

**1. Endpoint**

- a. POST /api/auth/register

**2. Descrição**

- a. Registra um novo usuário no sistema

**3. Resposta esperada**

- a. { "message": "Usuário registrado com sucesso!" }

**Body usado:**

json

Copiar  
Editar

```
{  
  "nome": "João Silva",  
  "email": "joao@email.com",  
  "telefone": "11999999999",  
  "cpf": "123.456.789-00",  
  "senhaHash": "123456",  
  "role": "usuario"  
}
```

**Resposta recebida:**

json

Copiar  
Editar

```
{  
  "message": "Usuário registrado com sucesso!"  
}
```

HTTP <http://localhost:5289/api/auth/register>

POST [▼](#) http://localhost:5289/api/auth/register

Params Authorization Headers (8) **Body** • Scripts Tests Settings

none  form-data  x-www-form-urlencoded  raw  binary  GraphQL [JSON](#) [▼](#)

```
1 {  
2     "nome": "João Silva",  
3     "email": "joao@email.com",  
4     "telefone": "11999999999",  
5     "cpf": "123.456.789-00",  
6     "senhaHash": "123456",  
7     "role": "usuario"  
8 }  
9
```

Body Cookies Headers (4) Test Results | [↻](#)

{ } [JSON](#) [▼](#) [▷ Preview](#) [🔗 Visualize](#) [▼](#)

```
1 {  
2     "message": "Usuário registrado com sucesso!"  
3 }
```

**4. Endpoint**

- a. POST/api/auth/login

**5. Descrição:**

- a. Autentica o usuário e retorna o token JWT.

**6. Resposta Esperada:**

- a. **Status:** 200 OK

- **Body (JSON):**

json

Copiar Editar

{

    "email": "joao@email.com",

    "senha": "123456"

}

◆

json

Copiar Editar

{

    "token": "eyJhbGciOiJIUzI1NilsInR5cCI6IkpXVCJ9..."

}

HTTP <http://localhost:5289/api/auth/login>

POST <http://localhost:5289/api/auth/login>

Params Authorization Headers (8) **Body** Scripts Tests Settings

none  form-data  x-www-form-urlencoded  raw  binary  GraphQL [JSON](#)

```
1 {  
2   "email": "joao@email.com",  
3   "senha": "123456"  
4 }  
5  
6
```

Body Cookies Headers (4) Test Results | ⏱

{ } [JSON](#) ▾ ▶ Preview ⚡ Visualize ▾

```
1 {  
2   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.  
eyJodHRwOi8vc2NoZW1hcyc54bWxzb2FwLm9yZy93cy8yMDA1LzA1L2lkZW50aXR5L2M  
WRlbnRpdmHkvY2xhaW1zL3JvbGUiOiJ1c3Vhcm1vIiwiZXhwIjoxNzQ2ODM0MTgxLCJp  
3 }
```

## **Endpoint: Consultar Mesas**

**Descrição** Retorna a lista de todas as mesas cadastradas, incluindo detalhes da sala associada.

**Método** GET

**URL** /api/Mesas

### ◆ Resposta Esperada

- **Status:** 200 OK
- **Body (JSON exemplo):**

json

Copiar  
Editar

```
[  
  {  
    "id": 11,  
    "numero": 301,  
    "salald": 4,  
    "sala": {  
      "id": 4,  
      "nome": "Sala Privativa 1",  
      "capacidade": 6,  
      "recursos": "Wi-Fi, Ar-condicionado"  
    }  
  },  
  {  
    "id": 12,  
    "numero": 302,  
    "salald": 4,  
    "sala": {  
      "id": 4,  
      "nome": "Sala Privativa 2",  
      "capacidade": 6,  
      "recursos": "Wi-Fi, Ar-condicionado"  
    }  
  }]
```

```

        "nome": "Sala Privativa 1",
        "capacidade": 6,
        "recursos": "Wi-Fi, Ar-condicionado"
    }
}
]

```

HTTP <http://localhost:5289/api/Mesas>

GET <http://localhost:5289/api/Mesas>

Params Authorization Headers (9) Body Scripts Tests Settings

Headers [8 hidden](#)

	Key	Value
<input checked="" type="checkbox"/>		eyJhbGciOiJIUzI1N
	Key	Value

Body Cookies Headers (4) Test Results | ⏱

{ } JSON ▾ ▷ Preview ⏷ Visualize | ▾

```

1  [
2   {
3     "id": 11,
4     "numero": 301,
5     "salaId": 4,
6     "sala": {
7       "id": 4,
8       "nome": "Sala Privativa 1",
9       "capacidade": 6,
10      "recursos": "Wi-Fi, Ar-condicionado"
11    }
12  },
13  {
14    "id": 12,
15    "numero": 302,
16    "salaId": 4,
17    "sala": {
18      "id": 4,
19      "nome": "Sala Privativa 1",
20      "capacidade": 6,
21      "recursos": "Wi-Fi, Ar-condicionado"
22    }
23  }

```

## **Endpoint: Atualizar Mesa**

**Descrição** Atualiza as informações de uma mesa existente no sistema, incluindo o número e a vinculação à sala correspondente.

**Método** PUT

**URL** /api/Mesas/{id}

---

◆ **Body (JSON exemplo):**

json

Copiar  
Editar

```
{  
    "id": 11,  
    "numero": 310,  
    "salalid": 4,  
    "sala": {  
        "id": 4,  
        "nome": "Sala Privativa 1",  
        "capacidade": 6,  
        "recursos": "Wi-Fi, Ar-condicionado"  
    }  
}
```

◆ **Resposta Esperada**

- **Status:** 204 No Content
- **Body:** Nenhum conteúdo retornado. A resposta indica que a mesa foi atualizada com sucesso.

The screenshot shows a Postman interface for a PUT request to `http://localhost:5289/api/Mesas/11`. The request body is a JSON object representing a room:

```
[{"id": 11, "name": "Sala 35B", "salaid": 4, "salal1": 4, "salal2": 4, "name": "Sala Privativa 1", "capacidad": 4, "recursos": "Wi-Fi, Ar-condicionado"}]
```

The response status is 204 No Content.

## **Endpoint: Consultar Mesa por ID**

**Descrição** Retorna os dados de uma mesa específica com os detalhes da sala vinculada.

**Método** GET

**URL** /api/Mesas/{id}

---

### ◆ Requisição

- **Headers:**

- **Parâmetro:**

- {id}: ID da mesa a ser consultada (exemplo: /api/Mesas/11).
- 

### ◆ Resposta Esperada

- **Status:** 200 OK

- **Body (JSON exemplo):**

json

Copiar Editar

```
{  
  "id": 11,  
  "numero": 310,  
  "salalid": 4,  
  "sala": {  
    "id": 4,  
    "nome": "Sala Privativa 1",  
    "capacidade": 6,  
    "recursos": "Wi-Fi, Ar-condicionado"  
  }  
}
```

- **Campos:**

- o id: ID da mesa,
- o numero: Número da mesa,
- o salaId: ID da sala vinculada,
- o sala: Objeto contendo os detalhes da sala (nome, capacidade e recursos).

HTTP <http://localhost:5289/api/Mesas/11>

GET <http://localhost:5289/api/Mesas/11>

Params Authorization Headers (9) Body Scripts Tests Settings

Headers (8 hidden)

	Key
<input checked="" type="checkbox"/>	
	Key

Body Cookies Headers (4) Test Results

{ } JSON ▾ Preview Visualize

```
1 {  
2   "id": 11,  
3   "numero": 310,  
4   "salaId": 4,  
5   "sala": {  
6     "id": 4,  
7     "nome": "Sala Privativa 1",  
8     "capacidade": 6,  
9     "recursos": "Wi-Fi, Ar-condicionado"  
10    }  
11 }
```

## Endpoint: Excluir Mesa por ID

**Descrição** Exclui uma mesa existente do sistema, identificada pelo seu ID.

**Método** DELETE

**URL** /api/Mesas/{id}

---

### ◆ Parâmetro:

- {id}: ID da mesa a ser excluída (exemplo: /api/Mesas/11).
  - **Body:** Nenhum.
- 

### ◆ Resposta Esperada

- **Status:** 204 No Content
- **Body:** Nenhum conteúdo retornado (confirmação de exclusão realizada com sucesso).



The screenshot shows the Postman interface with a DELETE request to `http://localhost:5288/api/Mesas/11`. The Headers tab contains a single entry: `Content-Type: application/json`. The Body tab is empty. The Response tab shows a successful `204 No Content` response with a timestamp of `2023-09-04T10:45:23.948Z`.



The screenshot shows the Postman interface with the same DELETE request. The Headers tab now includes an Authorization header with a long token value: `eyJhbGciOiJIUzI1NiJ9.eyJodHRwOi8vYmVzZW1hcy54bWx12FvtLm9yZy93cigbMDA1LzA1L2kZ...`. The Response tab shows a successful `204 No Content` response with a timestamp of `2023-09-04T10:45:23.948Z`.

## Endpoint: Consultar Reservas

**Descrição** Retorna a lista de todas as reservas cadastradas, com detalhes do usuário, sala e mesa vinculados.

**Método** GET

**URL** /api/Reservas

---

### ◆ Requisição

- **Headers:**

Key	Value
-----	-------

Content-Type application/json

Authorization Bearer eyJhbGciOiJIUzI1NilsInR5cCl6IkpxVCJ9... (token JWT do admin)

- **Body:** Nenhum.
- 

### ◆ Resposta Esperada (200 OK)

json

Copiar Editar

[

{

"id": 4,  
"usuarioid": 14,  
"usuario": {  
 "id": 14,  
 "nome": "Fillipe Gabriel Izidorio Serafim",  
 "email": "fillipe.serafim@email.com",  
 "telefone": "(11) 99999-1111",  
 "cpf": "123.456.789-01",  
 "senhaHash": "\*\*\*\*\*",

```
        "role": "admin"

    },
    "salaid": 1,
    "sala": {
        "id": 1,
        "nome": "Sala de Reunião 1",
        "capacidade": 10,
        "recursos": "Wi-Fi, TV, Quadro Branco"
    },
    "mesaid": 1,
    "mesa": {
        "id": 1,
        "numero": 101,
        "salaid": 1,
        "sala": {
            "id": 1,
            "nome": "Sala de Reunião 1",
            "capacidade": 10,
            "recursos": "Wi-Fi, TV, Quadro Branco"
        }
    },
    "dataHora": "2025-05-09T23:45:14.593Z",
    "status": "Confirmada"
}
]
```

- **Campos:**

- **usuario:** Detalhes completos do usuário que fez a reserva,
- **sala:** Dados da sala vinculada,

- o mesa: Dados da mesa e da sala correspondente,
- o dataHora: Data e hora da reserva,
- o status: Status atual da reserva.

http://localhost:5289/api/Reservas

GET http://localhost:5289/api/Reservas

Headers (9) Body Scripts Tests Settings

Headers < 8 hidden

Key	Value	Description
<input checked="" type="checkbox"/> Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5cI6IkpXVCJ9eyJodHRwOi8vc2NoZWlhcyc54bWxb2FwLm9yZy93cy8yMDA1LzA...	
Key	Value	Description

Body Cookies Headers (4) Test Results ⓘ

{ } JSON ▾ ▶ Preview ⏷ Visualize ▾

```

1 [
2   {
3     "id": 4,
4     "usuarioId": 14,
5     "usuario": {
6       "id": 14,
7       "name": "Fillipe Gabriel Iridorio Serafim",
8       "email": "fillipe.serafim@email.com",
9       "telefone": "(31) 99999-1111",
10      "cpf": "123.456.789-01",
11      "senhash": "92a81190nwYvc2GU27f9Lh88oxF.6R88pRP70jOEP81WMha99T8f1uVnAw.",
12      "role": "admin"
13    },
14    "salaid": 1,
15    "sala": {
16      "id": 1,
17      "name": "Sala de Reuniao 1",
18      "capacidade": 18,
19      "status": "Disponivel"
20    }
21  ]

```

## Endpoint: Criar Reserva

**Descrição** Cria uma nova reserva associando o usuário autenticado a uma sala e mesa específicos, em uma data e hora definidos.

**Método** POST

**URL** /api/Reservas

---

### ◆ Requisição

- **Headers:**

**Key** **Value**

Content-Type application/json

Authorization Bearer eyJhbGciOiJIUzI1NilsInR5cCI6IkpXVCJ9... (token JWT do usuário ou admin)

---

- **Body (JSON exemplo):**

```
json
CopiarEditar
{
  "salald": 4,
  "mesald": 12,
  "dataHora": "2025-05-10T10:30:00"
}
```

- **Campos:**

- salald: ID da sala onde a reserva será feita,
  - mesald: ID da mesa específica,
  - dataHora: Data e hora da reserva (no formato ISO 8601).
- 

### ◆ Resposta Esperada (201 Created)

json

Copiar Editar

```
{  
  "message": "Sua reserva foi confirmada! ID: 10"  
}
```

- O sistema retorna uma mensagem de confirmação informando o ID da nova reserva criada.

The screenshot shows a POST request to `http://localhost:5288/api/Reservas`. The JSON body of the request is:

```
1 {  
2   "salalId": 4,  
3   "reservaId": 12,  
4   "dataReserv": "2020-05-10T19:30:00"  
5 }
```

The response shows a successful creation with status `201 Created`, time `412 ms`, and size `250 B`. The response body is:

```
1 {  
2   "message": "Sua reserva foi confirmada! ID: 10"  
3 }
```

## **Endpoint: Consultar Reserva por ID**

**Descrição** Retorna os detalhes completos de uma reserva específica, identificada pelo seu ID.

**Método** GET

**URL** /api/Reservas/{id}

---

### ◆ Requisição

- **Headers:**

**Key** **Value**

Content-Type application/json

Authorization Bearer eyJhbGciOiJIUzI1NilsInR5cCI6IkpXVCJ9... (token JWT do usuário ou admin)

- **Parâmetro:**

- {id}: ID da reserva que deseja consultar (exemplo: /api/Reservas/10).

- **Body:** Nenhum.
- 

### ◆ Resposta Esperada (200 OK)

json

Copiar Editar

```
{  
  "id": 10,  
  "usuarioid": 22,  
  "usuario": null,  
  "salaid": 4,  
  "sala": null,  
  "mesaid": 12,  
  "mesa": null,
```

```
        "dataHora": "2025-05-10T10:30:00",
        "status": "Confirmada"
    }
```

- **Campos:**

- - usuarioid: ID do usuário,
  - salald: ID da sala,
  - mesald: ID da mesa,
  - dataHora: Data e hora da reserva,
  - status: Status atual da reserva.

---

- ◆ **Resposta em caso de erro**

- **401 Unauthorized:** Token inválido ou ausente.
- **403 Forbidden:** Usuário sem permissão.
- **404 Not Found:** A reserva com o ID informado não foi encontrada.

HTTP <http://localhost:5289/api/Reservas/10>

GET <http://localhost:5289/api/Reservas/10>

Params Authorization Headers (9) Body Scripts Tests Settings

none  form-data  x-www-form-urlencoded  raw  binary  GraphQL [JSON](#)

```
1  {
2    "salaId": 4,
3    "mesaId": 12,
4    "dataHora": "2025-05-10T10:30:00"
5  }
6
7
8
9
10
11
12
13
14
15
16
```

Body Cookies Headers (4) Test Results | ⚙️

{ } JSON ▾ ▷ Preview ⚙️ Visualize ▾

```
1  {
2    "id": 10,
3    "usuarioId": 22,
4    "usuario": null,
5    "salaId": 4,
6    "sala": null,
7    "mesaId": 12,
8    "mesa": null,
9    "dataHora": "2025-05-10T10:30:00",
10   "status": "Confirmada"
11 }
```

## Endpoint: Atualizar Reserva

**Descrição** Atualiza todos os detalhes de uma reserva específica, identificada pelo seu ID.

**Método** PUT

**URL** /api/Reservas/{id}

---

### ◆ Requisição

- **Headers:**

Key	Value
-----	-------

Content-Type	application/json
--------------	------------------

Authorization	Bearer eyJhbGciOiJIUzI1NilsInR5cCI6IkpXVCJ9... (token JWT do usuário ou admin)
---------------	--

- **Parâmetro:**

- {id}: ID da reserva que deseja atualizar (exemplo: /api/Reservas/10).
- 

- **Body (JSON completo exemplo):**

json

Copiar  
Editar

```
{  
    "id": 10,  
    "usuarioid": 22,  
    "usuario": {  
        "id": 22,  
        "nome": "Admin Teste",  
        "email": "admin@email.com",  
        "telefone": "11988887777",  
        "cpf": "999.999.999-99",  
    },  
}
```

```
"senhaHash": "*****",
"role": "admin"
},
"salald": 4,
"sala": {
"id": 4,
"nome": "Sala Privativa 1",
"capacidade": 6,
"recursos": "Wi-Fi, Ar-condicionado"
},
"mesald": 12,
"mesa": {
"id": 12,
"numero": 302,
"salald": 4,
"sala": {
"id": 4,
"nome": "Sala Privativa 1",
"capacidade": 6,
"recursos": "Wi-Fi, Ar-condicionado"
}
},
"dataHora": "2025-05-10T11:30:00",
"status": "Confirmada"
}
```

---

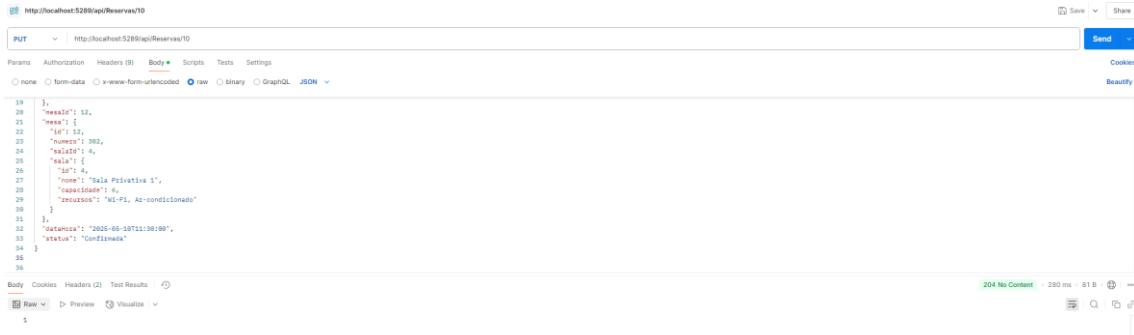
◆ **Resposta Esperada (204 No Content)**

- Atualização realizada com sucesso, sem retorno de conteúdo.

---

#### ◆ Resposta em caso de erro

- **400 Bad Request:** Dados inválidos ou JSON incompleto.
- **401 Unauthorized:** Token inválido ou ausente.
- **403 Forbidden:** Usuário sem permissão.
- **404 Not Found:** A reserva não foi encontrada.



The screenshot shows a Postman request to `http://localhost:5280/api/Reservas/10` using the `PUT` method. The request body is a JSON object representing a room reservation. The response status is `204 No Content`, indicating the operation was successful.

```
1,
2   "numero": 12,
3   "mesa": {
4     "id": 12,
5     "numero": 302,
6     "lateral": 4,
7     "sala": {
8       "id": 4,
9       "nome": "Sala Privativa 1",
10      "capacidade": 4,
11      "reservado": "Nº-F1, Accondicionado"
12    },
13    "datahora": "2020-08-10T11:30:00",
14    "status": "Confirmada"
15  }
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
```

## **Endpoint: Excluir Reserva**

**Descrição** Exclui uma reserva específica, identificada pelo seu ID.

**Método** DELETE

**URL** /api/Reservas/{id}

---

### ◆ Requisição

- **Headers:**

Key	Value
-----	-------

Content-Type	application/json
--------------	------------------

Authorization	Bearer eyJhbGciOiJIUzI1NilsInR5cCl6IkpxVCJ9... (token JWT do usuário ou admin)
---------------	--

- **Parâmetro:**

- {id}: ID da reserva que deseja excluir (exemplo: /api/Reservas/10) .

- **Body:** Nenhum.
- 

### ◆ Resposta Esperada (204 No Content)

- Reserva excluída com sucesso; a resposta não contém conteúdo.
- 

### ◆ Resposta em caso de erro

- **401 Unauthorized:** Token inválido ou ausente.
- **403 Forbidden:** Usuário sem permissão para excluir.
- **404 Not Found:** A reserva com o ID informado não foi encontrada.

http://localhost:5289/api/Reservas/10

DELETE http://localhost:5289/api/Reservas/10

Save Share

Params Authorization Headers (9) Body Scripts Tests Settings Cookies

Headers > 8 hidden

Key	Value	Description	Bulk Edit	Preset
<input checked="" type="checkbox"/> Authorization	Bearer eyJhbGciOiJIUzI1NiJ9.SCB0IkpxVC.JIeyJodHRwOi8vc2NoZWVtcy54dWxzb2FwLmdyZy93cylbMDA1LzA...	Description		
Key	Value	Description		

Body Cookies Headers (2) Test Results

Raw Preview Visualize

1

204 No Content · 280 ms · 81 B · ⏺ ⏻ ⏴ ⏵

## **Endpoint: Consultar Salas**

**Descrição** Retorna uma lista de todas as salas cadastradas no sistema.

**Método** GET

**URL** /api/Salas

---

### ◆ Requisição

- **Headers:**

Key	Value
-----	-------

Content-Type	application/json
--------------	------------------

Authorization	Bearer eyJhbGciOiJIUzI1NilsInR5cCl6IkpxVCJ9... (token JWT do usuário ou admin)
---------------	--

- **Body:** Nenhum.
- 

### ◆ Resposta Esperada (200 OK)

json

Copiar Editar

[

{

    "id": 1,

    "nome": "Sala de Reunião 1",

    "capacidade": 10,

    "recursos": "TV, Wi-Fi, Ar-condicionado"

},

{

    "id": 2,

    "nome": "Sala de Reunião 2",

    "capacidade": 8,

```

    "recursos": "TV, Wi-Fi, Quadro Branco, Ar-condicionado"
},
{
    "id": 3,
    "nome": "Sala de Reunião 3",
    "capacidade": 6,
    "recursos": "TV, Wi-Fi, Ar-condicionado"
}
]

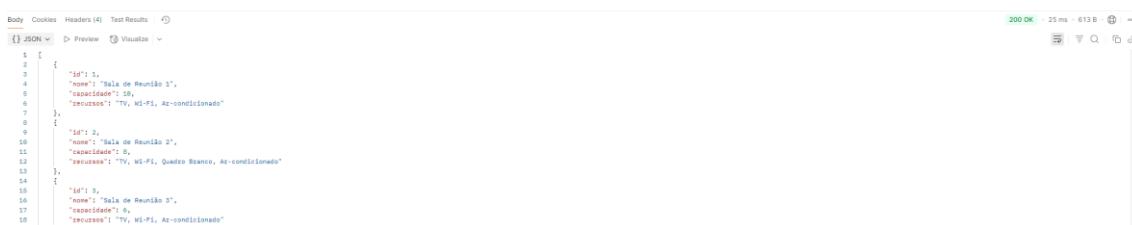
```

- **Campos:**

- **id:** ID único da sala,
- **nome:** Nome da sala,
- **capacidade:** Capacidade máxima da sala,
- **recursos:** Lista dos recursos disponíveis na sala.

◆ **Resposta em caso de erro**

- **401 Unauthorized:** Token inválido ou ausente.
- **403 Forbidden:** Usuário sem permissão para consultar.



## **Endpoint: Criar Sala**

**Descrição** Cria uma nova sala no sistema com nome, capacidade e recursos informados.

**Método** POST

**URL** /api/Salas

---

### ◆ Requisição

- **Headers:**

**Key** Value

Content-Type application/json

Authorization Bearer eyJhbGciOiJIUzI1NilsInR5cCl6IkpxVCJ9... (token JWT do admin)

---

- **Body (JSON exemplo):**

json

Copiar Editar

{

    "nome": "Sala de Reunião 4",  
    "capacidade": 12,  
    "recursos": "Wi-Fi, TV, Ar-condicionado"

}

- **Campos:**

- nome: Nome da nova sala,
- capacidade: Capacidade máxima de pessoas,
- recursos: Recursos disponíveis na sala.

---

### ◆ Resposta Esperada (201 Created)

json

## Copiar Editar

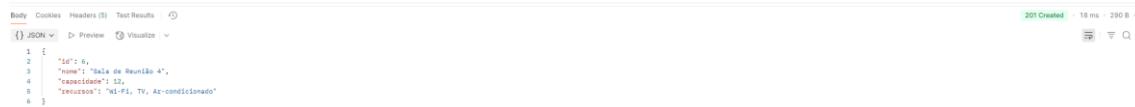
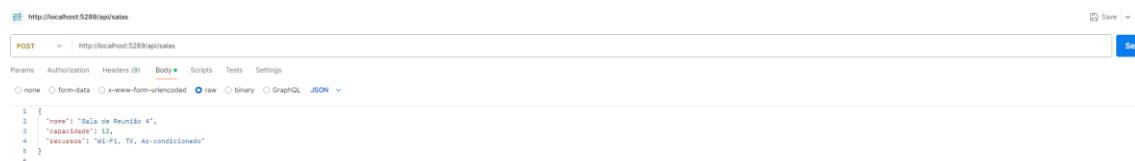
```
{  
  "id": 6,  
  "nome": "Sala de Reunião 4",  
  "capacidade": 12,  
  "recursos": "Wi-Fi, TV, Ar-condicionado"  
}
```

- A resposta retorna a nova sala criada, com seu id gerado automaticamente pelo sistema.

---

### ◆ Resposta em caso de erro

- **400 Bad Request:** Dados inválidos,
- **401 Unauthorized:** Token inválido ou ausente,
- **403 Forbidden:** Usuário sem permissão para criar sala.



## Endpoint: Consultar Sala

**Descrição** Retorna os detalhes de uma sala específica, identificada pelo seu ID.

**Método** GET

**URL** /api/Salas/{id}

---

### ◆ Requisição

- **Headers:**

**Key** **Value**

Content-Type application/json

Authorization Bearer eyJhbGciOiJIUzI1NilsInR5cCl6IkpxVCJ9... (token JWT do usuário ou admin)

- **Parâmetro:**

○ {id}: ID da sala a ser consultada (exemplo: /api/Salas/6).

- **Body:** Nenhum.
- 

### ◆ Resposta Esperada (200 OK)

json

Copiar Editar

```
{  
  "id": 6,  
  "nome": "Sala de Reunião 4",  
  "capacidade": 12,  
  "recursos": "Wi-Fi, TV, Ar-condicionado"  
}
```

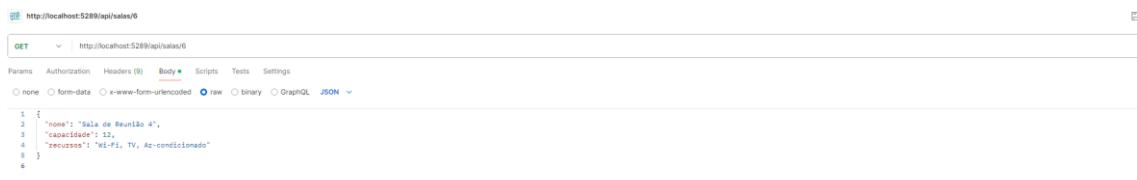
- **Campos:**

○ id: ID único da sala,

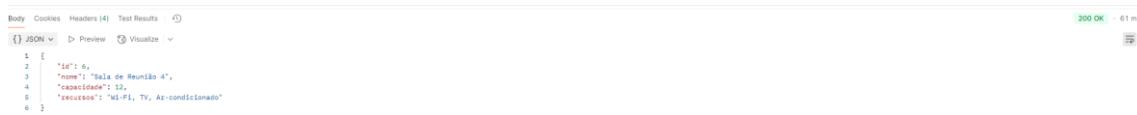
- o nome: Nome da sala,
  - o capacidade: Capacidade máxima,
  - o recursos: Lista dos recursos disponíveis.
- 

#### ◆ Resposta em caso de erro

- **401 Unauthorized:** Token inválido ou ausente,
- **403 Forbidden:** Usuário sem permissão para consultar,
- **404 Not Found:** Sala não encontrada.



```
1 {  
2   "id": 6,  
3   "name": "Sala de Reunião 4",  
4   "capacity": 12,  
5   "resources": "Wi-Fi, TV, Ar-condicionado"  
6 }
```



```
1 {  
2   "id": 6,  
3   "name": "Sala de Reunião 4",  
4   "capacity": 12,  
5   "resources": "Wi-Fi, TV, Ar-condicionado"  
6 }
```

## Endpoint: Atualizar Sala

**Descrição** Atualiza os dados (nome, capacidade e recursos) de uma sala existente pelo ID.

**Método** PUT

**URL** /api/Salas/{id}

---

◆ Requisição

- Headers:

**Key** Value

Content-Type application/json

Authorization Bearer {token válido do admin}

- Parâmetro:

○ {id}: ID da sala a ser atualizada (exemplo: /api/Salas/6).

---

- Body (exemplo usado):

json

Copiar Editar

```
{  
    "id": 6,  
    "nome": "Sala de Reunião 4 - Atualizada",  
    "capacidade": 15,  
    "recursos": "Wi-Fi, TV, Ar-condicionado, Projetor"  
}
```

---

◆ Resposta Esperada

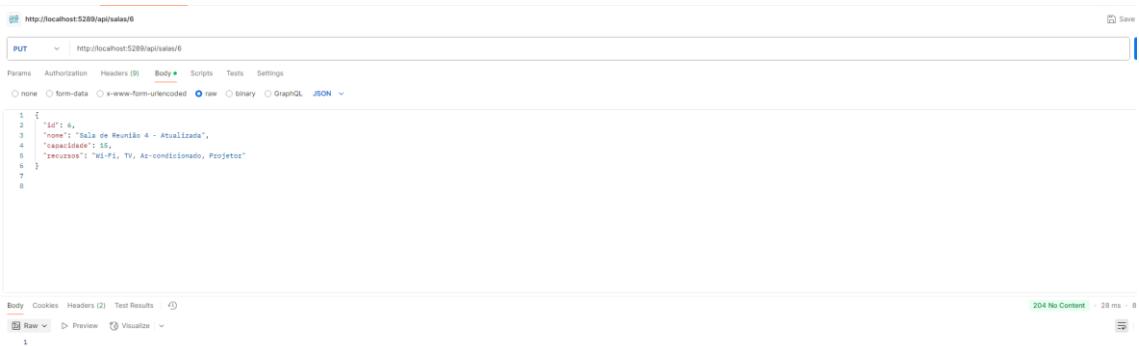
**Status** Descrição

204 No Content Atualização realizada com sucesso (sem retorno).

---

◆ Resposta em caso de erro

Status	Descrição
400 Bad Request	Dados inválidos enviados.
401 Unauthorized	Token não enviado ou inválido.
403 Forbidden	Usuário sem permissão para atualizar (exige role admin).
404 Not Found	Sala não encontrada para o ID informado.



The screenshot shows a POSTman interface with the following details:

- URL: `http://localhost:5289/api/sales/6`
- Method: `PUT`
- Body tab selected
- Request Body (Raw JSON):

```
1 {
2   "id": 6,
3   "name": "Sala de Reunião 4 - Atualizada",
4   "capacity": 15,
5   "resources": "Wi-Fi, TV, Ar-condicionado, Projetor"
6 }
```

- Response status: `204 No Content`
- Response time: `28 ms`
- Response size: `8`

## **Endpoint: Deletar Sala**

**Descrição** Remove uma sala existente usando seu ID.

**Método** DELETE

**URL** /api/Salas/{id}

---

◆ Requisição

- Headers:

<b>Key</b>	<b>Value</b>
------------	--------------

Authorization Bearer {token válido com role admin}

- Parâmetro de Rota:

<b>Nome</b>	<b>Tipo</b>	<b>Descrição</b>
-------------	-------------	------------------

id int ID da sala que será excluída.

**Exemplo:**

DELETE http://localhost:5289/api/Salas/6

---

◆ Resposta Esperada

<b>Status</b>	<b>Descrição</b>
---------------	------------------

204 No Content Sala deletada com sucesso (nenhum conteúdo retornado).

---

◆ Cenários de Erro

<b>Status</b>	<b>Descrição</b>
---------------	------------------

401 Unauthorized Token não enviado ou inválido.

403 Forbidden Token sem permissão de admin.

404 Not Found Sala não encontrada para o ID informado.

http://localhost:5289/api/sales/6

DELETE http://localhost:5289/api/sales/6

Params Authorization Headers (9) Body Scripts Tests Settings

Headers (0) Hidden

Key Value Description

Authorization Bearer eyJhbGciOiJIUzI1NiJ9.eyJodHRwOi8vZWlhcy54bWxzL2FwLm9yZy93cyByMDA1LzA... Value Description

Key

Body Cookies Headers (2) Test Results

Raw Preview Visualize 1 204 No Content

## Endpoint: Obter Todos os Usuários

- **Método:** GET
  - **URL:** /api/Usuarios
- 

### ◆ Descrição

Retorna uma lista de todos os usuários cadastrados no sistema. Este endpoint é protegido e **somente usuários com papel admin podem acessar.**

---

### ◆ Requisição

#### Headers obrigatórios:

Key	Value
-----	-------

Authorization Bearer *{token JWT válido com permissão admin}*

#### Exemplo:

GET http://localhost:5289/api/Usuarios

---

### ◆ Resposta (200 OK)

Retorna um array JSON com os dados dos usuários:

Campo	Tipo	Descrição
-------	------	-----------

id int ID único do usuário.

nome string Nome completo.

email string E-mail do usuário.

telefone string Número de telefone.

cpf string CPF do usuário.

senhaHash string Hash da senha (não retorna a senha em texto).

role string Papel do usuário (usuario ou admin).

#### Exemplo real de resposta:

json

[Copiar](#)[Editar](#)

```
[  
  {  
    "id": 14,  
    "nome": "Fillipe Gabriel Izidorio Serafim",  
    "email": "fillipe.serafim@email.com",  
    "telefone": "(11) 99999-1111",  
    "cpf": "123.456.789-01",  
    "senhaHash": "$2a$11$...",  
    "role": "admin"  
  },  
  {  
    "id": 15,  
    "nome": "Josenilson Silva Santos",  
    "email": "josenilson@email.com",  
    "telefone": "(11) 98888-2222",  
    "cpf": "987.654.321-00",  
    "senhaHash": "$2a$11$...",  
    "role": "usuario"  
  }  
]
```

---

#### ◆ Respostas de Erro

Status Code	Motivo
-------------	--------

401 Unauthorized Token ausente ou inválido.

403 Forbidden      Token válido mas sem permissão admin para listar os usuários.

http://localhost:5229/api/usuarios

Save ▾

GET http://localhost:5229/api/usuarios

Params Authorization Headers (9) Body Scripts Tests Settings

Headers ↗ 8 Hidden

Key	Value	Description	Bulk Edit
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInRSiwith3C0kpwVC2yeyJodHRwOi8vc2NoZWVtcy54bWxzb2fmlmByZy93cy8yMDA/LzA...	Description	
Key	Value	Description	

Body Cookies Headers (4) Test Results ⓘ

{} JSON ⓘ Preview ⓘ Visualize ⓘ

200 OK 23 ms · 2.07 KB

```
[{"id": 1, "name": "Filipe Gabriel Iriozio Barreto", "email": "filipe-serafim@email.com", "password": "$2a$11$OmVvCZg2779LH8QvF.6998cpRP78jOEPB1WhMa997St1u/nAw.", "role": "admin"}, {"id": 19, "name": "Dossenilson Silva Santos", "email": null}], [{"text": "User successfully registered."}]
```

## POST /api/Usuarios

### → Descrição:

Cria um novo usuário no sistema.

---

### → Requisição:

- **Endpoint:**

bash

CopiarEditar

## POST /api/Usuarios

- **Headers:**

Nome	Valor
------	-------

Authorization Bearer {token JWT}

Content-Type application/json

- **Body:**

json

CopiarEditar

{

```
"nome": "Joana Oliveira",
"email": "joana@example.com",
"telefone": "(11) 98888-7777",
"cpf": "123.456.789-10",
"senhaHash": "senhaSuperSecreta",
"role": "usuario"
```

}

---

### → Resposta (201 Created):

json

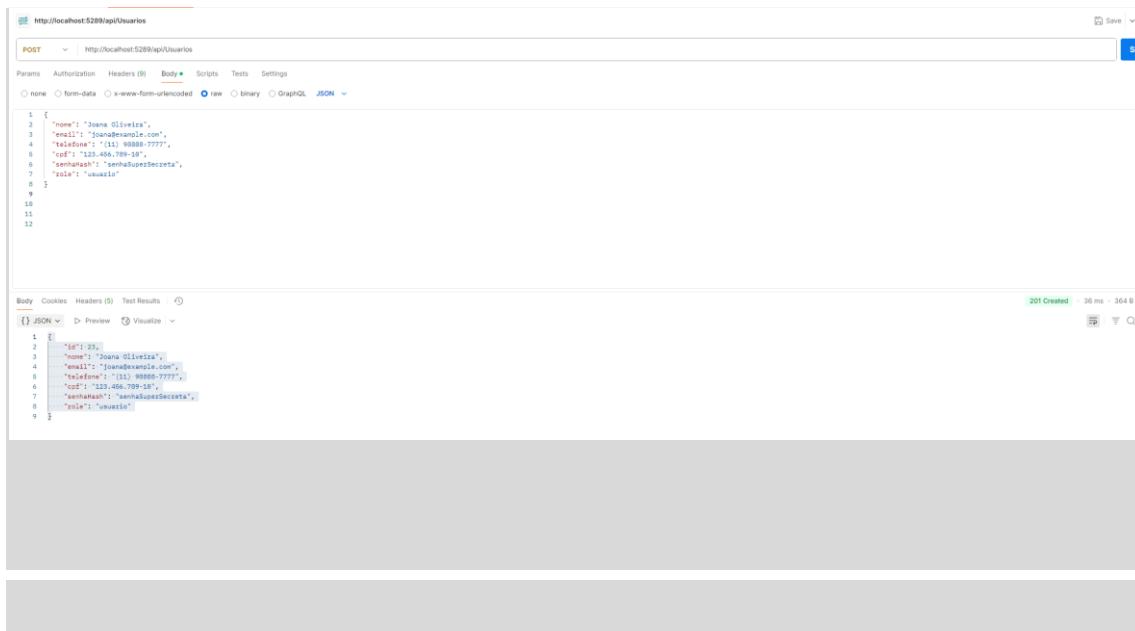
CopiarEditar

```
{
  "id": 23,
  "nome": "Joana Oliveira",
  "email": "joana@example.com",
  "telefone": "(11) 98888-7777",
  "cpf": "123.456.789-10",
  "senhaHash": "senhaSuperSecreta",
  "role": "usuario"
}
```

### → Possíveis erros:

#### Código Descrição

- 400 Falha de validação ou JSON mal formatado
- 401 Unauthorized – token inválido ou ausente
- 500 Erro interno do servidor



The screenshot shows a Postman interface with the following details:

- URL:** http://localhost:5289/api/Users
- Method:** POST
- Body:** JSON (checkbox checked)
- JSON Data:**

```

1  {
2   "nome": "Joana Oliveira",
3   "email": "joana@example.com",
4   "telefone": "(11) 98888-7777",
5   "cpf": "123.456.789-10",
6   "senhaHash": "senhaSuperSecreta",
7   "role": "usuario"
8 }
9
10
11
12

```
- Response:**
  - Status: 201 Created
  - Time: 36 ms
  - Size: 364 B
  - Content-Type: application/json; charset=utf-8
  - Content:

```

1 {
2   "id": 23,
3   "nome": "Joana Oliveira",
4   "email": "joana@example.com",
5   "telefone": "(11) 98888-7777",
6   "cpf": "123.456.789-10",
7   "senhaHash": "senhaSuperSecreta",
8   "role": "usuario"
9 }

```

## GET /api/Usuarios/{id}

### → Descrição:

Este endpoint retorna os dados detalhados de um usuário específico, localizado através do seu id.

---

### → Requisição:

- **Endpoint:**

bash

CopiarEditar

## GET /api/Usuarios/{id}

- **Parâmetros de rota:**

Nome	Tipo	Descrição
------	------	-----------

id	int	ID único do usuário
----	-----	---------------------

- **Headers:**

Nome	Valor
------	-------

Authorization	Bearer {token JWT}
---------------	--------------------

---

### → Exemplo de resposta (200 OK):

json

CopiarEditar

```
{  
  "id": 23,  
  "nome": "Joana Oliveira",  
  "email": "joana@example.com",  
  "telefone": "(11) 98888-7777",  
  "cpf": "123.456.789-10",  
  "senhaHash": "senhaSuperSecreta",  
  "role": "usuario"
```

}

---

## → Possíveis erros:

### Código Descrição

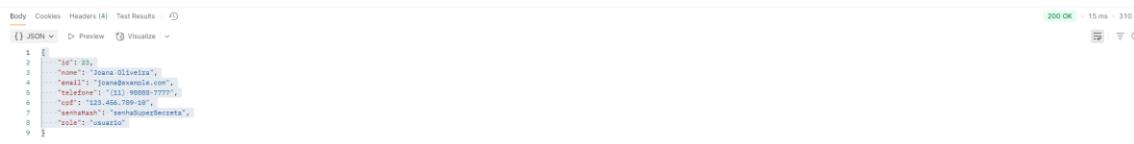
401      Unauthorized – token inválido ou ausente

403      Forbidden – sem permissão para acessar

404      Usuário não encontrado



The screenshot shows the Postman interface with a GET request to `http://localhost:5289/api/Users/23`. The Authorization header is set to a Bearer token: `Bearer eyJhbGciOiJIUzI1NiJ9.R5cCh6kgkVJC9eyJzdWIiOiJvc2NvZWlncy54ZWxzI2FwLm9yZy93cyByMDA1LzA..`. The response status is 200 OK.



The screenshot shows the Postman interface with a successful GET request to `http://localhost:5289/api/Users/23`, returning a JSON response with user details:

```
[{"id": 23, "name": "Joana Oliveira", "email": "joanaoliveira.com", "password": "senhasecreta", "conf": "123.456.789-09", "senhaHash": "senhahusersecreta", "role": "usuário"}]
```

The response status is 200 OK.

## PUT /api/Usuarios/{id}

### → Descrição:

Atualiza os dados de um usuário específico baseado no id informado. O corpo da requisição deve conter todos os dados atualizados.

---

### → Requisição:

- **Endpoint:**

bash

Copiar  
Editar

## PUT /api/Usuarios/{id}

- **Parâmetros de rota:**

### Nome Tipo Descrição

id int ID único do usuário que será atualizado

- **Headers:**

### Nome Valor

Authorization Bearer {token JWT}

- **Body (JSON):**

### Campo Tipo Descrição

id int ID do usuário (mesmo valor da rota)

nome string Nome completo

email string E-mail válido

telefone string Telefone formatado

cpf string CPF do usuário

senhaHash string Hash da senha

role string Papel do usuário (admin ou usuario)

---

→ Exemplo de body (JSON):

```
json  
CopiarEditar  
{  
    "id": 23,  
    "nome": "Joana Oliveira - Atualizada",  
    "email": "joana.atualizada@example.com",  
    "telefone": "(11) 97777-8888",  
    "cpf": "123.456.789-10",  
    "senhaHash": "novaSenhaHashSuperSecreta",  
    "role": "usuario"  
}
```

---

→ Resposta:

- **Código:** 204 No Content
  - **Descrição:** O usuário foi atualizado com sucesso. Nenhum conteúdo é retornado.
- 

→ Possíveis erros:

**Código** **Descrição**

- |     |  |
|-----|--|
| 400 | Bad Request – Corpo inválido ou campos obrigatórios ausentes |
| 401 | Unauthorized – Token ausente ou inválido                     |
| 403 | Forbidden – Permissões insuficientes                         |
| 404 | Not Found – Usuário não encontrado                           |

http://localhost:5289/api/Users/23

PUT http://localhost:5289/api/Users/23

Params Authorization Headers (0) **Body** Scripts Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1
2   "id": 23,
3   "name": "Joana Oliveira - Atualizada",
4   "email": "joana.oliveiradeexample.com",
5   "telefone": "(11) 99999-9999",
6   "cod": "123.456.799-58",
7   "senhahash": "novaSenhaHashSuperSecreta",
8   "role": "usuario"
9 }
10
11
12
13
```

Body Cookies Headers (2) Test Results

Raw Preview Visualize

204 No Content - 300 i

## DELETE /api/Usuarios/{id}

### → Descrição:

Exclui um usuário específico com base no id informado. A operação remove o registro permanentemente do sistema.

---

### → Requisição:

- **Endpoint:**

bash

CopiarEditar

## DELETE /api/Usuarios/{id}

- **Parâmetros de rota:**

Nome	Tipo	Descrição
------	------	-----------

id	int	ID único do usuário a ser excluído
----	-----	------------------------------------

- **Headers:**

Nome	Valor
------	-------

Authorization	Bearer {token JWT}
---------------	--------------------

- **Body:**

*(nenhum body é necessário para essa requisição)*

---

### → Exemplo:

bash

CopiarEditar

## DELETE /api/Usuarios/23

- **Authorization header:**

css

CopiarEditar

Authorization: Bearer {token}

---

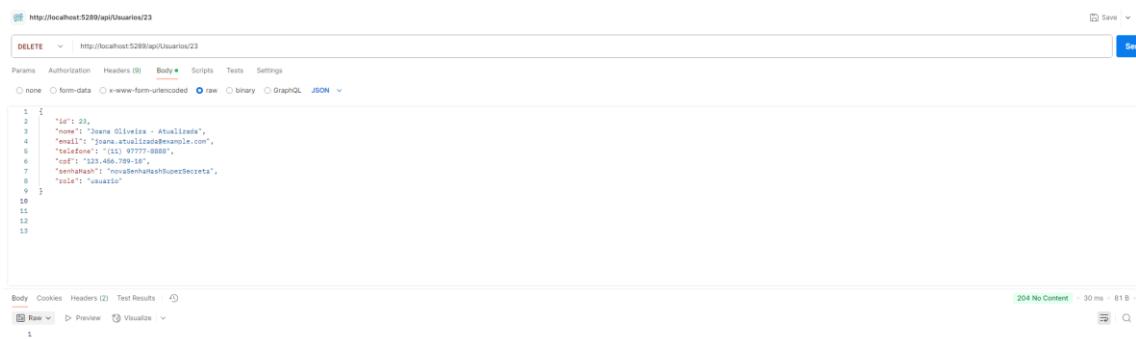
## → Resposta:

- **Código:** 204 No Content
  - **Descrição:** O usuário foi excluído com sucesso. Nenhum conteúdo é retornado.
- 

## → Possíveis erros:

### Código Descrição

- 400 Bad Request – ID inválido ou malformado
- 401 Unauthorized – Token ausente ou inválido
- 403 Forbidden – Permissões insuficientes
- 404 Not Found – Usuário não encontrado



The screenshot shows a Postman request to `http://localhost:5289/api/Users/23` using the `DELETE` method. The request body contains the following JSON payload:

```
1 {  
2     "id": 23,  
3     "name": "Joana Oliveira - Atualizada",  
4     "email": "joana.atualizada@example.com",  
5     "telephone": "97777-8888",  
6     "code": "123_Abc_789",  
7     "senhaHash": "novaSenhaHashBemSecreta",  
8     "role": "usuaria"  
9 }  
10  
11  
12  
13
```

The response status is `204 No Content`, indicating success. The response body is empty.