

API Rest com node



Professor Mário de Jesus

O que é arquitetura REST?

REST significa Transferência de Estado Representacional. REST é uma arquitetura baseada em padrões da web e usa protocolo HTTP. Ele gira em torno de recurso onde cada componente é um recurso acessado por uma interface comum usando métodos padrão HTTP. O REST foi introduzido pela primeira vez por Roy Fielding em 2000.

Um servidor REST simplesmente fornece acesso a recursos e o cliente REST acessa e modifica os recursos usando o protocolo HTTP. Aqui, cada recurso é identificado por URIs / IDs globais. REST usa várias representações para representar um recurso como texto, JSON, XML, mas JSON é o mais popular.



Métodos HTTP

Os quatro métodos HTTP a seguir são comumente usados na arquitetura baseada em REST.

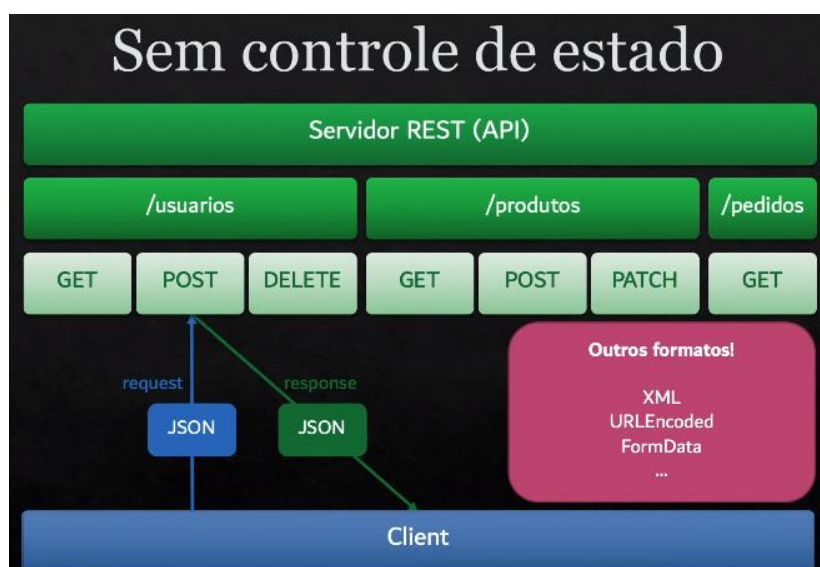
GET - Isso é usado para fornecer acesso somente leitura a um recurso.

PUT - Isso é usado para criar um novo recurso.

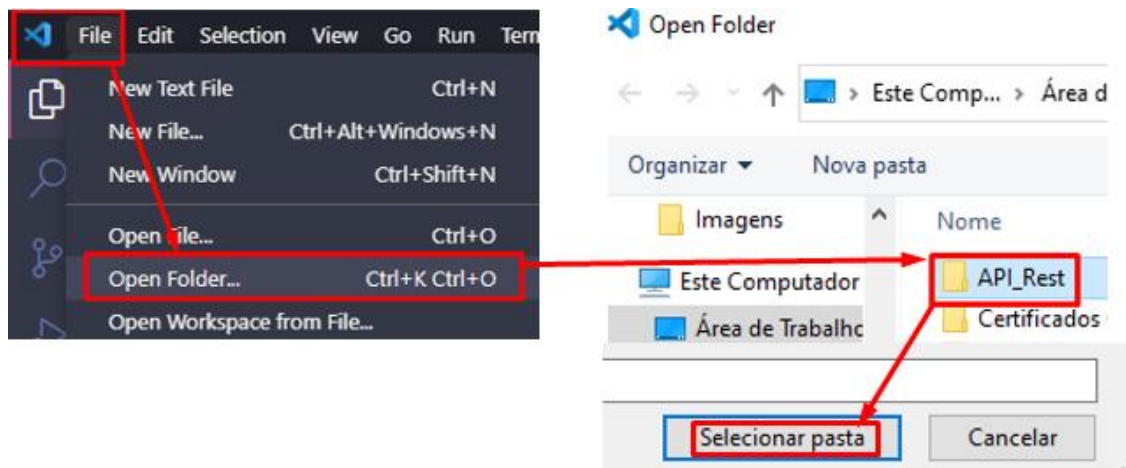
DELETE - Isso é usado para remover um recurso.

POST - Isso é usado para atualizar um recurso existente ou criar um novo recurso.

Para finalizar, o principal objetivo da Rest API é fornecer dados



Crie uma pasta chamada API_Rest e abra ela no Visual Studio Code



Abra o terminal do Visual Studio Code e faça as instalações abaixo:

npm init -y

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS
PS C:\Users\mario\Desktop\API_Rest> npm init -y
Wrote to C:\Users\mario\Desktop\API_Rest\package.json:
```

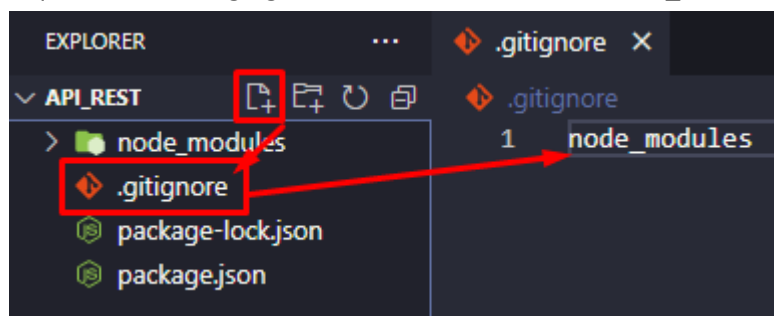
npm install express -save

```
PS C:\Users\mario\Desktop\API_Rest> npm install express -save
added 57 packages, and audited 58 packages in 8s
7 packages are looking for funding
run `npm fund` for details
```

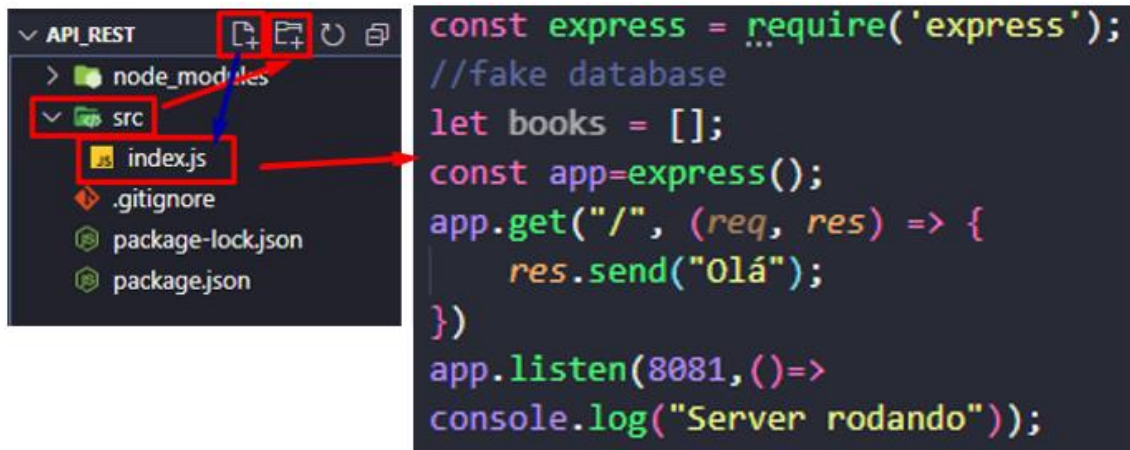
npm install -g nodemon

```
PS C:\Users\mario\Desktop\API_Rest> npm install -g nodemon
added 32 packages, and audited 33 packages in 5s
3 packages are looking for funding
run `npm fund` for details
```

Crie um arquivo chamado .gitignore e escreva dentro dele node_modules



Agora crie a pasta **src** e dentro dela crie o arquivo **index.js**. Digite o código abaixo:



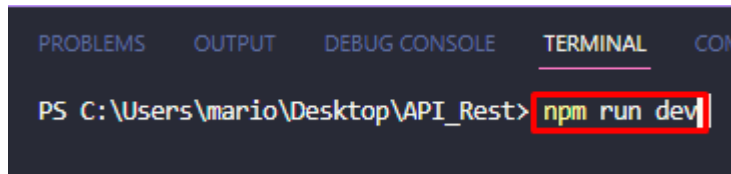
Código do **index.js**

```
const express = require('express');
//fake database
let books = [];
const app=express();
app.get("/", (req, res) => {
  res.send("Olá");
})
app.listen(8081,()=>
console.log("Server rodando"));
```

Abra o arquivo **package.json** e acrescente o código **"dev": "nodemon src/index.js"** para automatizar a inicialização o servidor.



Agora vamos iniciar o servidor no terminal com o comando **npm run dev**



```
PS C:\Users\mario\Desktop\API_Rest> npm run dev
```

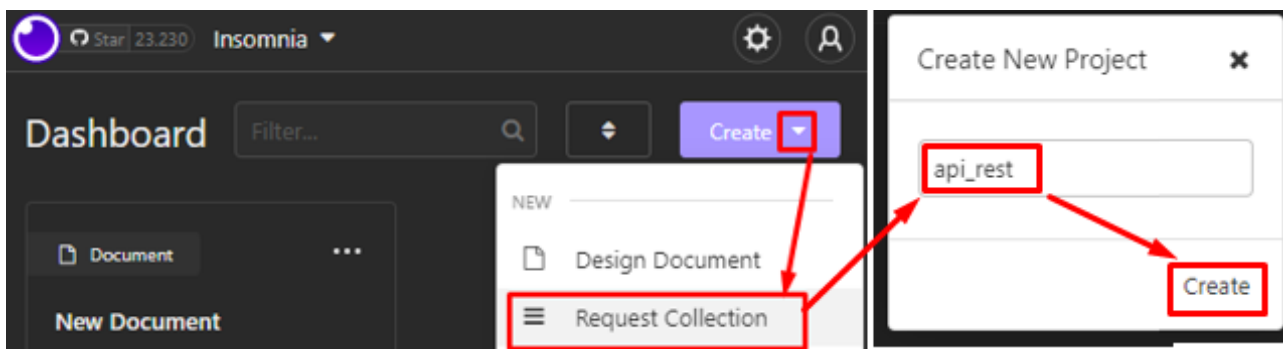
Faça os testes no browser.



Faça download do insomnia no site abaixo.

<https://insomnia.rest/download>

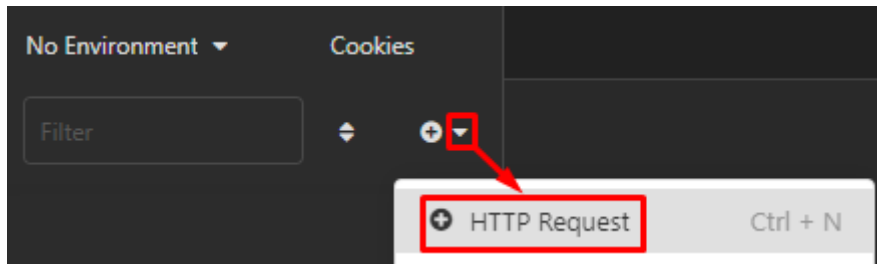
Crie um request collection chamado **api_rest**



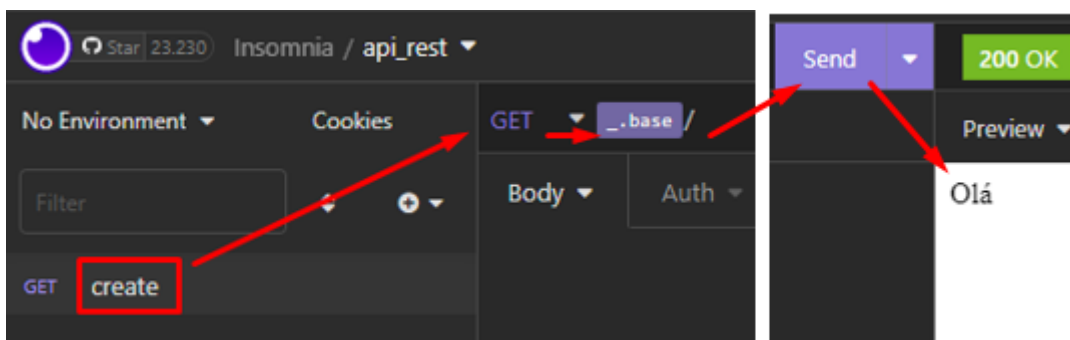
Crie uma variável chamada **"base"** com a instrução do nosso servidor **"localhost:8081"**



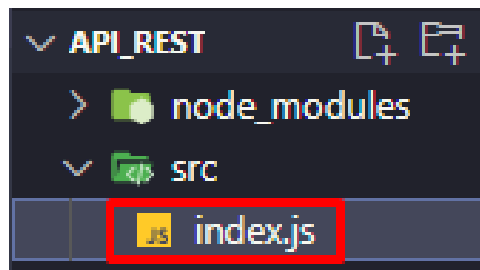
Agora crie um método HTTP Request



Chame esse método **request** de **create**, use a função **get**, chame a variável **_.base** e clique em **send** para ver o olá



Agora altere o index.js com os métodos abaixo:



```
const express = require('express');
//fake database
let books = [];
const app = express();

//Aplicando o middleware
app.use(express.json());

//rotas
app.get("/", (req, res) => {
  res.send("Olá");
});

//Cadastro de livros
app.post('/books_cad', (req, res) => {
  const { id, titulo, autor, publicacao } = req.body;
  const book = { id, titulo, autor, publicacao }
  books.push(book);
  //200 é o status de criado
  return res.status(200).json(book);
});

//Lista todos os livros
app.get('/books_lista', (req, res) => {
  const allBooks = books;
  return res.status(200).json(allBooks);
});

//Busca um livros específico
app.get('/books_lista/:book_id', (req, res) => {
  const { book_id } = req.params;
  const book = books.find((book) => book.id === book_id);
  if (!book) {
    res.status(404).json("nenhum livro encontrado");
  } else {
    return res.status(200).json(book);
  }
}
```

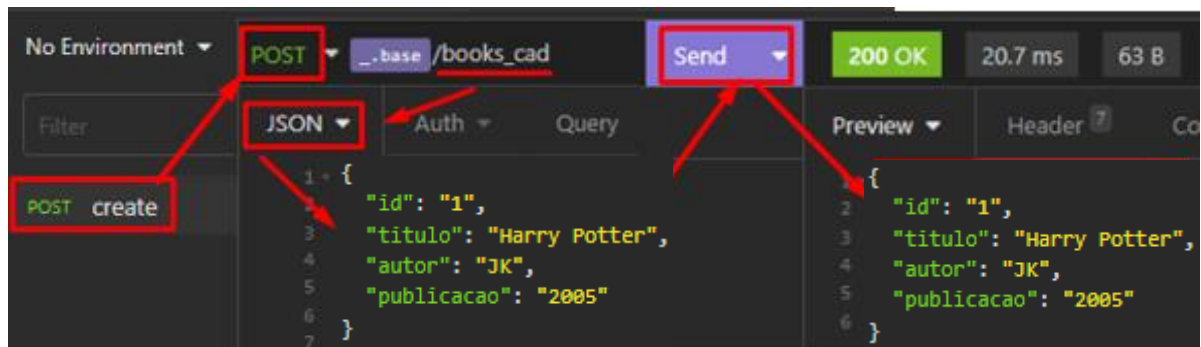
```
});

//Deleta um livros específico
app.delete('/books_delete/:book_id', (req, res) => {
  const { book_id } = req.params;
  const procura_book = books.filter(book => book.id !== book_id);
  books = procura_book;
  return res.status(204).json("deletado");
});

//Deleta um livros específico
app.patch('/books_update/:book_id', (req, res) => {
  const { titulo, autor, publicacao } = req.body;
  const { book_id } = req.params;
  const book = books.find(book => book.id === book_id);
  book.id = book_id;
  book.titulo = titulo ? titulo : book.titulo;
  book.autor = autor ? autor : book.autor;
  book.publicacao = publicacao ? publicacao : book.publicacao;
  return res.status(200).json(book);
});

app.listen(8081, () =>
  console.log("Server rodando"));
```

Teste os métodos no insomnia



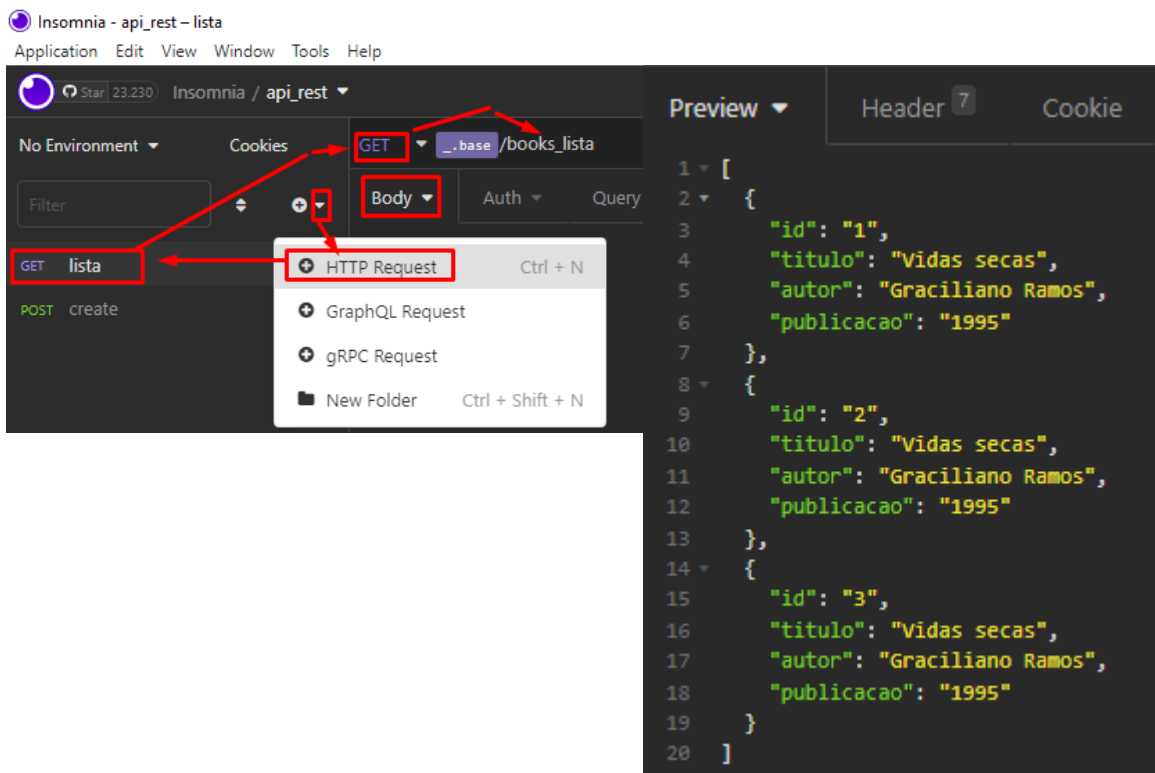
```
{
  "id": "1",
  "titulo": "Harry Potter",
  "autor": "JK",
  "publicacao": "2005"
}
```


Faça mais alguns cadastros

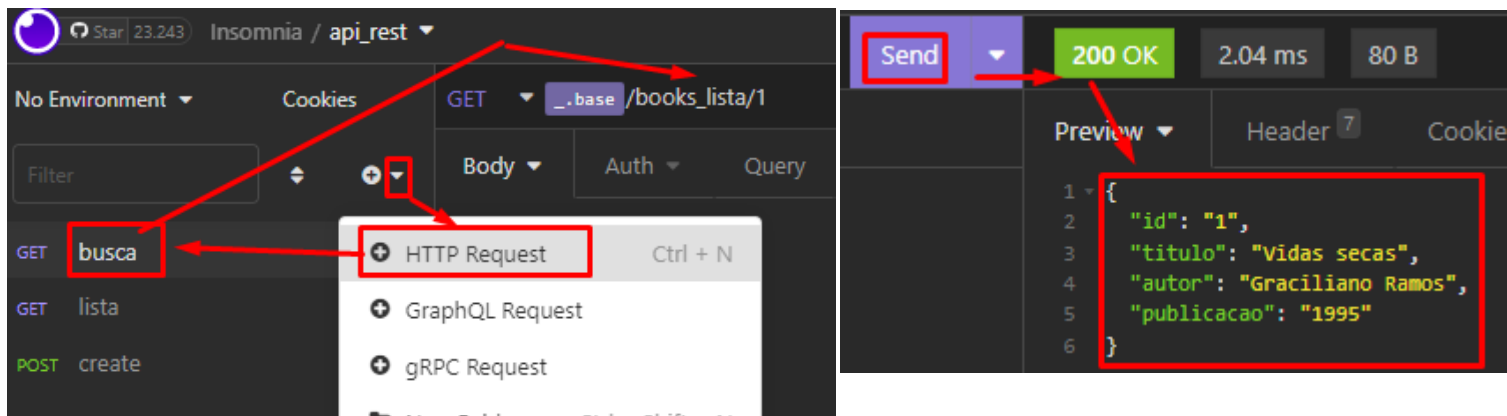
```
{
  "id": "2",
  "titulo": "Vidas secas",
  "autor": "Graciliano Ramos",
  "publicacao": "1995"
}
```

```
{
  "id": "2",
  "titulo": "Vidas secas",
  "autor": "Graciliano Ramos",
  "publicacao": "1995"
}
```

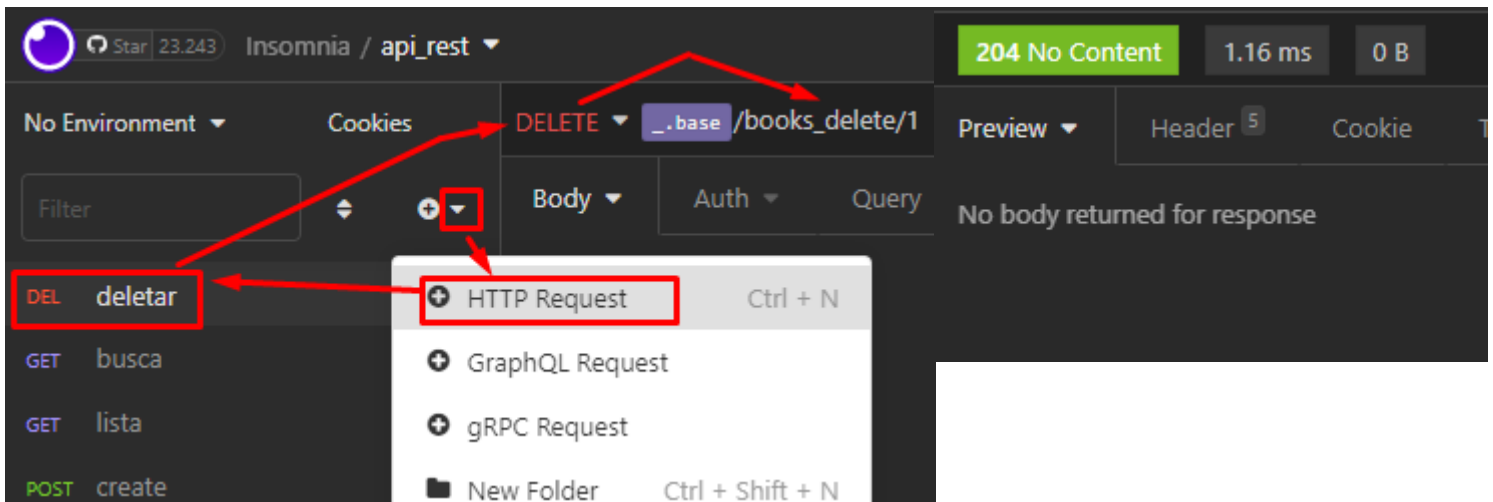
Vamos adicionar um método no insomnia chamado lista, com a função get e o body para mostrar os itens cadastros.



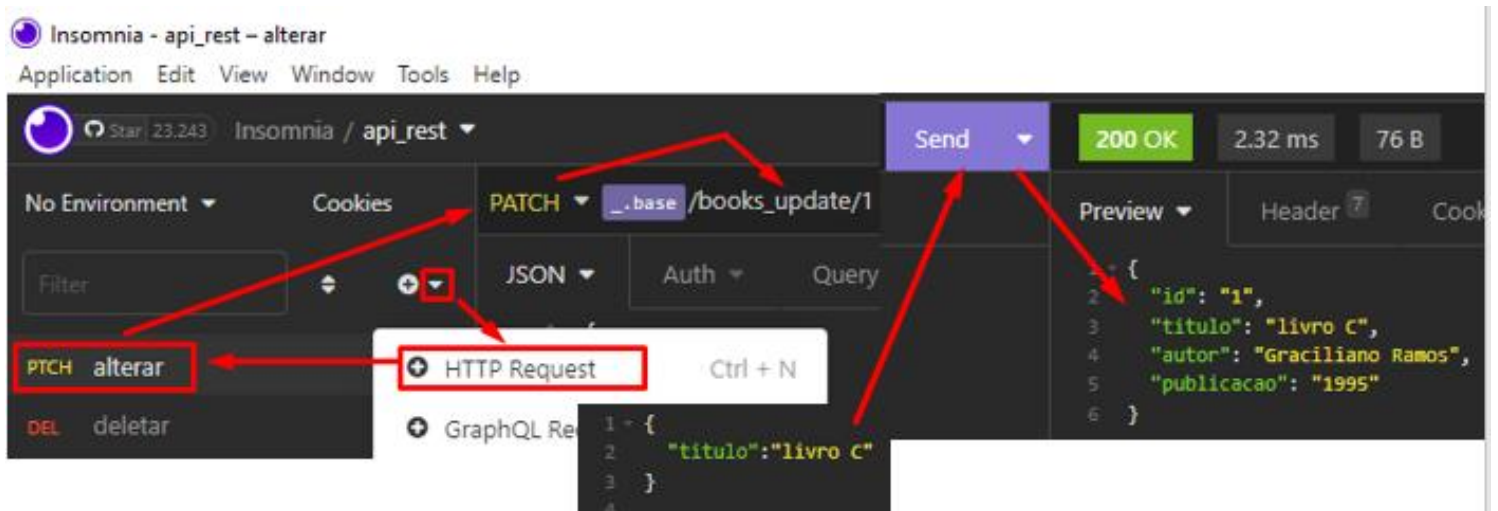
Agora vamos buscar os livros cadastrados



Agora vamos deletar um livro



Agora vamos alterar um livro



Referencias



YouTube

...

Como fazer uma API Rest NodeJS #fácil #rápido

Dev \u0026 Design | 2.1K views | one year ago



<https://youtu.be/qGpOKeYSzqc>