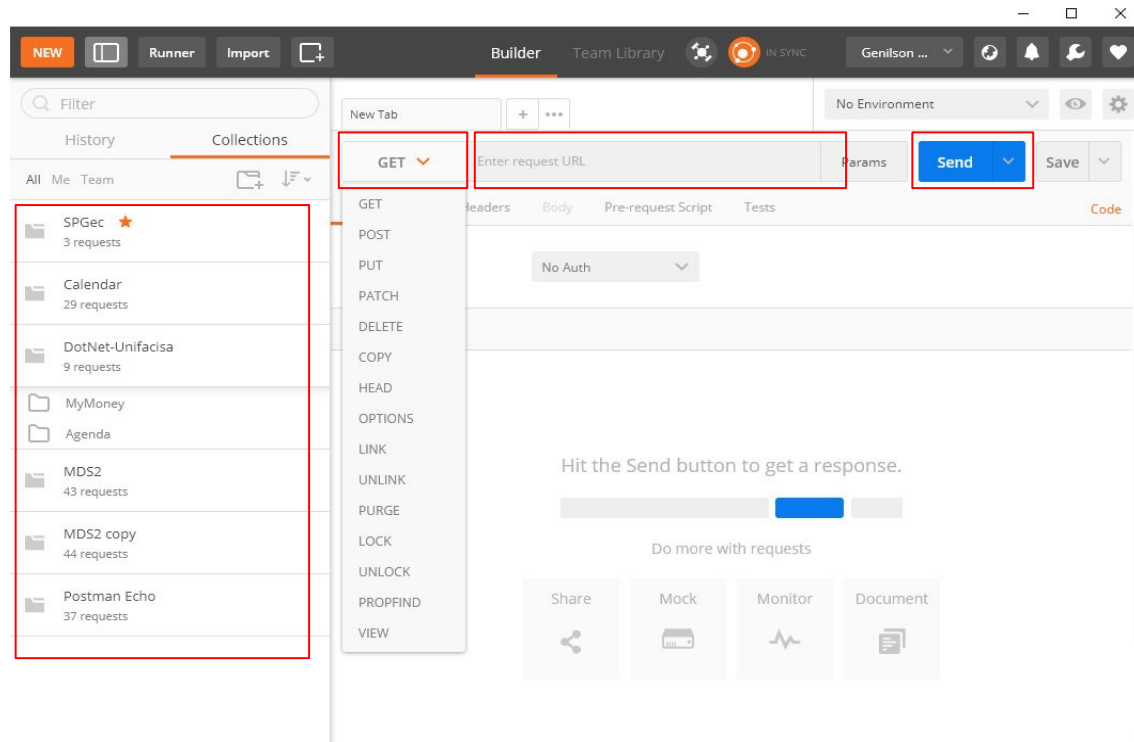


# Sistemas de Informação II

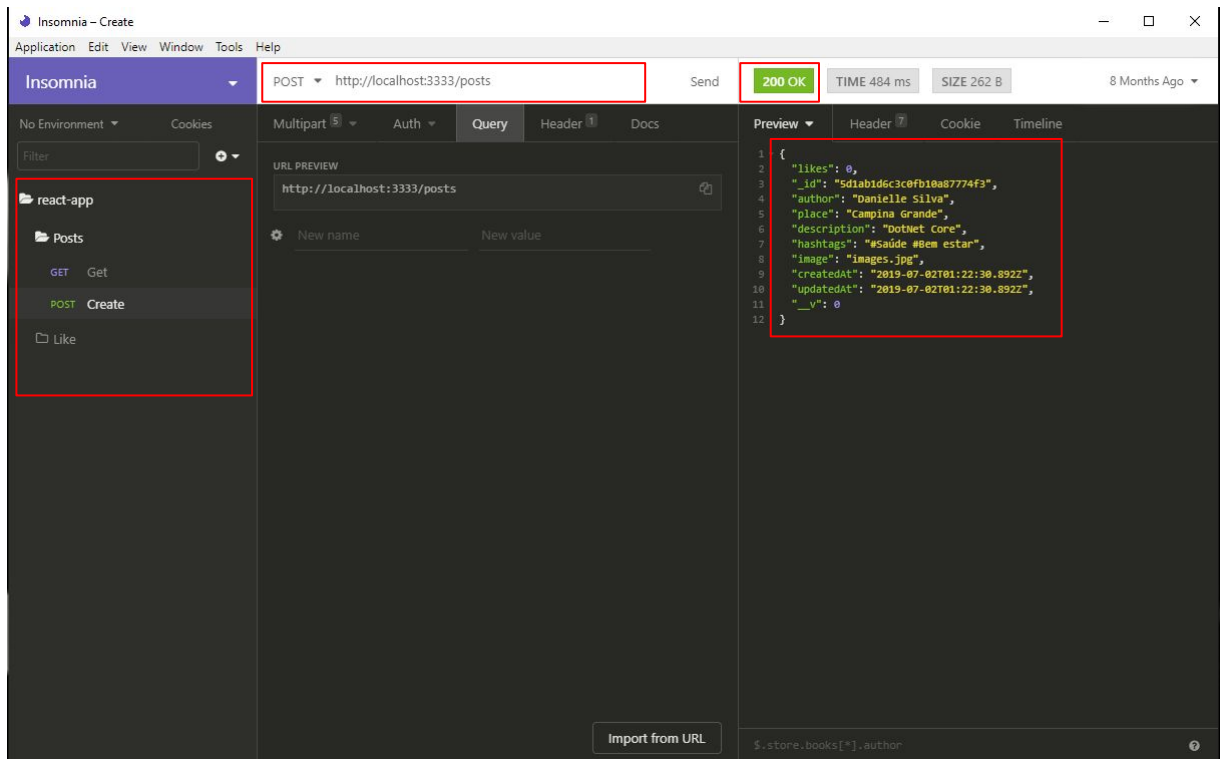
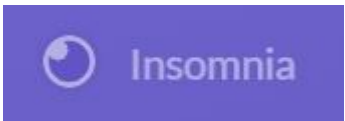
<https://github.com/genilsonmm/sistemas-de-informacao-2>

## Testando seu Web Service

Postman, Inc.  
Empresa

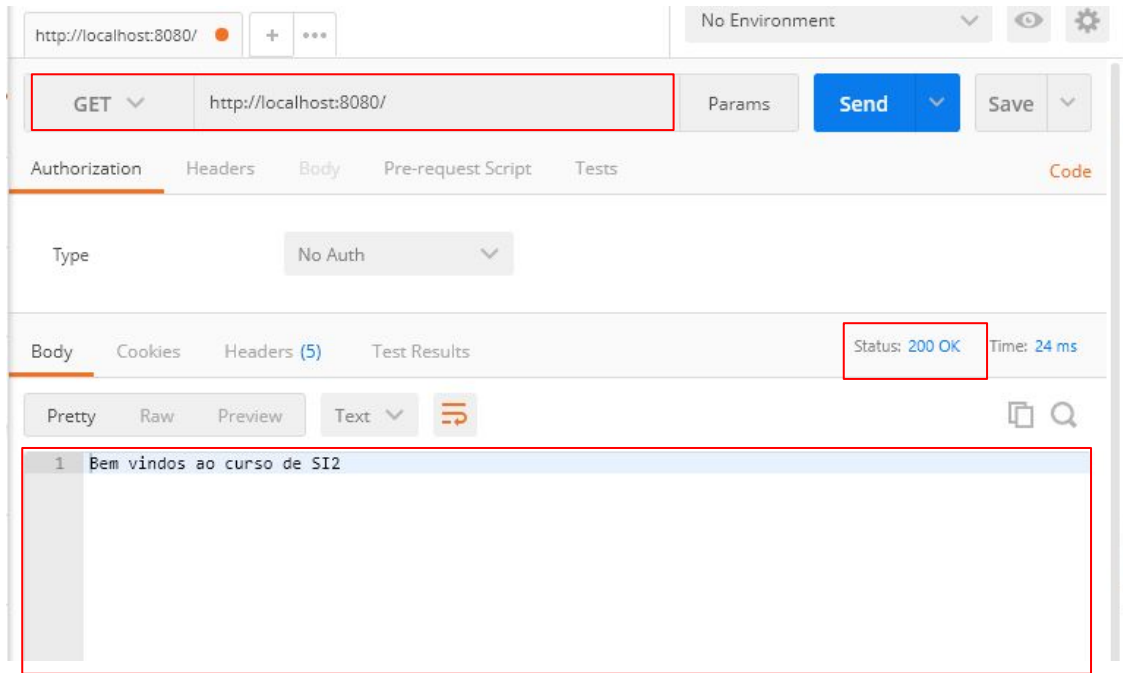


# Testando seu Web Service



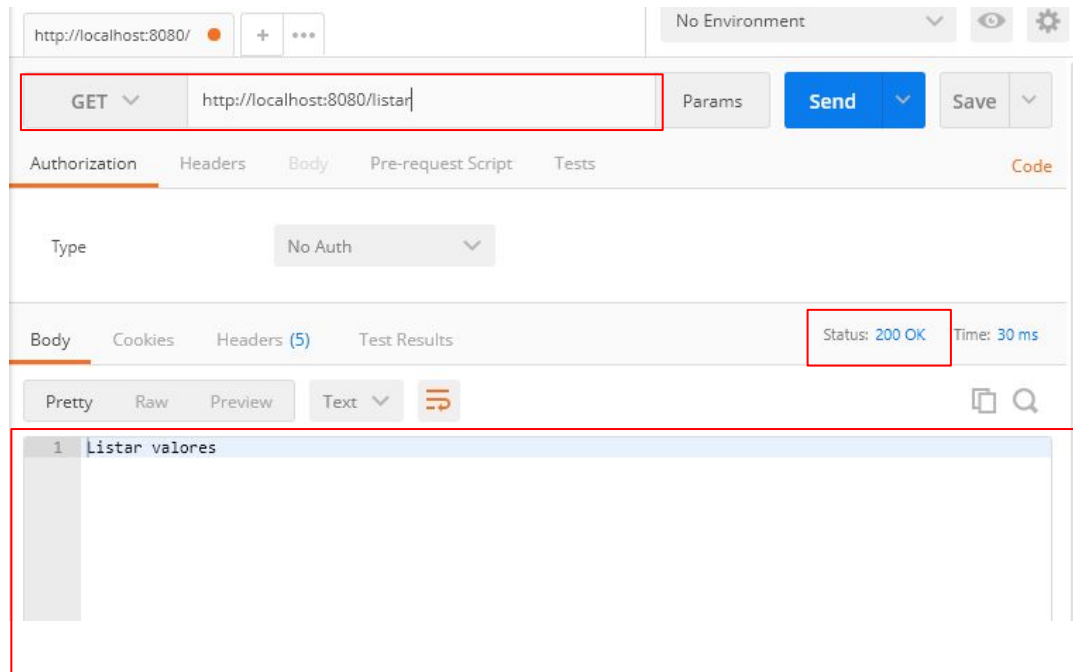
## Testando seu Web Service

```
@RequestMapping("/")  
public String bemVindo() {  
    return "Bem vindos ao curso de SI2";  
}
```



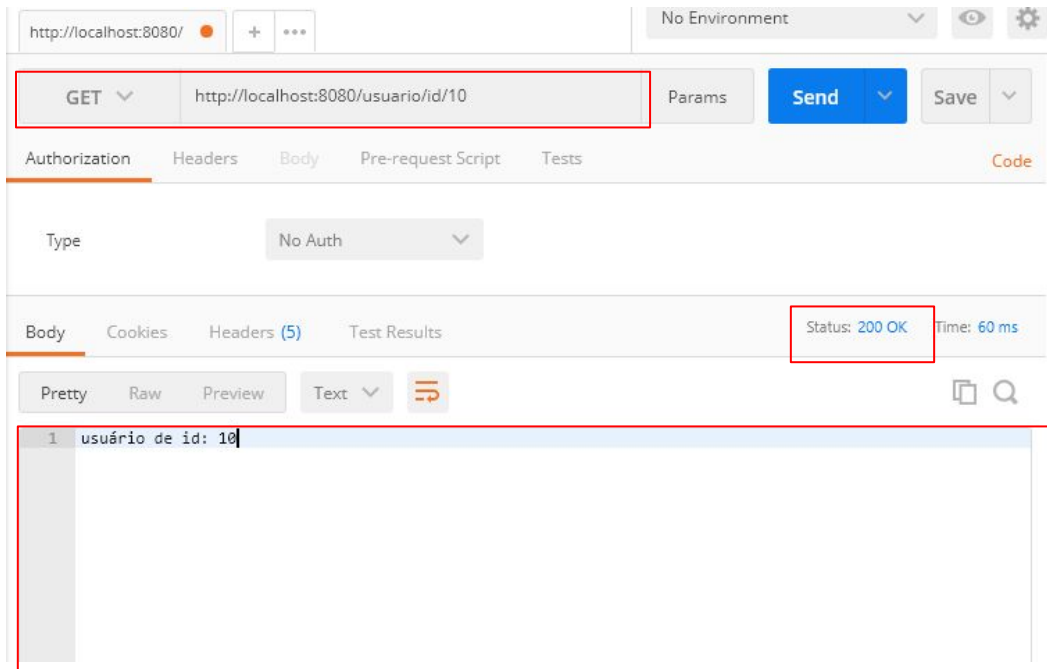
## Testando seu Web Service

```
@RequestMapping(value="/listar", method = RequestMethod.GET)
public String listar() {
    return "Listar valores";
}
```

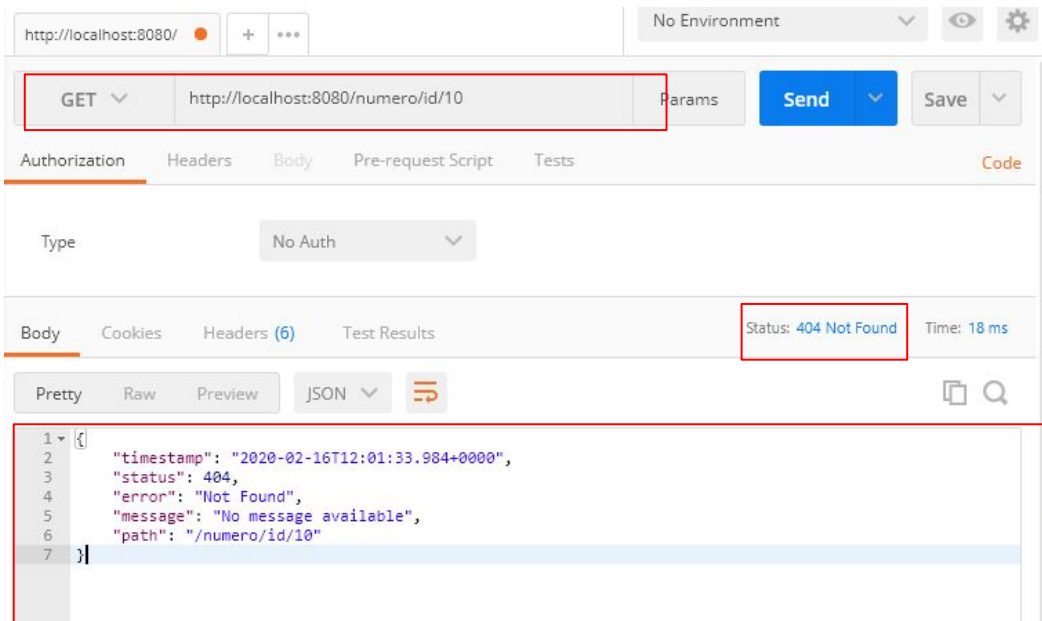


## Testando seu Web Service

```
@GetMapping("/usuario/id/{numero}")  
public String user(@PathVariable String numero) {  
    return "usuário de id: " + numero;  
}
```



## Testando o Web Service





# JSON

JavaScript Object Notation



## JSON

Um **acrônimo** de JavaScript Object Notation, é um **formato** compacto, de **padrão aberto** independente, de troca de dados simples e rápida (parsing) entre sistemas, especificado por Douglas Crockford em 2000, que utiliza **texto legível a humanos**, no formato atributo-valor (natureza auto-descritiva)

# JSON

```
{  
  "nome": "Pedro",  
  "altura": 1.90  
}
```

## JSON - Lista de alunos com suas notas

```
1 {"Alunos":[  
2   { "nome": "Edson Sales Arantes", "notas": [ 8, 9, 5 ] },  
3   { "nome": "Luiz Livelli ", "notas": [ 8, 10, 7 ] },  
4   { "nome": "Caique Caicedo De Plata", "notas": [ 10, 10, 9 ] }  
5 ]}
```

# JSON

```
1 {  
2   "nome": "Maria",  
3   "idade": 33,  
4   "livros": ["livro1", "livro2", "livro3"],  
5   "primos": [{  
6     "nome": "Pedro",  
7     "idade": 10  
8   },  
9   {  
10    "nome": "João",  
11    "idade": 20  
12  },  
13  {  
14    "nome": "Danielle",  
15    "idade": 30  
16  }  
17 ]  
18 }
```

## JSON - Validador

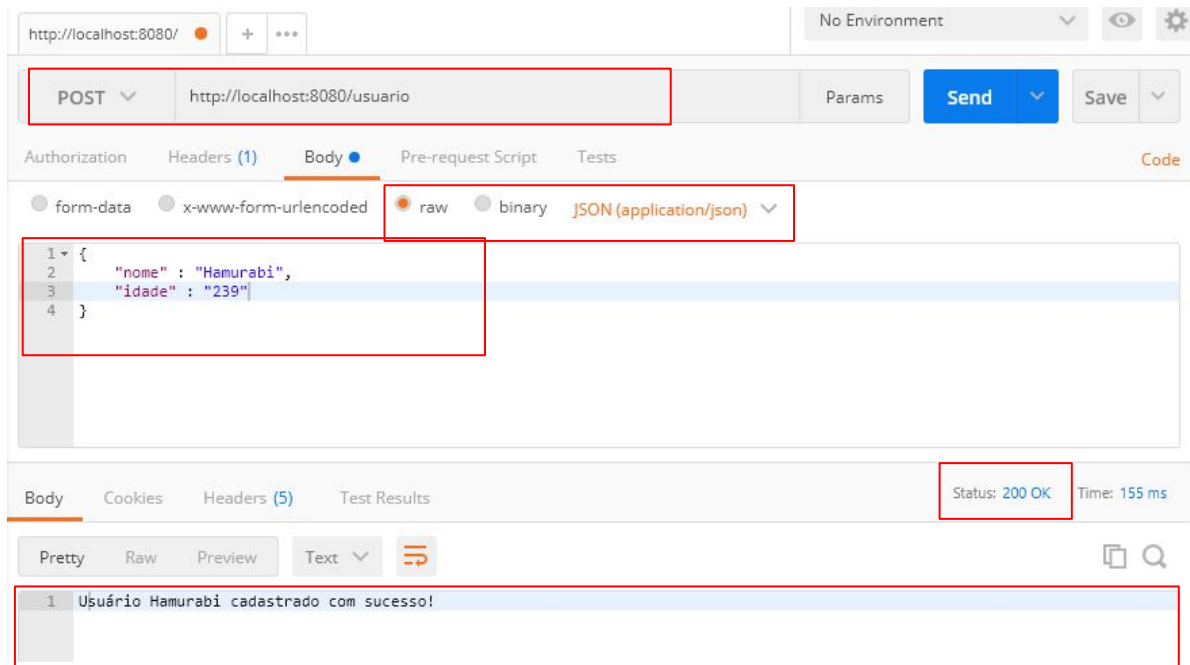
<https://jsonlint.com/>

## POST

```
@PostMapping("/usuario")
public String cadastrarUsuario(@RequestBody Usuario usuario) {
    return "Usuário " + usuario.getNome() + " cadastrado com sucesso!";
}
```

```
1 package com.si2.api.model;
2
3 public class Usuario {
4
5     private int id;
6     private String nome;
7     private int idade;
8
9     public int getId() {
10         return id;
11     }
12     public void setId(int id) {
13         this.id = id;
14     }
15     public String getNome() {
16         return nome;
17     }
18     public void setNome(String nome) {
19         this.nome = nome;
20     }
21     public int getIdade() {
22         return idade;
23     }
24     public void setIdade(int idade) {
25         this.idade = idade;
26     }
27 }
```

# POST



The screenshot displays a REST client interface with the following components:

- Address Bar:** Shows the URL `http://localhost:8080/`.
- Method and URL:** The method is set to **POST** and the URL is `http://localhost:8080/usuario`.
- Body Tab:** The **Body** tab is selected, showing the request body as raw JSON: 

```
{  "nome" : "Hamurabi",  "idade" : "239"}
```
- Response Tab:** The **Body** tab is selected, showing the response body as text: 

```
1  Usuário Hamurabi cadastrado com sucesso!
```
- Status:** The status is **200 OK** with a response time of **155 ms**.

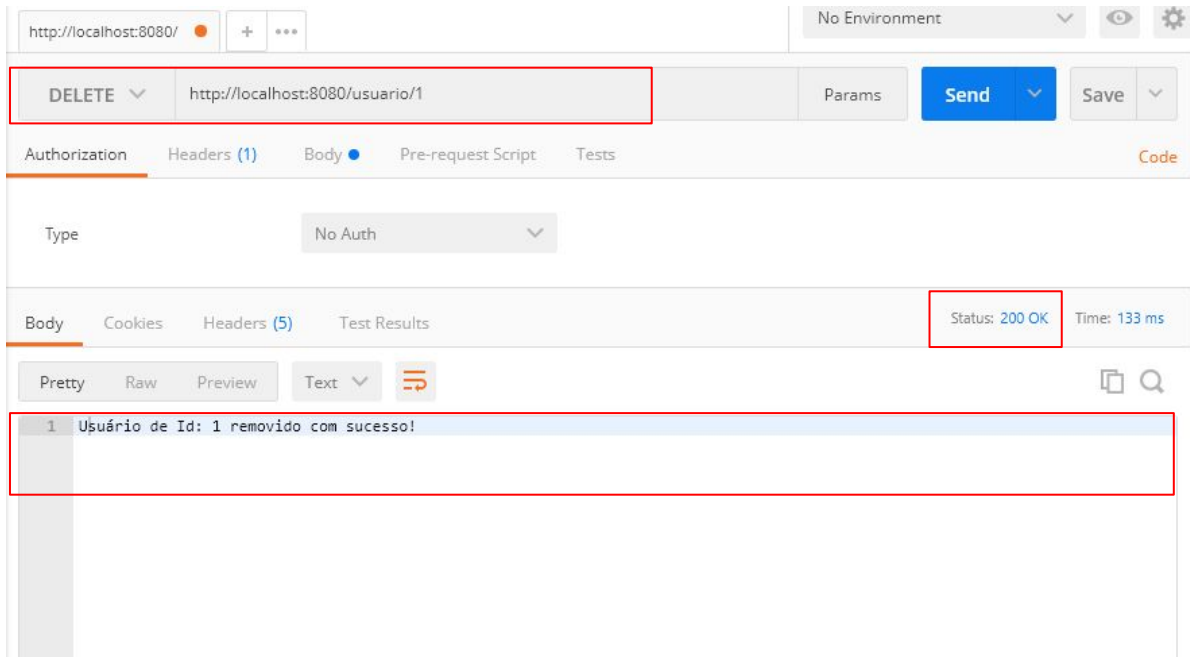
## DELETE

```
@DeleteMapping("/usuario/{id}")  
public String deletarUsuario(@PathVariable int id) {  
    return "Usuário de Id: " + id + " removido com sucesso!";  
}
```





## DELETE



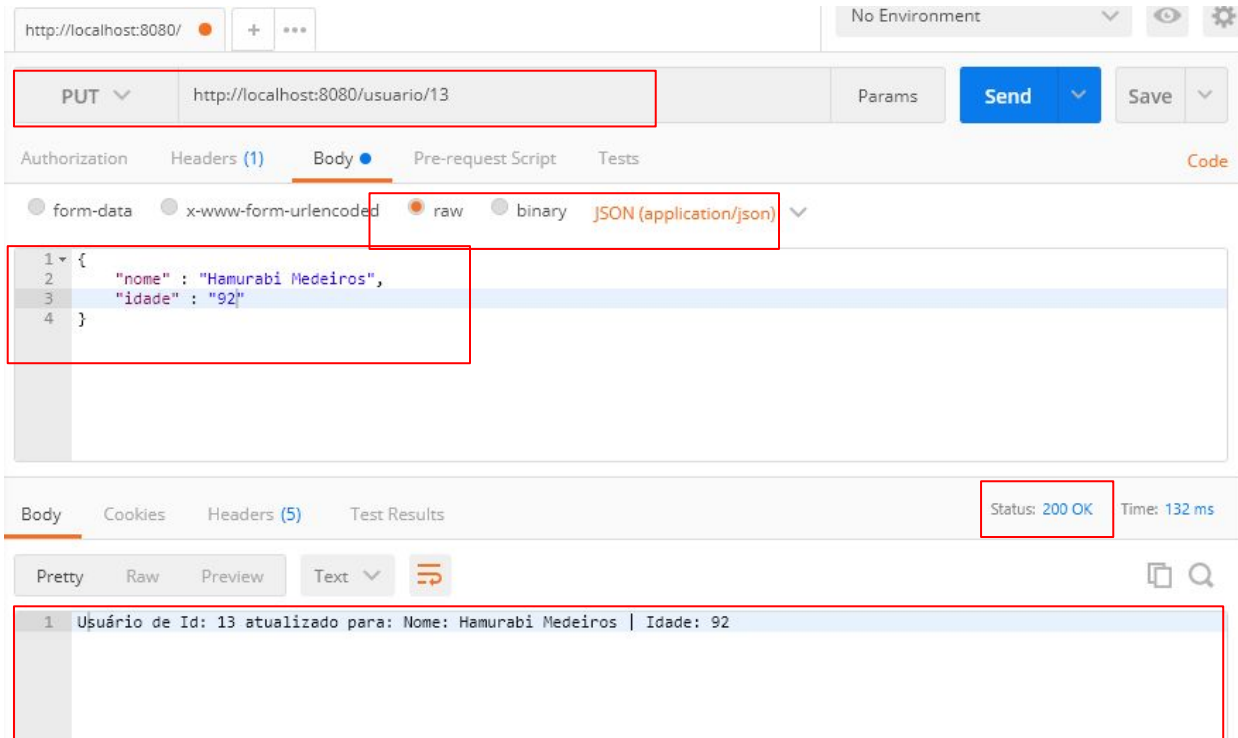
The screenshot displays a REST client interface with the following details:

- URL:** `http://localhost:8080/`
- Method:** `DELETE`
- Path:** `http://localhost:8080/usuario/1`
- Authorization:** No Auth
- Status:** 200 OK
- Time:** 133 ms
- Response Body:** `1 Usuário de Id: 1 removido com sucesso!`

## PUT

```
@PutMapping("/usuario/{id}")  
public String atualizarUsuario(@PathVariable int id, @RequestBody Usuario usuario) {  
    return "Usuário de Id: " + id + " atualizado para: " + usuario;  
}
```

# PUT



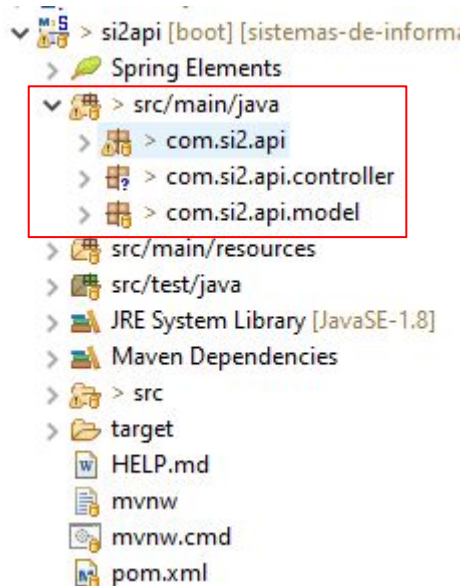
The screenshot displays a REST client interface with the following details:

- URL:** `http://localhost:8080/`
- Method:** `PUT`
- Path:** `http://localhost:8080/usuario/13`
- Body Type:** `JSON (application/json)`
- Body Content:**

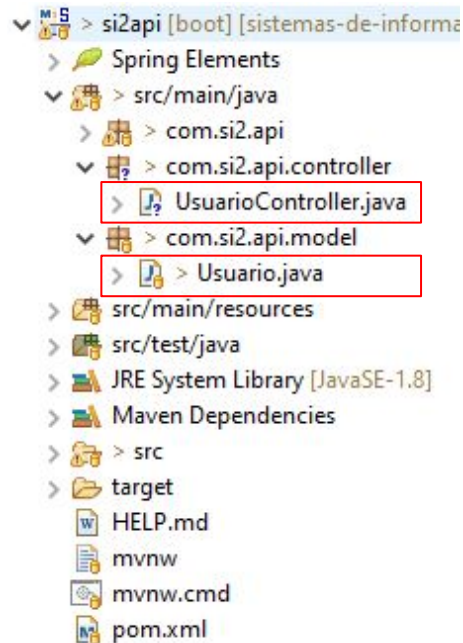
```
1 {  
2   "nome" : "Hamurabi Medeiros",  
3   "idade" : "92"  
4 }
```
- Status:** `200 OK`
- Time:** `132 ms`
- Response Content:**

```
1 Usuário de Id: 13 atualizado para: Nome: Hamurabi Medeiros | Idade: 92
```

## Nova estrutura do projeto



## Nova estrutura do projeto



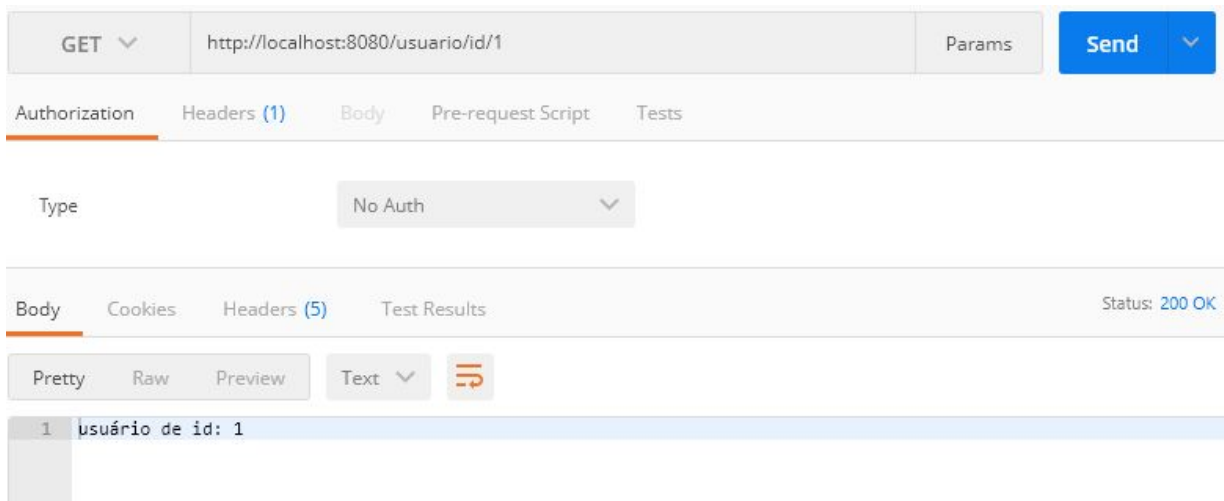
## Controller

```
@RestController()  
@RequestMapping("/usuario")  
public class UsuarioController {  
    @GetMapping("/id/{numero}")  
    public String user(@PathVariable String numero) {  
        return "usuário de id: " + numero;  
    }  
  
    @PostMapping()  
    public String cadastrarUsuario(@RequestBody Usuario usuario) {  
        return "Usuário " + usuario.getNome() + " cadastrado com sucesso!";  
    }  
  
    @DeleteMapping("/{id}")  
    public String deletarUsuario(@PathVariable int id) {  
        return "Usuário de Id: " + id + " removido com sucesso!";  
    }  
  
    @PutMapping("/{id}")  
    public String atualizarUsuario(@PathVariable int id, @RequestBody Usuario usuario) {  
        return "Usuário de Id: " + id + " atualizado para: " + usuario;  
    }  
}
```

## ResponseEntity

```
@GetMapping("/id/{numero}")
public ResponseEntity<String> user(@PathVariable int numero) {
    if(numero == 1)
    {
        String mensagem = "usuário de id: " + numero;
        return new ResponseEntity<String>(mensagem, HttpStatus.OK);
    }else {
        return new ResponseEntity<String>(HttpStatus.NOT_FOUND);
    }
}
```

## ResponseEntity



The screenshot displays a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:8080/usuario/id/1
- Params:** (empty)
- Buttons:** Send, Authorization, Headers (1), Body, Pre-request Script, Tests
- Type:** No Auth
- Body:** (empty)
- Cookies:** (empty)
- Headers (5):** (empty)
- Test Results:** (empty)
- Status:** 200 OK
- Response Body:** Pretty, Raw, Preview, Text (selected), JSON icon
- Response Content:** 1 usuário de id: 1



## ResponseEntity

GET ▾

http://localhost:8080/usuario/id/2

Params

Send ▾

Authorization

Headers (1)

Body

Pre-request Script

Tests

Type

No Auth ▾

Body

Cookies

Headers (4)

Test Results


Status: 404 Not Found

Pretty

Raw

Preview

Text ▾



1

## Retornando uma coleção de usuários

```
@RestController()
@RequestMapping("/usuario")
public class UsuarioController {

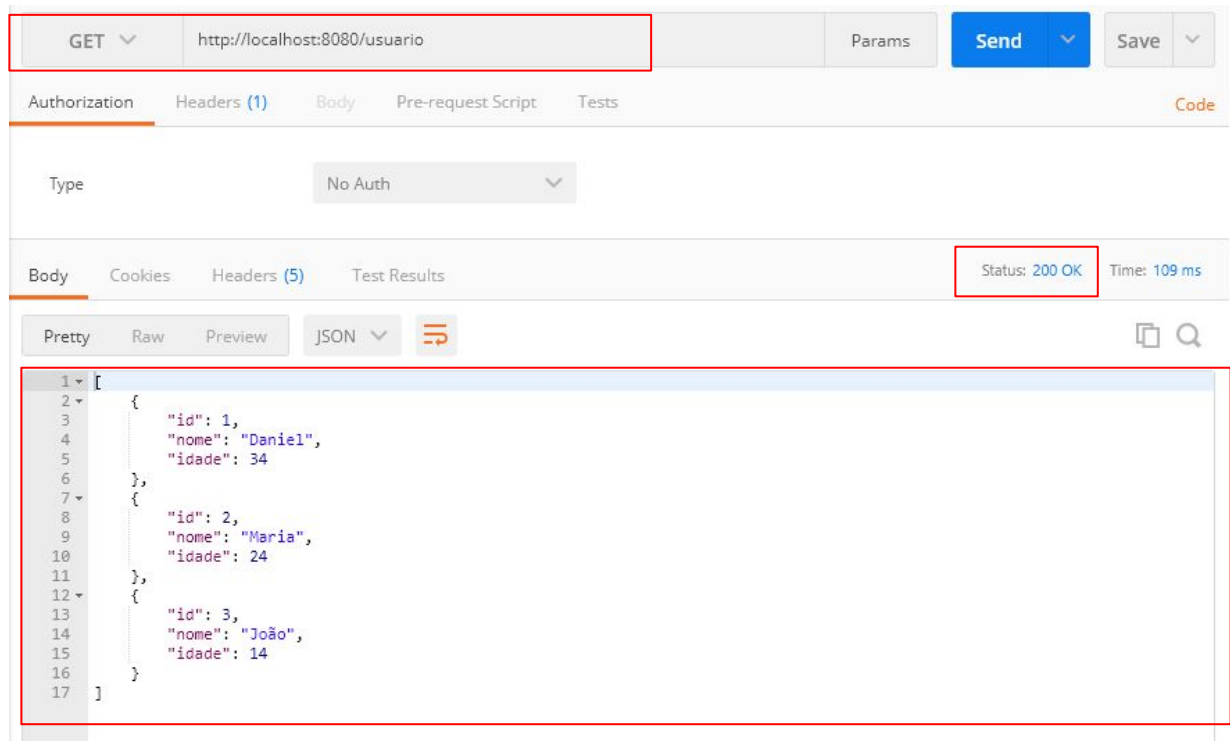
    private List<Usuario> usuarios;

    public UsuarioController() {
        usuarios = new ArrayList<Usuario>();

        usuarios.add(new Usuario(1, "Daniel", 34));
        usuarios.add(new Usuario(2, "Maria", 24));
        usuarios.add(new Usuario(3, "João", 14));
    }

    @GetMapping()
    public ResponseEntity<List<Usuario>> todos(){
        return new ResponseEntity<List<Usuario>>(usuarios, HttpStatus.OK);
    }
}
```

## Retornando uma coleção de usuários



GET `http://localhost:8080/usuario` Params Send Save

Authorization Headers (1) Body Pre-request Script Tests Code

Type No Auth

Body Cookies Headers (5) Test Results Status: 200 OK Time: 109 ms

Pretty Raw Preview JSON

```
1 [
2   {
3     "id": 1,
4     "nome": "Daniel",
5     "idade": 34
6   },
7   {
8     "id": 2,
9     "nome": "Maria",
10    "idade": 24
11  },
12  {
13    "id": 3,
14    "nome": "João",
15    "idade": 14
16  }
17 ]
```