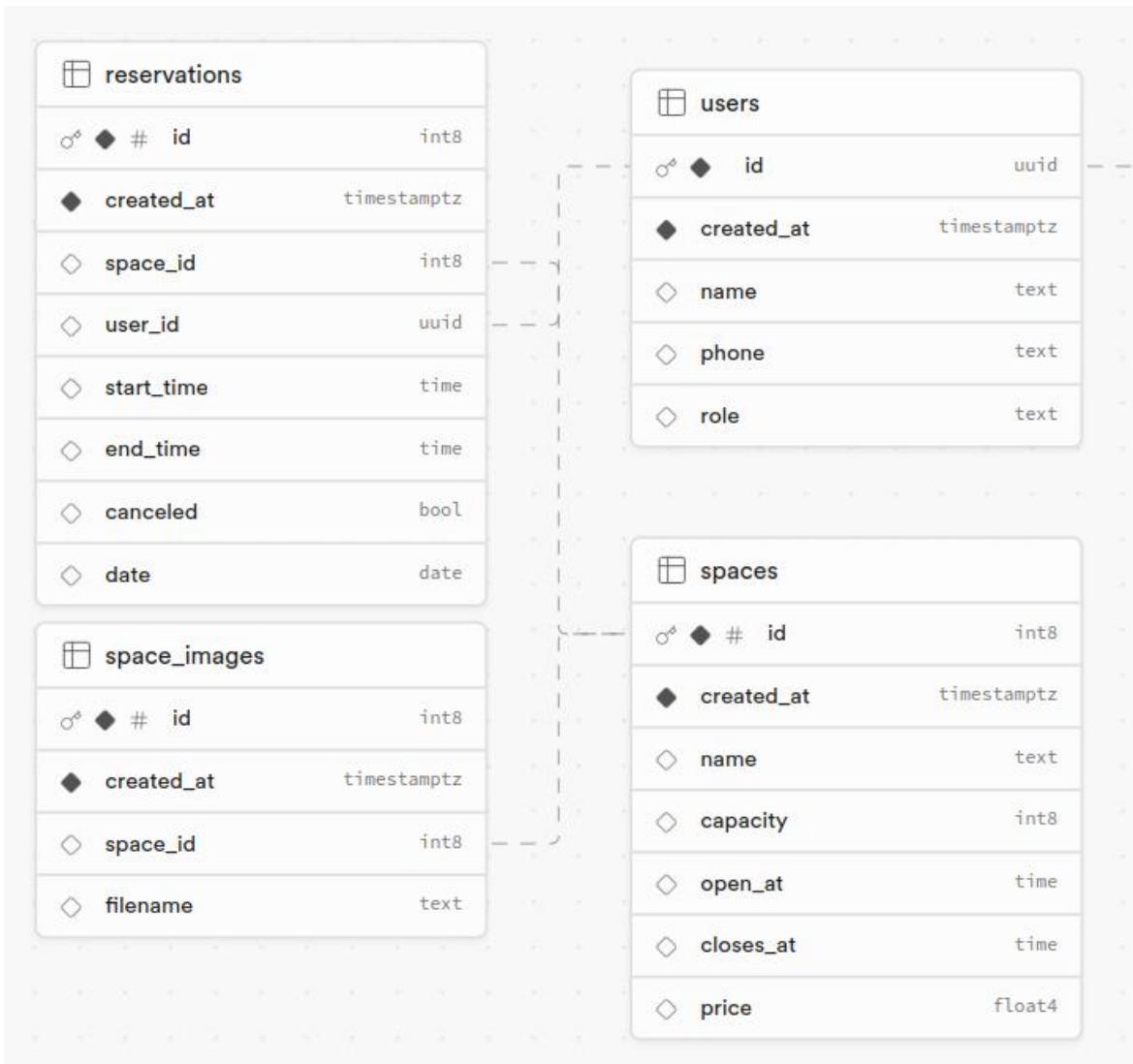
2 - Criação do Wireframe:

Desenvolvemos wireframes detalhados para a nossa aplicação, proporcionando uma visualização clara da interface do usuário e da experiência de navegação. Esses protótipos ajudaram a alinhar as expectativas da equipe e dos stakeholders.



3 - Modelagem do Banco de Dados:

Projetamos a estrutura das tabelas do banco de dados para suportar as funcionalidades definidas. A modelagem cuidadosa garantiu a integridade dos dados e a eficiência das operações de consulta e atualização.



4 - Definição das Rotas da API

Especificamos as rotas da API necessárias para a comunicação entre o front-end e o back-end da aplicação. As rotas foram projetadas para cobrir todas as operações CRUD (Create, Read, Update, Delete) e outras interações essenciais.

Stack escolhida

Front-end

Optamos por React devido à sua capacidade de componentização, que facilita a reutilização de código e a manutenção do projeto.

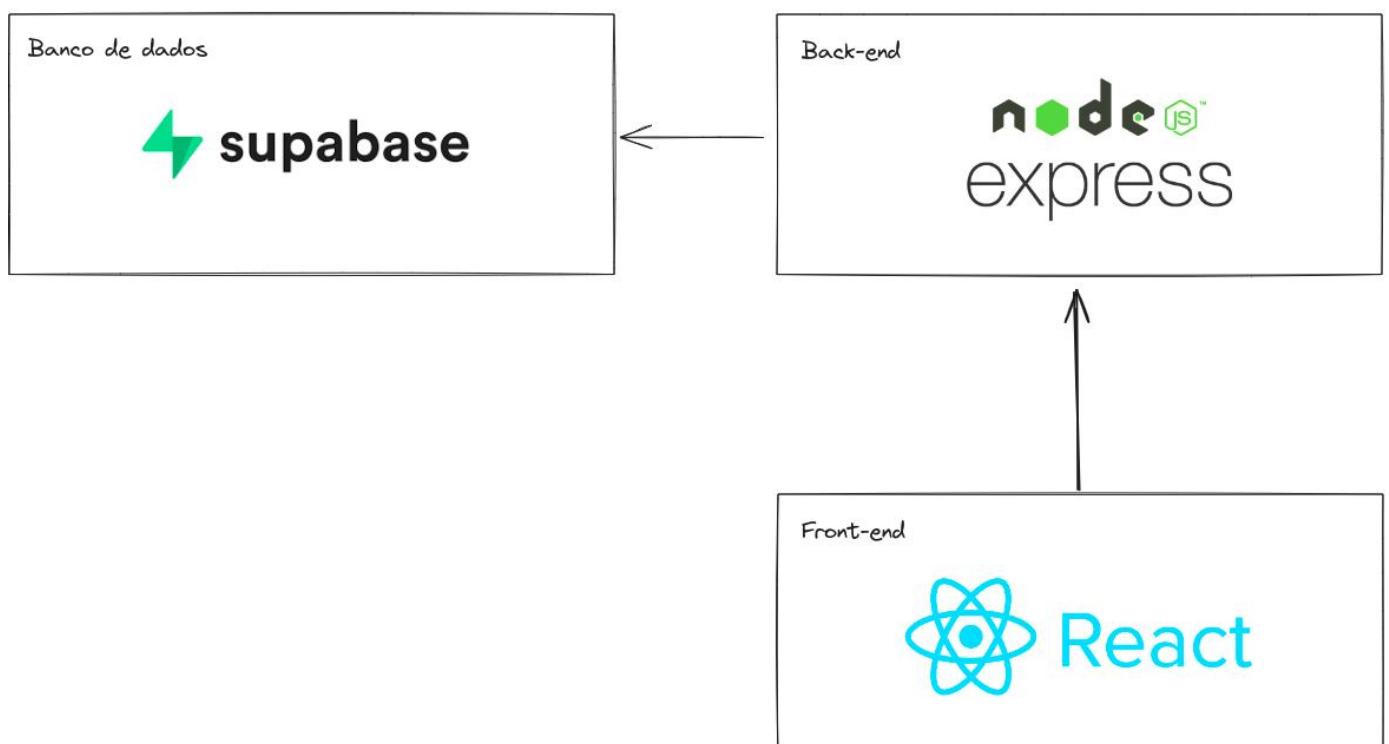
Back-end

Escolhemos Express e Node.js, aproveitando o conhecimento prévio da equipe, o que agiliza o desenvolvimento e resolução de problemas.

Banco de Dados

Implementamos o Supabase, reconhecido por sua facilidade de deploy e integração com projetos baseados em Node.js.

Essa abordagem garantiu um desenvolvimento organizado e eficiente da nossa aplicação, resultando em um produto fácil de usar, tanto para os clientes quanto para os administradores dos espaços.



Desafios encontrados

Durante o desenvolvimento da nossa aplicação, enfrentamos diversos desafios e superamos obstáculos para garantir um produto final. O processo incluiu os seguintes aspectos:

Criação da Estrutura do Banco de Dados:

Desenvolvemos uma estrutura de banco de dados eficiente que facilita o armazenamento e a gestão dos horários escolhidos pelos usuários, assegurando a integridade e acessibilidade dos dados.

Implementação das Rotas da API:

Criamos uma rota específica que retorna os horários disponíveis para reserva. Esta funcionalidade é essencial para que os usuários possam visualizar em tempo real os horários ainda não ocupados.

Desenvolvimento do Componente Visual:

Implementamos um componente visual que impede os usuários de reservarem horários já ocupados, melhorando a usabilidade e prevenindo conflitos de agendamento.

Design de Interface de Usuário Intuitiva:

Projetamos uma interface de usuário que é fácil de navegar para todos os tipos de usuários, independentemente de sua familiaridade com a tecnologia. Isso incluiu a criação de um design claro e acessível que facilita a experiência do usuário.

Desafios Durante o Desenvolvimento:

Enfrentamos um desafio significativo quando um dos membros da equipe, residente em Porto Alegre, Rio Grande do Sul, ficou sem internet, água e luz por uma semana devido às enchentes que assolaram o estado. Este evento inesperado exigiu uma readequação das tarefas e colaboração entre os demais membros para garantir o cumprimento dos prazos e a continuidade do desenvolvimento.

Documentação das APIs desenvolvidas

POST /signin

Autentica um usuário no sistema.

Request - JSON

```
{  
  "email": "string",  
  "password": "string"  
}
```

Response

200

```
{  
  "user": { ... }  
}
```

POST /signup

Cria uma nova conta de usuário.

Request - JSON

```
{  
  "name": "Jose",  
  "phone": "5435346785",  
  "email": "admin@reserve aqui.com",  
  "password": "12345678"  
}
```

Response

200

```
{  
  "user": { ... }  
}
```

GET /user

Obtém informações do usuário autenticado.

Request - Headers

```
Authorization: Bearer {token}
```

Response

200

```
{  
  "user": { ... }  
}
```

POST /spaces

Cria um novo espaço para reserva.

Request - Multipart FormData

```
{  
  "name": "Campo de futebol",  
  "capacity": 60,  
  "open_at": "09:00:00",  
  "closes_at": "18:00:00",  
  "price": 100,  
  "images": Files  
}
```

Response

201

```
{  
  "space": { ... }  
}
```

GET /spaces

Obtém a lista de todos os espaços disponíveis.

Response

```
[  
  {  
    "id": 15,  
    "created_at": "2024-05-01T23:37:02.045907+00:00",  
    "name": "Campo de futebol",  
    "capacity": 60,  
    "open_at": "09:00:00",  
    "closes_at": "18:00:00",  
    "price": 50,  
    "space_images": []  
  }  
]
```

200

GET /spaces/:id

Obtém informações de um espaço específico.

Response

```
{  
  "id": 15,  
  "created_at": "2024-05-01T23:37:02.045907+00:00",  
  "name": "Campo de futebol",  
  "capacity": 60,  
  "open_at": "09:00:00",  
  "closes_at": "18:00:00",  
  "price": 50,  
  "space_images": []  
}
```

200

DELETE /spaces/:id

Deleta um espaço específico.

Request - Headers

```
Authorization: Bearer {token}
```

Response

```
200
```

PUT /spaces/:id

Atualiza as informações de um espaço específico.

Request - JSON

```
{
  "name": "Campo de futebol",
  "capacity": 60
}
```

Response

```
201
```

```
{
  "space": { ... }
}
```

GET /spaces/:id/hours

Obtém os horários disponíveis de um espaço específico.

Request - Query

```
date: 2024-05-22
```

Response**200**

```
[  
  {  
    "hour": "09:00",  
    "available": true  
  },  
  {  
    "hour": "10:00",  
    "available": true  
  }  
]
```

GET /spaces/:id/reservations

Obtém as reservas de um espaço específico.

Response**200**

```
[  
  {  
    "id": 17,  
    "created_at": "2024-05-21T23:37:33.464537+00:00",  
    "space_id": 15,  
    "user_id": "ed4acc9d-2941-489a-8d68-553f5bd33466",  
    "start_time": "13:00:00",  
    "end_time": "15:00:00",  
    "canceled": false,  
    "date": "2024-05-22",  
    "user": []  
  }  
]
```

GET /reservations

Obtém a lista de todas as reservas do usuário autenticado.

Response

```
[  
  {  
    "space_id": 15,  
    "start_time": "10:00:00",  
    "end_time": "12:00:00",  
    "date": "2024-05-22"  
  }  
]
```

200**POST /reservations**

Cria uma nova reserva.

Request - JSON

```
{  
  "space_id": 15,  
  "start_time": "10:00:00",  
  "end_time": "12:00:00",  
  "date": "2024-05-22"  
}
```

Response**200****POST /reservations/:id/cancel**

Cancela uma reserva existente.

Response**200**