Documentação de EndPoints API

Esta documentação tem como objetivo deixar mais claros todos os endpoints da API até o momento. Todos estão separados em suas devidas pastas, e são marcados pelos métodos que exercem como – GET, POST, PUT e DELETE (DEL).

1. Preparação de Ambiente Postman....2
1.1. Baixar arquivo json no GitHub....2
1.2. Importar Collection no Postman....2
2. Autenticação....3
2.1. Token....3
2.1.1. Requisição do token....3
2.1.2. Adicionando o token a requisição....4
3. Métodos Login / Cadastrar....6
3.1. POST CadastrarUsuario....6
3.2. POST SolicitarNovoCodigoVerificação....7
3.3. POST VerificaEmail....10
3.4. POST Login....11
3.5. POST OAuth2Google....13
3.6. POST RefreshToken....15
3.7. POST SolicitaçãoParaResetarSenha....15
3.8. POST ConfirmarResetarSenha....17
4. Métodos Usuario....19
4.1. GET BuscarTodosUsuarios....19
4.2. GET BuscarUsuario....19
4.3. GET BuscarInfoUsuarioLogado....20
4.4. PUT AtualizarUsuario....21
4.5. DELETE DeletarUsuario....22
4.6. DELETE DeletarUsuario....23
4.7. PATCH SetAtivoUsuario....24
5. Métodos Servico....26
5.1. GET BuscarTodosServicos....26
5.2. GET BuscarServico....27
6. Métodos Prestação de Serviços....29
6.1. GET BuscarTodosServicos....29
6.2. GET BuscarServicoPeloId....30
6.3. GET BuscarServicoPeloUsuarioId....32
6.4. GET BuscarServicoPeloServiçoNome....33
6.5. POST CadastrarServiçoPrestado....34
6.6. PUT AtualizarServiçoPrestado....35
6.7. PUT DesativarServicoPorId....36
6.8. PUT AtivarServicoPorId....37
6.9. DELETE DeletarServicoPorId....37

1. Preparação de Ambiente Postman

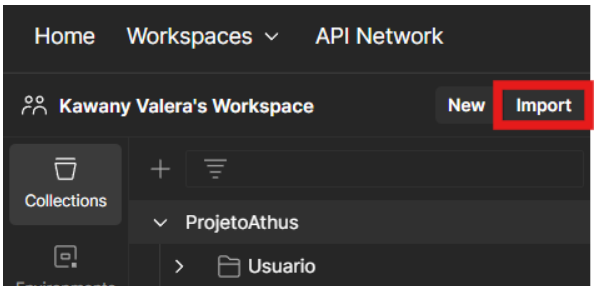
ATENÇÃO: Para usar a API localmente, deve-se seguir os passos da documentação principal no github do projeto https://github.com/felipemariano2511/ProjetoAthus/blob/main/Documents/GuiaRequisitos%26ConfiguracaoAmbiente.pdf, antes de seguir os itens a seguir.

1.1. Baixar arquivo json no GitHub

https://github.com/felipemariano2511/ProjetoAthus/blob/main/Postman/ProjetoAthus.postman_collection.json

1.2. Importar Collection no Postman

Clicar em Import e depois selecionar o arquivo ProjetoAthus.postman_collection.json



2. Autenticação

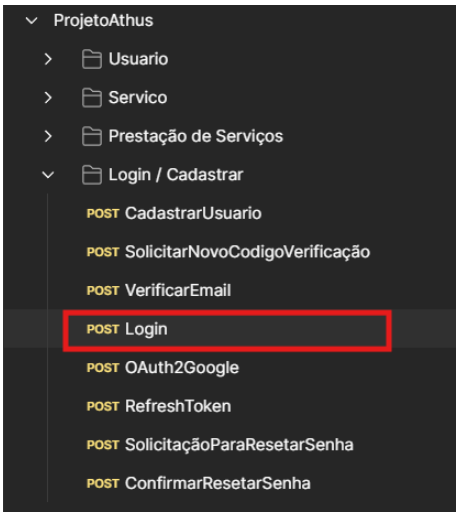
2.1. Token

2.1.1. Requisição do token

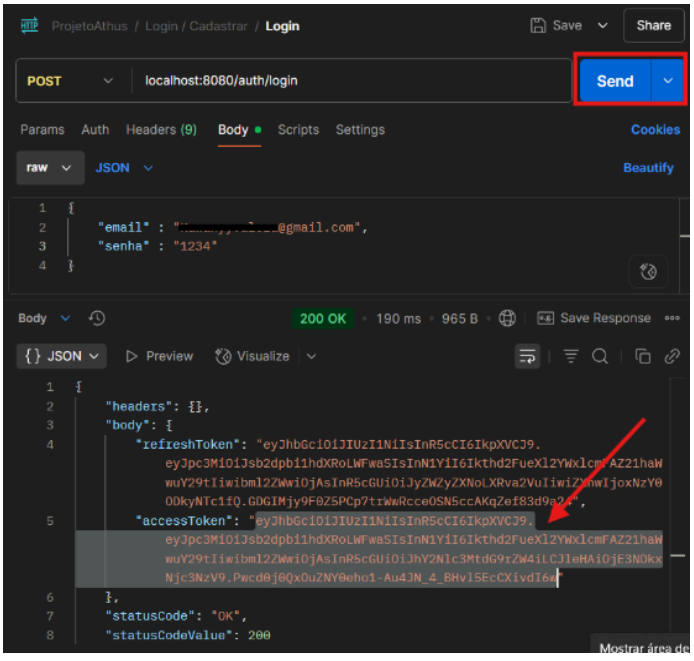
Algumas requisições necessitam de um token.

Caso já tenha usuário cadastrado usando seu e-mail e senha siga os passos abaixo, se ainda não tiver, primeiro siga os passos do item ***3. Métodos Login / Cadastrar*** e depois volte a esse item.

1. No caminho ProjetoAthus - Login / Cadastrar - POST Login:



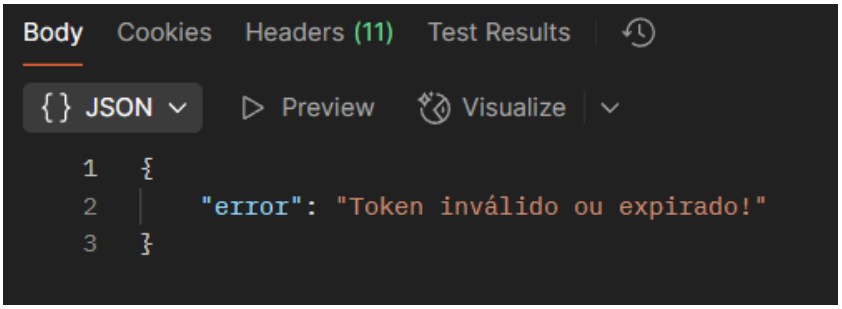
2. Preencher o body com seu e-mail e senha cadastrados, depois clicar em Send para mandar a requisição:



3. Como resposta vai receber o “accessToken”, copiar para utilização.
4. **Atenção:** “refreshToken” acima é o código a ser utilizado no item ***3.5. POST RefreshToken***.

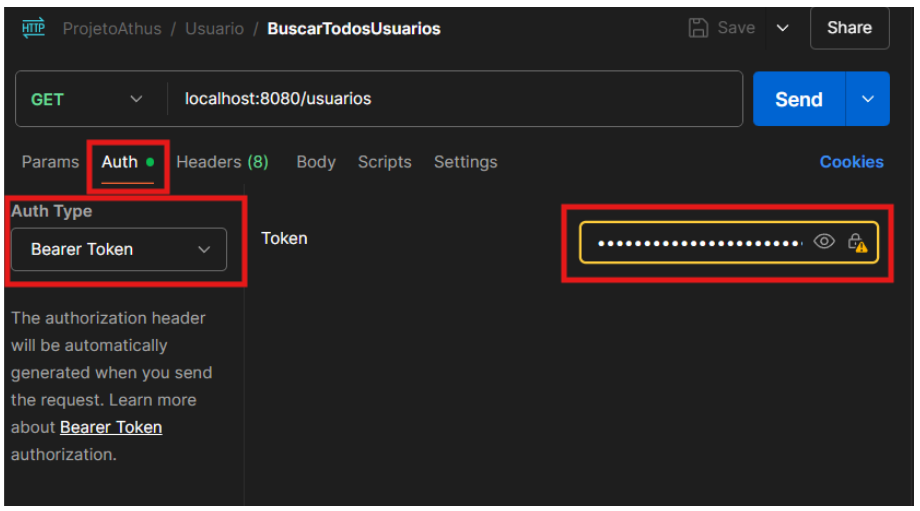
Possíveis respostas:

Quando mandar uma requisição e receber o erro abaixo, significa que precisa atualizar ou adicionar o token, ***lembre-se que ele expira***.



2.1.2. Adicionando o token a requisição

Antes de clicar em Send na requisição que necessita de autenticação (devidamente sinalizadas nos itens de cada uma nessa documentação), deve ir a aba “Authorization”, selecionar o “Auth Type” como “Bearer Token” e colar o token recebido no passo ***2.1.1. Requisição do token*** no campo protegido. Após esses passos pode-se seguir com a utilização da requisição normalmente.



Se ainda assim houver o *“error: Token inválido ou expirado!”*, deverá repetir os passos *2.1.1. Requisição do token* e *2.1.2. Adicionando o token a requisição*, respectivamente. Provavelmente se deve pela expiração do mesmo.

3. Métodos Login / Cadastrar

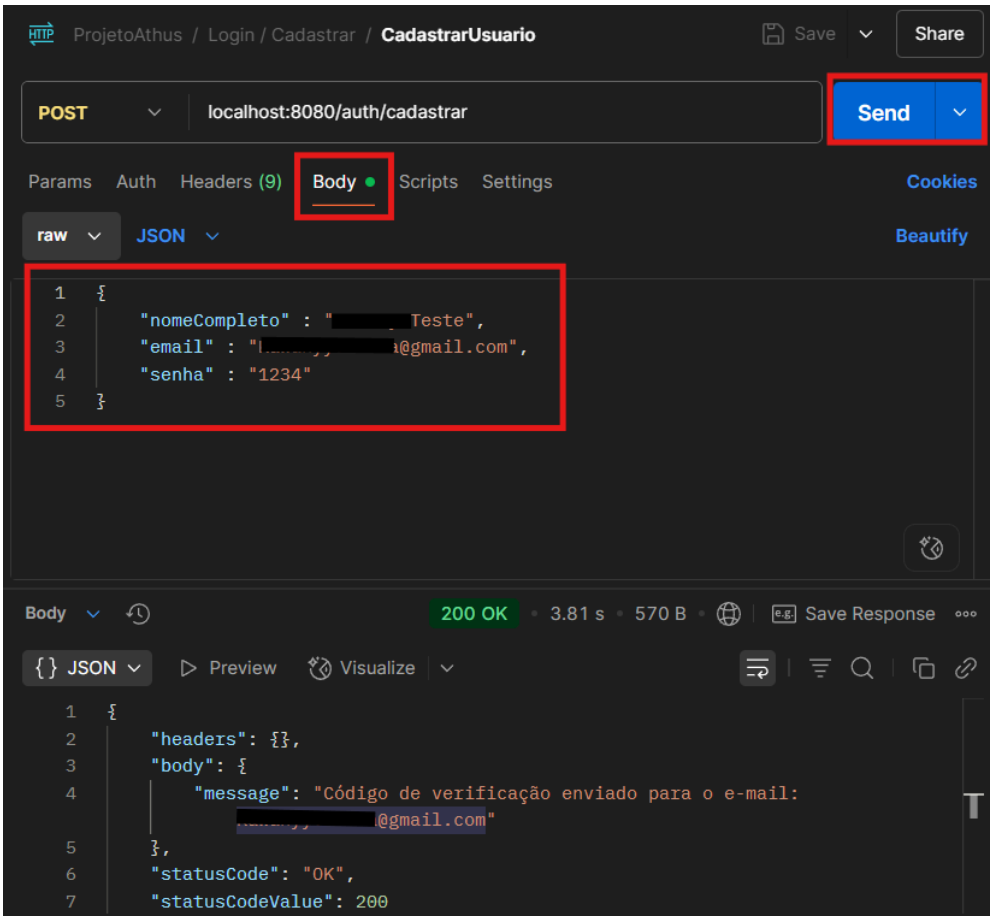
3.1. POST CadastrarUsuario

Descrição: Criar um novo usuário (login e senha).

Body: Na aba “Body”, ajustar json para parâmetros desejados, *nomeCompleto*, *email* e a *senha* se referem aos dados do novo usuário, alterar campos dentro dos parênteses.

Path: auth/cadastrar.

Autenticação: Não requer autenticação.



Exemplo:

{

“nomeCompleto” : “João Ninguém”,

“email” : “joao.ninguem@email.com”,

“senha” : “123456”

}

Email a ser recebido:

Possíveis respostas:

200 OK:

Requisição bem sucedida, um código foi enviado para e-mail fornecido no body para verificação.

```
{
    "headers": {},
    "body": {
            "message": "Código de verificação enviado para o e-mail: **********@gmail.com"
    },
    "statusCode": "OK",
    "statusCodeValue": 200
}
```

409 Conflict:

Requisição com problema, provavelmente conflito de chaves, campos que não podem ser repetidos.

{

"message": "Não é possível cadastrar o usuário, pois o e-mail já está em uso."

}

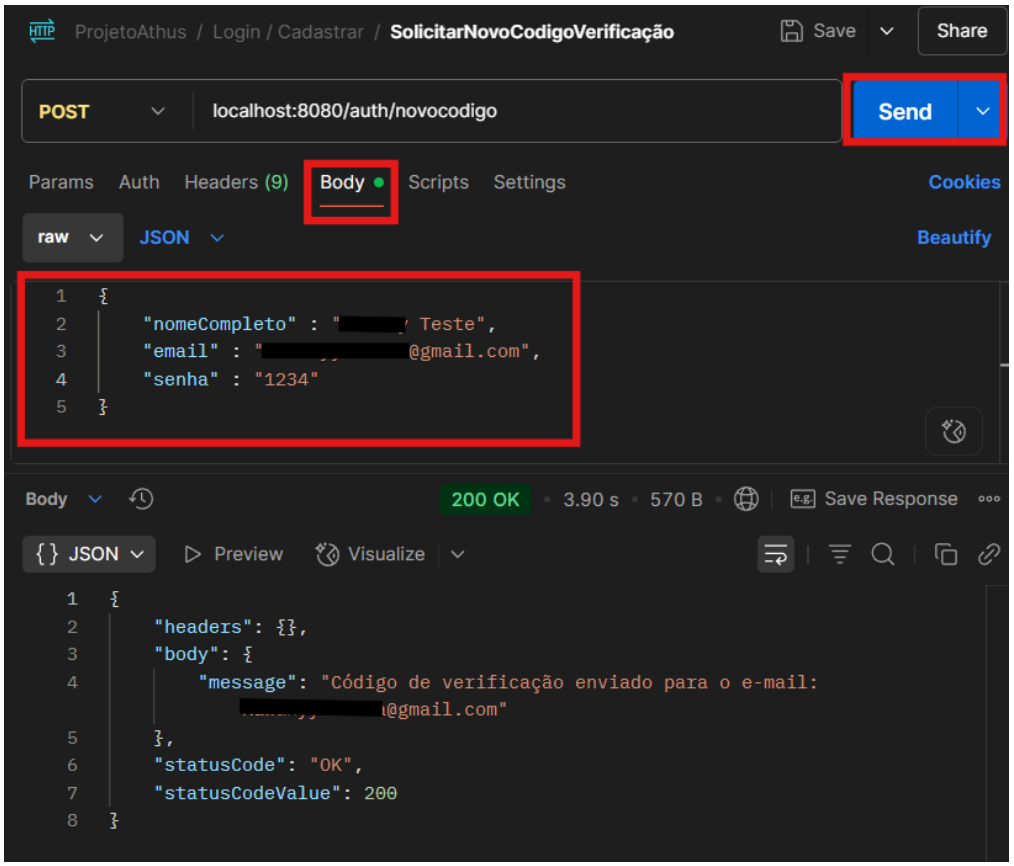
3.2. POST SolicitarNovoCodigoVerificação

Descrição: O código para verificação enviado ao e-mail no item ***3.1. POST CadastrarUsuario***, tem validade, após expirar é necessário solicitar outro caso não tenha sido autenticado o novo usuário ainda.

Body: Na aba “Body”, ajustar json para parâmetros informados no item ***3.1. POST CadastrarUsuario***, *nomeCompleto*, *email* e a *senha*, alterar campos dentro dos parênteses.

Path: auth/novocodigo.

Autenticação: Não requer autenticação.



Exemplo:

{

“nomeCompleto” : “João Ninguém”,

“email” : “joao.ninguem@email.com”,

“senha” : “123456”

}

Email a ser recebido:

Possíveis respostas:

200 OK:

Requisição bem sucedida, um código foi enviado para e-mail fornecido no body para verificação.

```
{
    "headers": {},
    "body": {
            "message": "Código de verificação enviado para o e-mail: ***********@gmail.com"
    },
    "statusCode": "OK",
    "statusCodeValue": 200
}
```

409 Conflict:

Requisição com problema, provavelmente conflito de chaves, campos que não podem ser repetidos.

```
{
        "message": "Não é possível cadastrar o usuário, pois o e-mail já está em uso."
 }
```

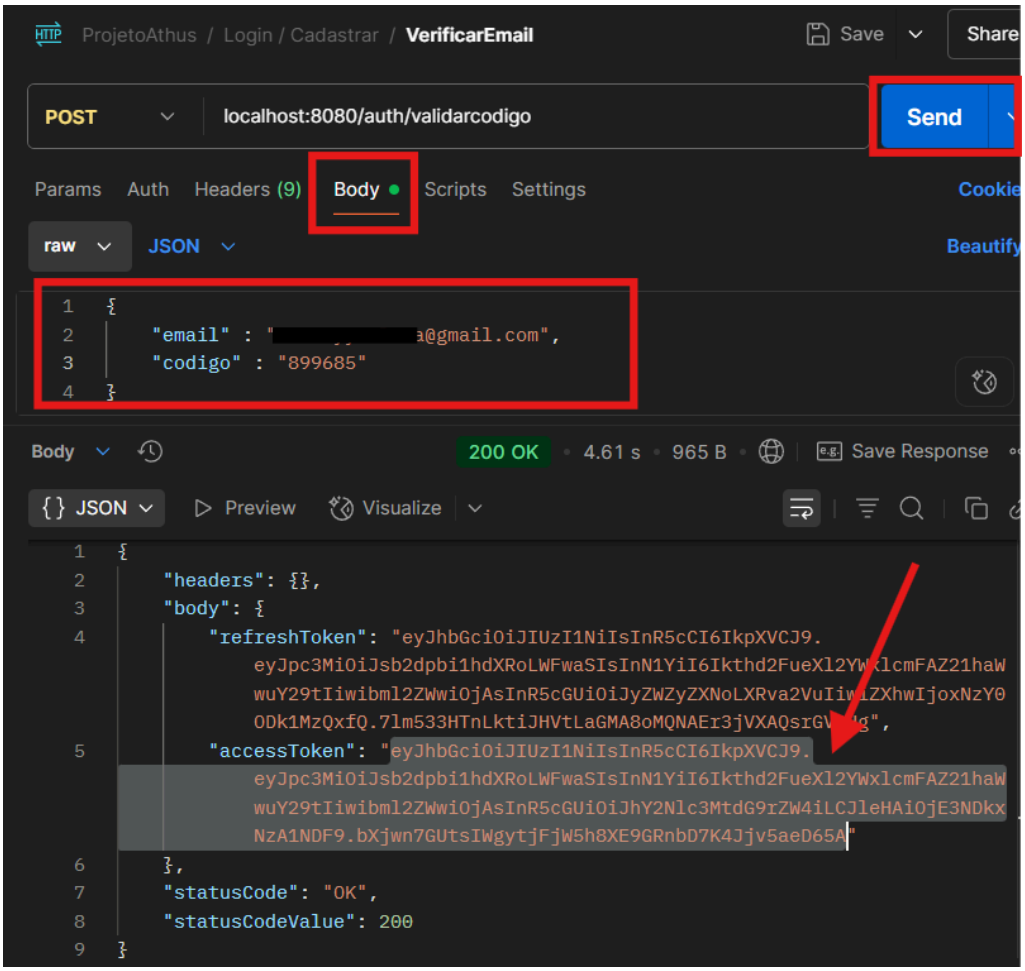
3.3. POST VerificaEmail

Descrição: Valida o código enviado por e-mail, ação realizada pelos itens ***3.1. POST CadastrarUsuario*** ou ***3.2. POST SolicitarNovoCodigoVerificação***.

Body: Na aba “Body”, ajustar json para parâmetro *email* o e-mail cadastrado que recebeu o código de verificação e *codigo* o código de verificação recebido.

Path: auth/validarcodigo.

Autenticação: Não requer autenticação.



Exemplo de Body:

```
{
        "email": "josesoaresgalvao21@gmail.com",
        "codigo": "113398"
}
```

Possíveis respostas:

200 OK:

Requisição bem sucedida:


```
{
    "headers": {},
    "body": {
        "refreshToken": "******",
        "accessToken": "******"
    },
    "statusCode": "OK",
    "statusCodeValue": 200
}
```


422 Unprocessable Entity:

Nenhum código foi gerado para esse e-mail:

400 Bad Request:

Código inválido. Tentativas restantes: 4

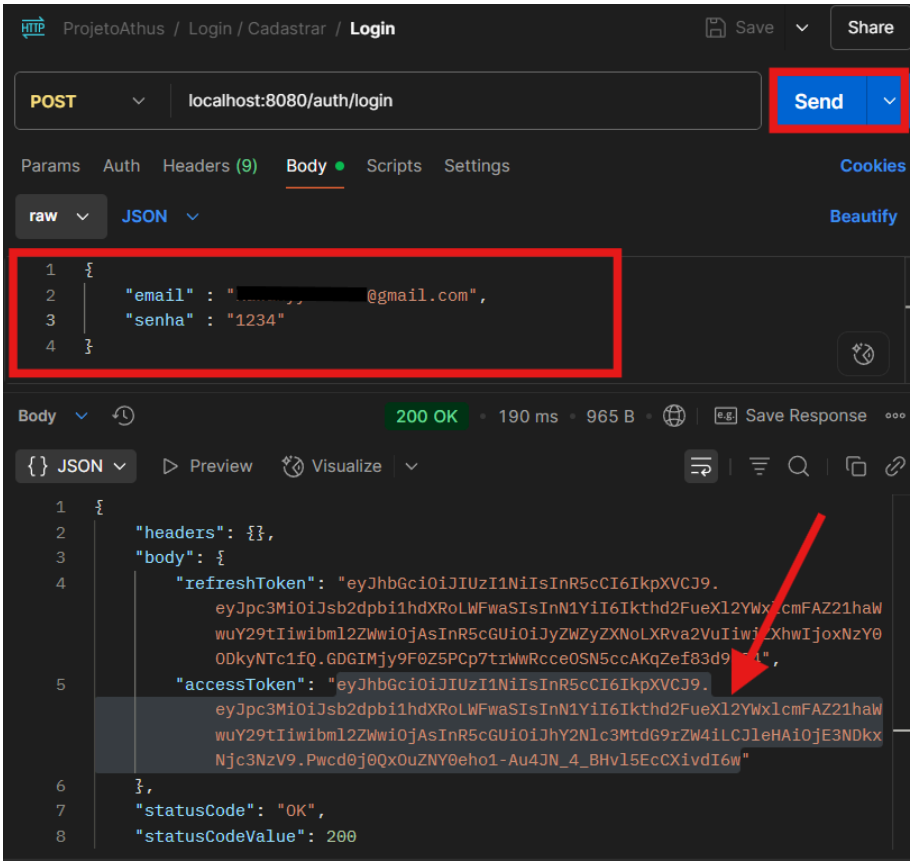
3.4. POST Login

Descrição: Realiza o login no sistema.

Body: Preencher *email* e *senha* válidos de usuários ativos para logar e receber o access token. Se não tiver cadastro no sistema seguir passos ***3.1. POST CadastrarUsuario*** e ***3.3. POST VerificaEmail*** para criar um usuário.

Path: auth/login.

Autenticação: Não requer autenticação.



Exemplo de Body:

```
{
    "email": "josesoaresgalvao21@gmail.com",
    "senha": "123"
}
```

Possíveis respostas:

200 OK:

Requisição bem sucedida:

```
{
    "headers": {},
    "body": {
        "refreshToken": "******",
        "accessToken": "******"
    },
    "statusCode": "OK",
    "statusCodeValue": 200
}
```

400 Bad Request:

Senha incorreta!

3.5. POST OAuth2Google

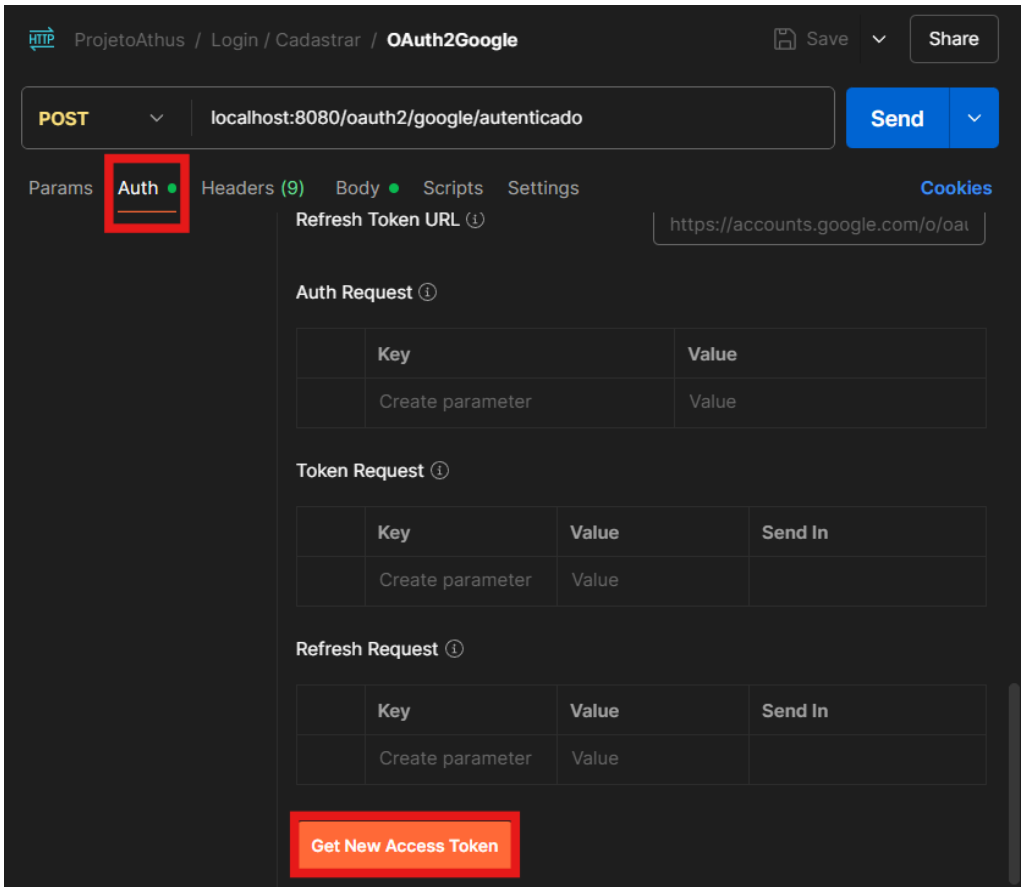
Descrição: Autenticação do Google.

Path: oauth2/google/autenticado.

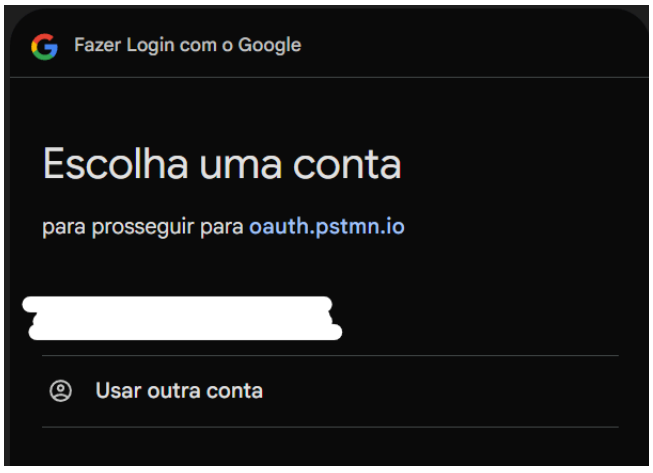
Autenticação: Requer Access Token do Google.

Passo a Passo:

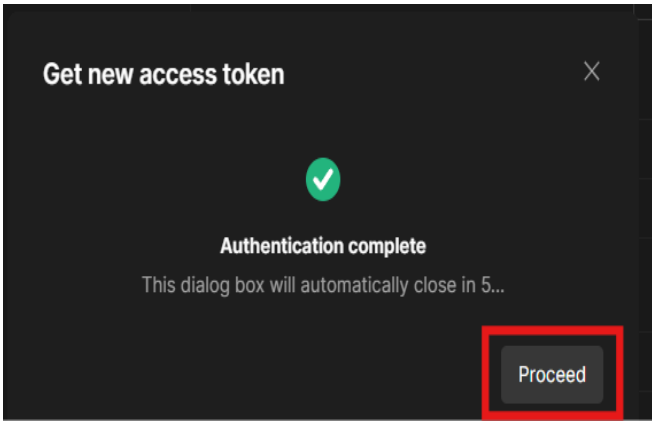
1. Entre em Authorization e no final desta página clique em Get New Access Token:



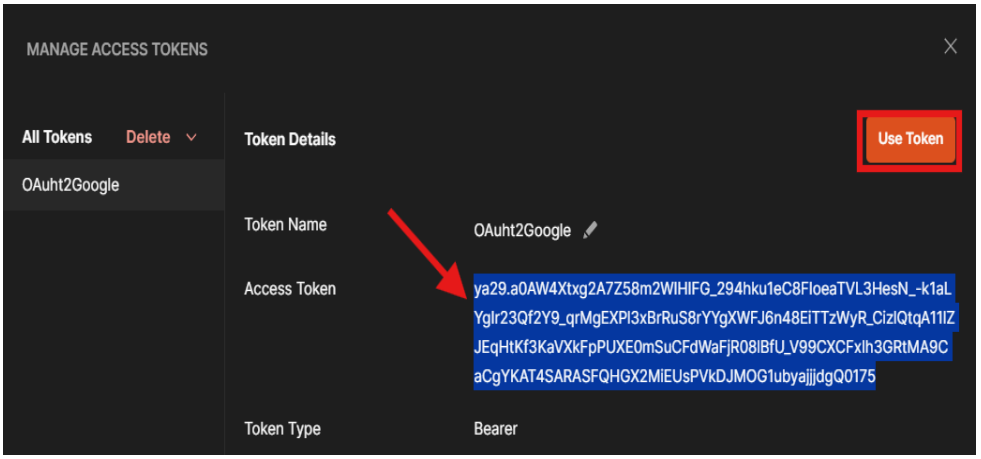
2. Escolher conta pela qual vai ser gerada o token:



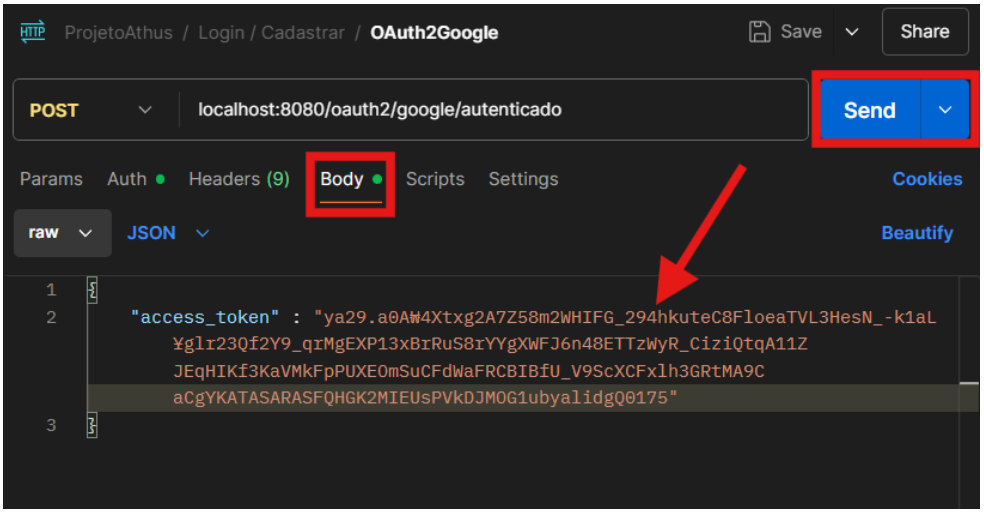
3. Após autenticação (importante: estar logado no postman).



4. Copiar Access Token e clicar em Use Token:



5. Por fim na aba Body, colar o Access Token e enviar a requisição.



Possíveis respostas:

200 OK:

Requisição bem sucedida.

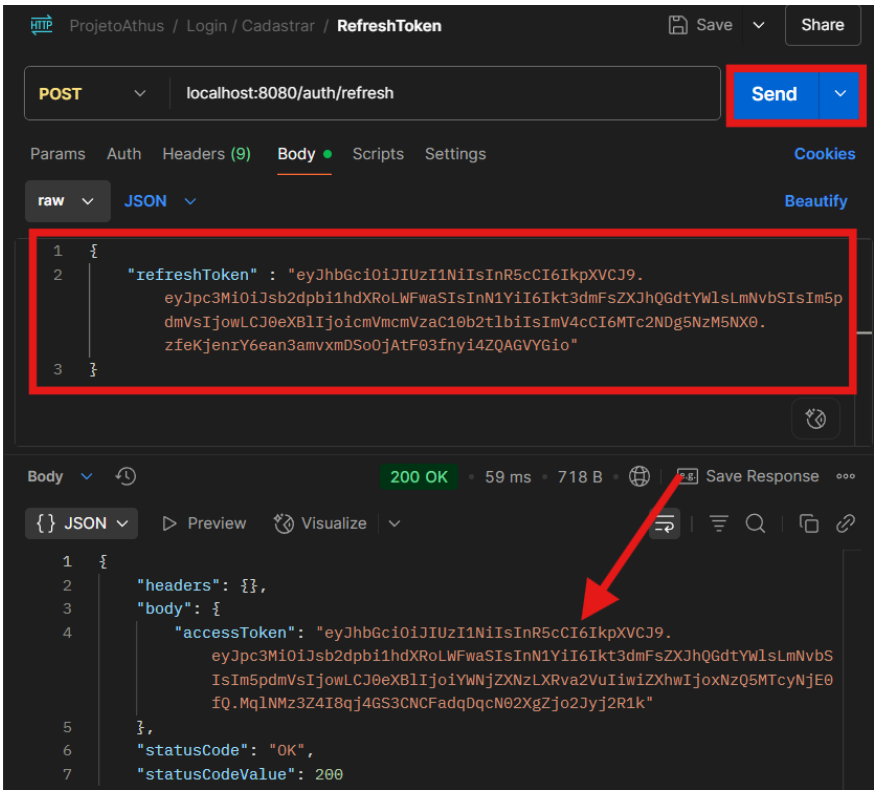
3.6. POST RefreshToken

Descrição: Atualizar Bearer Token, sem precisar logar novamente.

Body: Preencher com “refreshToken” que foi salvo durante o Login.

Path: auth/refresh.

Autenticação: Não requer atuenticação.



Possíveis respostas:

200 OK:

Será recebido um novo accessToken.

```
{
    "headers": {},
    "body": {
        "accessToken": "******"
    },
    "statusCode": "OK",
    "statusCodeValue": 200
}
```

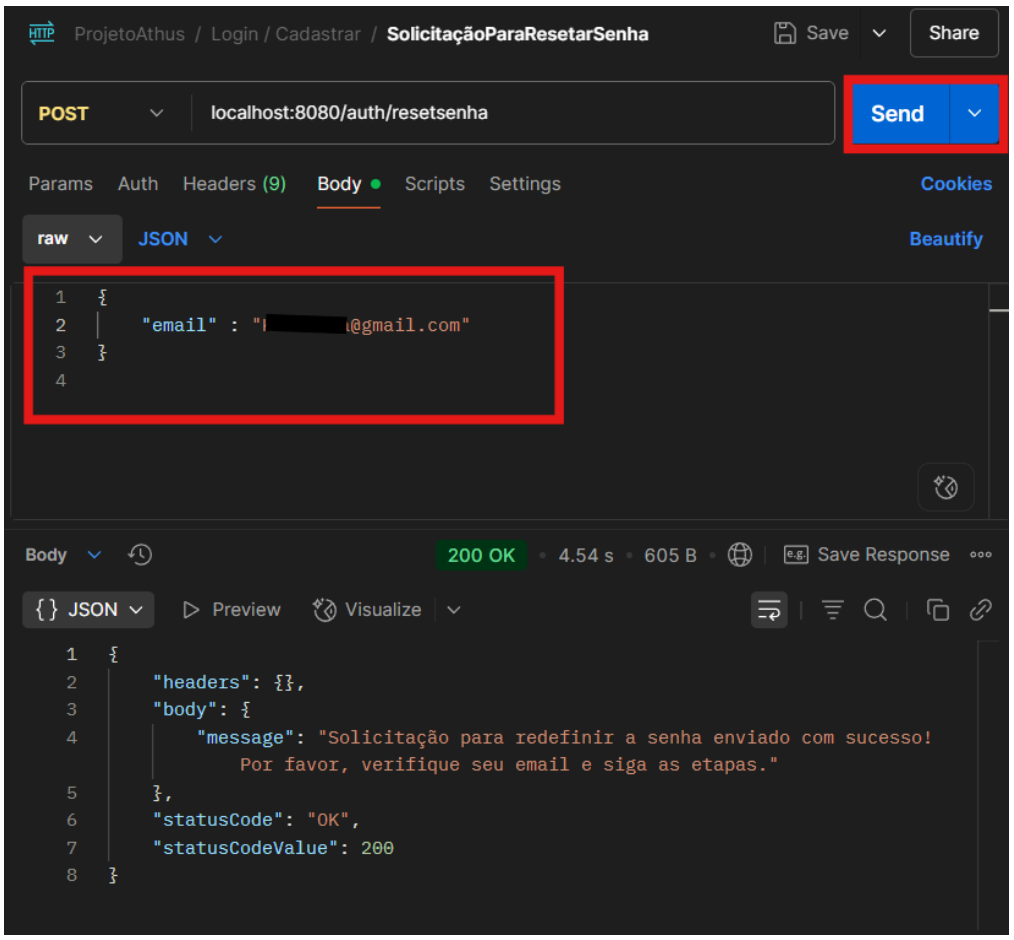
3.7. POST SolicitaçãoParaResetarSenha

Descrição: Enviar uma solicitação para resetar a senha do usuário.

Body: Preencher com “email” com e-mail do usuário que deseja resetar.

Path: auth/resetsenha.

Autenticação: Não requer atuenticação.



E-mail esperado:

Possíveis respostas:

200 OK:

Requisição bem sucedida.


```
{
    "headers": {},
    "body": {
          "message": "Solicitação para redefinir a senha enviado com sucesso! Por favor, verifique seu email e siga as etapas."
    },
    "statusCode": "OK",
    "statusCodeValue": 200
}
```


3.8. POST ConfirmarResetarSenha

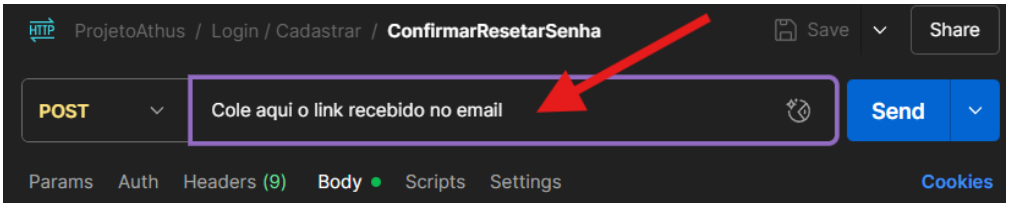
Descrição: Resetar senha pela API ao invés de acessar a página do link enviado no e-mail do item ***3.7. SolicitaçãoParaResetarSenha***.

Body: Preencher em “novaSenha” a senha nova que deseja utilizar.

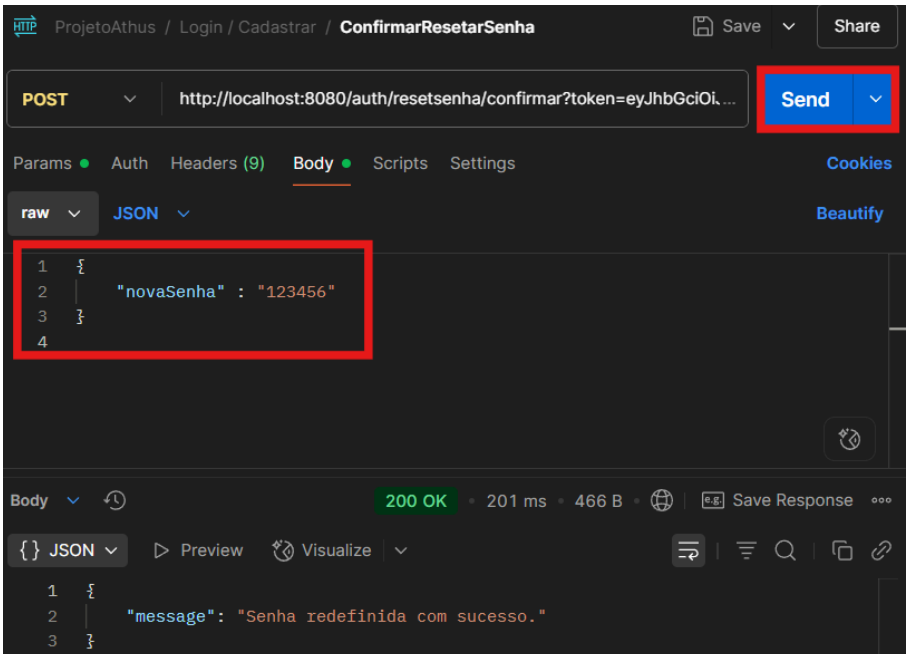
Path: *o link que foi enviado no e-mail após realizar o item ***3.7. SolicitaçãoParaResetarSenha***.

Autenticação: Não requer atuenticação.

1. Preencher no local indicado o link que foi enviado no e-mail após realizar o item ***3.7. SolicitaçãoParaResetarSenha***.



2. Preencher senha nova no body e clicar em Send.



Possíveis respostas:

200 OK:

Requisição bem sucedida.

```
{
    "message": "Senha redefinida com sucesso."

}
```

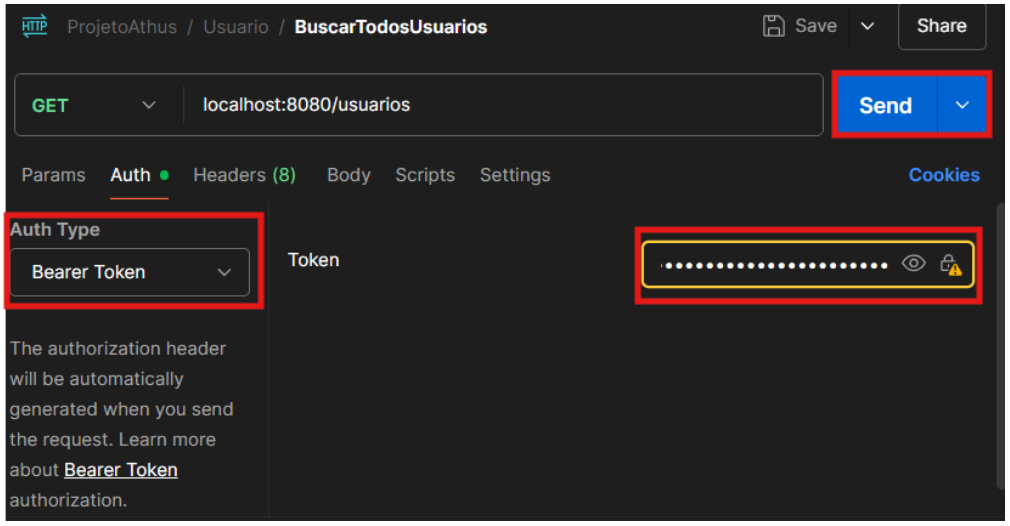
4. Métodos Usuario

4.1. GET BuscarTodosUsuarios

Descrição: Lista todos os usuários.

Path: usuarios.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Possíveis respostas:

200 OK:

Requisição bem sucedida:

```
[{
        "id": 1,
        "nome": "José",
        "email": "josesoaresgalvao21@gmail.com",
        ...
 }]
```

403 Forbidden:

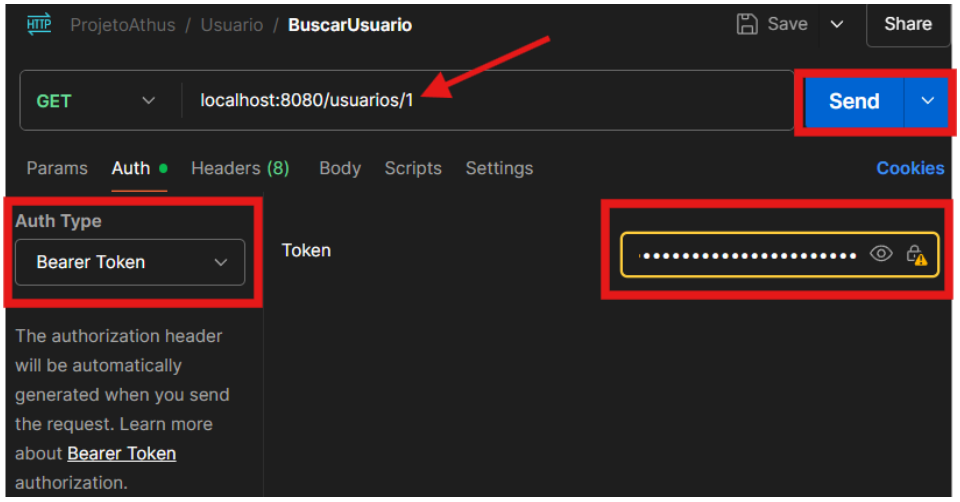
Acesso negado.

4.2. GET BuscarUsuario

Descrição: Retorna dados de um usuário por ID.

Path: usuarios/{id}.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID do usuário (alterar no {id} pelo ID do usuário desejado).

Possíveis respostas:

200 OK:

Usuário encontrado, requisição bem sucedida.

403 Forbidden:

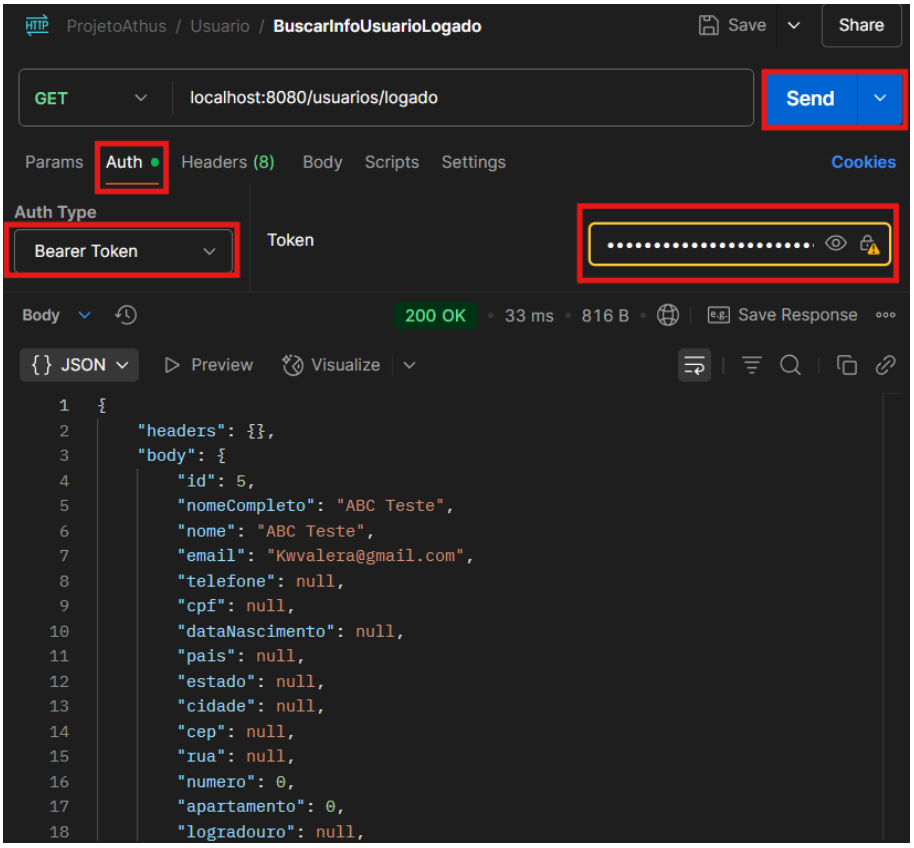
Acesso negado.

4.3. GET BuscarInfoUsuarioLogado

Descrição: Retorna informações cadastradas do usuário logado.

Path: usuarios/logado.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Possíveis respostas:

200 OK:

Requisição bem sucedida.

403 Forbidden:

Acesso negado.

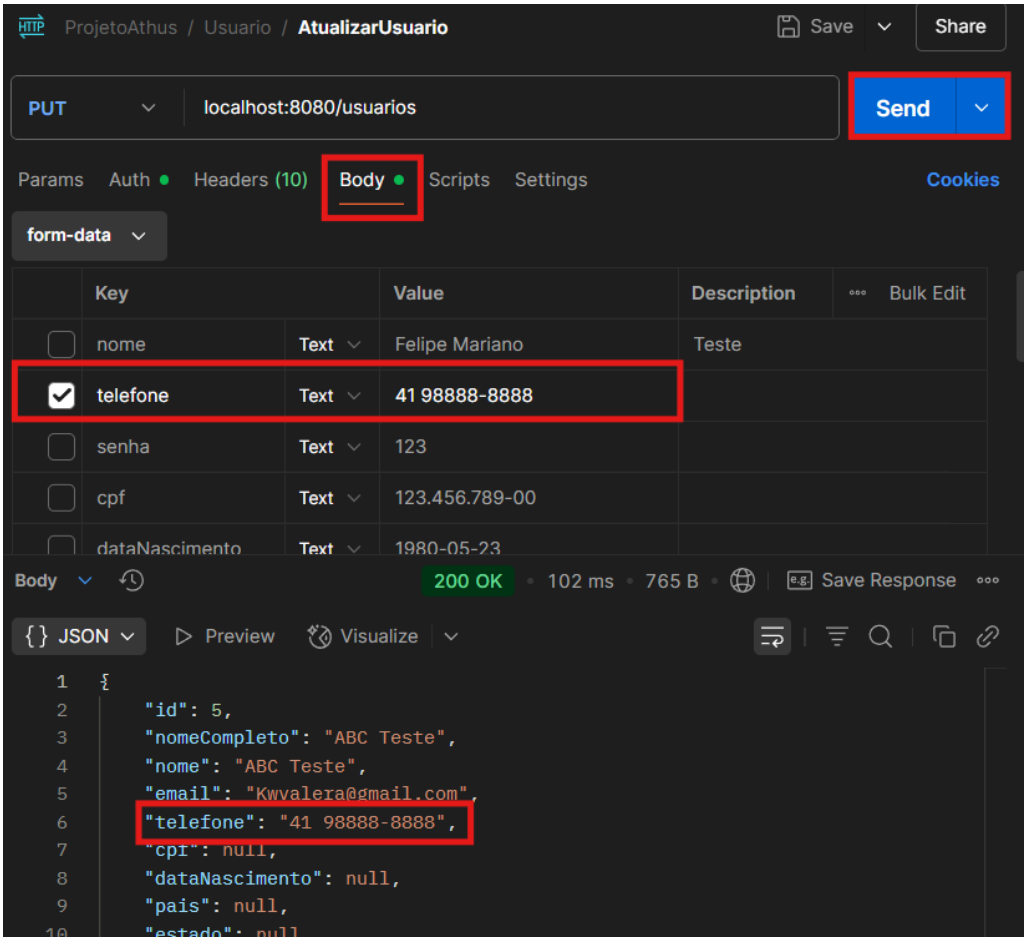
4.4. PUT AtualizarUsuario

Descrição: Atualiza dados do usuário.

Body: O body possui as informações do cadastro do usuário, mandar informações que deseja modificar. Somente é possível mudar informações do seu próprio usuário (que está logado). **Atenção aos campos obrigatórios, sempre os enviar junto.**

Path: usuarios.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Possíveis respostas:

200 OK:

Informações do usuário atualizadas com sucesso.

403 Forbidden:

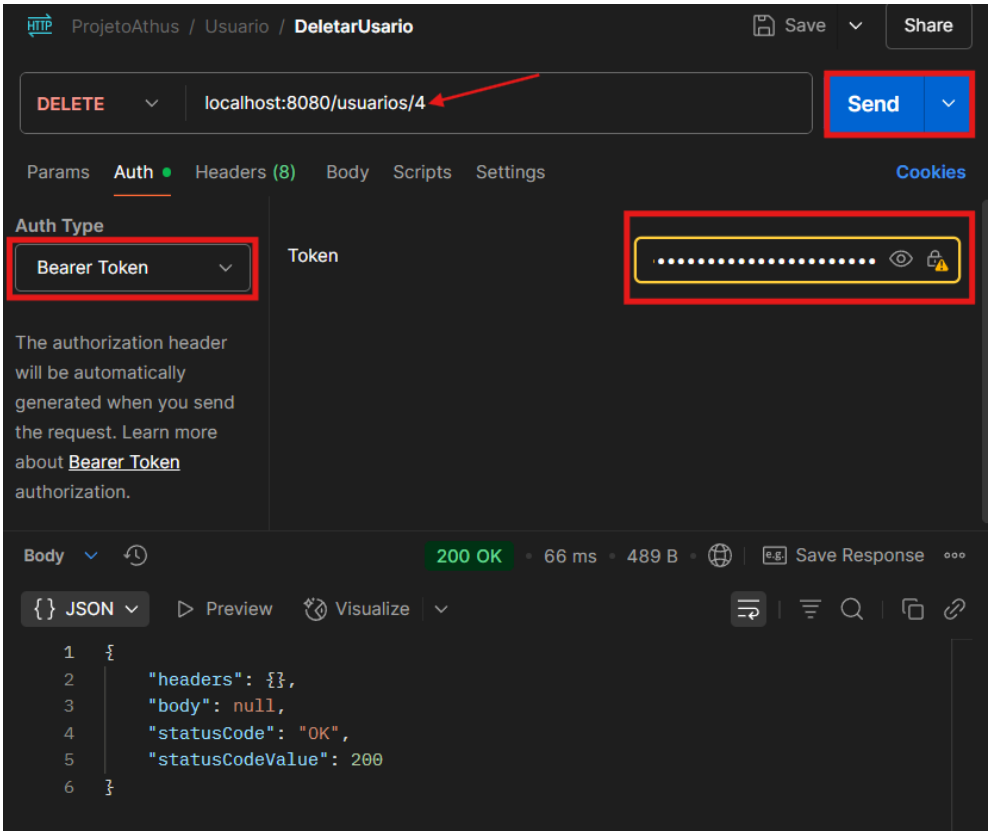
Acesso negado.

4.5. DELETE DeletarUsuario

Descrição: Deletar usuário informado por ID.

Path: usuarios/{id}.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item *2.1 Token*.



Parâmetros:

- id - ID do usuário (alterar no {id} pelo ID do usuário desejado).

Possíveis respostas:

200 OK:

Usuário deletado com sucesso, requisição bem sucedida.

```
{
    "headers": {},
    "body": null,
    "statusCode": "OK",
    "statusCodeValue": 200
}
```

403 Forbidden:

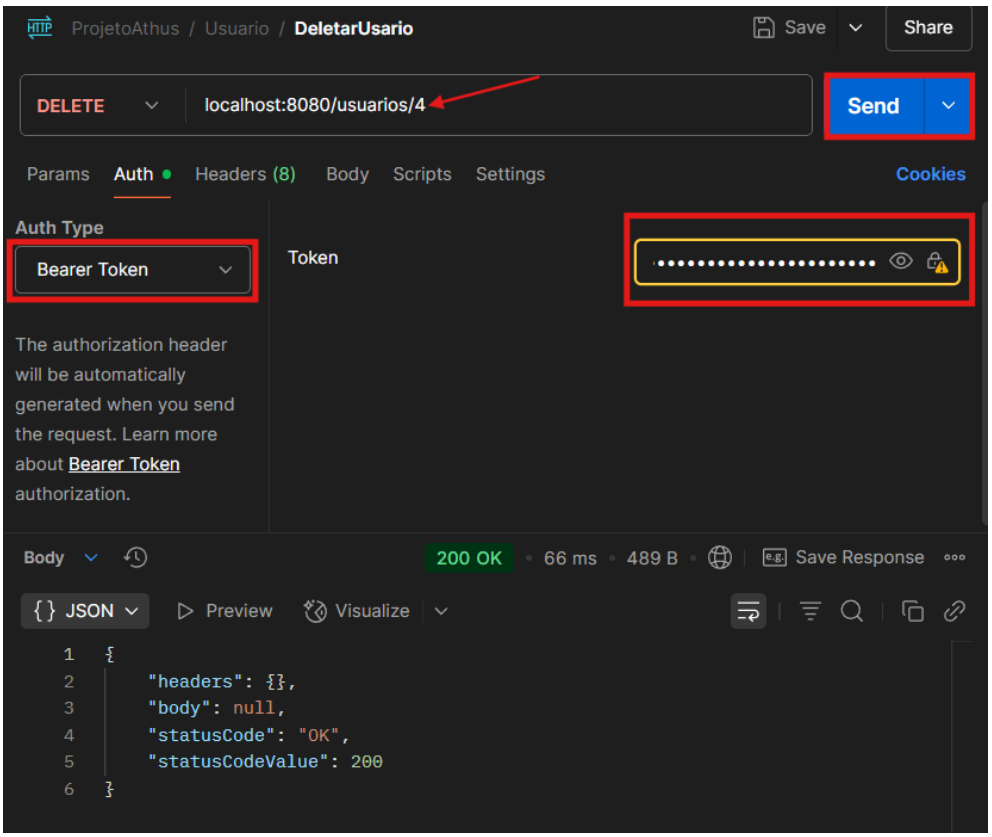
Acesso negado.

4.6. DELETE DeletarUsuario

Descrição: Deletar usuário informado por ID.

Path: usuarios/{id}.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID do usuário (alterar no {id} pelo ID do usuário desejado).

Possíveis respostas:

200 OK:

Usuário deletado com sucesso, requisição bem sucedida.

```
{
    "headers": {},
    "body": null,
    "statusCode": "OK",
    "statusCodeValue": 200
}
```

403 Forbidden:

Acesso negado

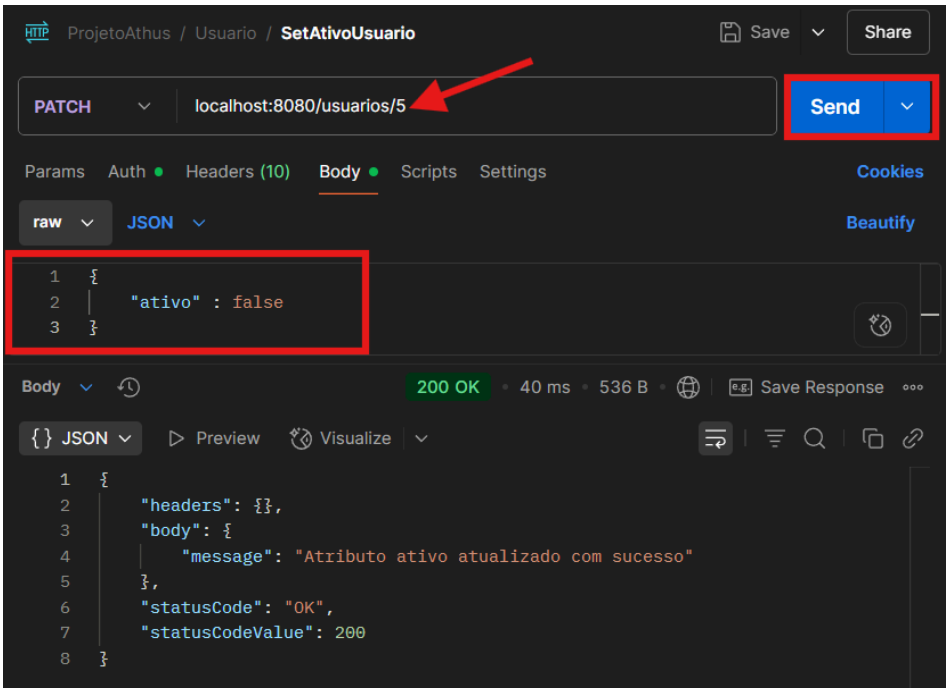
4.7. PATCH SetAtivoUsuario

Descrição: Ativa e desativa usuários.

Body: setar *false* para desativar o usuário escolhido e *true* para ativar o mesmo.

Path: usuarios/{id}.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID do usuário (alterar no {id} pelo ID do usuário desejado).

Possíveis respostas:

200 OK:

Usuário deletado com sucesso, requisição bem sucedida.

```
{
    "headers": {},
    "body": null,
    "statusCode": "OK",
    "statusCodeValue": 200
}
```

403 Forbidden:

Acesso negado.

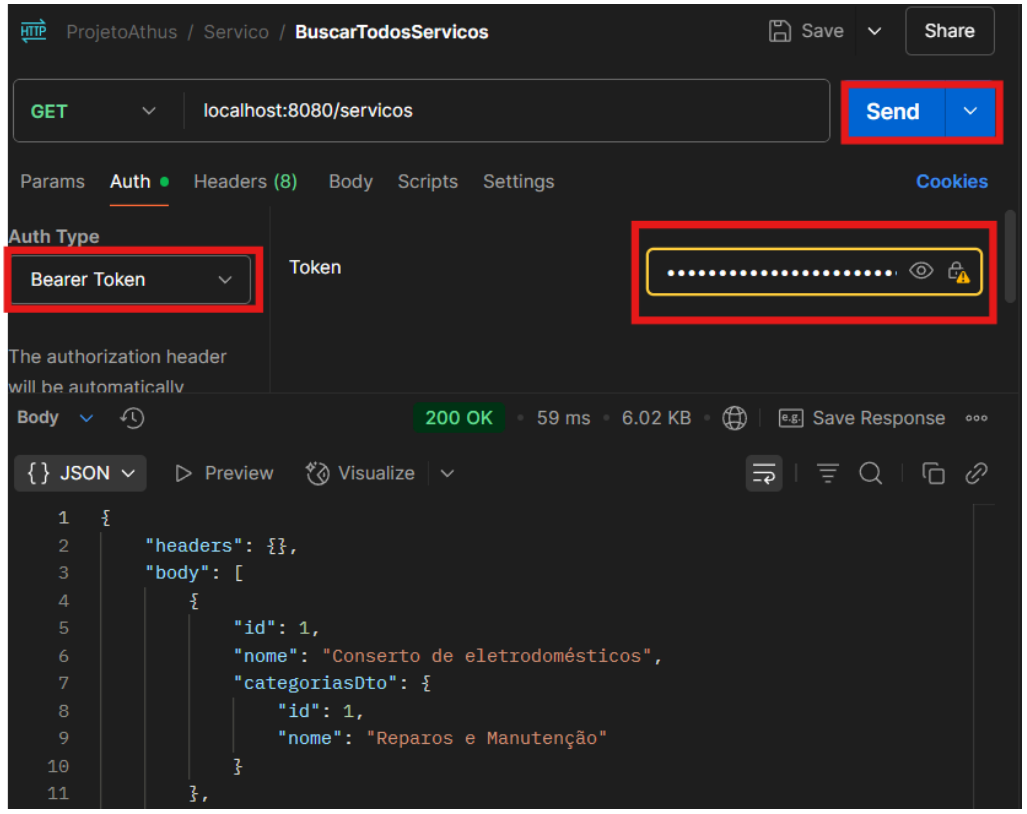
5. Métodos Servico

5.1. GET BuscarTodosServicos

Descrição: Lista todos os serviços cadastrados.

Path: servicos.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Possíveis respostas:

200 OK:

Requisição bem sucedida:

[{

"id": 1,

"nome": "Conserto de eletrodomésticos",

"categoriasDto": ...

}]

403 Forbidden:

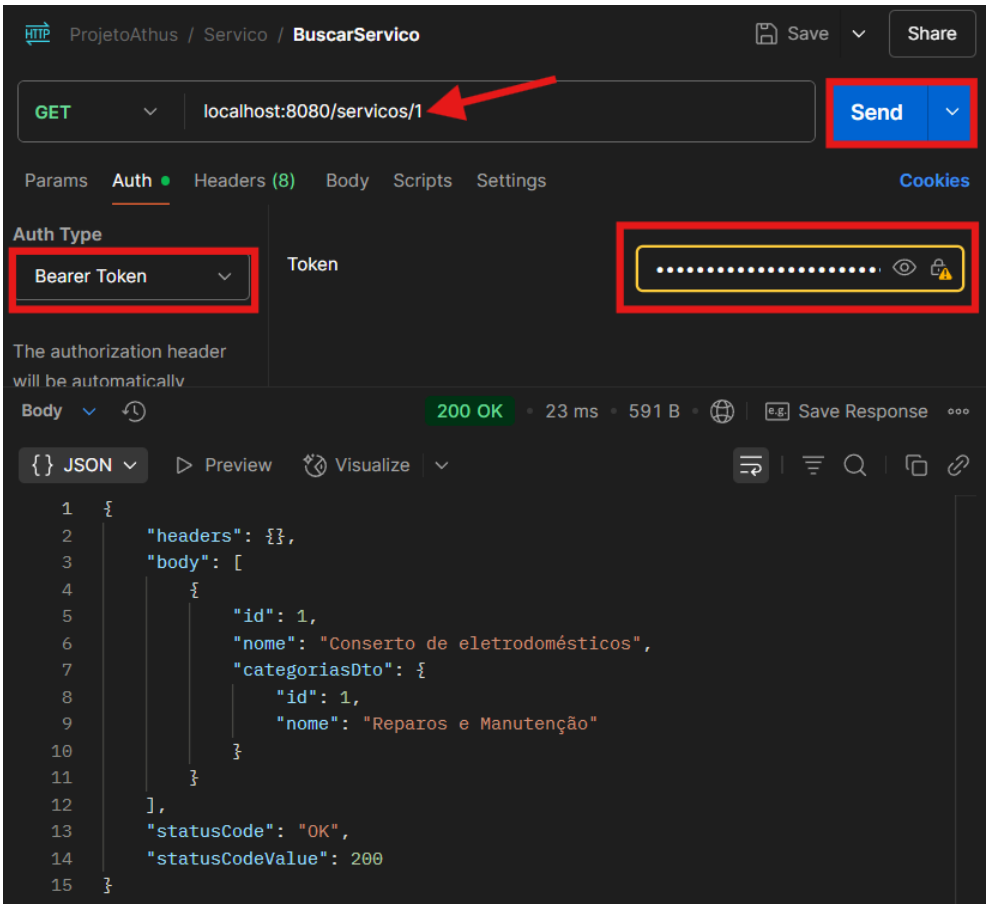
Acesso negado.

5.2. GET BuscarServico

Descrição: Busca serviço por ID.

Path: servicos/{id}.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID do usuário (alterar no {id} pelo ID do usuário desejado).

Possíveis respostas:

200 OK:

Requisição bem sucedida, retorna dados do serviço solicitado.

```
{
    "headers": {},
    "body": [
        {
            "id": 1,
            "nome": "Conserto de eletrodomésticos",
            "categoriasDto": {
```

```
                "id": 1,
                "nome": "Reparos e Manutenção"
            }
        }
    ],
    "statusCode": "OK",
    "statusCodeValue": 200
}
```


403 Forbidden:

Acesso negado.

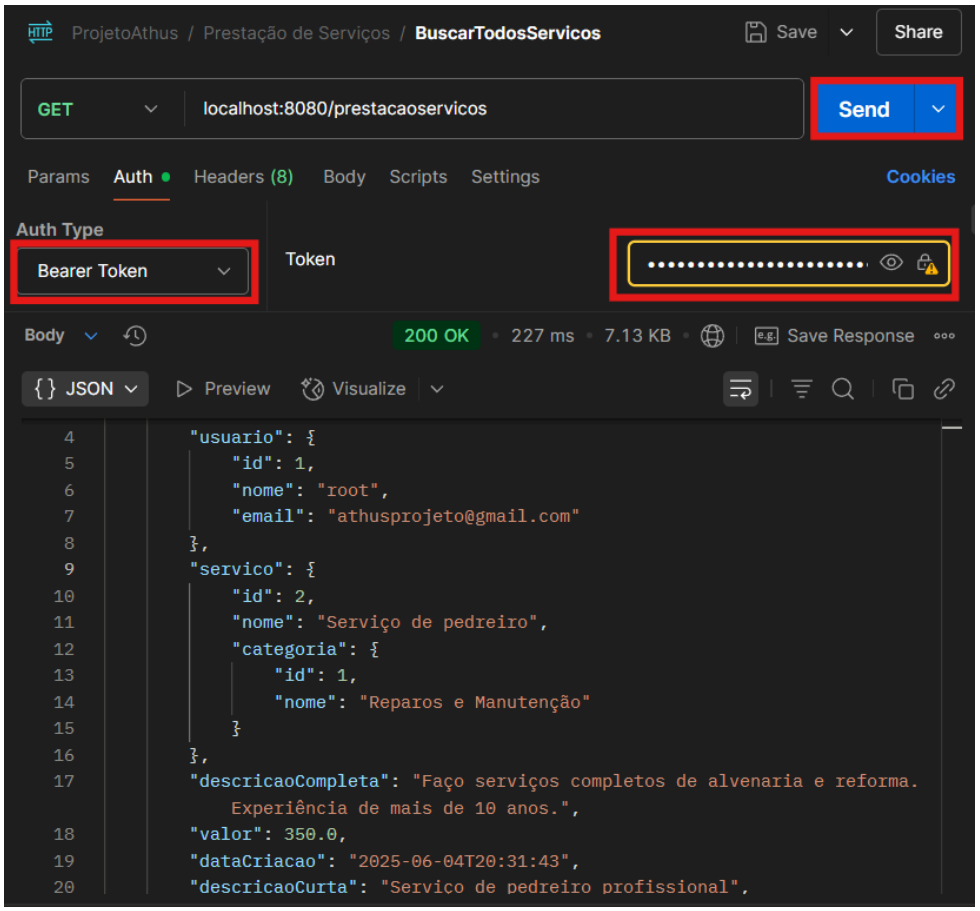
6. Métodos Prestação de Serviços

6.1. GET BuscarTodosServicos

Descrição: Retorna todos os serviços que estão sendo prestados e cadastrados no sistema, separados por ID e usuário.

Path: prestacaoservicos

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Possíveis respostas:

200 OK:

Requisição bem sucedida:

```
[
    {
        "id": 1,
        "usuario": {
            "id": 1,
            "nome": "root",
            "email": "athusprojeto@gmail.com"
        },
```

```
        "servico": {
            "id": 2,
            "nome": "Serviço de pedreiro",
            "categoria": {
                "id": 1,
                "nome": "Reparos e Manutenção"
            }
        },
         "descricaoCompleta": "Faço serviços completos de alvenaria e
reforma. Experiência de mais de 10 anos.",
        "valor": 350.0,
        "dataCriacao": "2025-06-04T20:31:43",
        "descricaoCurta": "Serviço de pedreiro profissional",
        "imagens": [
            "[\"storage/imagens/anuncios/pedreiro1.jpg\"",
            " \"storage/imagens/anuncios/pedreiro2.jpg\"]"
        ],
        "ativo": true
    },
```


403 Forbidden:

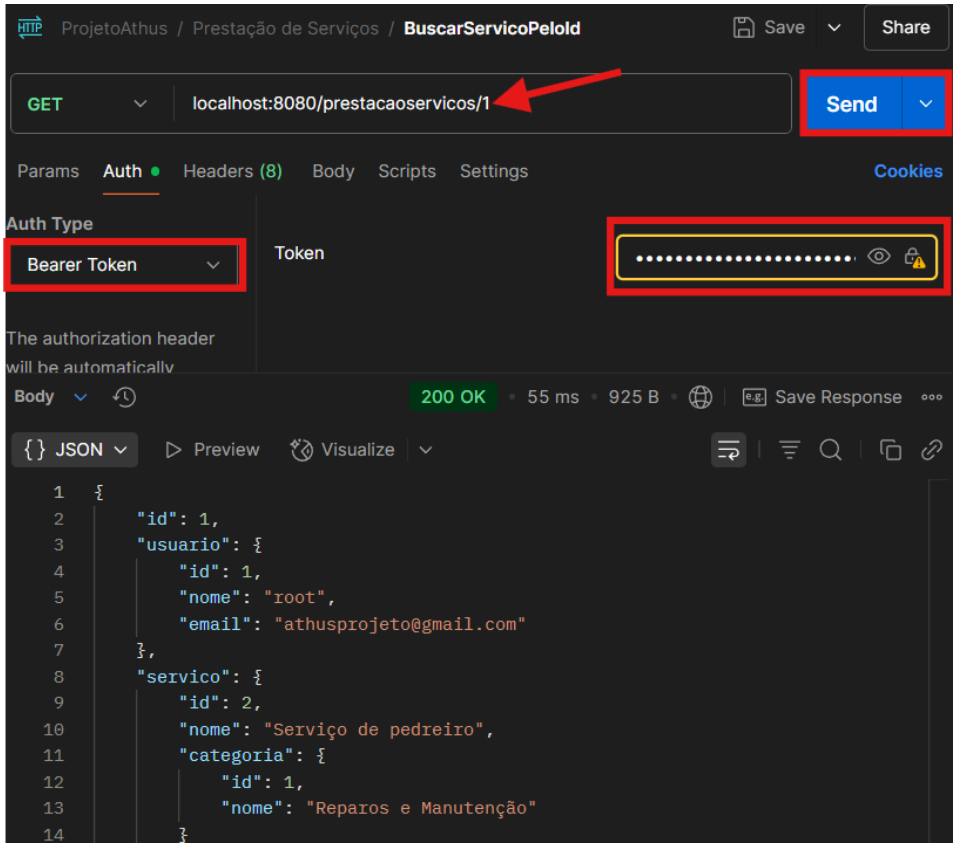
Acesso negado

6.2. GET BuscarServicoPeloId

Descrição: Retorna todos os serviços de determinada prestação de serviço por ID da mesma.

Path: prestacaoservicos/{id}

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID da prestação de serviço desejado (alterar no {id} pelo ID desejado).

Possíveis respostas:

200 OK:

Requisição bem sucedida:

```
{
    "id": 1,
    "usuario": {
        "id": 1,
        "nome": "root",
        "email": "athusprojeto@gmail.com"
    },
    "servico": {
        "id": 2,
        "nome": "Serviço de pedreiro",
        "categoria": {
            "id": 1,
            "nome": "Reparos e Manutenção"
        }
    },
```

```
        "descricaoCompleta": "Faço serviços completos de alvenaria e
reforma. Experiência de mais de 10 anos.",
    "valor": 350.0,
    "dataCriacao": "2025-06-04T20:31:43",
    "descricaoCurta": "Serviço de pedreiro profissional",
    "imagens": [
        "[\"storage/imagens/anuncios/pedreiro1.jpg\"",
        " \"storage/imagens/anuncios/pedreiro2.jpg\"]"
    ],
    "ativo": true
}
```


403 Forbidden:

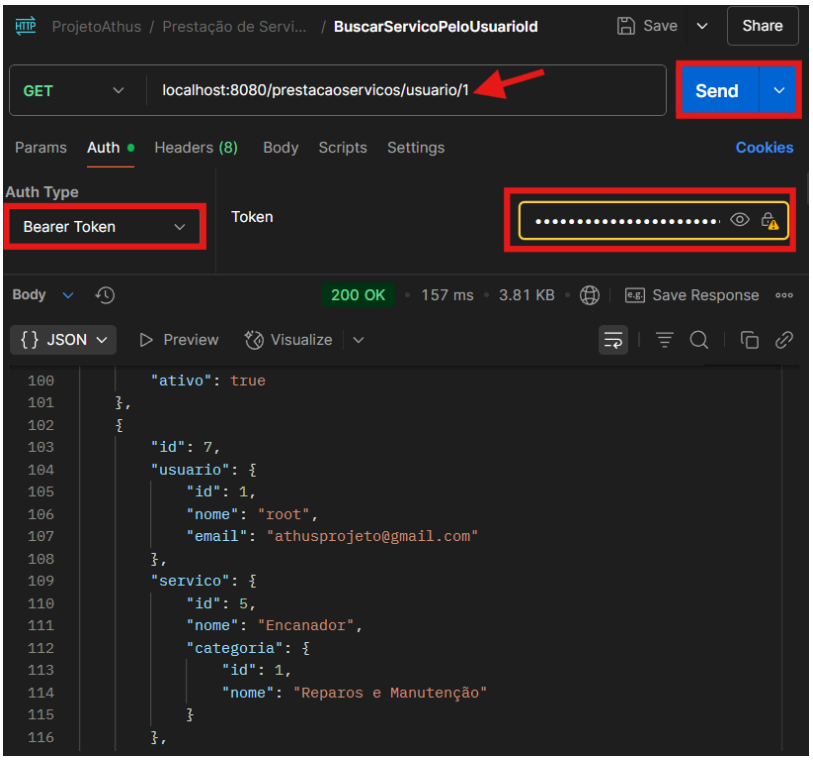
Acesso negado

6.3. GET BuscarServicoPeloUsuarioId

Retorna todos os serviços de determinada prestação de serviço por usuário de acordo com o parâmetro ID do usuário.

Path: prestacaoservicos/usuario/{id}

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID do usuário (alterar no {id} pelo ID do usuário desejado).

Possíveis respostas:

200 OK:

Requisição bem sucedida:

403 Forbidden:

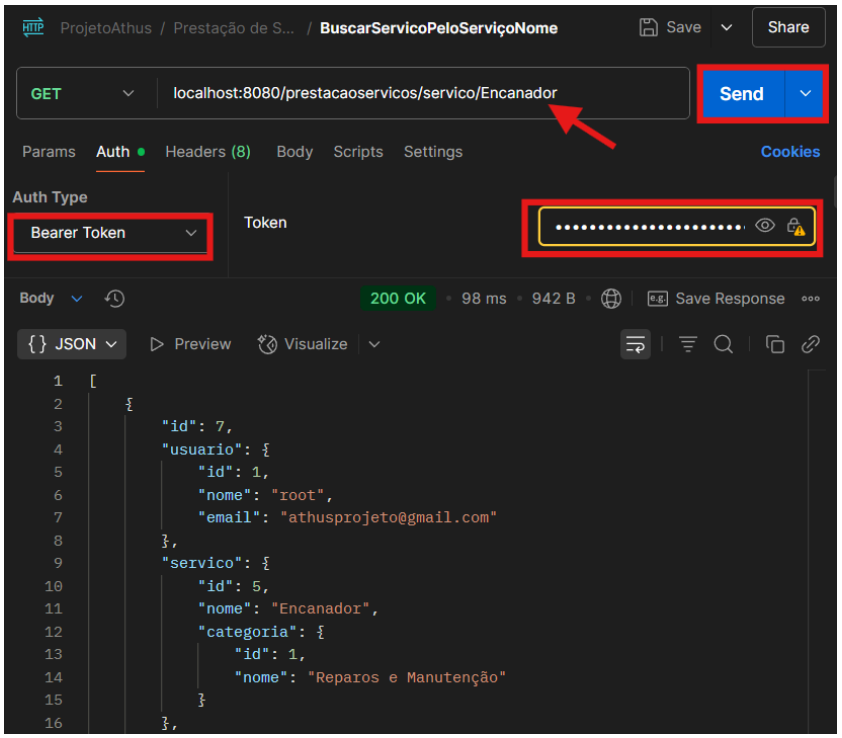
Acesso negado

6.4. GET BuscarServicoPeloServiçoNome

Descrição: Retorna prestação de serviço cadastrada pelo nome do serviço.

Path: prestacaoservicos/servico/{nome_servico}

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- nome_servico - Nome do serviço desejado (alterar no {nome_servico} pelo nome que deseja pesquisar).

Possíveis respostas:

200 OK:

Requisição bem sucedida:

```
{
```

```
        "id": 7,
        "usuario": {
            "id": 1,
            "nome": "root",
            "email": "athusprojeto@gmail.com"
        },
        "servico": {
            "id": 5,
            "nome": "Encanador",
            "categoria": {
                "id": 1,
                "nome": "Reparos e Manutenção"
            }
```


403 Forbidden:

Acesso negado

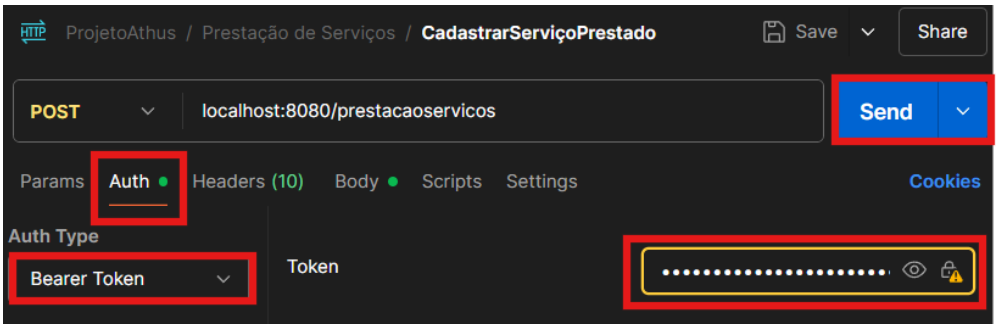
6.5. POST CadastrarServiçoPrestado

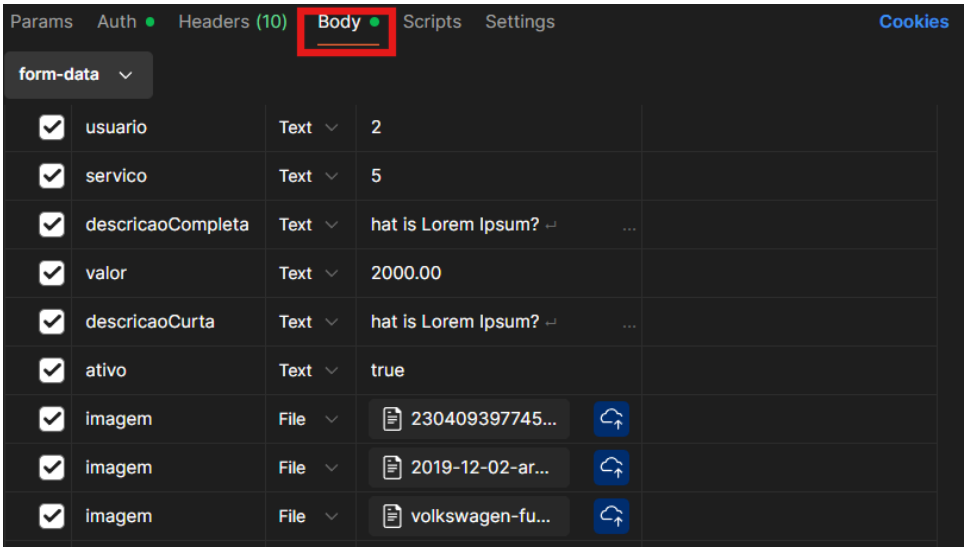
Descrição: Cadastrar um serviço prestado pelo usuário dentro da categoria serviço.

Body: usuário a qual será ligado a prestação de serviço, a categoria de serviço que se encaixa, uma descrição sobre a prestação de serviço, o valor cobrado, resumo da descrição, se está ativo ou não e imagens para exemplificar.

Path: prestacaoservicos.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.





Possíveis respostas:

200 OK:

Requisição bem sucedida:

403 Forbidden:

Acesso negado

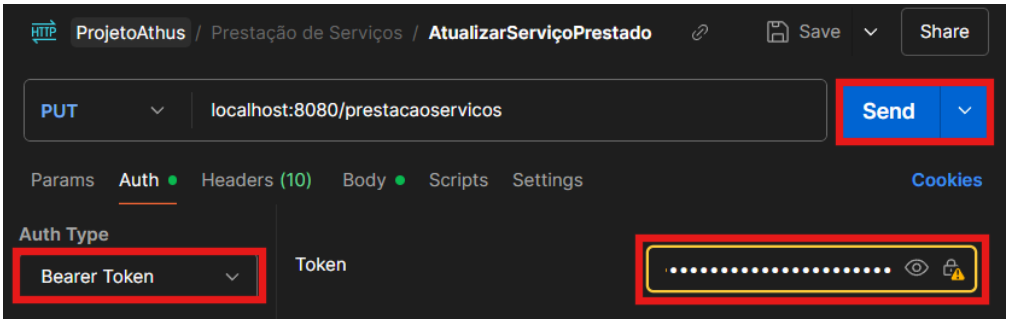
6.6. PUT AtualizarServiçoPrestado

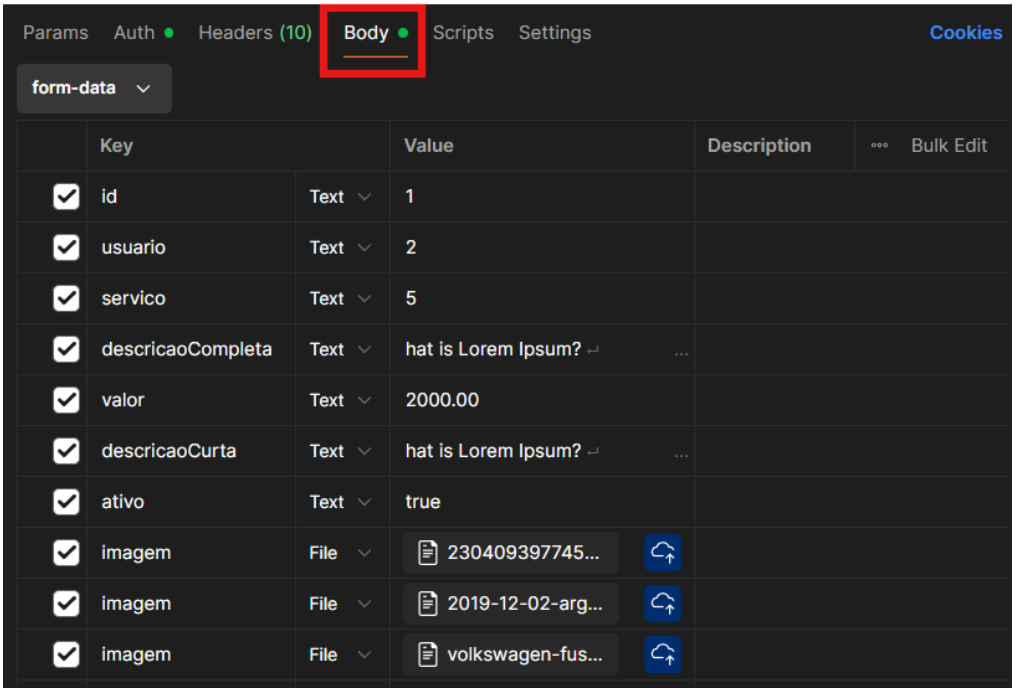
Descrição: Alterar ou atualizar algum dado do cadastro de prestação de serviço desejado.

Body: Necessário informar o *id* da prestação de serviço a ser modificada no body e o campo e conteúdo a ser modificado.

Path: prestacaoservicos.

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.





Possíveis respostas:

200 OK:

Requisição bem sucedida:

403 Forbidden:

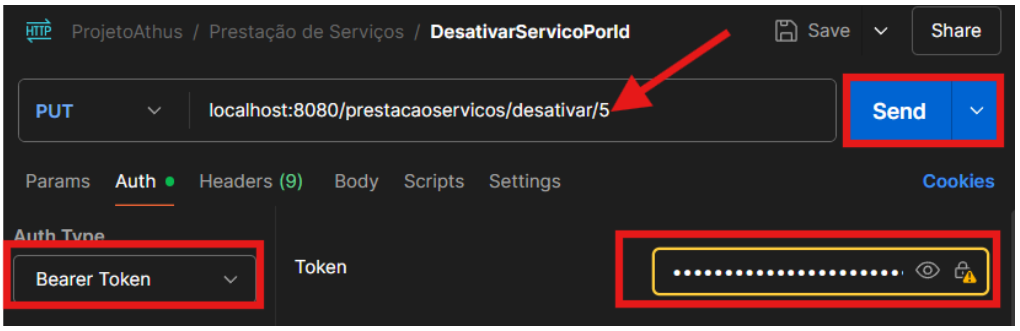
Acesso negado

6.7. PUT DesativarServicoPorId

Descrição: Desativar uma prestação de serviço oferecida por algum usuário por ID de prestação de serviço.

Path: prestacaoservicos/desativar/{id}

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID da prestação de serviço que deseja desativar (alterar no {id} pelo ID que deseja desativar).

Possíveis respostas:

200 OK:

Requisição bem sucedida, prestação de serviço informada desativada.

403 Forbidden:

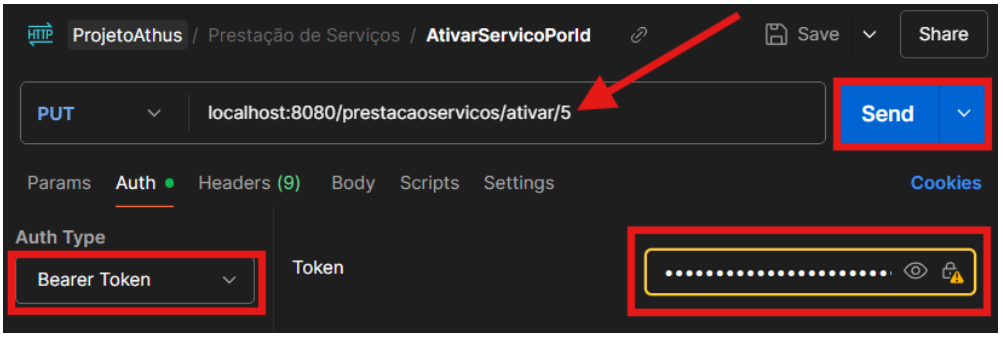
Acesso negado

6.8. PUT AtivarServicoPorId

Descrição: Ativar uma prestação de serviço oferecida por algum usuário por ID de prestação de serviço.

Path: prestacaoservicos/ativar/{id}

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID da prestação de serviço que deseja ativar (alterar no {id} pelo ID que deseja ativar).

Possíveis respostas:

200 OK:

Requisição bem sucedida, prestação de serviço informada ativada.

403 Forbidden:

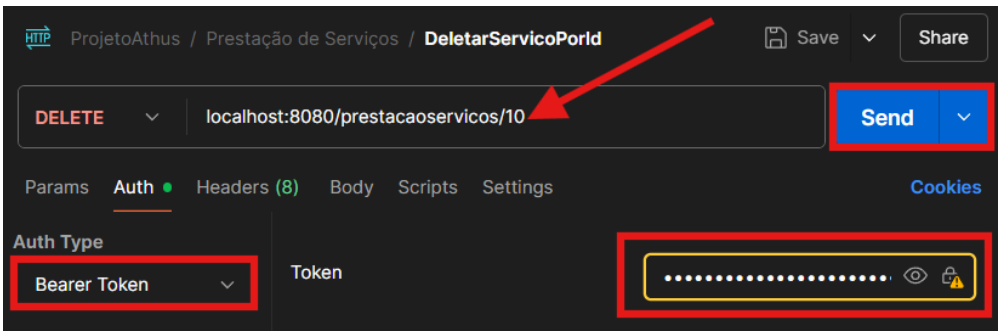
Acesso negado

6.9. DELETE DeletarServicoPorId

Descrição: Deletar uma prestação de serviço oferecida por algum usuário por ID de prestação de serviço.

Path: prestacaoservicos/{id}

Autenticação: Requer autenticação Bearer Token. Para gerar o token, favor seguir os passos do item ***2.1 Token***.



Parâmetros:

- id - ID da prestação de serviço que deseja deletar (alterar no {id} pelo ID que deseja deletar).

Possíveis respostas:

200 OK:

Requisição bem sucedida, prestação de serviço informada deletada.

403 Forbidden:

Acesso negado