

# PROJETO RESTFUL

## SUMÁRIO

<b>PROJETO RESTFUL.....</b>	<b>1</b>
DADOS PARA A DISCIPLINA.....	6
ATIVIDADE 1.....	6
OBJETIVO.....	6
O QUE É ESSENCIAL?.....	6
CRITÉRIOS DE AVALIAÇÃO (Rúbricas do Google Class).....	7
ENTREGA.....	7
CONTATOS.....	7
<b>PROJETO CONCEITUAL.....</b>	<b>8</b>
DEPENDÊNCIAS.....	8
CASO.....	8
DIAGRAMA DE CLASSE.....	9
TESTES A SEREM REALIZADOS NO POSTMAN.....	10
BUSCA PAGINADA DE ALUNOS.....	10
BUSCA DE ALUNO POR ID.....	10
INSERIR NOVO ALUNO.....	10
ATUALIZAR ALUNO.....	10
DELETAR ALUNO.....	10
<b>CRIAR O PROJETO API_RESTFUL.....</b>	<b>11</b>
DEFINIR O WORKSPACE PARA O PROJETO.....	11
CRIAR O PROJETO COM O INITIALIZR.....	11
ACESSAR A URL.....	11
GERAR O PROJETO.....	11
Salvar dentro do workspace.....	12
Descompactar e Renomear.....	12
IMPORTAR O PROJETO PARA O STS.....	13
Rodar o projeto.....	15
TESTAR NO BROWSER.....	15
Github-1.....	16
Criar o projeto no Github.....	16
Sincronizar com o projeto local via git.....	17

Visualizar no GitHub.....	19
Adicionar o README no GitHub.....	19
Baixar para o projeto.....	20
<b>INICIAR O DESENVOLVIMENTO DO PROJETO.....</b>	<b>21</b>
STUDENT_CLASSE.....	21
DIAGRAMA DE CLASSE.....	21
IMPLEMENTAR A ESTRUTURA E A CLASSE STUDENT NO STS.....	22
Criar a estrutura e classe.....	22
Definir o Serializable e os atributos da Classe.....	22
Criar o construtor.....	24
Criar os Getters and Setters.....	25
Criar o hashCode and equals.....	27
Github-2.....	28
STUDENT_RESOURCE.....	29
CONCEITUAL.....	29
IMPLEMENTAR A ESTRUTURA E O RECURSO STUDENT_RESOURCE.....	30
Criar a estrutura e o recurso REST.....	30
Implementar as Notações Rest.....	30
Criar o Endpoint findAll.....	31
Rodar o projeto.....	31
Testar com o Postman.....	32
Github-3.....	33
STUDENT_REPOSITORY.....	34
CONCEITUAL.....	34
IMPLEMENTAR A ESTRUTURA E O STUDENT_REPOSITORY.....	35
Criar a estrutura e a interface de acesso ao banco.....	35
Implementar a notação e estender a JPA.....	35
Github-4.....	36
STUDENT_SERVICE.....	37
CONCEITUAL.....	37
IMPLEMENTAR A ESTRUTURA E O STUDENT_SERVICE.....	38
Criar a estrutura e a classe de serviço.....	38
Implementar a lógica para o findAll.....	39
Github-5.....	40
INTEGRAÇÃO COM O BANCO.....	41
AJUSTES.....	41
Implementar o StudentResource.....	41
Implementar a classe Student.....	42
Rodar o projeto.....	42
Testar com o Postman.....	42
IMPLEMENTAR O BANCO H2.....	43

Configurar o perfil de teste.....	43
Criar o arquivo de configuração para teste.....	44
Copiar o código de configuração para dentro do arquivo:.....	45
Rodar o projeto.....	45
Acessar o banco H2, via web.....	46
Teste de inserção.....	47
Testar no postman.....	47
Github-6.....	48
SEEDING DA BASE DE DADOS.....	49
IMPLEMENTAR A CARGA PARA O BANCO.....	49
Criar o import.sql, no src/main/resources.....	49
Implementar os inserts para a carga inicial.....	50
Rodar o projeto.....	50
TESTAR A CARGA.....	51
Testar no banco H2.....	51
Testar no Postman.....	52
Github-7.....	53
DTO.....	54
CONCEITUAL.....	54
IMPLEMENTAR A ESTRUTURA E O STUDENT_DTO.....	55
Criar a estrutura e a classe DTO.....	55
Implementar o Serializable e os mesmos atributos da Classe Student.....	55
Implementar os construtores.....	56
Vazio.....	56
de Classe.....	56
De Entidade.....	57
Implementar os Getters and Setters.....	57
REALIZAR OS AJUSTES PARA O DTO.....	59
Implementar o DTO na classe StudentService.....	59
Implementar o DTO na classe StudentResource.....	59
Rodar o projeto.....	60
TESTES.....	60
Testar com o Postman.....	60
Github-8.....	61
ENDPOINT - FIND_BY_ID.....	62
BUSCAR ALUNOS POR ID COM GET.....	62
Implementar busca por Id, no StudentResource.....	62
Implementar o método findById, no StudentService.....	62
Rodar o projeto.....	63
TESTES.....	64
Testar no Postman.....	64

Github-9.....	64
TRATAMENTO DE EXCEÇÕES PARA O FIND_BY_ID.....	66
Rodar o projeto.....	66
Simular erro no Postman.....	66
Verificar o erro no console.....	66
Criar a estrutura de exceções no service e uma exceção personalizada.....	67
Implementar o ResourceNotFoundException.....	67
Implementar a exceção no findById, do StudentService.....	68
Criar a estrutura de exceções no resource e uma classe personalizada.....	68
Implementar o Serializable e os atributos do StandardError.....	69
Implementar um construtor vazio.....	69
Implementar os Getters and Setters.....	70
Criar um controller advice para manipular a exceção.....	72
Implementar o ResourceExceptionHandler.....	72
Rodar o projeto.....	73
TESTES.....	73
Testar no Postman.....	73
Github-10.....	74
PAGINAÇÃO.....	75
AJUSTAR O FIND_ALL PARA BUSCA PAGINADA.....	75
Implementar a busca paginada, no StudentResource.....	75
Ajustar a busca paginada, no StudentService.....	75
Expandir o seed do banco para teste.....	76
Rodar o projeto.....	76
TESTES.....	77
Testar no Postman.....	77
Github-11.....	78
ENDPOINT - INSERT.....	79
INSERIR NOVO ALUNO COM POST.....	79
Implementar o insert, no StudentResource.....	79
Implementar a metodologia REST ao método.....	79
Implementar o método insert convertendo o DTO para uma entidade, no StudentService....	80
Rodar o projeto.....	80
TESTES.....	81
Testar no Postman.....	81
Inserir.....	81
Buscar por Id.....	81
Github-12.....	82
ENDPOINT - UPDATE.....	83
ATUALIZAR ALUNO COM PUT.....	83
Implementar o update, no StudentResource.....	83

---

Implementar o método update, no StudentService.....	83
TRATAMENTO DE ERRO.....	84
Implementar o tratamento para ID Não encontrado.....	84
Rodar o projeto.....	84
TESTES.....	85
Testar no Postman.....	85
Update.....	85
Busca por ID.....	85
Github-13.....	86
ENDPOINT - DELETE.....	87
DELETAR UM ALUNO COM DELETE.....	87
Implementar o update, no StudentResource.....	87
Implementar o método update, no StudentService.....	87
TRATAMENTO DE ERRO DO DELETE.....	88
Implementar o tratamento para ID Não encontrado.....	88
Rodar o projeto.....	88
TESTES.....	88
Testar no Postman.....	88
Github-14.....	89

## DADOS PARA A DISCIPLINA

- **DISCIPLINA:** Arquitetura e Frameworks para Desenvolvimento Web
- **EQUIPE:**
  - Auricelio Freitas Moreira
  - Miguel Stenio Soares Cavalcante

## ATIVIDADE 1

### OBJETIVO

Criar uma API RESTful que abstraia os detalhes de implementação, expondo claramente as funcionalidades e operações que oferece, independentemente da linguagem de programação utilizada. Este projeto é uma chance de aplicar conceitos de abstração, garantindo que a interface da API seja intuitiva e fácil de usar, sem revelar os mecanismos internos de como as operações são realizadas.

### O QUE É ESSENCIAL?

- A API deve ser desenhada seguindo rigorosamente os princípios REST, focando em expor "o quê" sua API faz, ao invés de "como" faz. Pode usar qualquer linguagem ou Framework. Uso de banco de dados, para esta atividade, é opcional.
- A documentação deve ser clara e detalhada, oferecendo a usuários/desenvolvedores todas as informações necessárias para entender e interagir com a API. Isso inclui a descrição das rotas, métodos HTTP, formatos de requisições/respostas e exemplos práticos de uso.

## CRITÉRIOS DE AVALIAÇÃO (Rúbricas do Google Class)

- **Critério 1** - Adesão aos Princípios REST: *20 pontos*
- **Critério 2** - Documentação: *15 Pontos*
- **Critério 3** - Funcionamento da API: *15 Pontos*
- **Total:** *50 pontos.*

## ENTREGA

- Formulário para entrega: <https://forms.gle/9yQ97vi9zF8WeWDn9>
  - Obs.: Certifique-se de incluir no repositório Git da sua API um arquivo README, que inclua instruções para configuração, instalação, e um guia de como consumir a API. O README deve também refletir a ênfase na abstração, destacando "o quê" a API faz.
- Prazo: 06/04.
- Faremos essa atividade em sala. Quem puder entregar antes, excelente. Quem optar por fazer em sala comigo, tudo bem também. Faremos juntos, utilizando Spring Boot no backend e React no frontend.

Boa sorte! Estou disponível para esclarecer quaisquer dúvidas e oferecer suporte durante o desenvolvimento do projeto.

## CONTATOS

- (85) 99614.5615 (Whatsapp)
- [marcoseduardoss@gmail.com](mailto:marcoseduardoss@gmail.com)

# PROJETO CONCEITUAL

## DEPENDÊNCIAS

O projeto será implementado conforme abaixo:

- **IDE:** Spring Tool Suite 4 (STS)
- **Banco de Dados:** H2
- **Gerenciador de Dependências:** Maven
- **Linguagem:** Java
- **Versionamento:** GitHub
- **Testes da API:** Postman

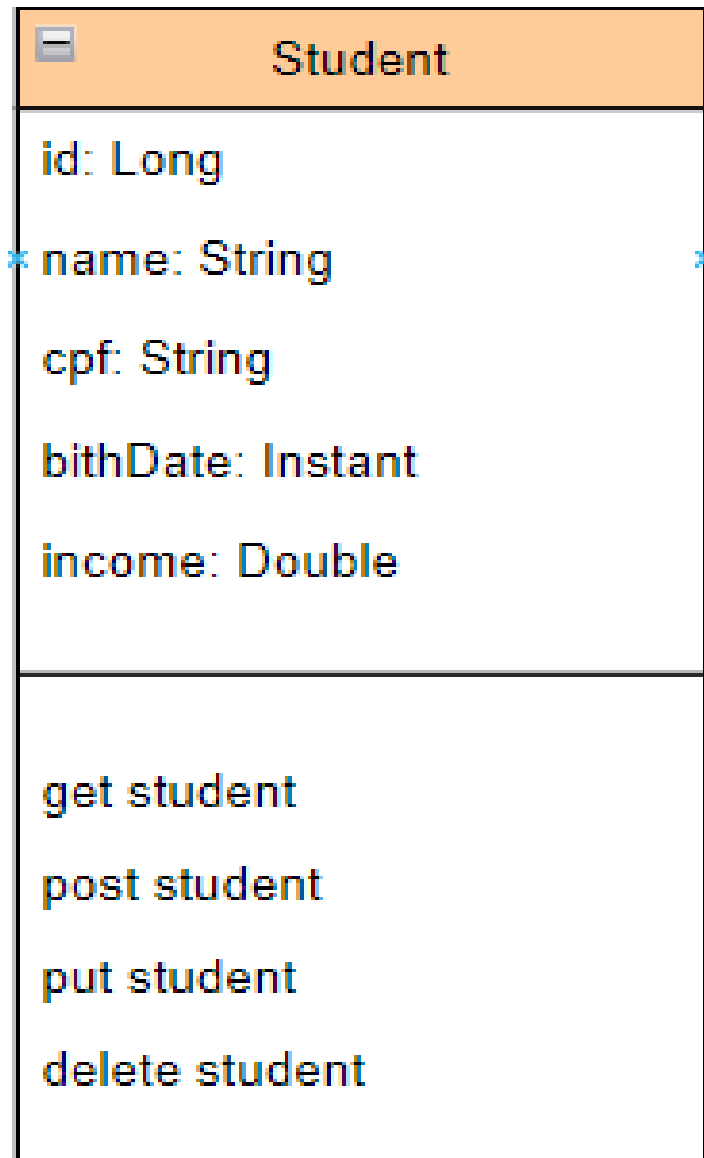
## CASO

Uma Universidade precisa cadastrar os seus alunos, conforme segue abaixo:

- NOME
- CPF
- DATA DE NASCIMENTO
- RENDA



## DIAGRAMA DE CLASSE



## TESTES A SEREM REALIZADOS NO POSTMAN

### BUSCA PAGINADA DE ALUNOS

- GET /students?page=0&sort=name,asc

### BUSCA DE ALUNO POR ID

- GET /students/1

### INSERIR NOVO ALUNO

- POST /students  
{  
 "name": "Auricelio Freitas",  
 "cpf": "12345678901",  
 "birthDate": "1982-08-28T10:30:00Z",  
 "income": 15089.0  
}

### ATUALIZAR ALUNO

- PUT /students/1  
{  
 "name": "Auricelio Moreira",  
 "cpf": "12345678901",  
 "birthDate": "1982-08-28T10:30:00Z",  
 "income": 15089.0  
}

### DELETAR ALUNO

- DELETE /students/1

# CRIAR O PROJETO API RESTFUL

## DEFINIR O WORKSPACE PARA O PROJETO

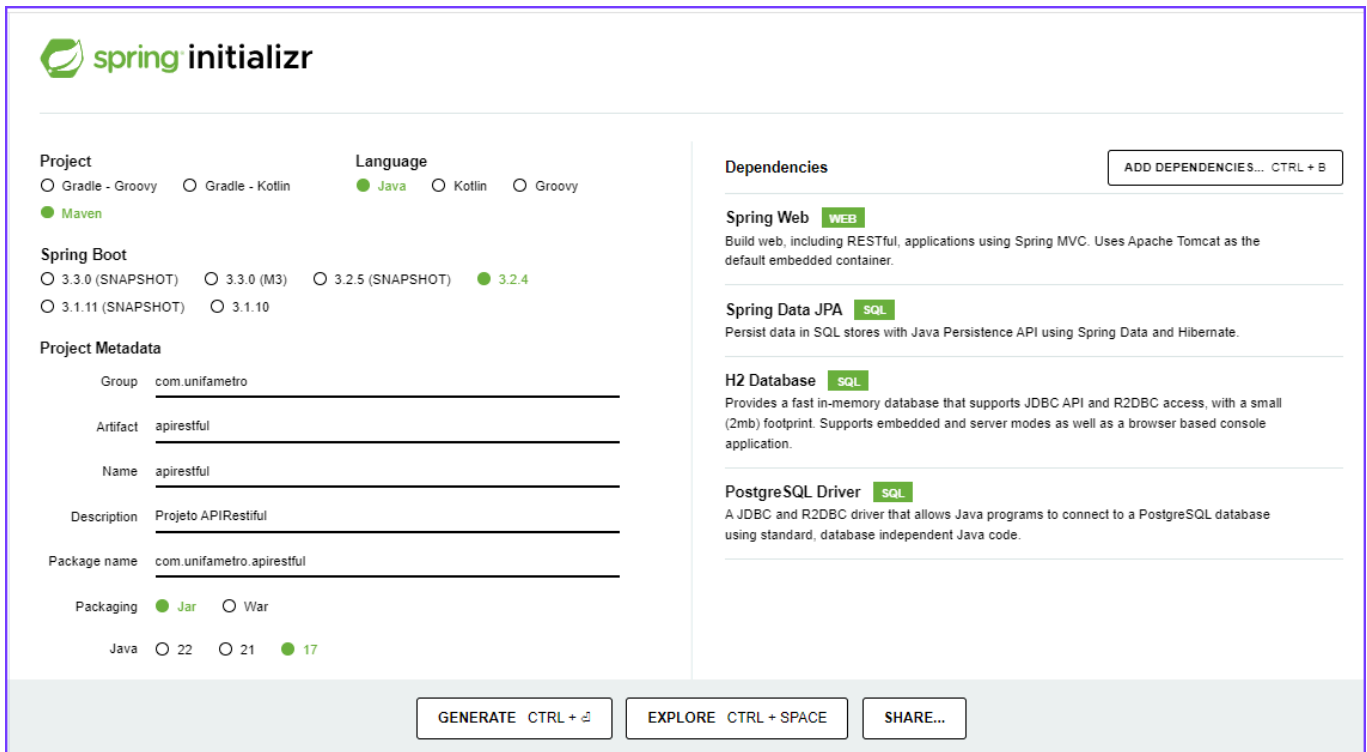
- C:\PROJETOS\UNIFAMETRO\APIRestful\

## CRIAR O PROJETO COM O INITIALIZR

### ACESSAR A URL

- <https://start.spring.io/>

### GERAR O PROJETO



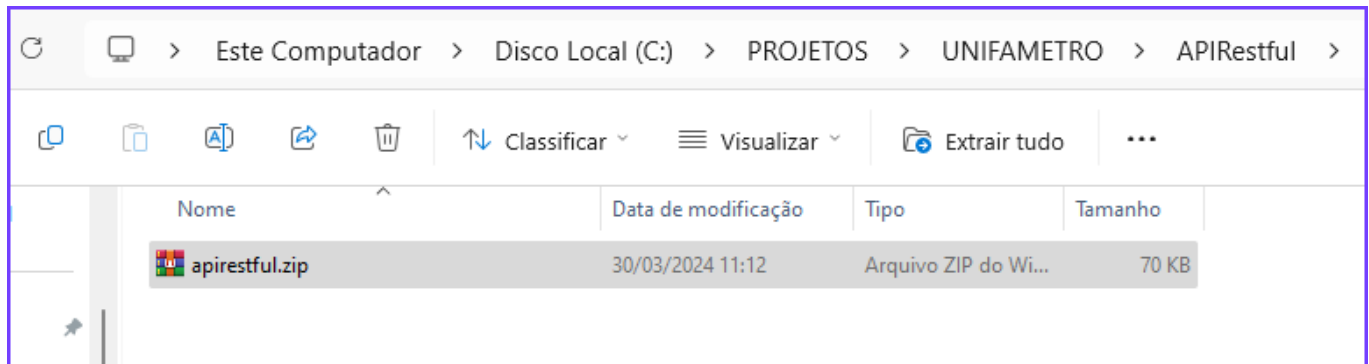
The image shows the Spring Initializr web interface. It is a form for generating a Spring project. The interface is divided into several sections:

- Project:** Includes radio buttons for **Gradle - Groovy**, **Gradle - Kotlin**, and **Maven** (selected).
- Language:** Includes radio buttons for **Java** (selected), **Kotlin**, and **Groovy**.
- Spring Boot:** Includes radio buttons for **3.3.0 (SNAPSHOT)**, **3.3.0 (M3)**, **3.2.5 (SNAPSHOT)**, **3.2.4** (selected), **3.1.11 (SNAPSHOT)**, and **3.1.10**.
- Project Metadata:** Includes text input fields for **Group** (com.unifametro), **Artifact** (apirestful), **Name** (apirestful), **Description** (Projeto APIRestful), and **Package name** (com.unifametro.apiestful).
- Packaging:** Includes radio buttons for **Jar** (selected) and **War**.
- Java:** Includes radio buttons for **22**, **21**, and **17** (selected).
- Dependencies:** Includes a button **ADD DEPENDENCIES... CTRL + B** and a list of dependencies:
  - Spring Web** (WEB): Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
  - Spring Data JPA** (SQL): Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.
  - H2 Database** (SQL): Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.
  - PostgreSQL Driver** (SQL): A JDBC and R2DBC driver that allows Java programs to connect to a PostgreSQL database using standard, database independent Java code.

At the bottom, there are three buttons: **GENERATE CTRL + G**, **EXPLORE CTRL + SPACE**, and **SHARE...**

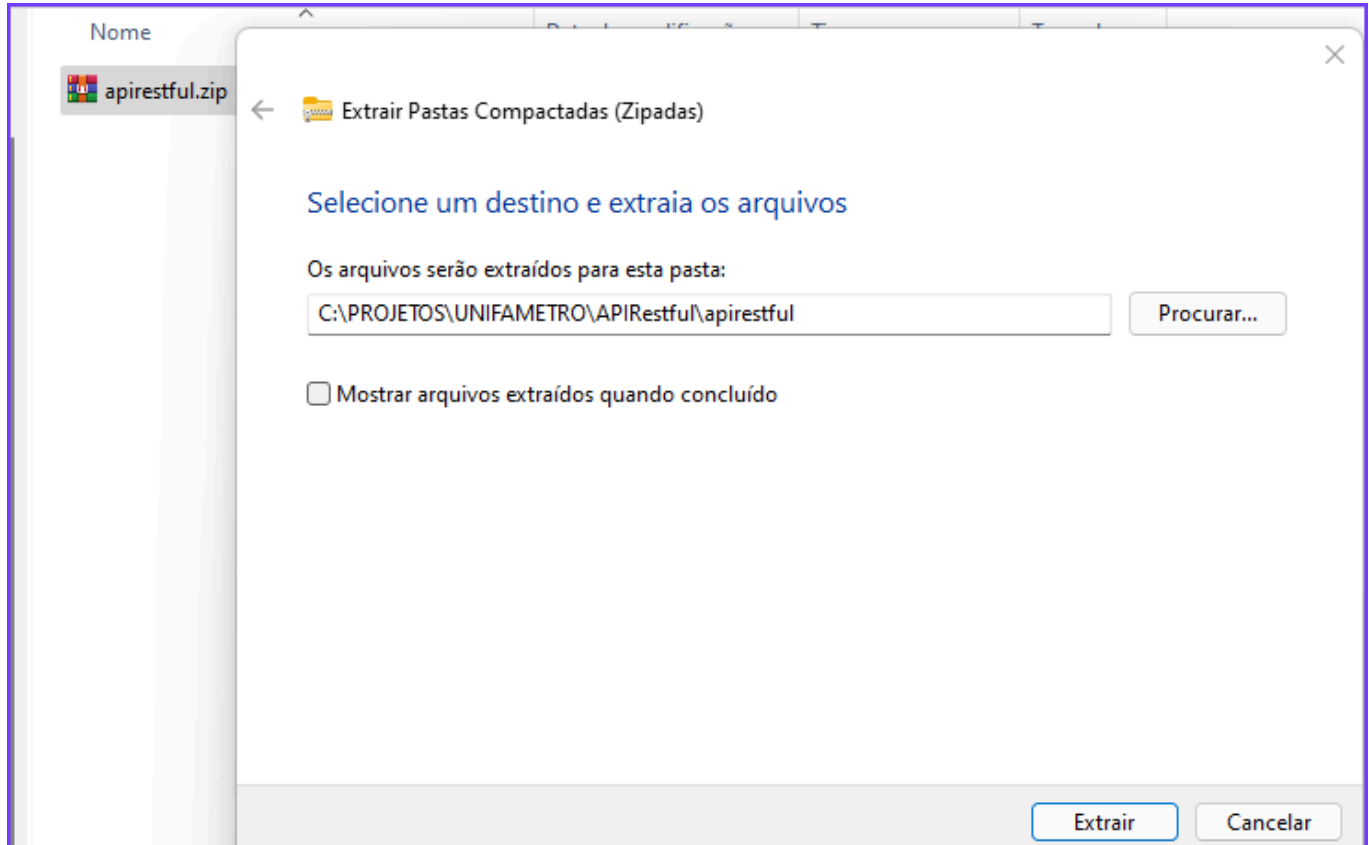
## Salvar dentro do workspace

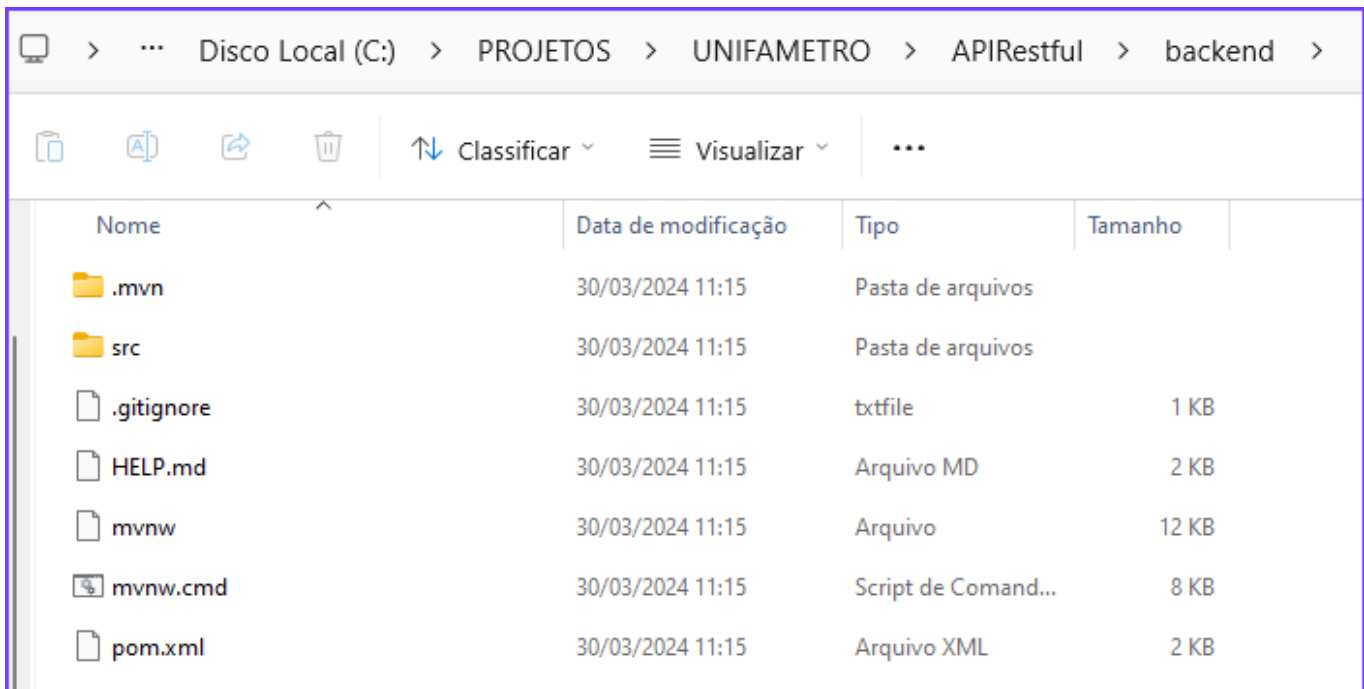
- C:\PROJETOS\UNIFAMETRO\APIRestful\



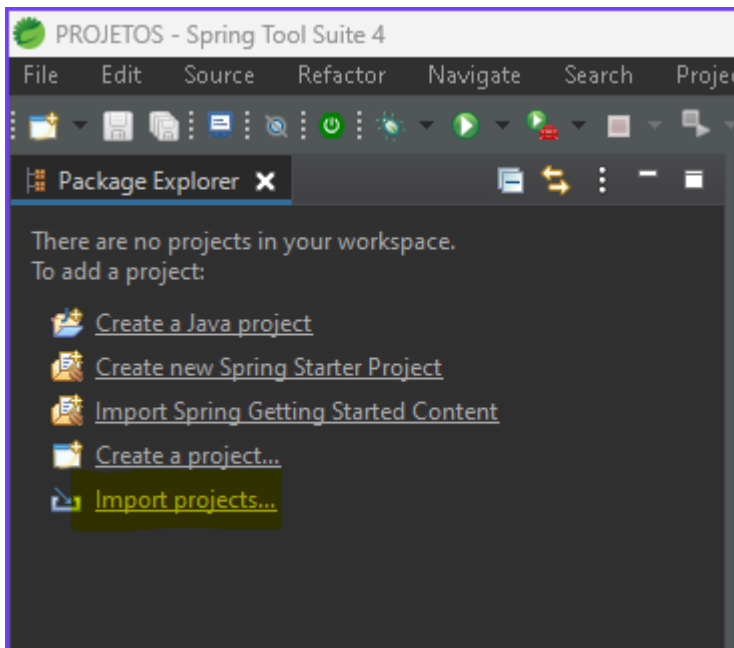
## Descompactar e Renomear

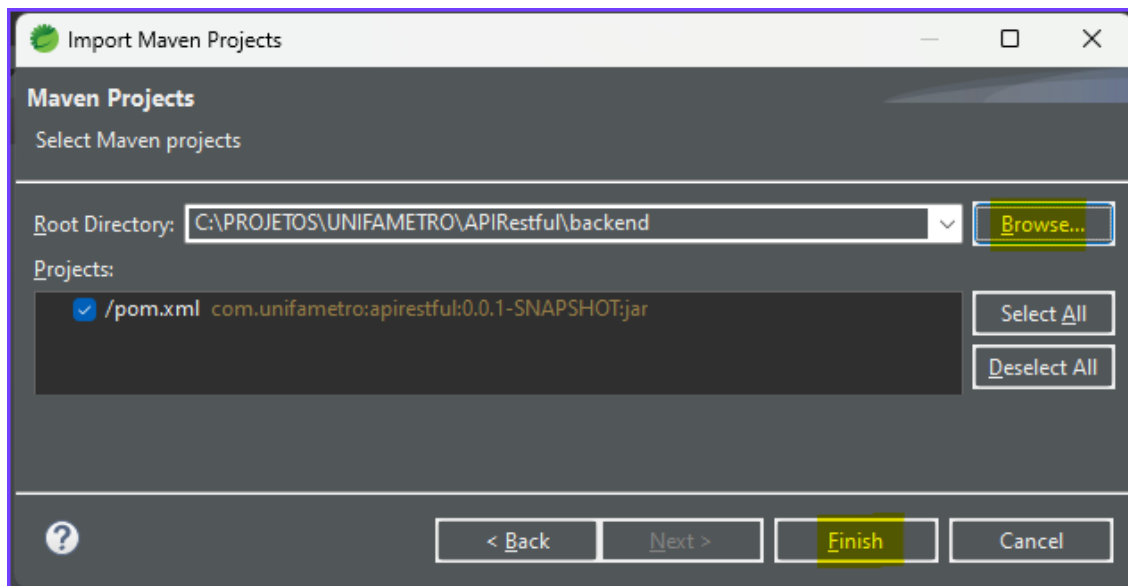
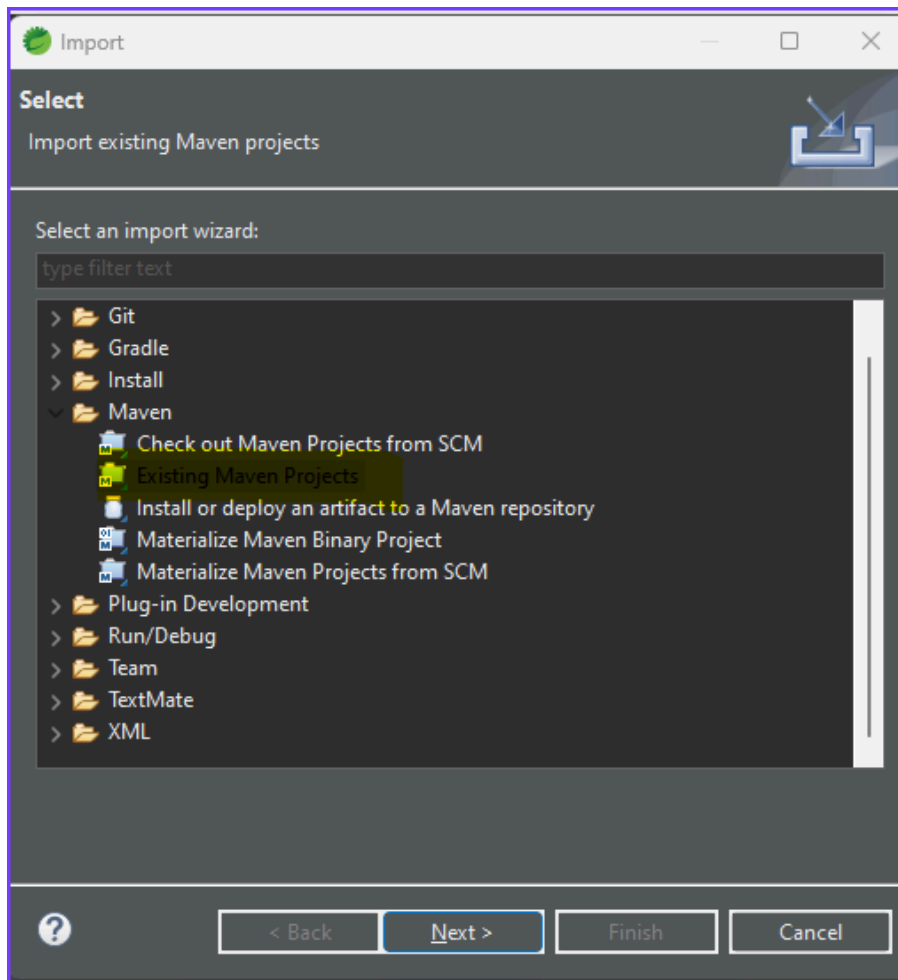
- backend



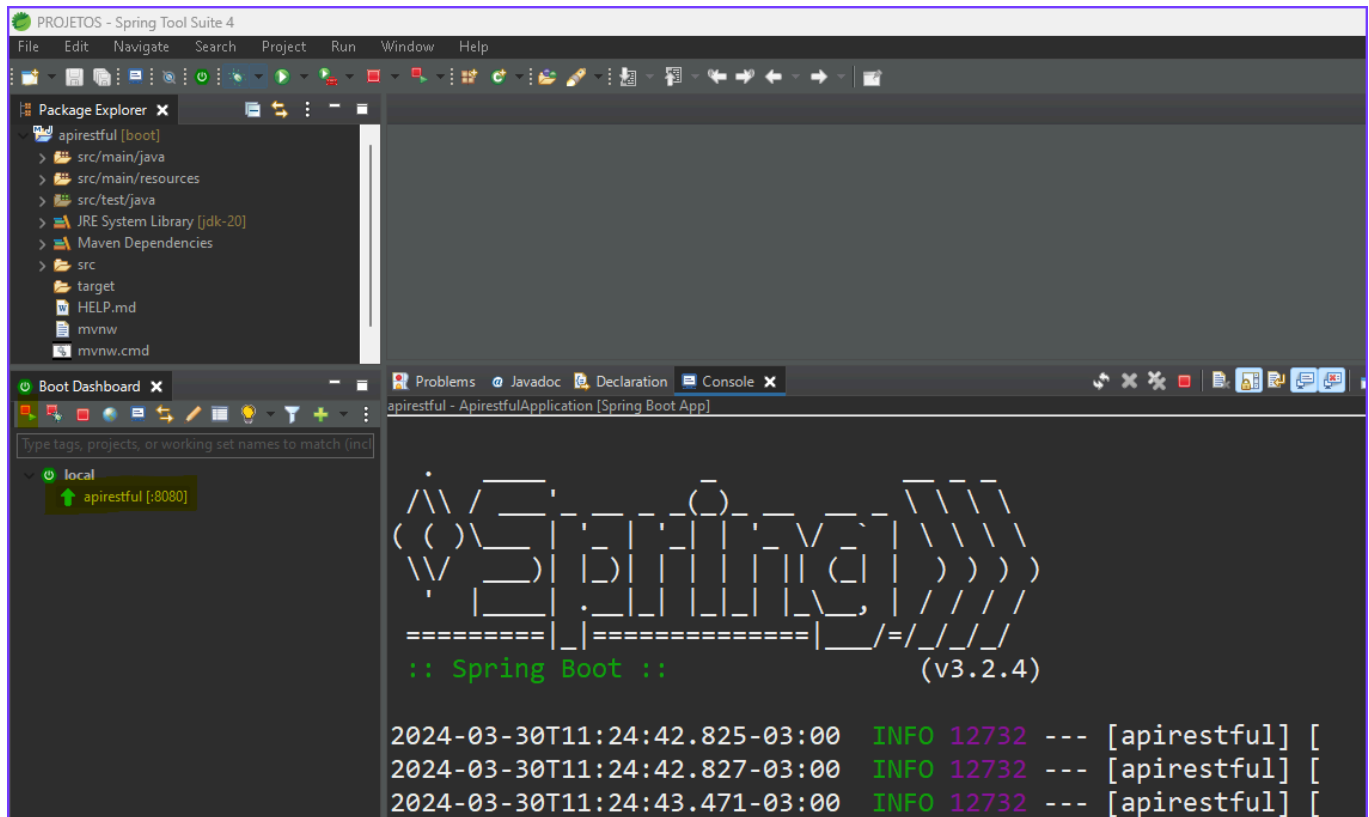


## IMPORTAR O PROJETO PARA O STS



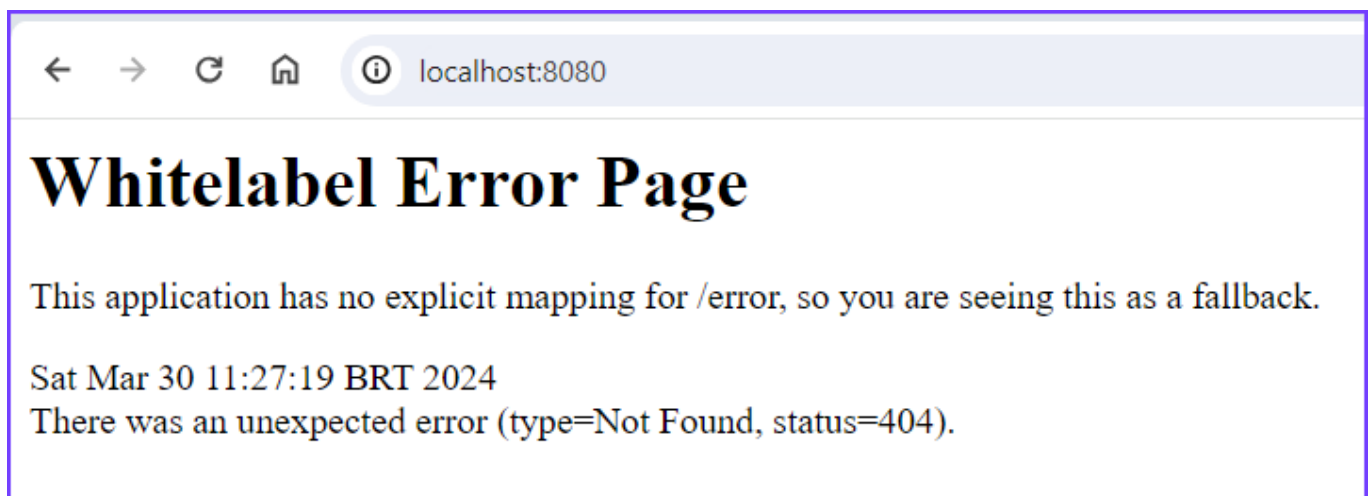


## Rodar o projeto



## TESTAR NO BROWSER

- URL: <http://localhost:8080/>



## Github-1

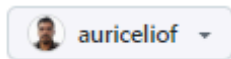
### Criar o projeto no Github

#### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (\*).

Owner \*



Repository name \*

/ unifametro-afdpw-APIRes

✓ unifametro-afdpw-APIRestful is available.

Great repository names are short and memorable. Need inspiration? How about [silver-journey](#) ?

Description (optional)

UNIFAMETRO - PÓS - ARQUITETURA E FRAMEWORKS PARA DESENVOLVIMENTO DE PLATAFORMAS WEB - AF



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.



unifametro-afdpw-APIRestful

Public



Unwatch 1



Fork 0



Star 0



##### Set up GitHub Copilot

Use GitHub's AI pair programmer to autocomplete suggestions as you code.

[Get started with GitHub Copilot](#)



##### Add collaborators to this repository

Search for people using their GitHub username or email address.

[Invite collaborators](#)

#### Quick setup — if you've done this kind of thing before

Set up in Desktop or ☐ HTTPS ☐ SSH

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

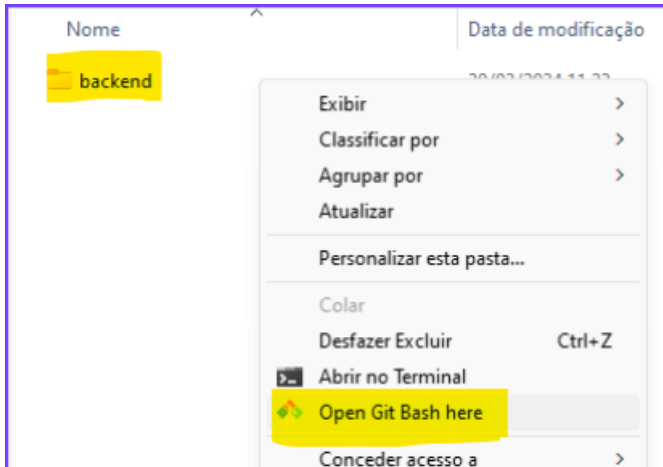
#### ...or create a new repository on the command line

```
echo "# unifametro-afdpw-APIRestful" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
git push -u origin main
```

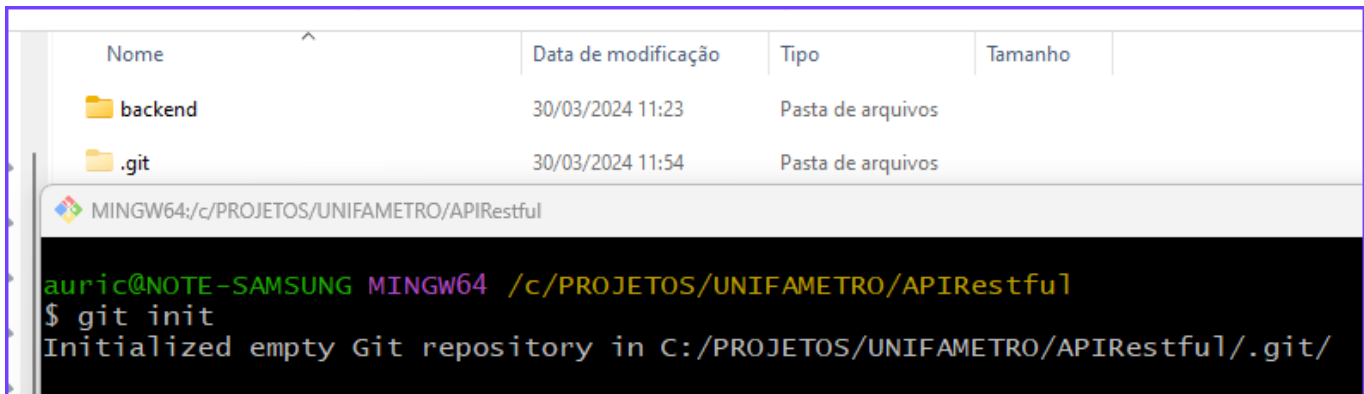


## Sincronizar com o projeto local via git

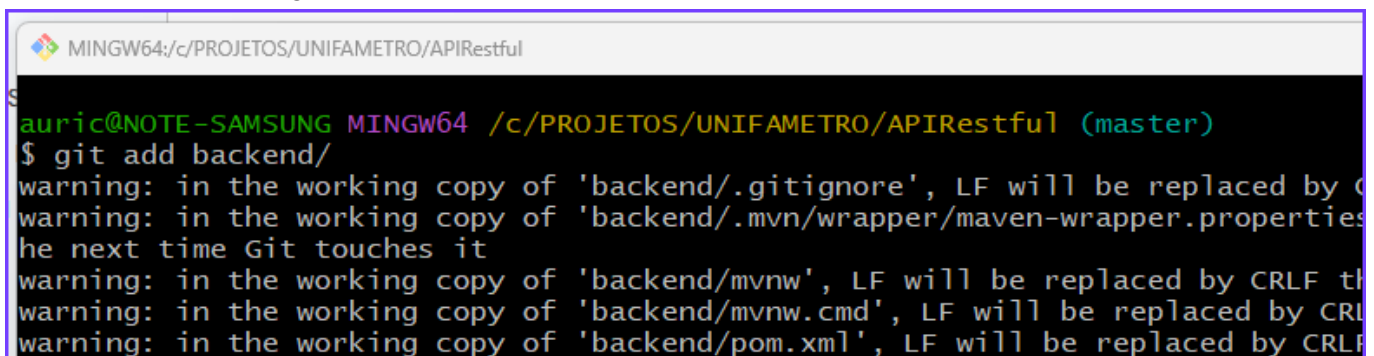
- “Open Git bash here” no diretório do projeto.



- git init



- git add backend



- `git commit -m "Initial backend"`

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (master)
$ git commit -m "initial backend"
[master (root-commit) e8fc979] initial backend
 9 files changed, 630 insertions(+)
 create mode 100644 backend/.gitignore
 create mode 100644 backend/.mvn/wrapper/maven-wrapper.jar
 create mode 100644 backend/.mvn/wrapper/maven-wrapper.properties
 create mode 100644 backend/mvnw
 create mode 100644 backend/mvnw.cmd
 create mode 100644 backend/pom.xml
```

- `git branch -M main`

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (master)
$ git branch -M main

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$
```

- `git remote add origin https://github.com/auriceliof/unifametro-afdpw-APIRestful.git`

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful

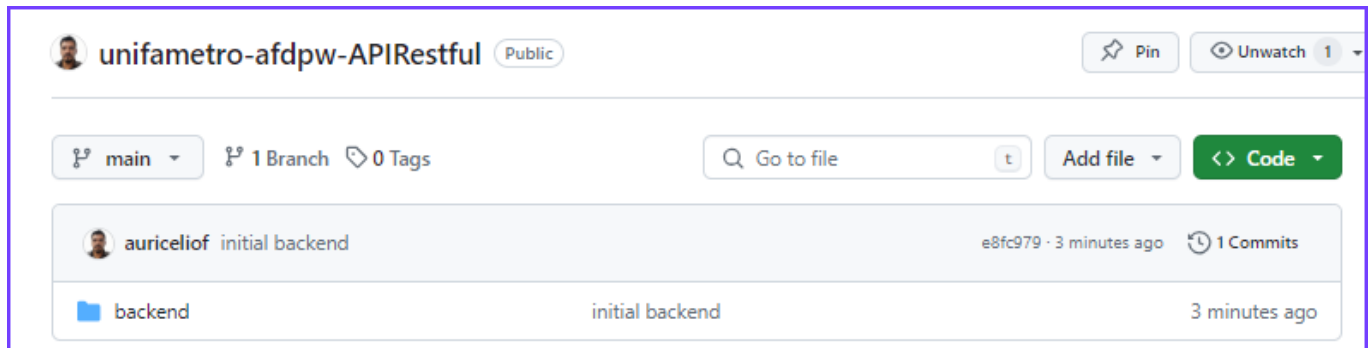
auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git remote add origin https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
```

- `git push -u origin main`

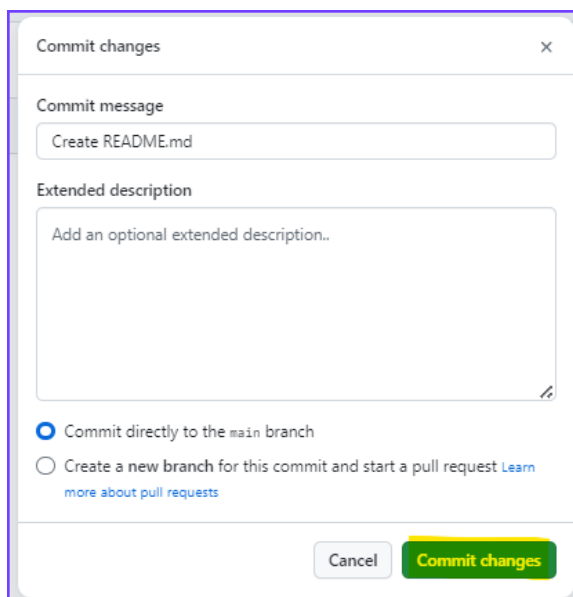
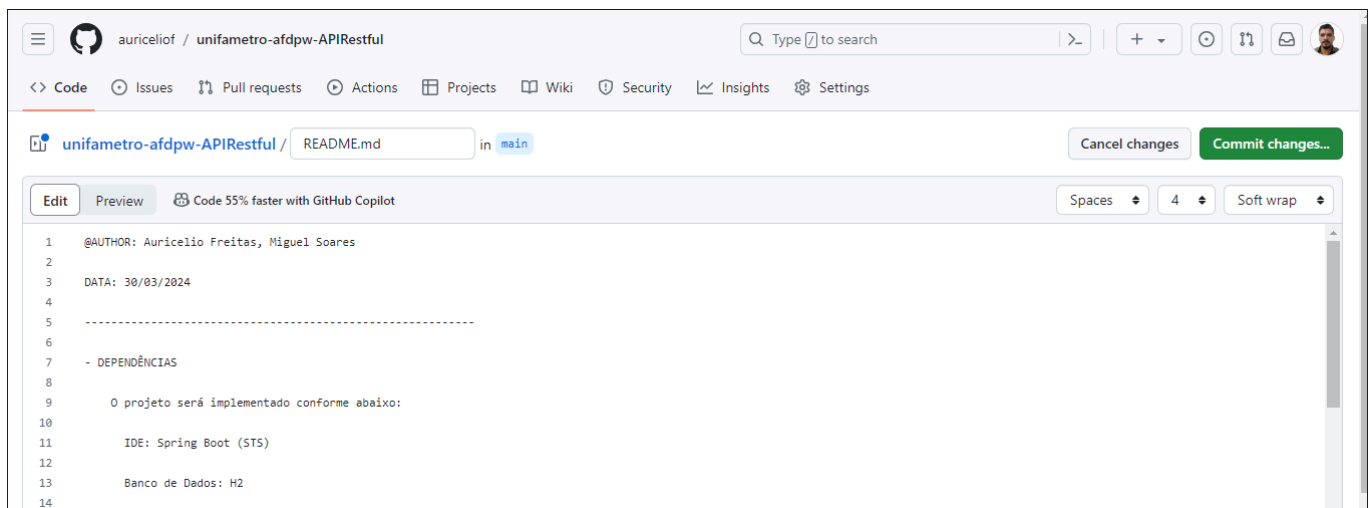
```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful

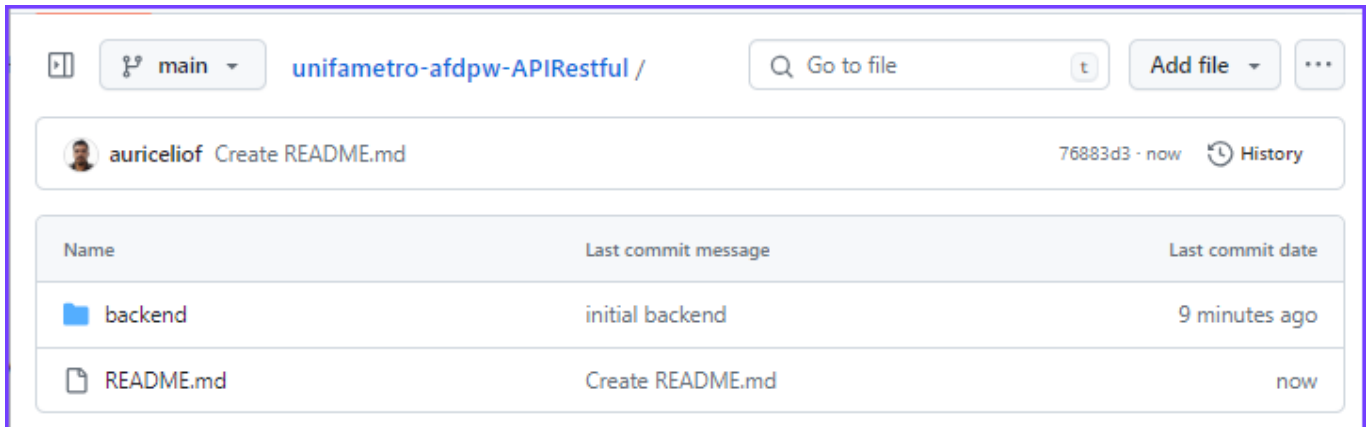
auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push -u origin main
Enumerating objects: 26, done.
Counting objects: 100% (26/26), done.
Delta compression using up to 8 threads
Compressing objects: 100% (16/16), done.
Writing objects: 100% (26/26), 63.01 KiB | 12.60 MiB/s, done.
Total 26 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

## Visualizar no GitHub



## Adicionar o README no GitHub



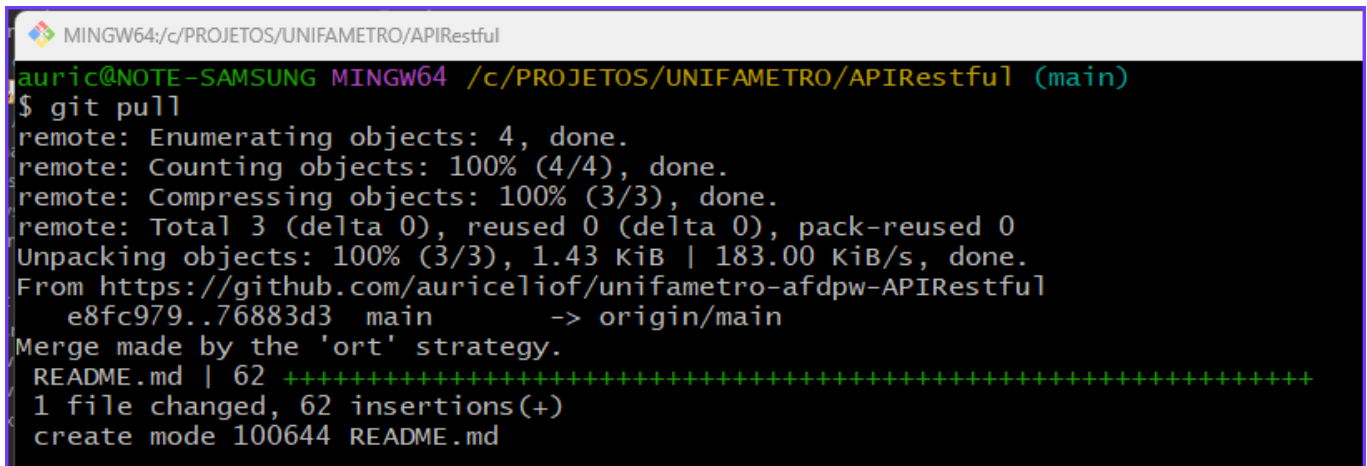


The screenshot shows the GitHub interface for the repository 'unifametro-afdpw-APIRestful'. At the top, there's a navigation bar with a dropdown menu set to 'main', a search bar 'Go to file', and buttons for 'Add file' and a three-dot menu. Below this, a commit summary shows 'auriceliof' created 'README.md' with commit hash '76883d3' 'now'. A 'History' link is next to it. A table below lists the commit history:

Name	Last commit message	Last commit date
backend	initial backend	9 minutes ago
README.md	Create README.md	now

## Baixar para o projeto

- git pull

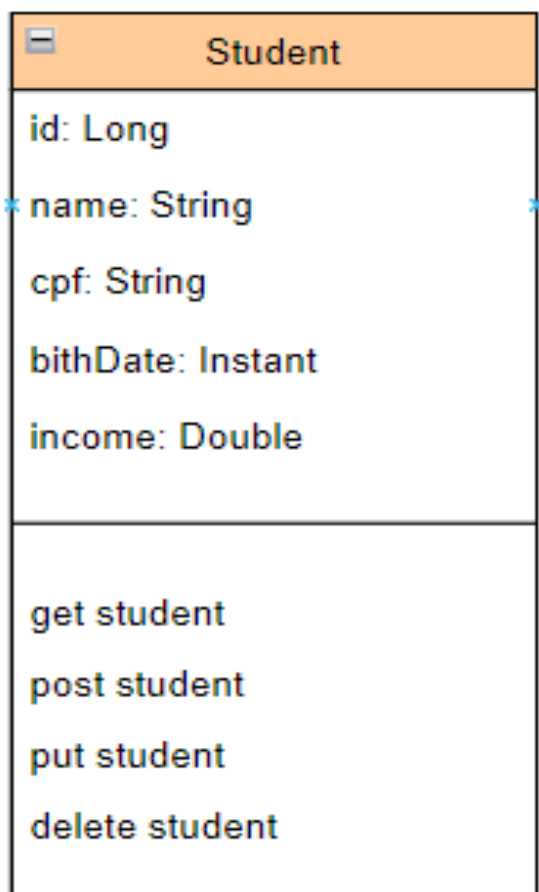


```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful
auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git pull
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 1.43 KiB | 183.00 KiB/s, done.
From https://github.com/auriceliof/unifametro-afdpw-APIRestful
   e8fc979..76883d3  main      -> origin/main
Merge made by the 'ort' strategy.
 README.md | 62 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 1 file changed, 62 insertions(+)
 create mode 100644 README.md
```

## INICIAR O DESENVOLVIMENTO DO PROJETO

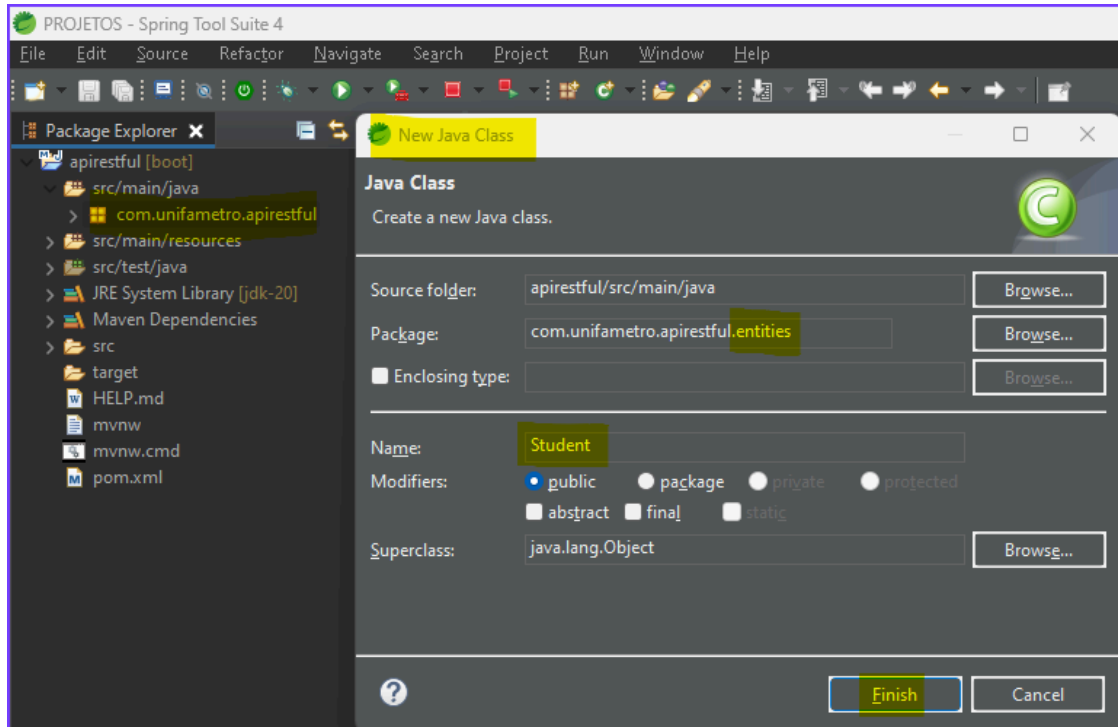
### STUDENT\_CLASSE

#### DIAGRAMA DE CLASSE

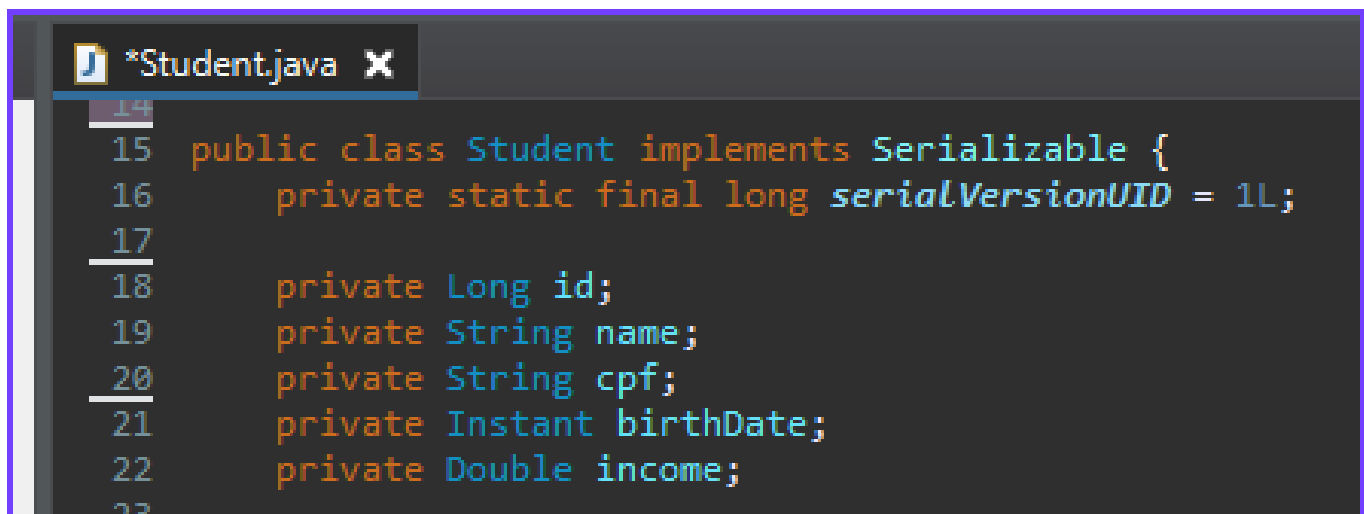


## IMPLEMENTAR A ESTRUTURA E A CLASSE STUDENT NO STS

### Criar a estrutura e classe



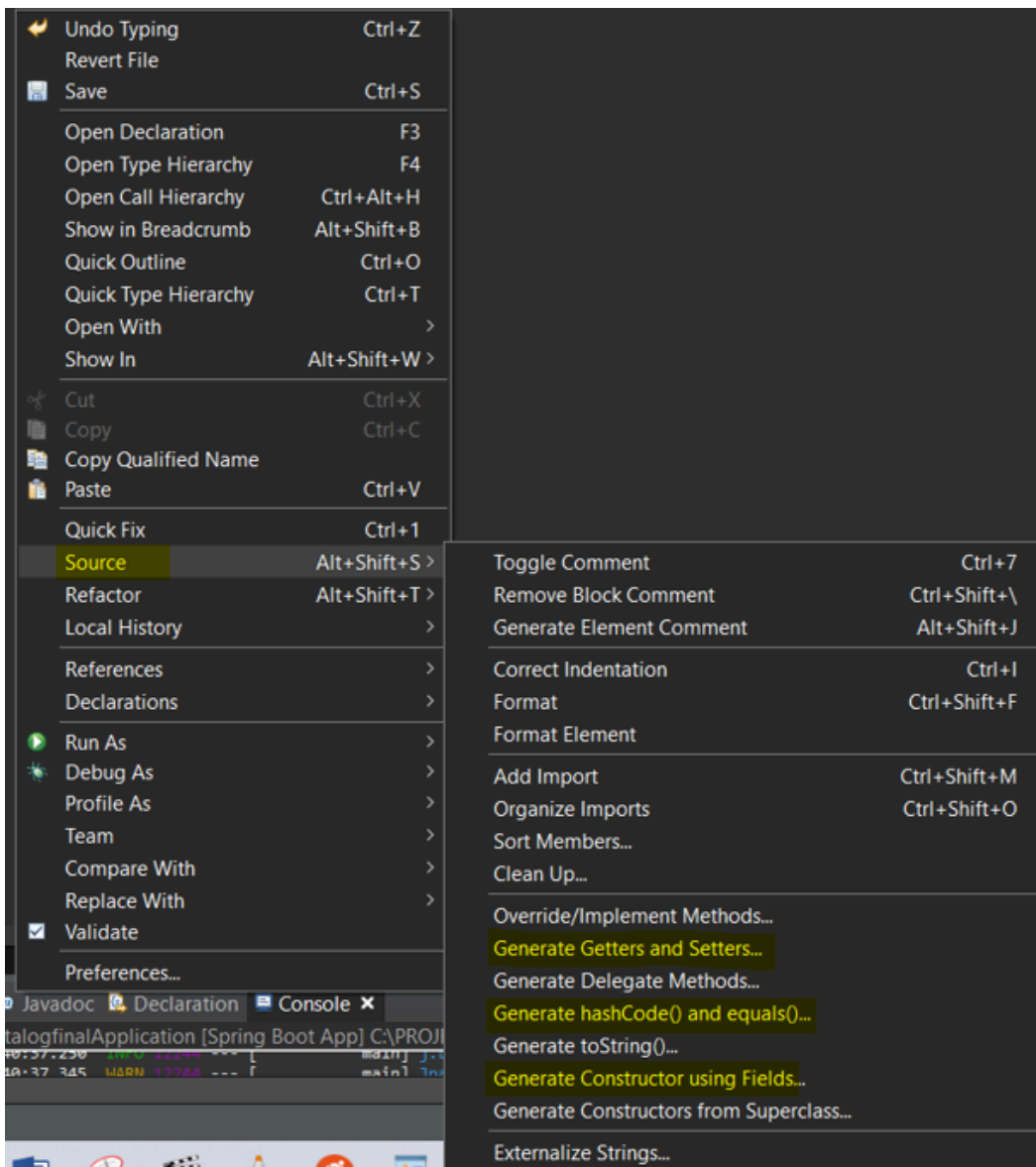
### Definir o Serializable e os atributos da Classe



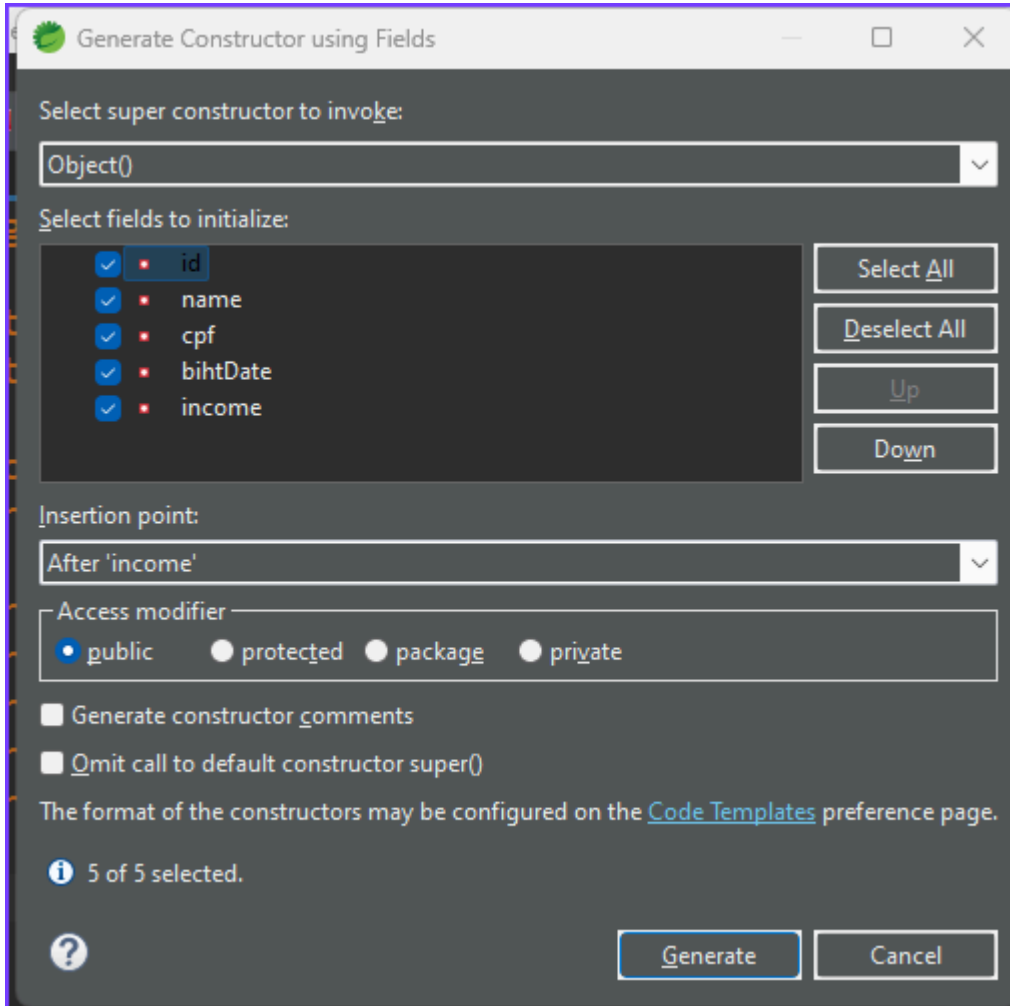
## DICA:

Para criar os construtores, getters e setters e o hashCode, proceder conforme segue:

- Clicar com o botão direito aonde quer inserir
  - Source
    - “Escolher o método que deseja implementar”
      - Selecionar os itens
        - Clicar em: Generate



## Criar o construtor



Generate Constructor using Fields

Select super constructor to invoke:

Object()

Select fields to initialize:

- ☒ id
- ☒ name
- ☒ cpf
- ☒ birthDate
- ☒ income

Select All  
Deselect All  
Up  
Down

Insertion point:

After 'income'

Access modifier:

☒ public ☐ protected ☐ package ☐ private

☐ Generate constructor comments  
☐ Omit call to default constructor super()

The format of the constructors may be configured on the [Code Templates](#) preference page.

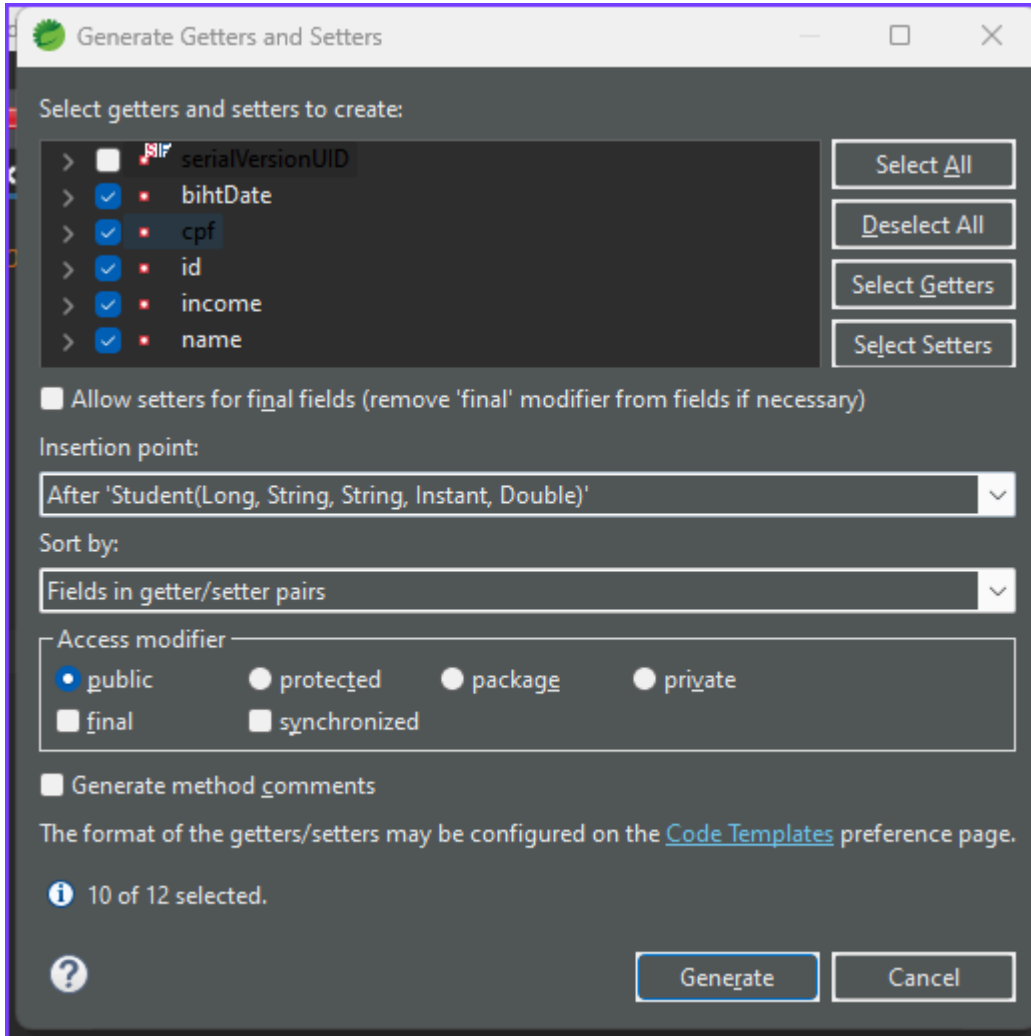
5 of 5 selected.

Generate Cancel

```
*Student.java x
28
29 public Student() {
30
31 }
32
33
34 public Student(Long id, String name, String cpf, Instant birthDate, Double income) {
35     super();
36     this.id = id;
37     this.name = name;
38     this.cpf = cpf;
39     this.birthDate = birthDate;
40     this.income = income;
41 }
42
```



## Criar os Getters and Setters



Generate Getters and Setters

Select getters and setters to create:

- ☐ serialVersionUID
- ☒ bihtDate
- ☒ cpf
- ☒ id
- ☒ income
- ☒ name

☐ Allow setters for final fields (remove 'final' modifier from fields if necessary)

Insertion point:  
After 'Student(Long, String, String, Instant, Double)'

Sort by:  
Fields in getter/setter pairs

Access modifier:  
☒ public ☐ protected ☐ package ☐ private  
☐ final ☐ synchronized

☐ Generate method comments

The format of the getters/setters may be configured on the [Code Templates](#) preference page.

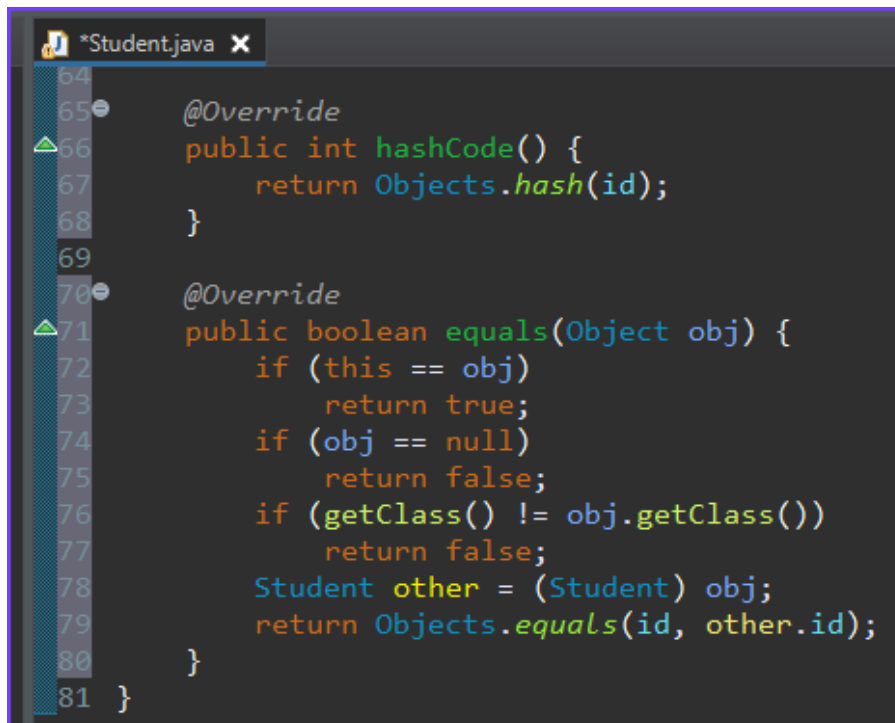
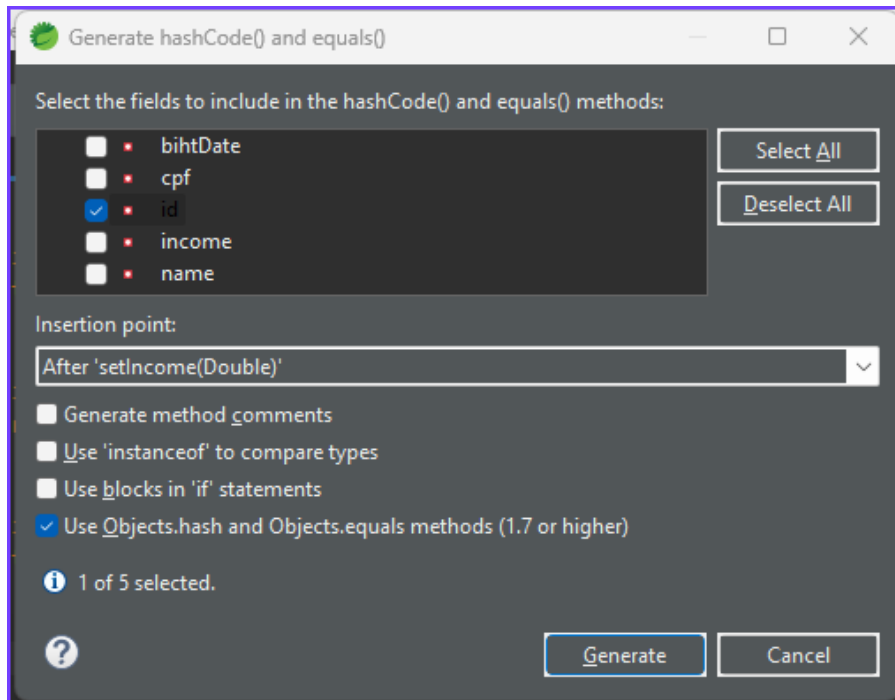
10 of 12 selected.

Generate Cancel

```
*Student.java x
23
24 public Long getId() {
25     return id;
26 }
27
28 public void setId(Long id) {
29     this.id = id;
30 }
31
32 public String getName() {
33     return name;
34 }
35
36 public void setName(String name) {
37     this.name = name;
38 }
39
```

```
Student.java x
57
58 public String getCpf() {
59     return cpf;
60 }
61
62 public void setCpf(String cpf) {
63     this.cpf = cpf;
64 }
65
66
67
68 public Instant getBirthDate() {
69     return birthDate;
70 }
71
72 public void setBirthDate(Instant birthDate) {
73     this.birthDate = birthDate;
74 }
75
76 public Double getIncome() {
77     return income;
78 }
79
80 public void setIncome(Double income) {
81     this.income = income;
82 }
83
```

## Criar o hashCode and equals




## Github-2


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Created class Student”
  - git push



```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 29, done.
Counting objects: 100% (22/22), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (13/13), 1.21 KiB | 1.21 MiB/s, done.
Total 13 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
76883d3..50ffa0d  main -> main
```

 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags  Add file Code

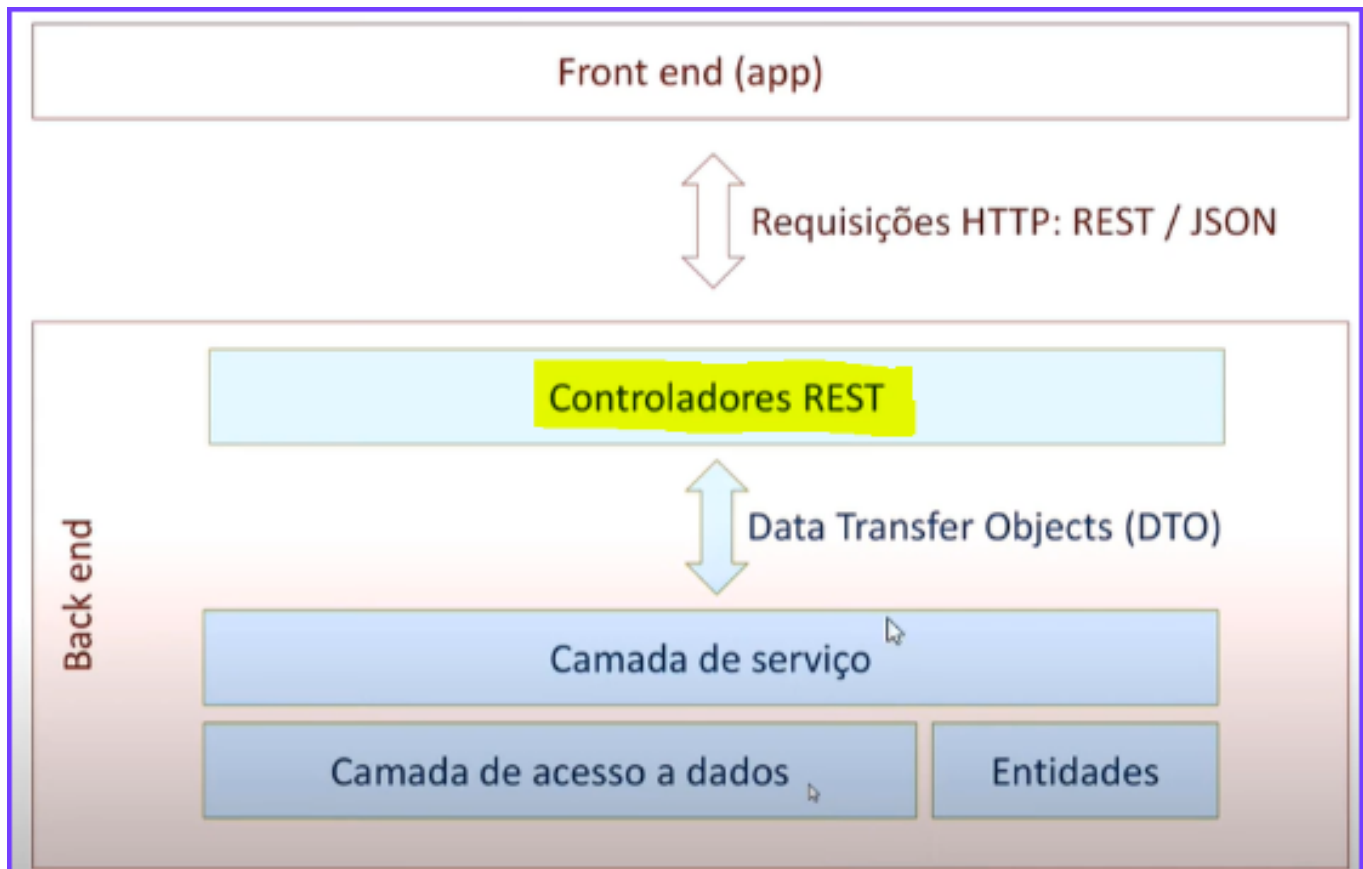
 **auriceliof** Merge branch 'main' of <https://github.com/auriceliof/unifametro-afdpw-APIRestful> 50ffa0d · 4 minutes ago 4 Commits

 backend	Created class Student	6 minutes ago
 README.md	Create README.md	40 minutes ago

## STUDENT\_RESOURCE

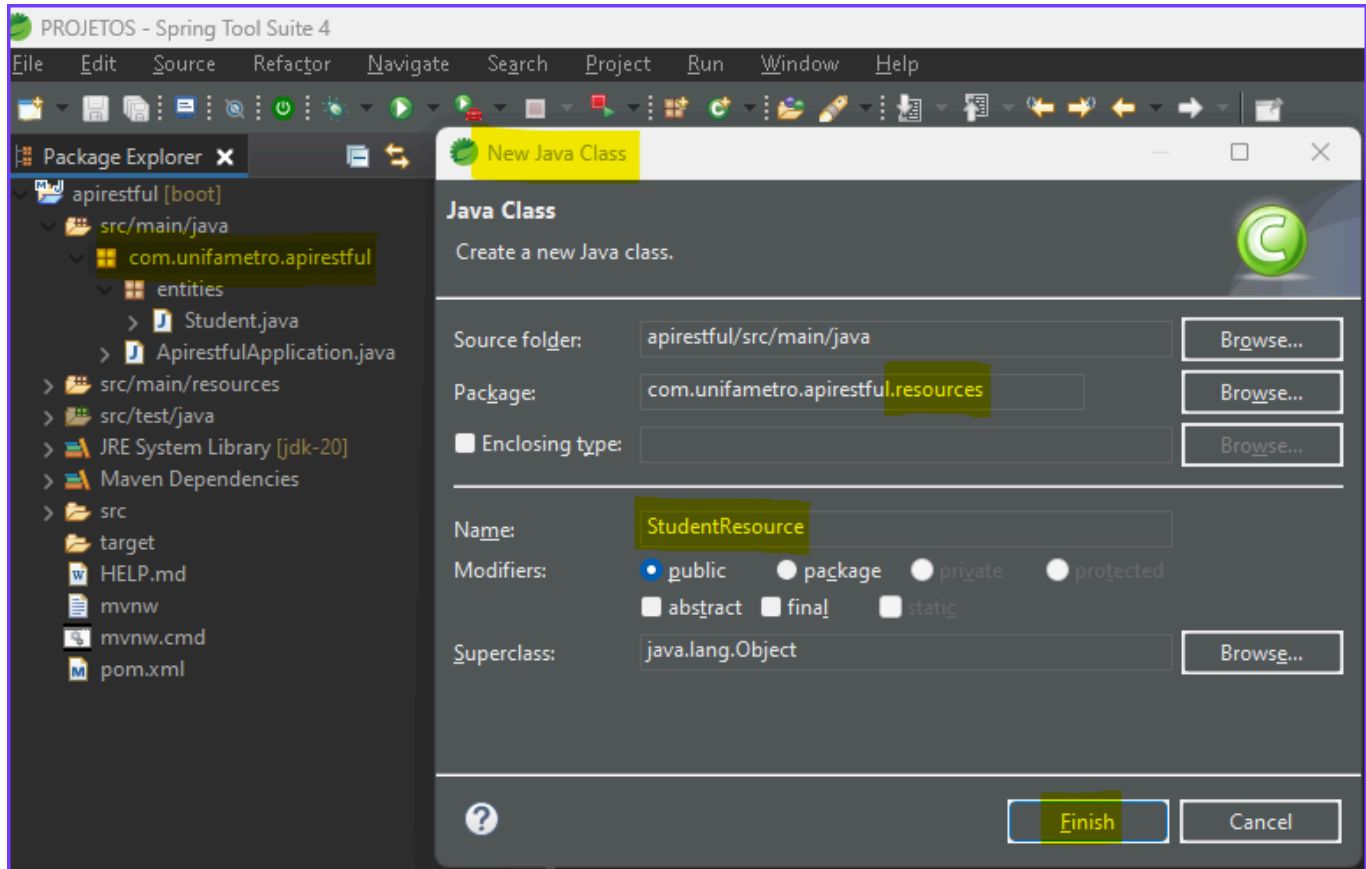
- O controlador é quem gerencia as requisições. Tanto pode ser chamado de Resources ou Controllers.

### CONCEITUAL

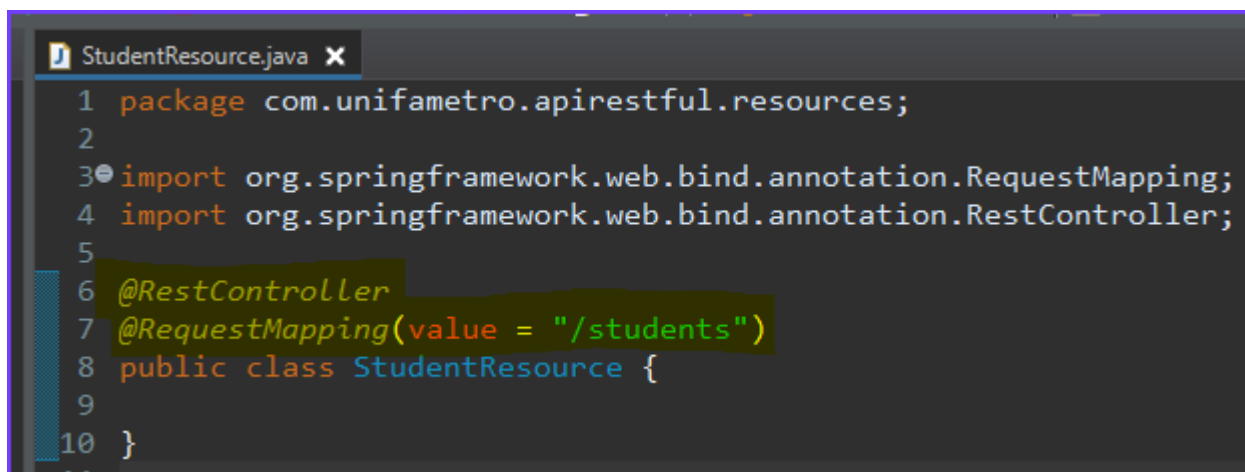


## IMPLEMENTAR A ESTRUTURA E O RECURSO STUDENT\_RESOURCE

### Criar a estrutura e o recurso REST



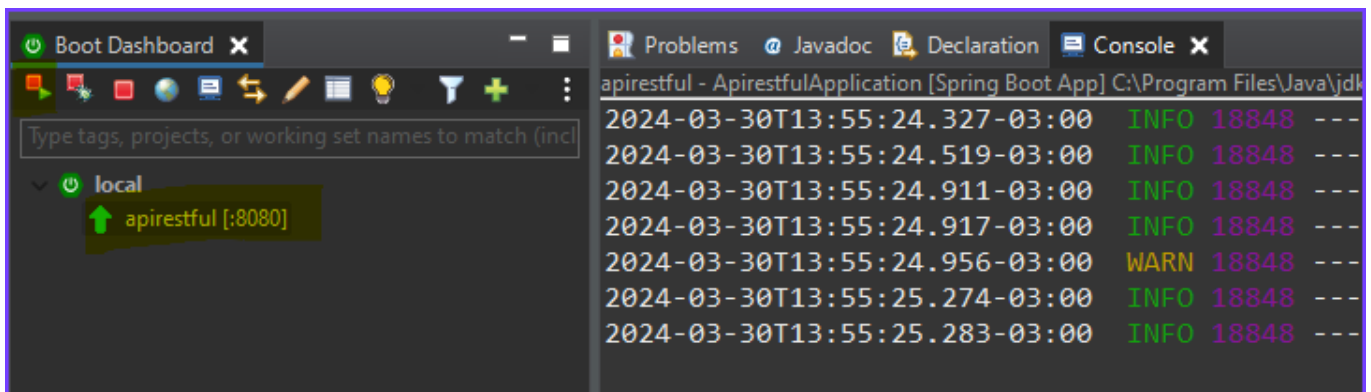
### Implementar as Notações Rest



## Criar o Endpoint findAll

```
StudentResource.java x
5
6 import org.springframework.http.ResponseEntity;
7 import org.springframework.web.bind.annotation.GetMapping;
8 import org.springframework.web.bind.annotation.RequestMapping;
9 import org.springframework.web.bind.annotation.RestController;
10
11 import com.unifametro.apirestful.entities.Student;
12
13 @RestController
14 @RequestMapping(value = "/students")
15 public class StudentResource {
16
17     @GetMapping
18     public ResponseEntity<List<Student>> findAll() {
19         List<Student> list = new ArrayList<>();
20
21         list.add(new Student(1L, "Auricelio", "123.456.789-00", null, 15049.0));
22         list.add(new Student(2L, "Miguel", "123.456.789-00", null, 20149.0));
23         list.add(new Student(3L, "Matheus", "123.456.789-00", null, 13049.0));
24         list.add(new Student(4L, "Rafel", "123.456.789-00", null, 13049.0));
25
26         return ResponseEntity.ok().body(list);
27     }
28 }
```

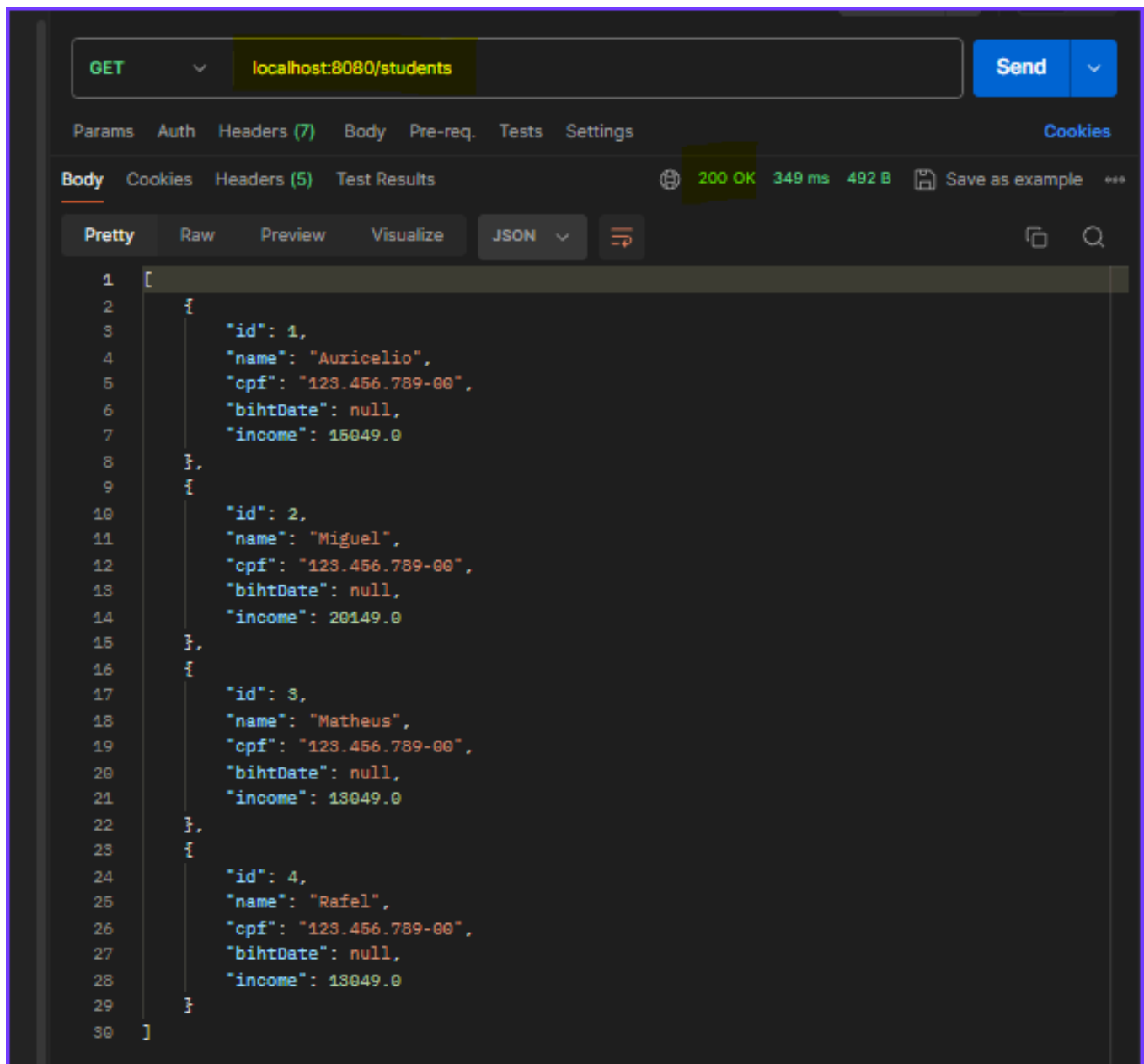
## Rodar o projeto



The screenshot shows the Spring Boot IDE interface. On the left, the 'Boot Dashboard' tab is active, displaying a list of running applications under the 'local' environment. The application 'apirestful' is running on port 8080. On the right, the 'Console' tab is active, showing the application's startup logs. The logs indicate that the application started successfully on 2024-03-30T13:55:24.327-03:00, with an INFO level message and a response time of 18848ms. Subsequent logs show the application is running and responding to requests.

Timestamp	Level	Message	Response Time (ms)
2024-03-30T13:55:24.327-03:00	INFO	18848	---
2024-03-30T13:55:24.519-03:00	INFO	18848	---
2024-03-30T13:55:24.911-03:00	INFO	18848	---
2024-03-30T13:55:24.917-03:00	INFO	18848	---
2024-03-30T13:55:24.956-03:00	WARN	18848	---
2024-03-30T13:55:25.274-03:00	INFO	18848	---
2024-03-30T13:55:25.283-03:00	INFO	18848	---

## Testar com o Postman





## Github-3


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Created StudentResources and the Endpoint findAll”
  - git push


```
MINGW64/c/PROJETOS/UNIFAMETRO/APIRestful


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Created StudentResources and the Endpoint findAll"
[main 34f2f6e] Created StudentResources and the Endpoint findAll
 2 files changed, 105 insertions(+), 1 deletion(-)
 create mode 100644 backend/src/main/java/com/unifametro/apirestful/resources/StudentResource.java

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (13/13), 1.73 KiB | 1.73 MiB/s, done.
Total 13 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
 6c6168d..34f2f6e  main -> main
```

 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1


 main


 1 Branch


 0 Tags

Go to file


Add file

 Code

 **auriceliof** Created StudentResources and the Endpoint findAll 34f2f6e - 1 minute ago 12 Commits

 backend

Created StudentResources and the Endpoint findAll 1 minute ago

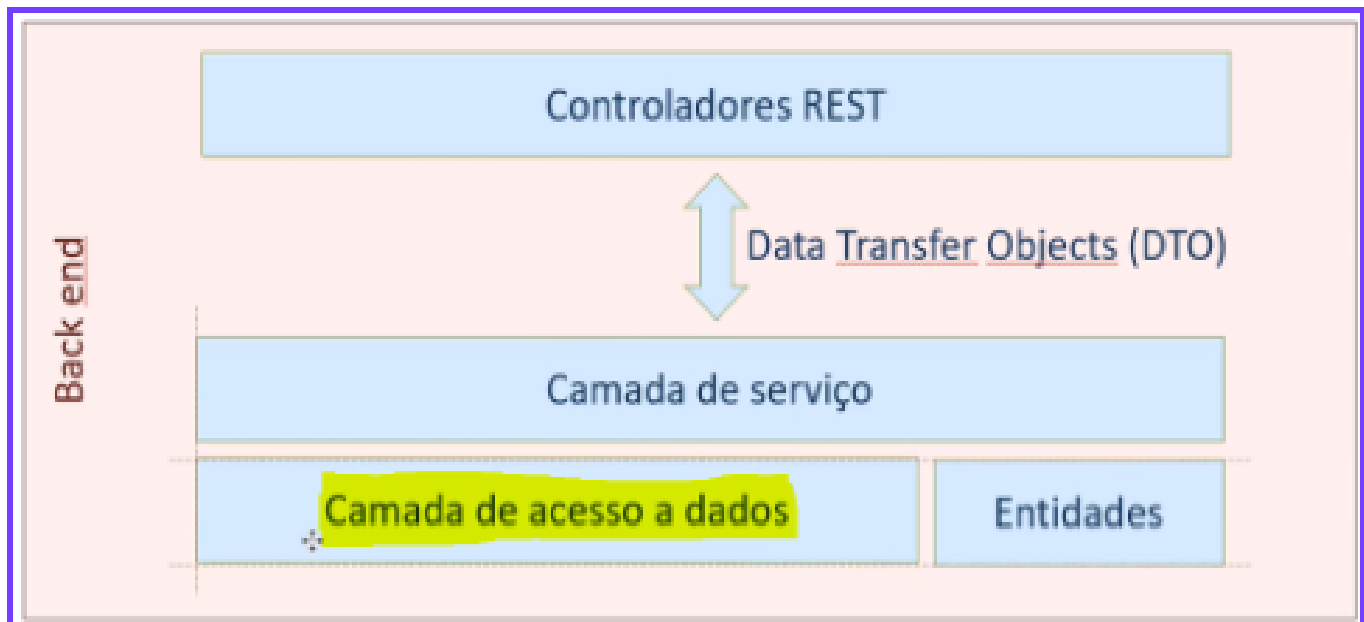
 README.md

Update README.md 52 minutes ago

## STUDENT\_REPOSITORY

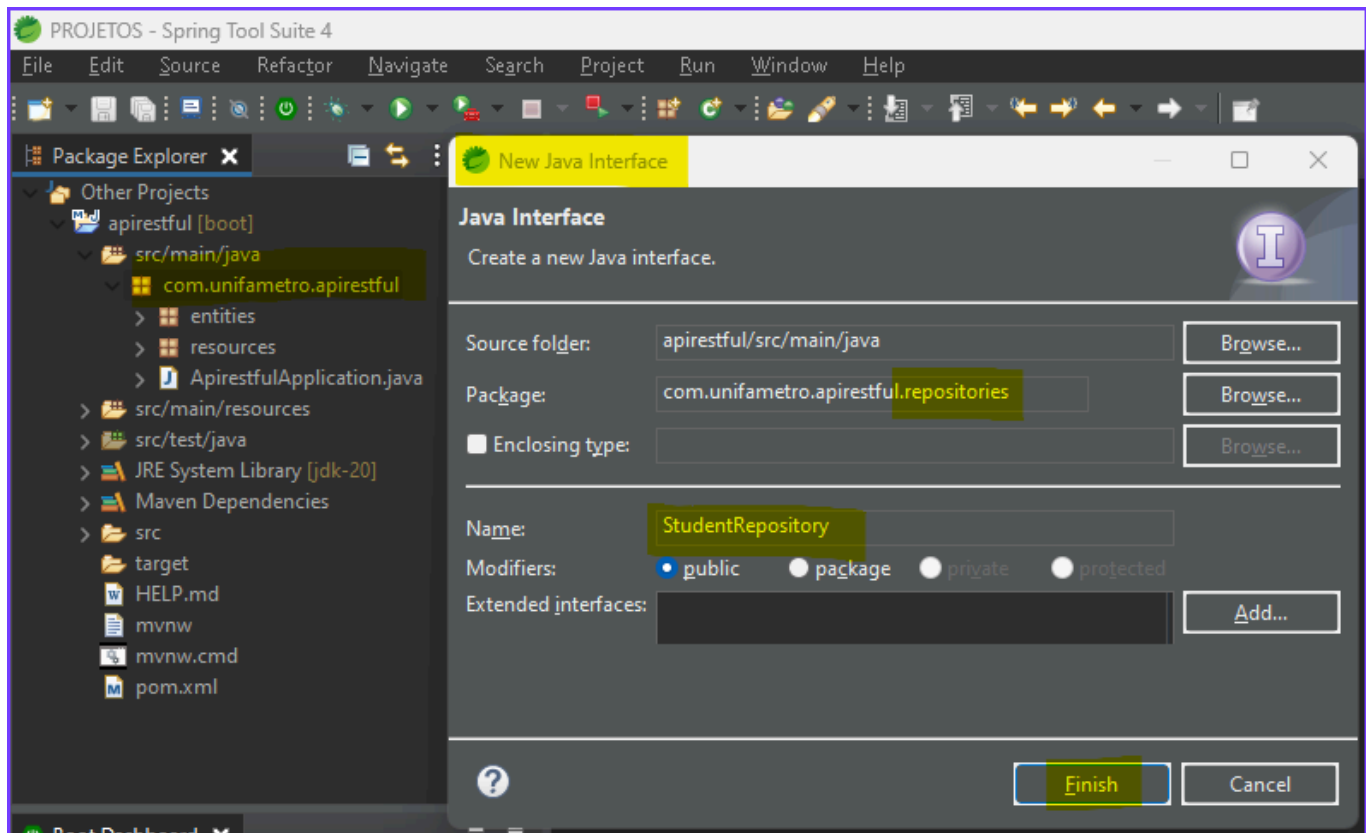
- Iremos implementar a “Camada de acesso a dados”, chamada de Repositories.

### CONCEITUAL

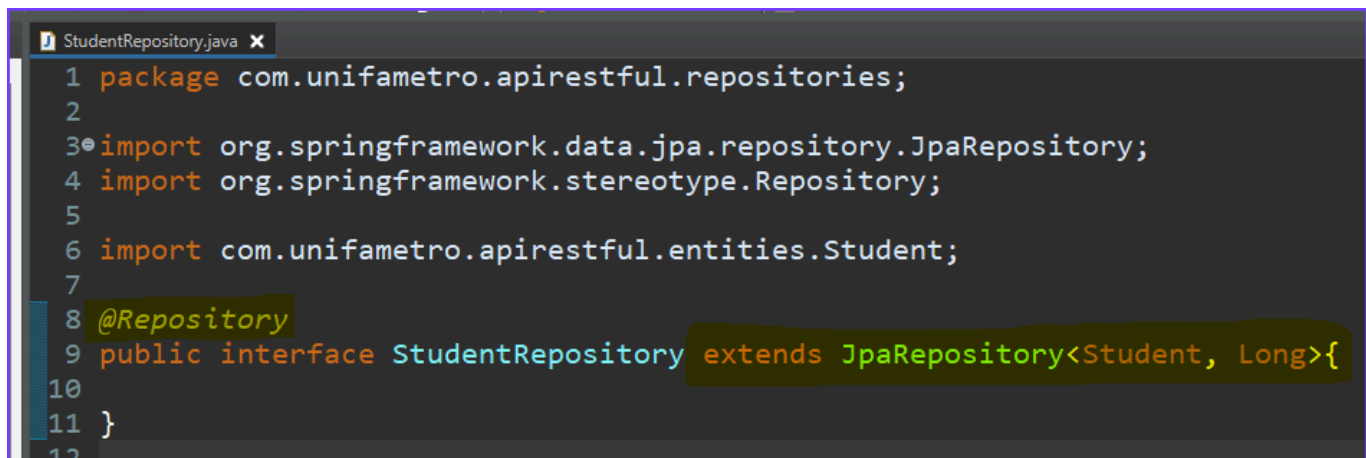


## IMPLEMENTAR A ESTRUTURA E O STUDENT\_REPOSITORY

### Criar a estrutura e a interface de acesso ao banco



### Implementar a notação e estender a JPA



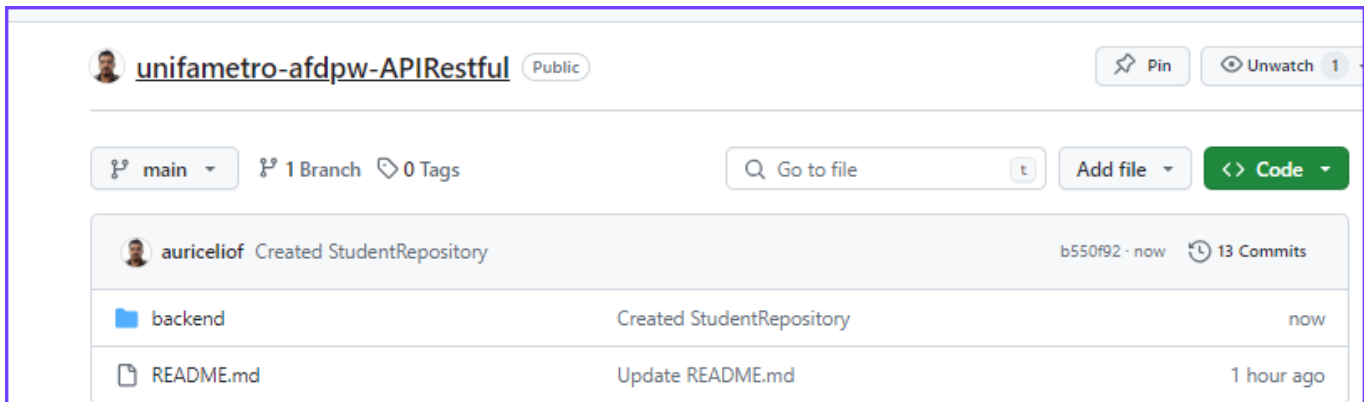
## Github-4

- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Created StudentRepository”
  - git push

```
MINGW64/c:/PROJETOS/UNIFAMETRO/APIRestful
auric@NOTE-SAMSUNG MINGW64 /c:/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .

auric@NOTE-SAMSUNG MINGW64 /c:/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Created StudentRepository"
[main b550f92] Created StudentRepository
 2 files changed, 9 insertions(+)
 create mode 100644 backend/src/main/java/com/unifametro/apirestful/repositories/StudentRepository.java

auric@NOTE-SAMSUNG MINGW64 /c:/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 23, done.
Counting objects: 100% (23/23), done.
Delta compression using up to 8 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (13/13), 1.03 KiB | 1.03 MiB/s, done.
Total 13 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
 34f2f6e..b550f92  main -> main
```

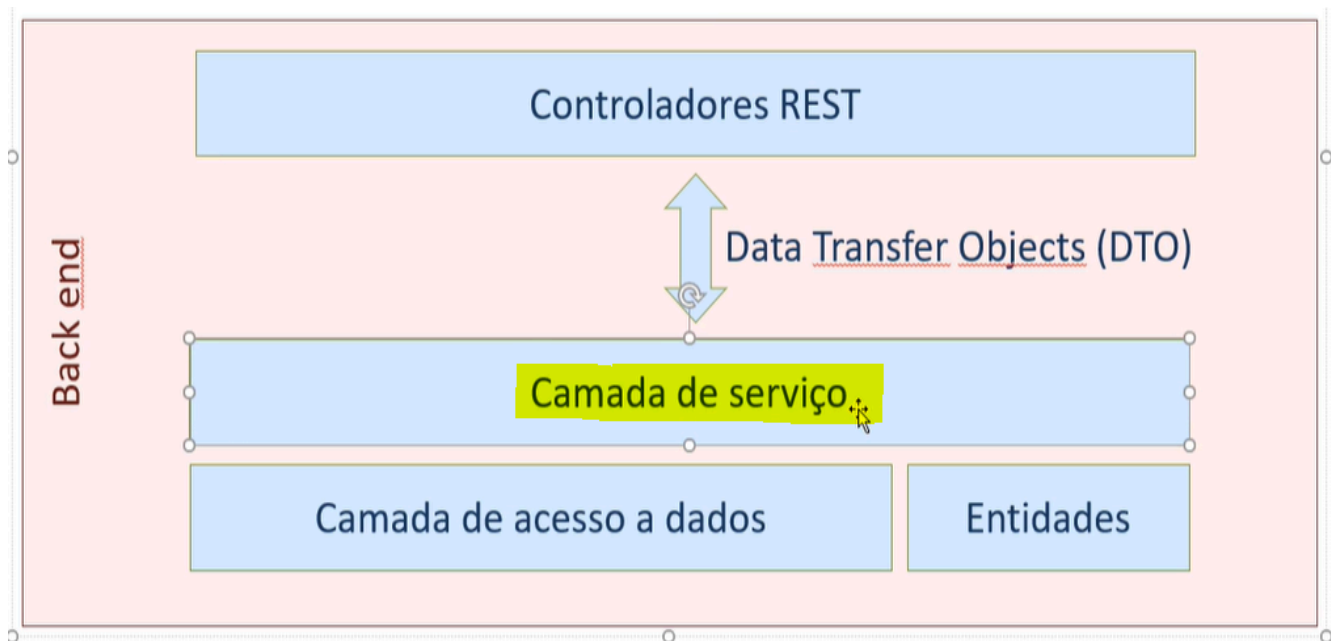


The screenshot shows the GitHub repository page for 'unifametro-afdpw-APIRestful'. The repository is public and has 1 branch (main) and 0 tags. The commit history shows a commit by 'auriceliof' titled 'Created StudentRepository' with the hash 'b550f92' and 13 commits in total. The file list shows a folder named 'backend' and a file named 'README.md', both created or updated 'now'.

## STUDENT\_SERVICE

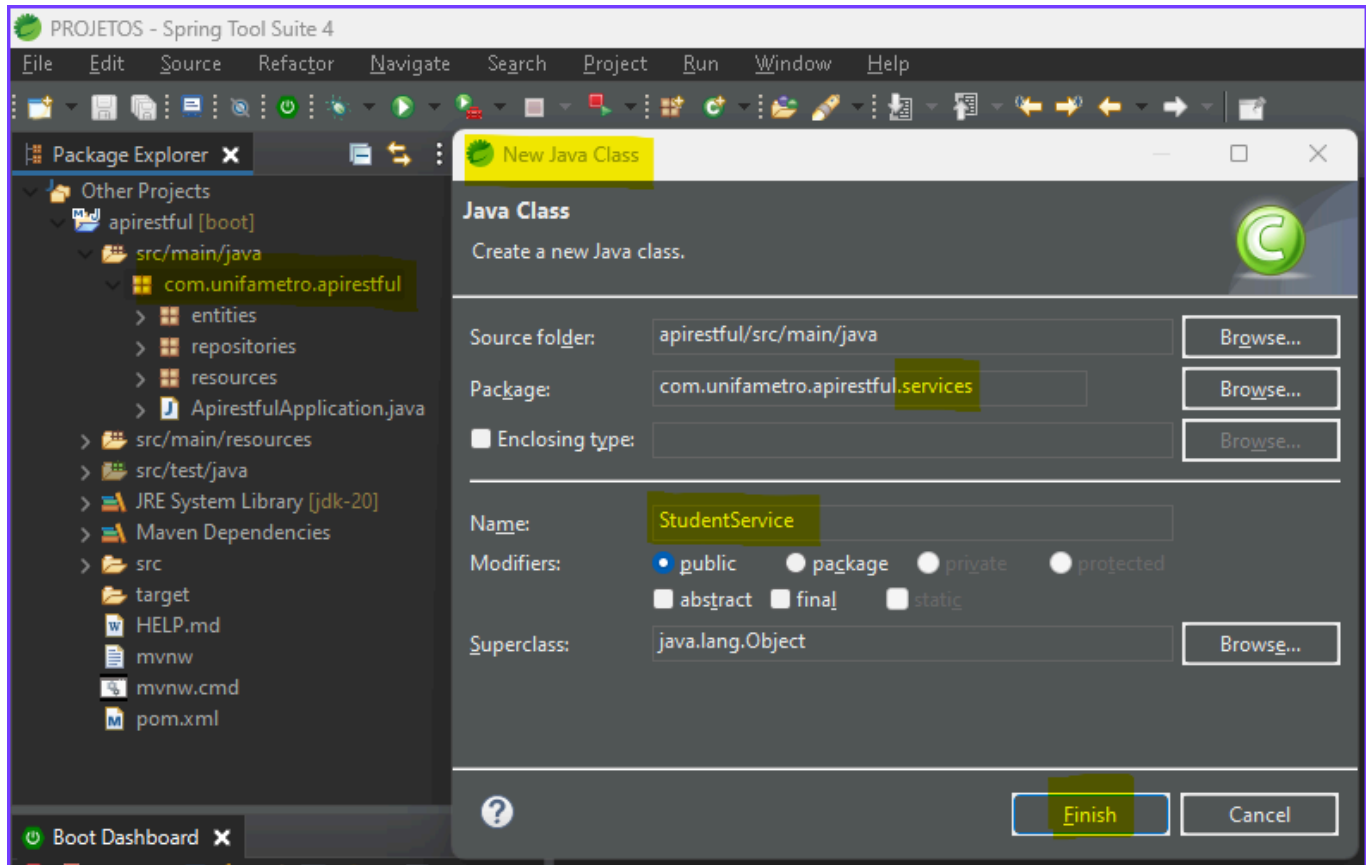
- Iremos implementar a “Camada de serviço”, onde concentramos toda a lógica do projeto.

### CONCEITUAL



## IMPLEMENTAR A ESTRUTURA E O STUDENT\_SERVICE

### Criar a estrutura e a classe de serviço



## Implementar a lógica para o findAll

```
*StudentService.java X
1 package com.unifametro.apirestful.services;
2
3 import java.util.List;
4
5 import org.springframework.beans.factory.annotation.Autowired;
6 import org.springframework.stereotype.Service;
7 import org.springframework.transaction.annotation.Transactional;
8
9 import com.unifametro.apirestful.entities.Student;
10 import com.unifametro.apirestful.repositories.StudentRepository;
11
12
13
14 @Service
15 public class StudentService {
16
17     @Autowired
18     private StudentRepository repository;
19
20     @Transactional(readonly = true)
21     public List<Student> findAll(){
22
23         return repository.findAll();
24     }
25 }
26 }
```

NOTA: O “@Transactional”, garante a integridade da transação de um método junto ao banco. No caso de pesquisa, utilizar o “readOnly” para evitar o locking ao banco de dados.

## Github-5


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Created StudentService”
  - git push

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Created StudentService"
[main 912d309] Created StudentService
3 files changed, 29 insertions(+), 7 deletions(-)
create mode 100644 backend/src/main/java/com/unifametro/apirestful/services/StudentService.java

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 27, done.
Counting objects: 100% (27/27), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (15/15), 1.47 KiB | 1.47 MiB/s, done.
Total 15 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
b550f92..912d309  main -> main
```

 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags

Go to file t Add file <> Code

 **auriceliof** Created StudentService 912d309 - 1 minute ago 14 Commits

backend	Created StudentService	1 minute ago
README.md	Update README.md	2 hours ago



## INTEGRAÇÃO COM O BANCO

### AJUSTES

#### Implementar o StudentResource

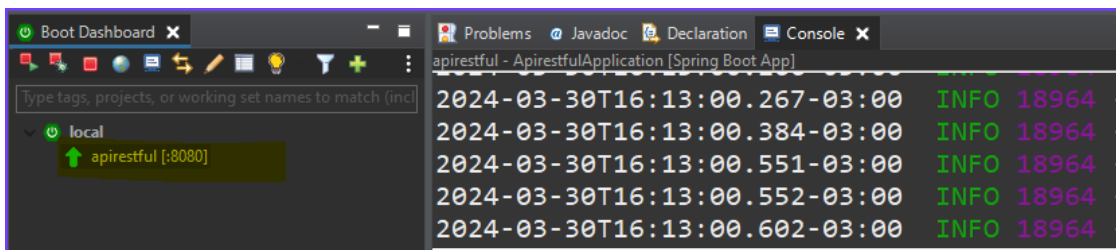
```
*StudentResource.java x
1 package com.unifametro.apirestful.resources;
2
3 import java.util.List;
13
14 @RestController
15 @RequestMapping(value = "/students")
16 public class StudentResource {
17
18     @Autowired
19     private StudentService service;
20
21     @GetMapping
22     public ResponseEntity<List<Student>> findAll(){
23
24         List<Student> list = service.findAll();
25
26         return ResponseEntity.ok().body(list);
27     }
28 }
```

## Implementar a classe Student

```
Student.java x
13
14 @Entity
15 @Table(name = "tb_student")
16 public class Student implements Serializable {
17     private static final long serialVersionUID = 1L;
18
19     @Id
20     @GeneratedValue(strategy = GenerationType.IDENTITY)
21     private Long id;
22     private String name;
23     private String cpf;
24
25     @Column(columnDefinition = "TIMESTAMP WITH TIME ZONE")
26     private Instant bihtDate;
27     private Double income;
```

**NOTA:** Ao instanciar as notações, sempre escolher o pacote referente a especificação “*jakarta.persistence*”.

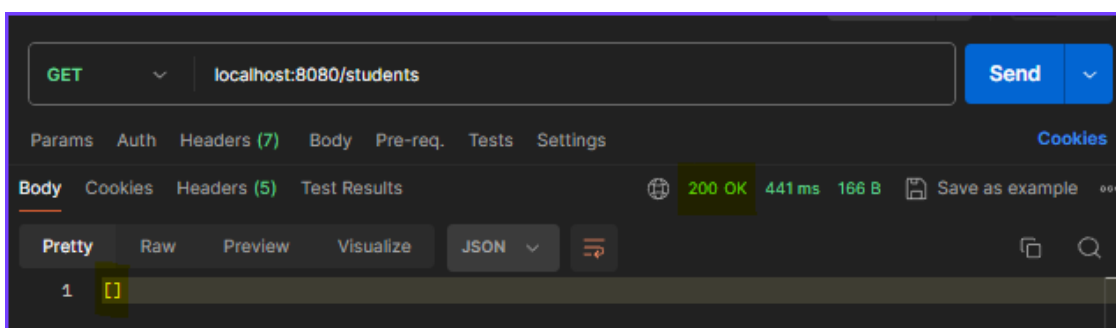
## Rodar o projeto



apirestful - ApirestfulApplication [Spring Boot App]

Timestamp	Log Level	Value
2024-03-30T16:13:00.267-03:00	INFO	18964
2024-03-30T16:13:00.384-03:00	INFO	18964
2024-03-30T16:13:00.551-03:00	INFO	18964
2024-03-30T16:13:00.552-03:00	INFO	18964
2024-03-30T16:13:00.602-03:00	INFO	18964

## Testar com o Postman



GET localhost:8080/students

Send

Params Auth Headers (7) Body Pre-req. Tests Settings Cookies

Body Cookies Headers (5) Test Results 200 OK 441 ms 166 B Save as example

Pretty Raw Preview Visualize JSON

1

## IMPLEMENTAR O BANCO H2

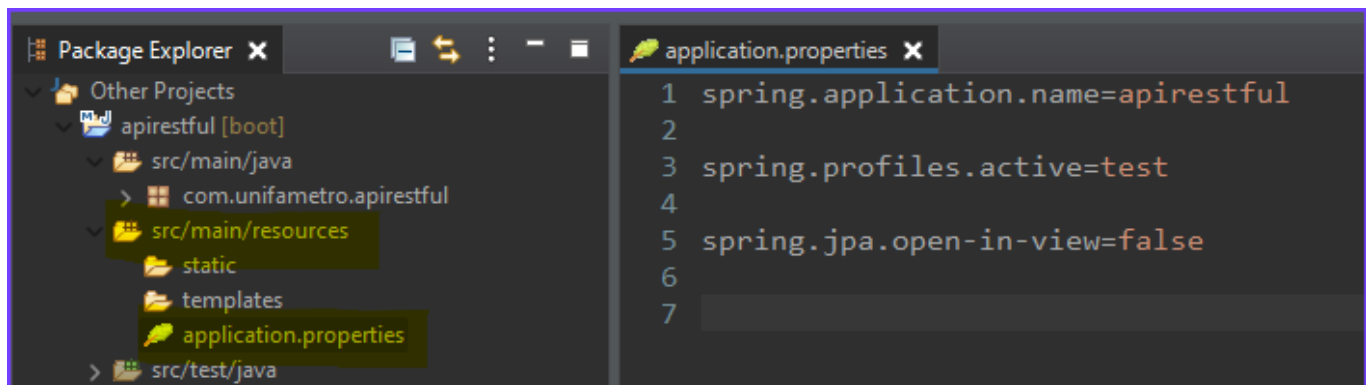
### Configurar o perfil de teste

#### Local

- src/main/resources
  - application.properties

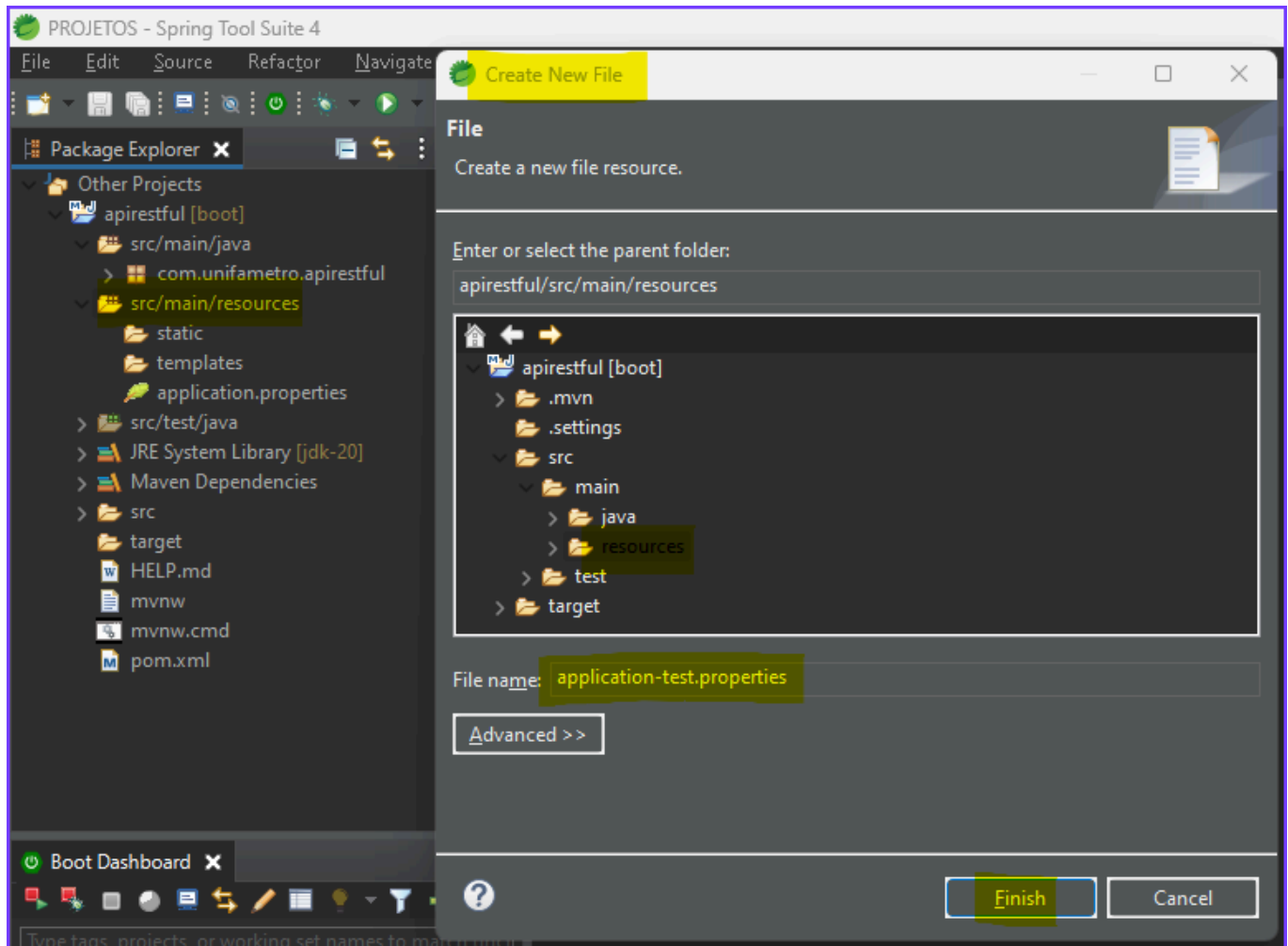
#### Acrescentar

- spring.profiles.active=test
- spring.jpa.open-in-view=false



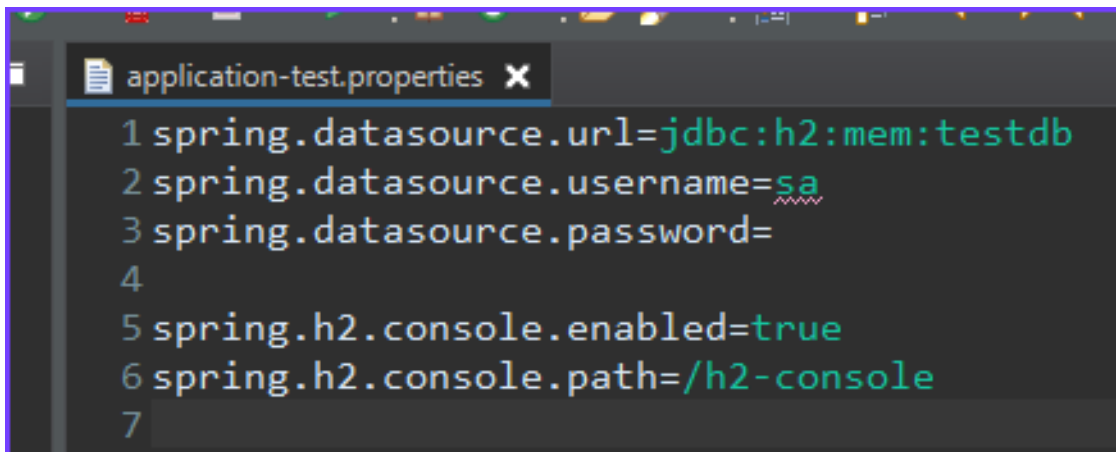
*NOTA: O “spring.jpa.open-in-view”, faz com que as transações ao banco com JPA, sejam encerradas na camada de Serviço. Não passando para a camada de Controle.*

## Criar o arquivo de configuração para teste



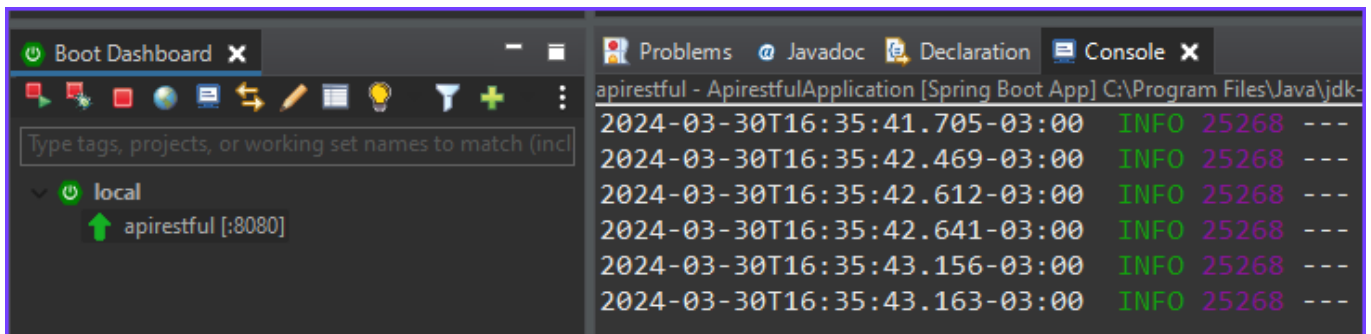
### Copiar o código de configuração para dentro do arquivo:

- `spring.datasource.url=jdbc:h2:mem:testdb`
- `spring.datasource.username=sa`
- `spring.datasource.password=`
  
- `spring.h2.console.enabled=true`
- `spring.h2.console.path=/h2-console`



```
application-test.properties
1 spring.datasource.url=jdbc:h2:mem:testdb
2 spring.datasource.username=sa
3 spring.datasource.password=
4
5 spring.h2.console.enabled=true
6 spring.h2.console.path=/h2-console
7
```

### Rodar o projeto



Boot Dashboard

Type tags, projects, or working set names to match (incl)

local

↑ apirestful [:8080]

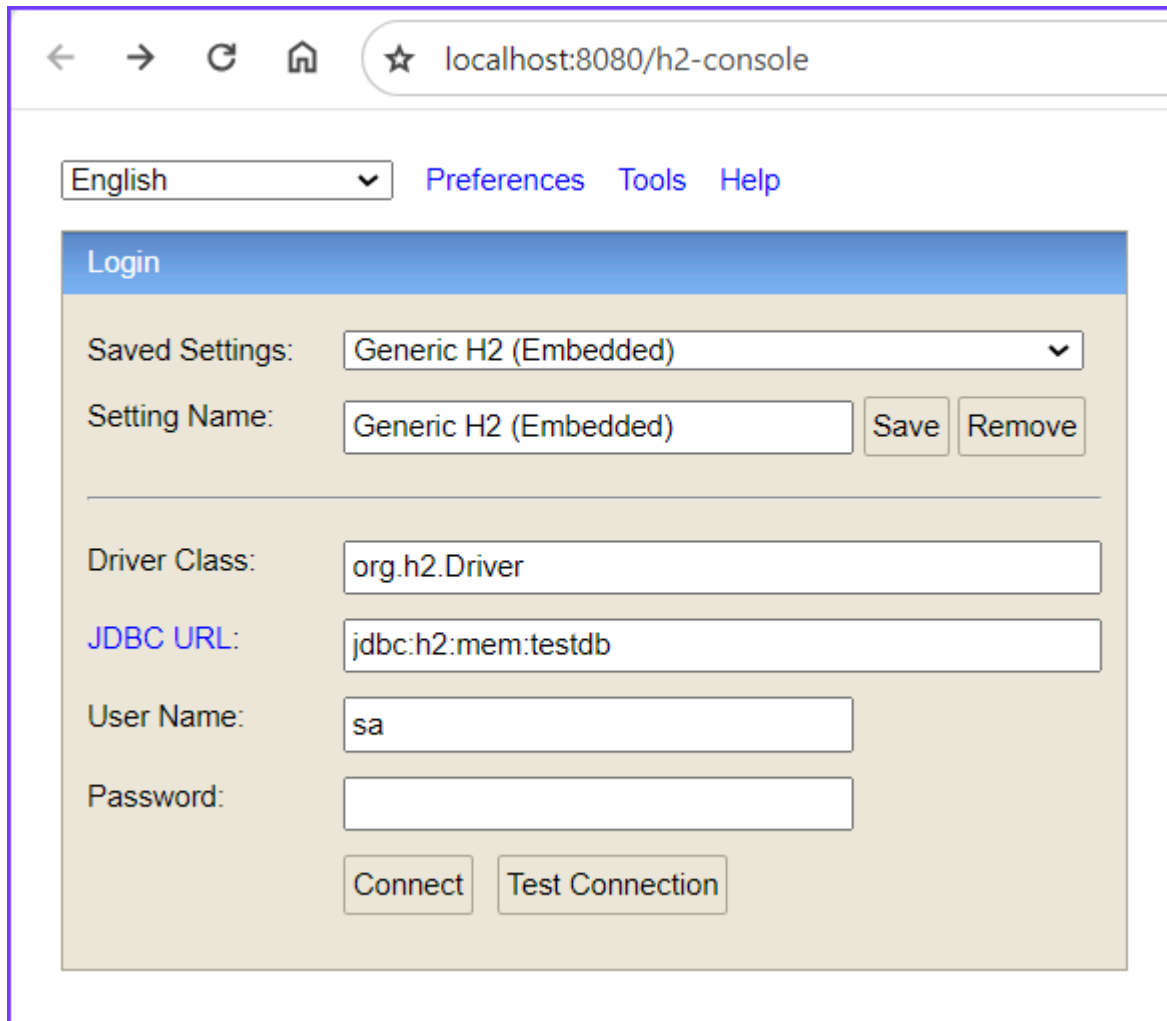
Problems Javadoc Declaration Console

apirestful - ApirestfulApplication [Spring Boot App] C:\Program Files\Java\jdk-

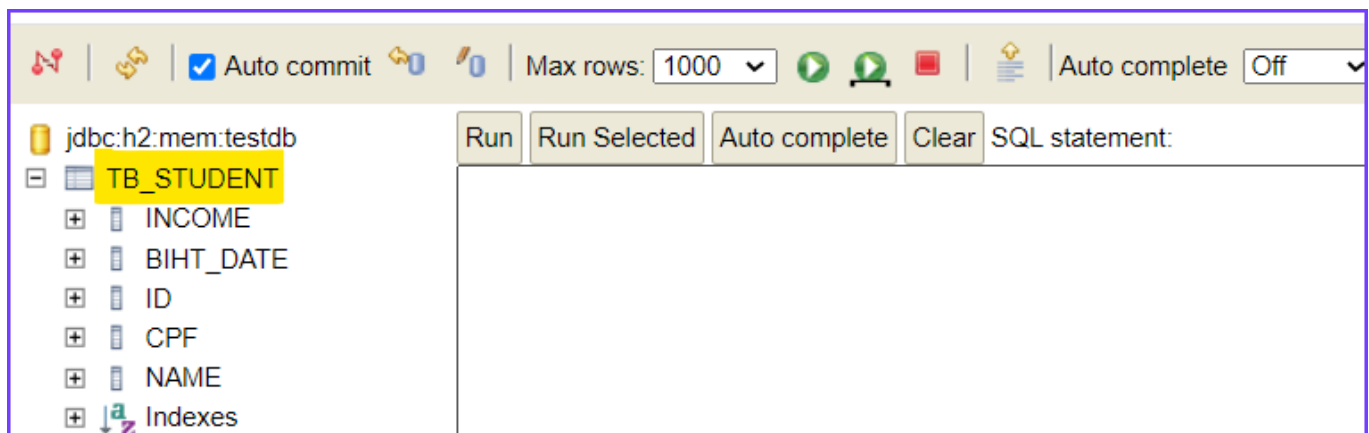
```
2024-03-30T16:35:41.705-03:00 INFO 25268 ---
2024-03-30T16:35:42.469-03:00 INFO 25268 ---
2024-03-30T16:35:42.612-03:00 INFO 25268 ---
2024-03-30T16:35:42.641-03:00 INFO 25268 ---
2024-03-30T16:35:43.156-03:00 INFO 25268 ---
2024-03-30T16:35:43.163-03:00 INFO 25268 ---
```

## Acessar o banco H2, via web

- <http://localhost:8080/h2-console>



The screenshot shows the H2 console login page in a web browser. The address bar displays "localhost:8080/h2-console". The page has a navigation bar with "English" (a dropdown menu), "Preferences", "Tools", and "Help". Below this is a "Login" section with a blue header. It contains a "Saved Settings:" dropdown menu set to "Generic H2 (Embedded)". Below that is a "Setting Name:" field also containing "Generic H2 (Embedded)", with "Save" and "Remove" buttons to its right. A horizontal line separates this from the connection fields. The "Driver Class:" field contains "org.h2.Driver". The "JDBC URL:" field contains "jdbc:h2:mem:testdb". The "User Name:" field contains "sa". The "Password:" field is empty. At the bottom of the login section are "Connect" and "Test Connection" buttons.



The screenshot shows the main interface of the H2 console. The top toolbar includes icons for undo, redo, and a checked "Auto commit" checkbox. It also shows "Max rows:" set to "1000" and "Auto complete" set to "Off". Below the toolbar, the database connection "jdbc:h2:mem:testdb" is selected. A tree view on the left shows the database schema: "TB\_STUDENT" (highlighted in yellow), "INCOME", "BIHT\_DATE", "ID", "CPF", "NAME", and "Indexes". To the right of the tree view are buttons for "Run", "Run Selected", "Auto complete", and "Clear". Further right is a text area labeled "SQL statement:".

## Teste de inserção

Run Run Selected Auto complete Clear SQL statement:

```
INSERT INTO TB_STUDENT (NAME) VALUES ('Auricelio')
```

INSERT INTO TB\_STUDENT (NAME) VALUES ('Auricelio');  
Update count: 1  
(3 ms)

Run Run Selected Auto complete Clear SQL statement:

```
SELECT * FROM TB_STUDENT
```

SELECT \* FROM TB\_STUDENT;

INCOME	BIRTH_DATE	ID	CPF	NAME
null	null	1	null	Auricelio

(1 row, 3 ms)

## Testar no postman

GET localhost:8080/students Send

Params Auth Headers (7) Body Pre-req. Tests Settings Cookies

Body Cookies Headers (5) Test Results 200 OK 51 ms 234 B Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "name": "Auricelio",
5     "cpf": null,
6     "birthDate": null,
7     "income": null
8   }
9 ]
```

## Github-6


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Integration with H2”
  - git push

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .
warning: in the working copy of 'backend/src/main/resources/application.properties'
LF the next time Git touches it


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Integration with H2"
[main b409b6f] Integration with H2
4 files changed, 39 insertions(+), 7 deletions(-)
create mode 100644 backend/src/main/resources/application-test.properties

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 30, done.
Counting objects: 100% (30/30), done.
Delta compression using up to 8 threads
Compressing objects: 100% (11/11), done.
Writing objects: 100% (16/16), 1.49 KiB | 1.49 MiB/s, done.
Total 16 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
912d309..b409b6f  main -> main
```


 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags  Add file Code

 **auriceliof** Integration with H2 b409b6f · now 15 Commits

 backend

 Integration with H2 now

 README.md

 Update README.md 3 hours ago

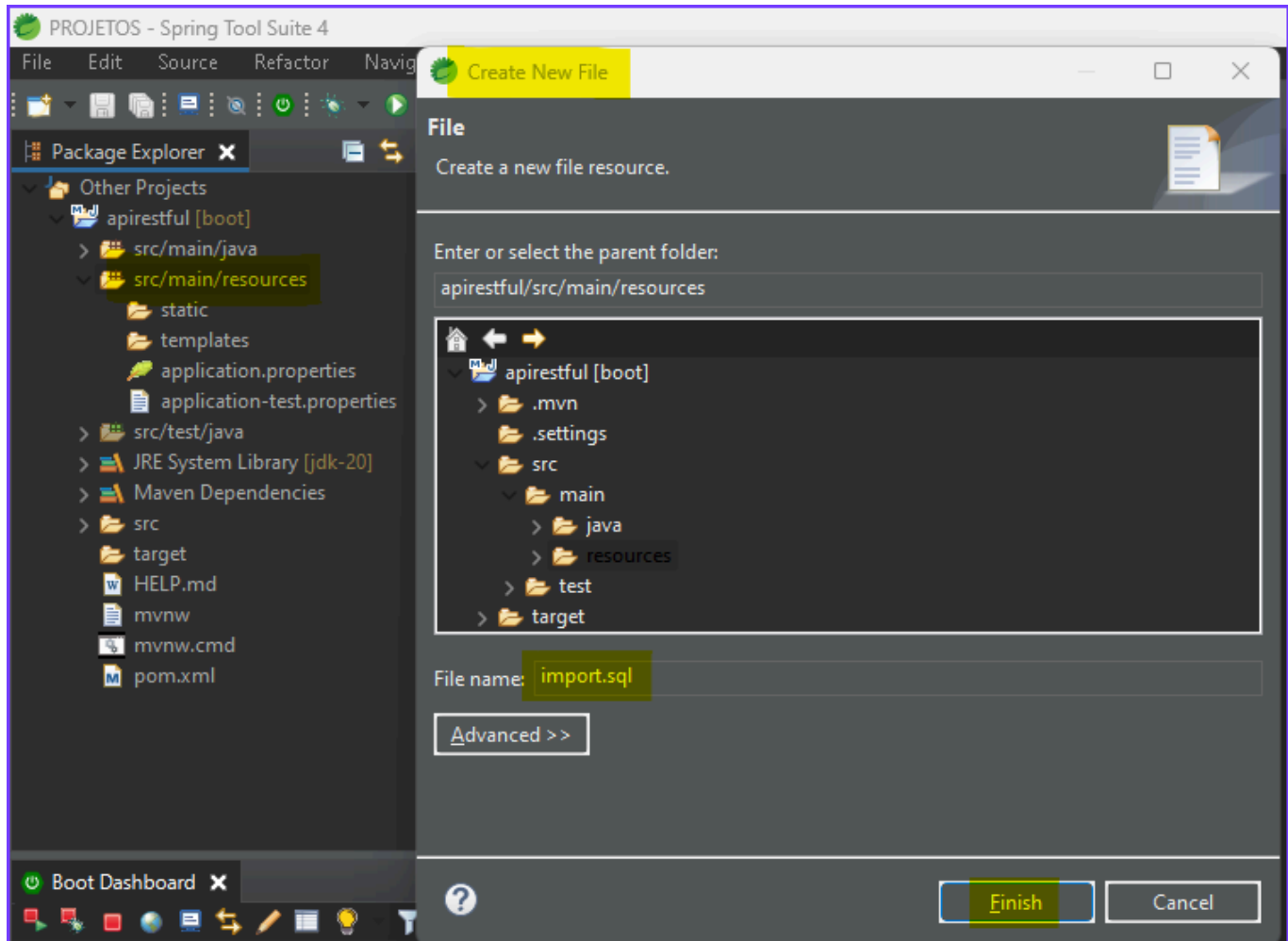


## SEEDING DA BASE DE DADOS

- É o termo utilizado para dar uma carga inicial de dados ao banco, a fim de conseguirmos realizar testes com dados.

### IMPLEMENTAR A CARGA PARA O BANCO

**Criar o import.sql, no src/main/resources**

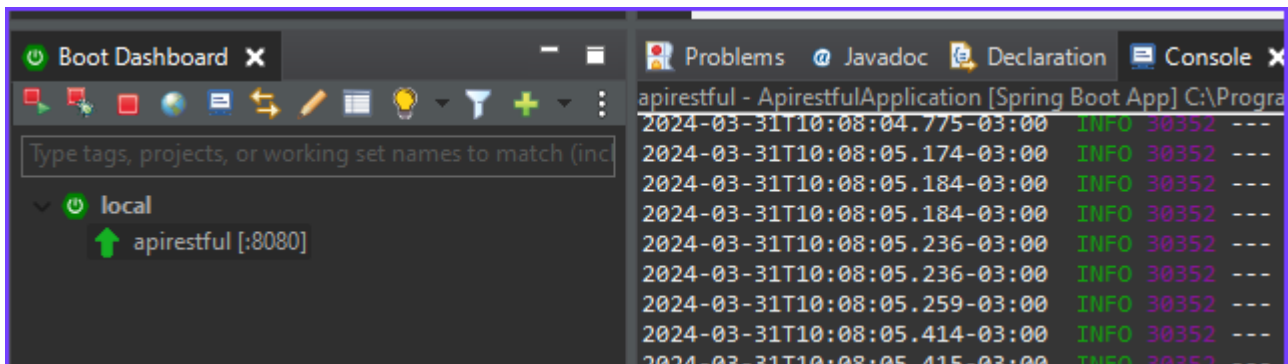


## Implementar os inserts para a carga inicial

```
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Auricelio Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1982-08-28T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Miguel Soares', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1989-02-10T10:30:00Z', 20700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Matheus Marques', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1992-11-23T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Rafael Sales', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1988-10-13T10:30:00Z', 10700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Bruno Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1987-04-22T10:30:00Z', 9700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Romulo Araujo', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1991-01-09T10:30:00Z', 8700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Ryan Sousa', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1994-03-26T10:30:00Z', 7700.5);
INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('test1', '000.111.222-33', null, 1024.0);
```

```
import.sql x
1 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Auricelio Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1982-08-28T10:30:00Z', 10700.5);
2 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Miguel Soares', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1989-02-10T10:30:00Z', 20700.5);
3 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Matheus Marques', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1992-11-23T10:30:00Z', 10700.5);
4 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Rafael Sales', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1988-10-13T10:30:00Z', 10700.5);
5 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Bruno Freitas', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1987-04-22T10:30:00Z', 9700.5);
6 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Romulo Araujo', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1991-01-09T10:30:00Z', 8700.5);
7 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Ryan Sousa', '000.111.222-33', TIMESTAMP WITH TIME ZONE '1994-03-26T10:30:00Z', 7700.5);
8 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('test1', '000.111.222-33', null, 1024.0);
```

## Rodar o projeto



The screenshot shows an IDE interface with two main panels. The left panel, titled 'Boot Dashboard', displays a list of running applications. The right panel, titled 'Console', shows the output of the application.

**Boot Dashboard:**

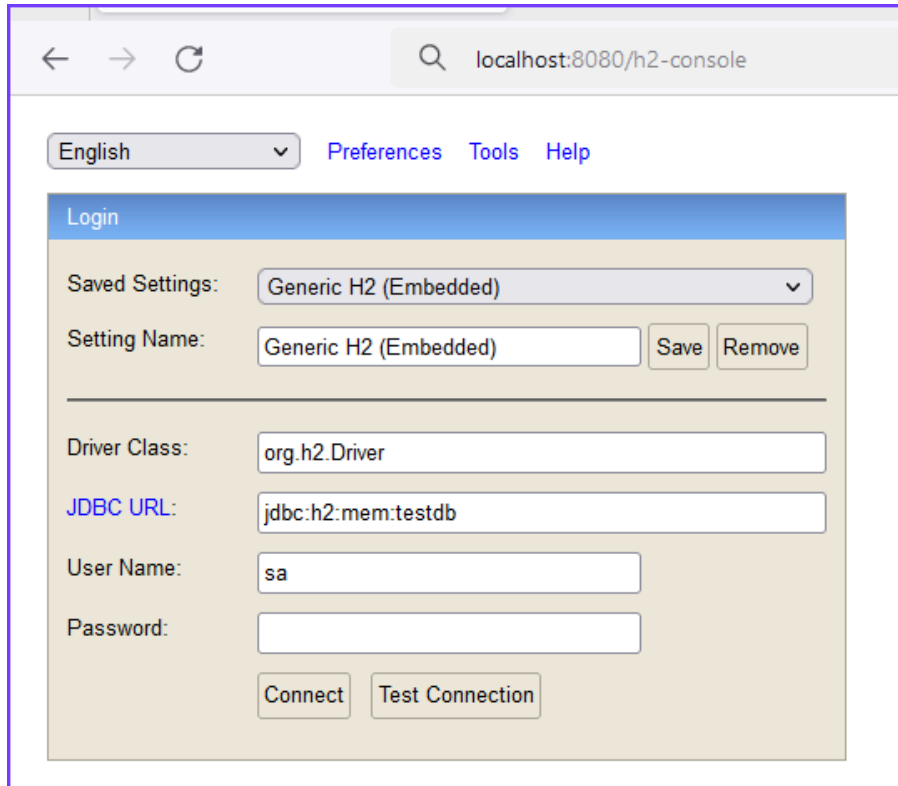
- local
- ↑ apirestful [:8080]

**Console:**

```
apirestful - ApirestfulApplication [Spring Boot App] C:\Progra
2024-03-31T10:08:04.775-03:00 INFO 30352 ---
2024-03-31T10:08:05.174-03:00 INFO 30352 ---
2024-03-31T10:08:05.184-03:00 INFO 30352 ---
2024-03-31T10:08:05.184-03:00 INFO 30352 ---
2024-03-31T10:08:05.236-03:00 INFO 30352 ---
2024-03-31T10:08:05.236-03:00 INFO 30352 ---
2024-03-31T10:08:05.259-03:00 INFO 30352 ---
2024-03-31T10:08:05.414-03:00 INFO 30352 ---
2024-03-31T10:08:05.415-03:00 INFO 30352 ---
```

## TESTAR A CARGA

### Testar no banco H2



localhost:8080/h2-console

English Preferences Tools Help

Login

Saved Settings: Generic H2 (Embedded)

Setting Name: Generic H2 (Embedded) Save Remove

Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:mem:testdb

User Name: sa

Password:

Connect Test Connection

jdbc:h2:mem:testdb

- TB\_STUDENT
  - INCOME
  - BIRTH\_DATE
  - ID
  - CPF
  - NAME
  - Indexes
- INFORMATION\_SCHEMA
- Users
- H2 2.2.224 (2023-09-17)

Run Run Selected Auto complete Clear SQL statement:

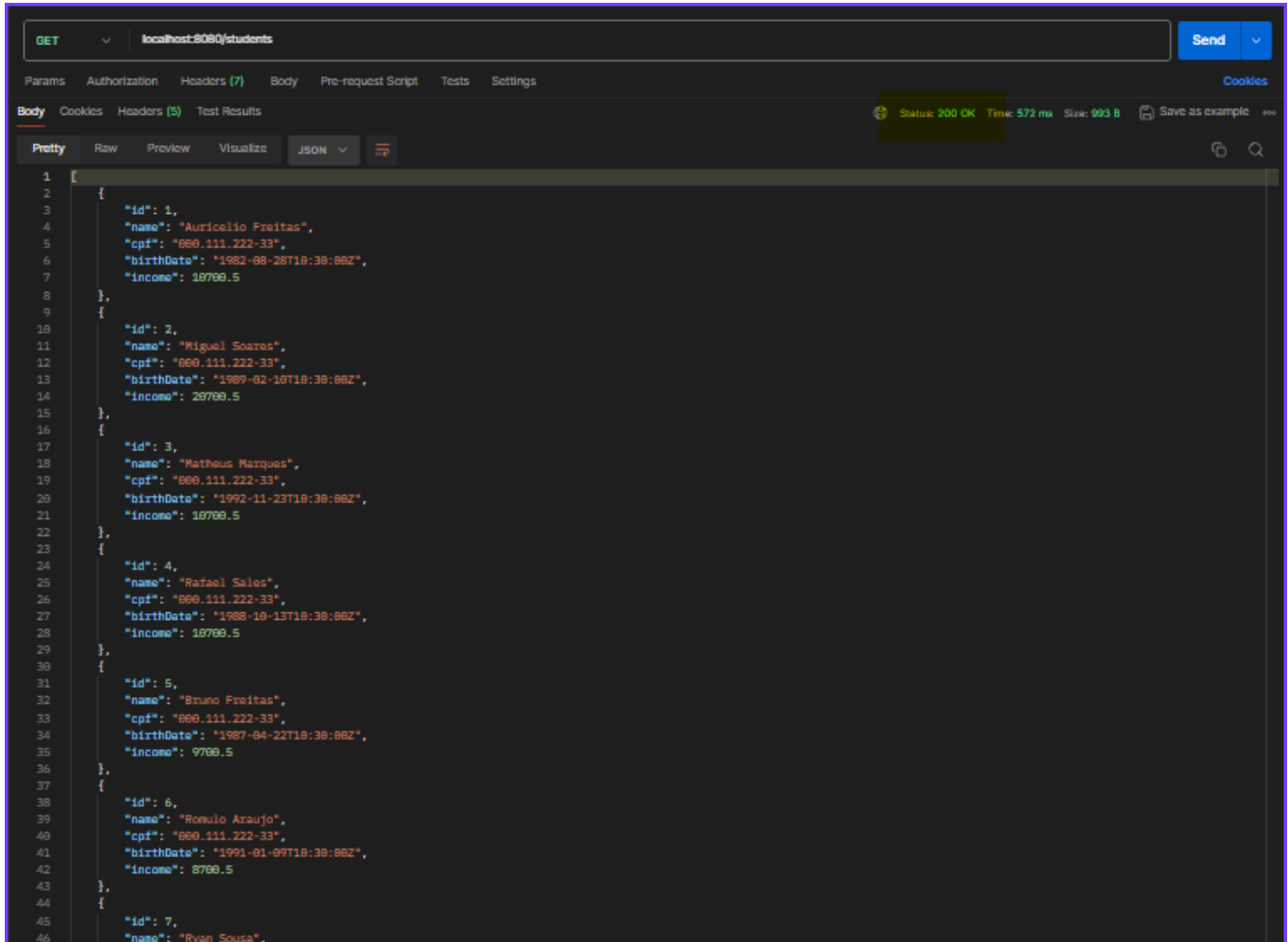
SELECT \* FROM TB\_STUDENT

SELECT \* FROM TB\_STUDENT;
 

INCOME	BIRTH_DATE	ID	CPF	NAME
10700.5	1982-08-28 10:30:00+00	1	000.111.222-33	Auricelio Freitas
20700.5	1989-02-10 10:30:00+00	2	000.111.222-33	Miguel Soares
10700.5	1992-11-23 10:30:00+00	3	000.111.222-33	Matheus Marques
10700.5	1988-10-13 10:30:00+00	4	000.111.222-33	Rafael Sales
9700.5	1987-04-22 10:30:00+00	5	000.111.222-33	Bruno Freitas
8700.5	1991-01-09 10:30:00+00	6	000.111.222-33	Romulo Araujo
7700.5	1994-03-26 10:30:00+00	7	000.111.222-33	Ryan Sousa
1024.0	null	8	000.111.222-33	test1

 (8 rows, 4 ms)

## Testar no Postman



The screenshot shows the Postman interface with a GET request to `localhost:8080/students`. The response status is 200 OK, with a time of 572 ms and a size of 993 B. The response body is a JSON array of 7 student objects, displayed in the 'Pretty' view.

```
1 [
2   {
3     "id": 1,
4     "name": "Auricelio Freitas",
5     "cpf": "999.111.222-33",
6     "birthDate": "1982-08-28T18:38:06Z",
7     "income": 19700.5
8   },
9   {
10    "id": 2,
11    "name": "Miguel Soares",
12    "cpf": "999.111.222-33",
13    "birthDate": "1989-02-18T18:38:06Z",
14    "income": 29700.5
15  },
16  {
17    "id": 3,
18    "name": "Matheus Marques",
19    "cpf": "999.111.222-33",
20    "birthDate": "1992-11-23T18:38:06Z",
21    "income": 19700.5
22  },
23  {
24    "id": 4,
25    "name": "Rafael Sales",
26    "cpf": "999.111.222-33",
27    "birthDate": "1988-10-13T18:38:06Z",
28    "income": 19700.5
29  },
30  {
31    "id": 5,
32    "name": "Bruno Freitas",
33    "cpf": "999.111.222-33",
34    "birthDate": "1987-04-22T18:38:06Z",
35    "income": 9700.5
36  },
37  {
38    "id": 6,
39    "name": "Romulo Araujo",
40    "cpf": "999.111.222-33",
41    "birthDate": "1991-01-09T18:38:06Z",
42    "income": 8700.5
43  },
44  {
45    "id": 7,
46    "name": "Ryan Sousa",
```

## Github-7


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Created seed”
  - git push

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Created seed"
[main 5349c37] Created seed
 3 files changed, 22 insertions(+), 9 deletions(-)
 create mode 100644 backend/src/main/resources/import.sql

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 28, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (15/15), 1.40 KiB | 1.40 MiB/s, done.
Total 15 (delta 5), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (5/5), completed with 5 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
 b409b6f..5349c37  main -> main
```

 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags

Go to file t Add file <> Code

 **auriceliof** Created seed 5349c37 · 39 minutes ago 16 Commits

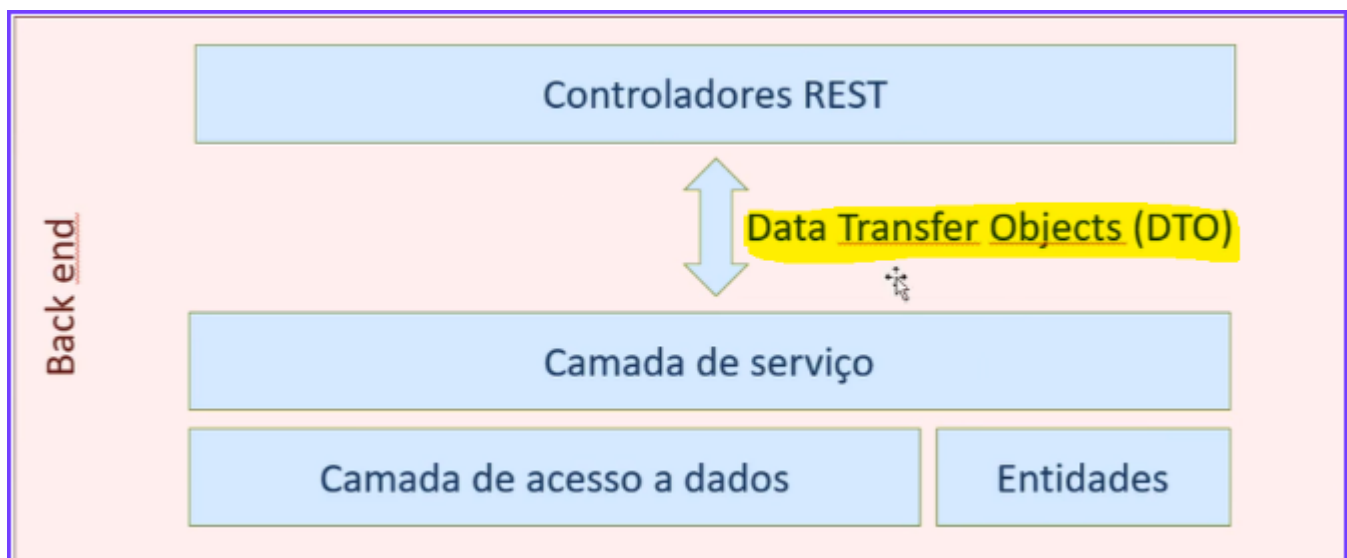
backend Created seed 39 minutes ago

README.md Update README.md yesterday

## DTO

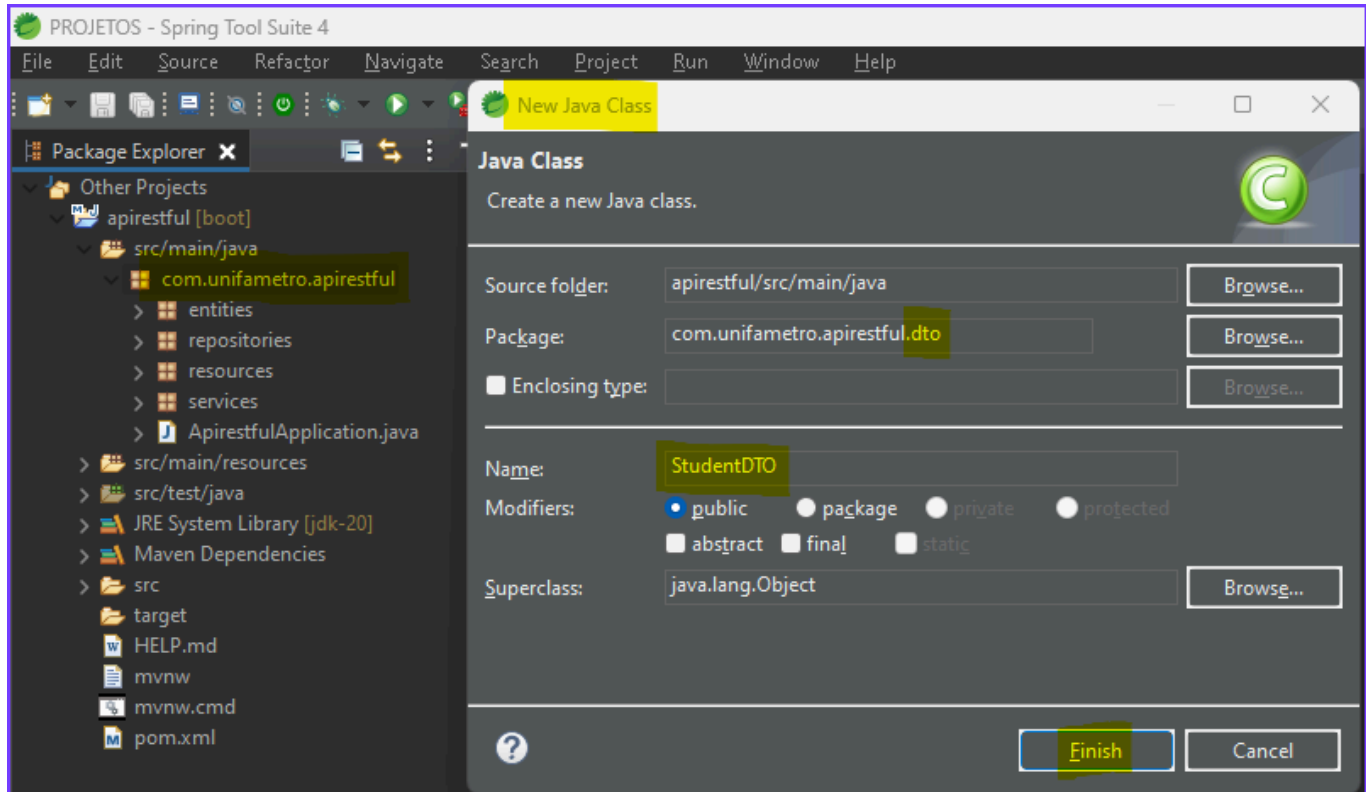
- O Controlador Rest não tem integração direta com a Entidade
- A comunicação entre o Controlador e o Serviço é feito por meio do DTO
- DTO é um objeto que serve, apenas, para transferência de dados.
- Não tem relação com a JPA.
- Podemos controlar quais dados serão entregues para a aplicação API

## CONCEITUAL



## IMPLEMENTAR A ESTRUTURA E O STUDENT DTO

### Criar a estrutura e a classe DTO



### Implementar o Serializable e os mesmos atributos da Classe Student

```
*StudentDTO.java x
6 public class StudentDTO implements Serializable {
7     private static final long serialVersionUID = 1L;
8
9     private Long id;
10    private String name;
11    private String cpf;
12    private Instant birthDate;
13    private Double income;
```

## Implementar os construtores

### Vazio

```
*StudentDTO.java x
15 public StudentDTO() {
16
17 }
```

### de Classe

Generate Constructor using Fields

Select super constructor to invoke:

Object()

Select fields to initialize:

<input checked="" type="checkbox"/>	id	Select All Deselect All Up
<input checked="" type="checkbox"/>	name	
<input checked="" type="checkbox"/>	cpf	
<input checked="" type="checkbox"/>	birthDate	
<input checked="" type="checkbox"/>	income	

Insertion point:

After 'StudentDTO()'

Access modifier

☒ public ☐ protected ☐ package ☐ private

☐ Generate constructor comments

☐ Omit call to default constructor super()

The format of the constructors may be configured on the [Code Templates](#) preference page.

5 of 5 selected.

Generate Cancel

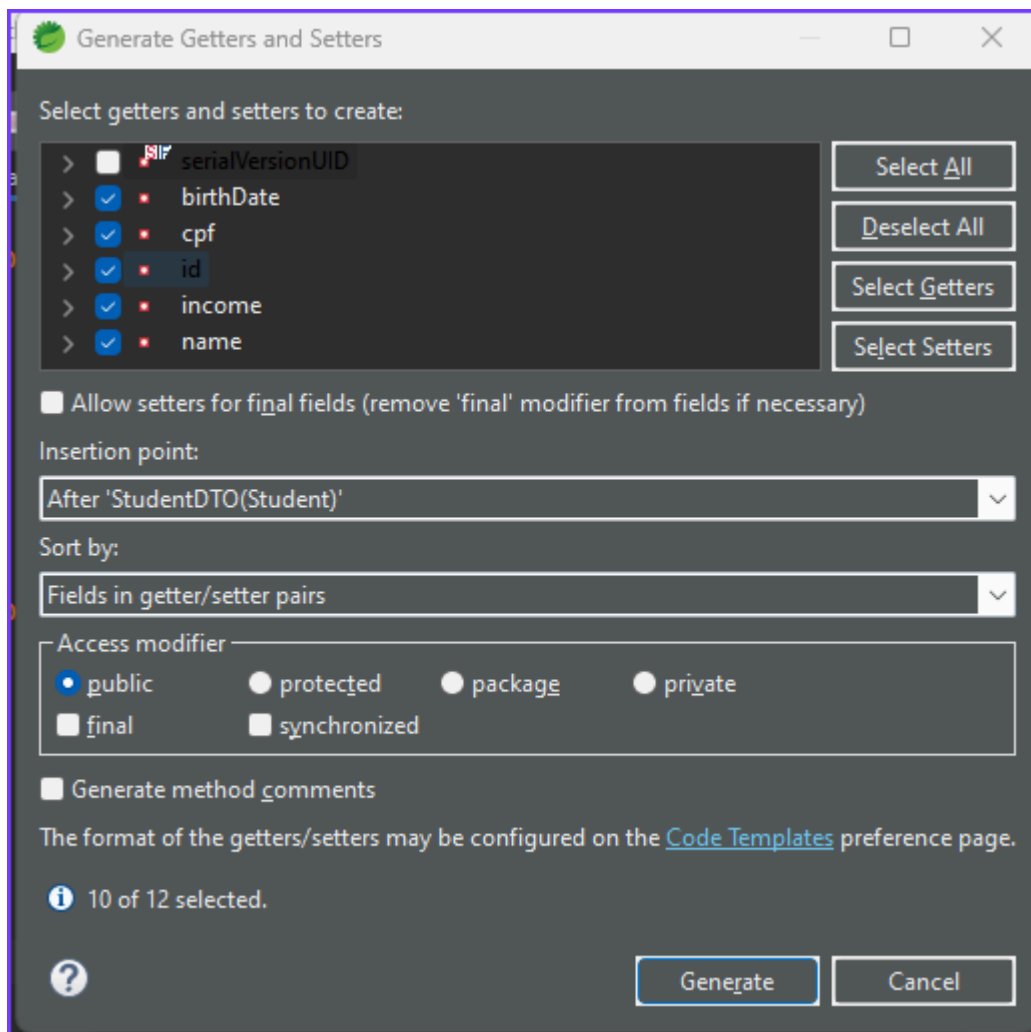
```
*StudentDTO.java x
20
21 public StudentDTO(Long id, String name, String cpf, Instant birthDate, Double income) {
22     this.id = id;
23     this.name = name;
24     this.cpf = cpf;
25     this.birthDate = birthDate;
26     this.income = income;
27 }
```



## De Entidade

```
StudentDTO.java x
29
30 public StudentDTO(Student entity) {
31     id = entity.getId();
32     name = entity.getName();
33     cpf = entity.getCpf();
34     birthDate = entity.getBirthDate();
35     income = entity.getIncome();
36 }
37
```

## Implementar os Getters and Setters



Generate Getters and Setters

Select getters and setters to create:

<input type="checkbox"/>	serialVersionUID
<input checked="" type="checkbox"/>	birthDate
<input checked="" type="checkbox"/>	cpf
<input checked="" type="checkbox"/>	id
<input checked="" type="checkbox"/>	income
<input checked="" type="checkbox"/>	name

☐ Allow setters for final fields (remove 'final' modifier from fields if necessary)

Insertion point:  
After 'StudentDTO(Student)'

Sort by:  
Fields in getter/setter pairs

Access modifier:  
☒ public ☐ protected ☐ package ☐ private  
☐ final ☐ synchronized

☐ Generate method comments

The format of the getters/setters may be configured on the [Code Templates](#) preference page.

10 of 12 selected.

Generate Cancel

```
*StudentDTO.java x
36
37 public Long getId() {
38     return id;
39 }
40
41 public void setId(Long id) {
42     this.id = id;
43 }
44
45 public String getName() {
46     return name;
47 }
48
49 public void setName(String name) {
50     this.name = name;
51 }
52
53 public String getCpf() {
54     return cpf;
55 }
56
57 public void setCpf(String cpf) {
58     this.cpf = cpf;
59 }
60
```

```
60
61 public Instant getBirthDate() {
62     return birthDate;
63 }
64
65 public void setBirthDate(Instant birthDate) {
66     this.birthDate = birthDate;
67 }
68
69 public Double getIncome() {
70     return income;
71 }
72
73 public void setIncome(Double income) {
74     this.income = income;
75 }
76 }
77
```

## REALIZAR OS AJUSTES PARA O DTO

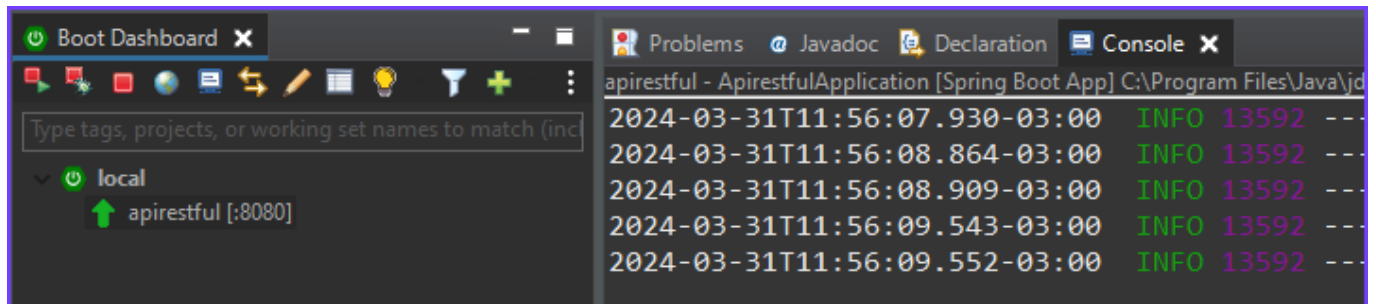
### Implementar o DTO na classe StudentService

```
StudentService.java x
1 package com.unifametro.apirestful.services;
2
3+ import java.util.List;
13
14 @Service
15 public class StudentService {
16
17+ @Autowired
18     private StudentRepository repository;
19
20+ @Transactional(readOnly = true)
21     public List<StudentDTO> findAll(){
22
23         List<Student> list = repository.findAll();
24
25         return list.stream().map(x -> new StudentDTO(x)).collect(Collectors.toList());
26     }
27 }
28
```

### Implementar o DTO na classe StudentResource

```
StudentResource.java x
1 package com.unifametro.apirestful.resources;
2
3+ import java.util.List;
13
14 @RestController
15 @RequestMapping(value = "/students")
16 public class StudentResource {
17
18+ @Autowired
19     private StudentService service;
20
21+ @GetMapping
22     public ResponseEntity<List<StudentDTO>> findAll(){
23
24         List<StudentDTO> list = service.findAll();
25
26         return ResponseEntity.ok().body(list);
27     }
28 }
29
```

## Rodar o projeto

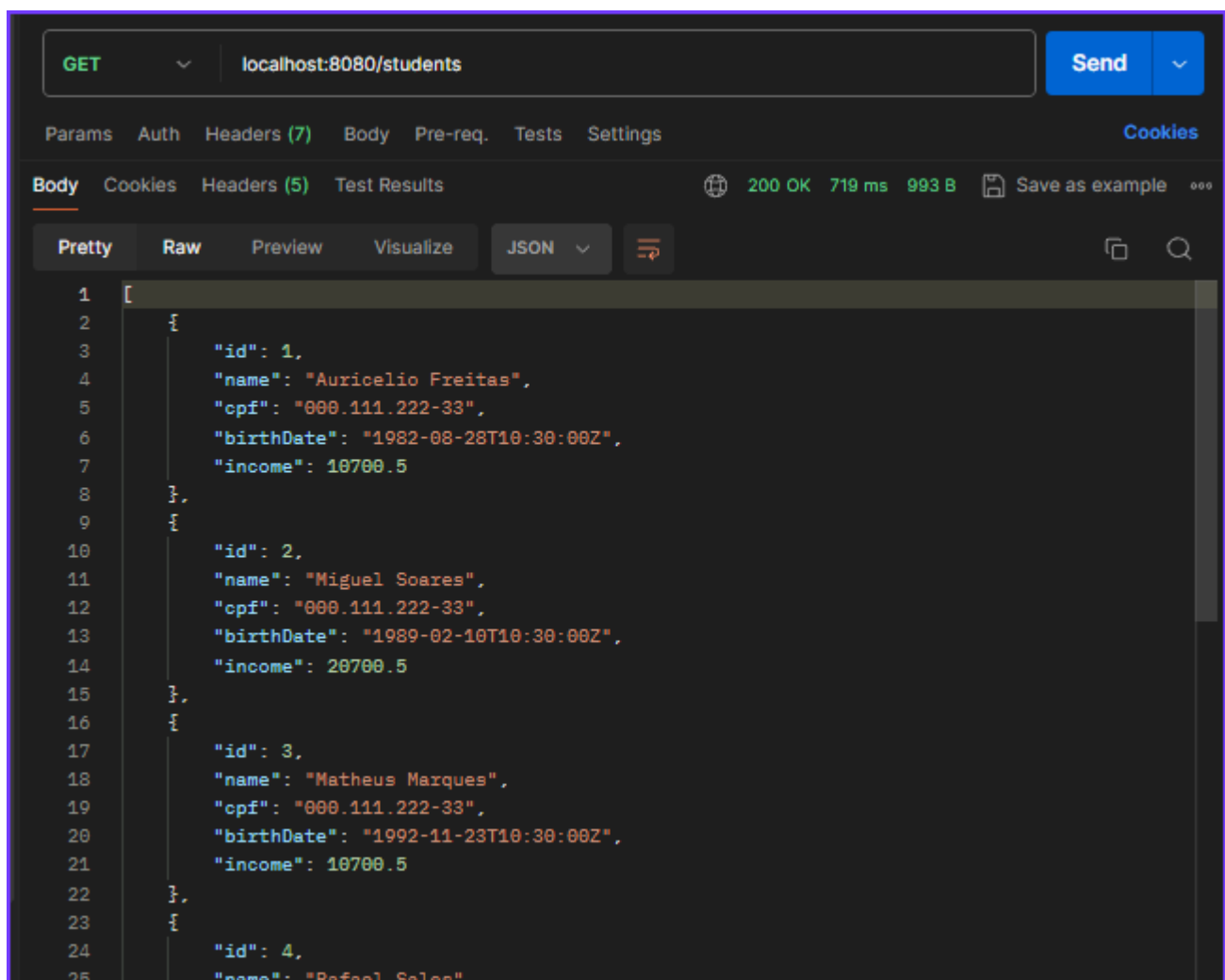


```
Boot Dashboard x
Type tags, projects, or working set names to match (inc
local
  apirestful [:8080]

Problems Javadoc Declaration Console x
apirestful - ApirestfulApplication [Spring Boot App] C:\Program Files\Java\jd
2024-03-31T11:56:07.930-03:00 INFO 13592 ---
2024-03-31T11:56:08.864-03:00 INFO 13592 ---
2024-03-31T11:56:08.909-03:00 INFO 13592 ---
2024-03-31T11:56:09.543-03:00 INFO 13592 ---
2024-03-31T11:56:09.552-03:00 INFO 13592 ---
```

## TESTES

### Testar com o Postman



```
GET localhost:8080/students Send
Params Auth Headers (7) Body Pre-req. Tests Settings Cookies
Body Cookies Headers (5) Test Results 200 OK 719 ms 993 B Save as example
Pretty Raw Preview Visualize JSON
1 [
2   {
3     "id": 1,
4     "name": "Auricelio Freitas",
5     "cpf": "000.111.222-33",
6     "birthDate": "1982-08-28T10:30:00Z",
7     "income": 10700.5
8   },
9   {
10    "id": 2,
11    "name": "Miguel Soares",
12    "cpf": "000.111.222-33",
13    "birthDate": "1989-02-10T10:30:00Z",
14    "income": 20700.5
15  },
16  {
17    "id": 3,
18    "name": "Matheus Marques",
19    "cpf": "000.111.222-33",
20    "birthDate": "1992-11-23T10:30:00Z",
21    "income": 10700.5
22  },
23  {
24    "id": 4,
25    "name": "Rafael Sales"
```

## Github-8


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Created DTO”
  - git push

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Created DTO"
[main 647f822] Created DTO
 3 files changed, 109 insertions(+), 7 deletions(-)
 create mode 100644 backend/src/main/java/com/unifametro/apirestful/dto/StudentDTO.java

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 27, done.
Counting objects: 100% (27/27), done.
Delta compression using up to 8 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (15/15), 1.59 KiB | 1.59 MiB/s, done.
Total 15 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
 5349c37..647f822  main -> main
```


 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags  Add file Code

 **auriceliof** Created DTO 647f822 · now 16 Commits

 backend

 Created DTO now

 README.md

 Update README.md yesterday

## ENDPOINT - FIND\_BY\_ID

### BUSCAR ALUNOS POR ID COM GET

Implementar busca por Id, no StudentResource

```
StudentResource.java x
35
36 @GetMapping(value =("/{id}")
37 public ResponseEntity<StudentDTO> findById(@PathVariable Long id){
38
39     StudentDTO dto = service.findById(id);
40
41     return ResponseEntity.ok().body(dto);
42 }
```

Implementar o método findById, no StudentService

*DICA: No StudentResource, clicando sobre o método e em sobre o Create method ..., será criado um método default no StudentService, onde podemos implementá-lo em seguida.*

```
StudentResource.java x
35
36 @GetMapping(value =("/{id}")
37 public ResponseEntity<StudentDTO> findById(@PathVariable Long id){
38
39     StudentDTO dto = service.findById(id);
40
41     return ResponseEntity.ok().body(dto);
42 }
43
44 @PostMapping
45 public ResponseEntity<Student
```

The method findById(Long) is undefined for the type StudentService

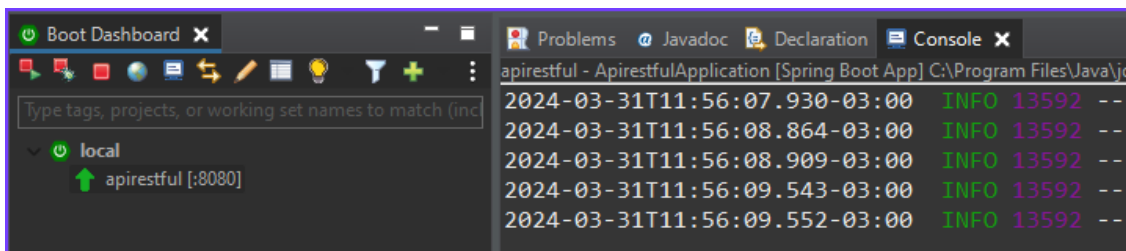
2 quick fixes available:

- Create method 'findById(Long)' in type 'StudentService'
- Add cast to 'service'

```
*StudentService.java x
27
28 public StudentDTO findById(Long id) {
29     // TODO Auto-generated method stub
30     return null;
31 }
```

```
StudentService.java x
28
29 @Transactional(readonly = true)
30 public StudentDTO findById(Long id) {
31
32     Optional<Student> obj = repository.findById(id);
33     Student entity = obj.get();
34
35     return new StudentDTO(entity);
36 }
```

## Rodar o projeto

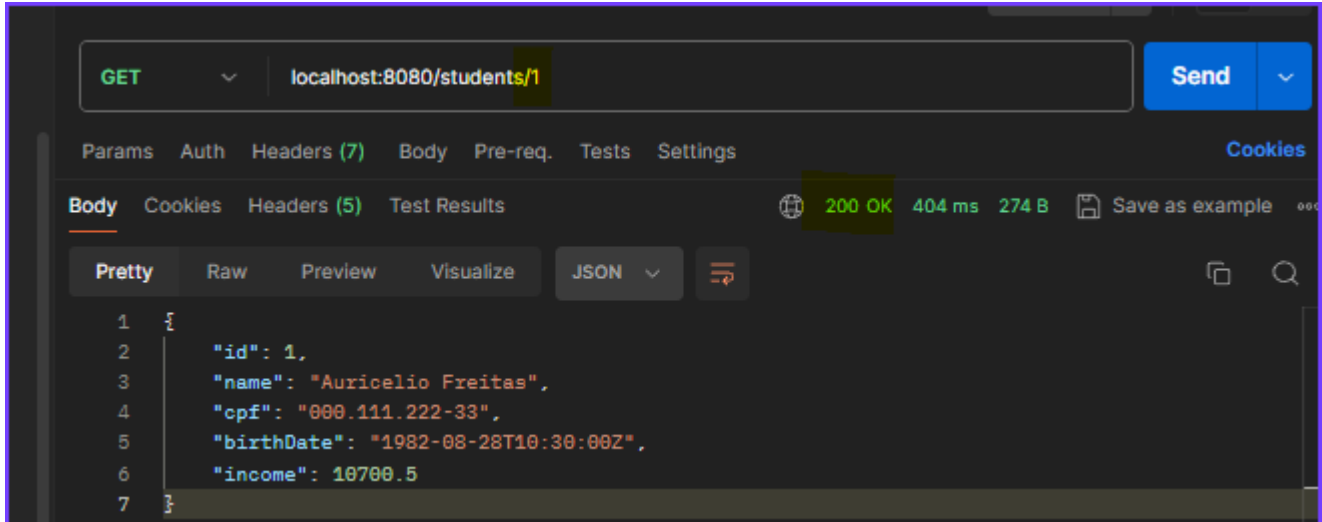


The screenshot shows the Spring Boot IDE interface. The left sidebar displays the 'Boot Dashboard' with a search bar and a list of running applications: 'local' and 'apirestful [:8080]'. The main area shows the 'Console' tab with the following output:

```
apirestful - ApirestfulApplication [Spring Boot App] C:\Program Files\Java\jdk
2024-03-31T11:56:07.930-03:00 INFO 13592 ---
2024-03-31T11:56:08.864-03:00 INFO 13592 ---
2024-03-31T11:56:08.909-03:00 INFO 13592 ---
2024-03-31T11:56:09.543-03:00 INFO 13592 ---
2024-03-31T11:56:09.552-03:00 INFO 13592 ---
```

## TESTES

### Testar no Postman



## Github-9

- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Implemented findByld”
  - git push






```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful




auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .



auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Implemented findById"
[main 743eeca] Implemented findById
2 files changed, 46 insertions(+)



auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (13/13), 1.12 KiB | 1.13 MiB/s, done.
Total 13 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
647f822..743eeca main -> main
```

 **unifametro-afdpw-APIRestful** Public

 Pin  Unwatch 1

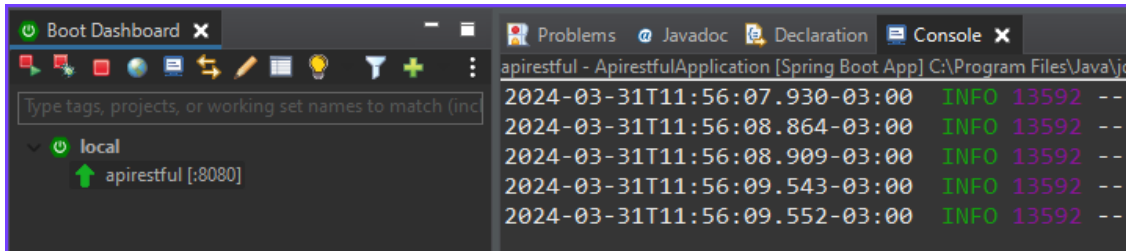
 main  1 Branch  0 Tags  t Add file <> Code

 **auriceliof** Implemented findById 743eeca · now  18 Commits

 backend	Implemented findById	now
 README.md	Update README.md	yesterday

## TRATAMENTO DE EXCEÇÕES PARA O FIND\_BY\_ID

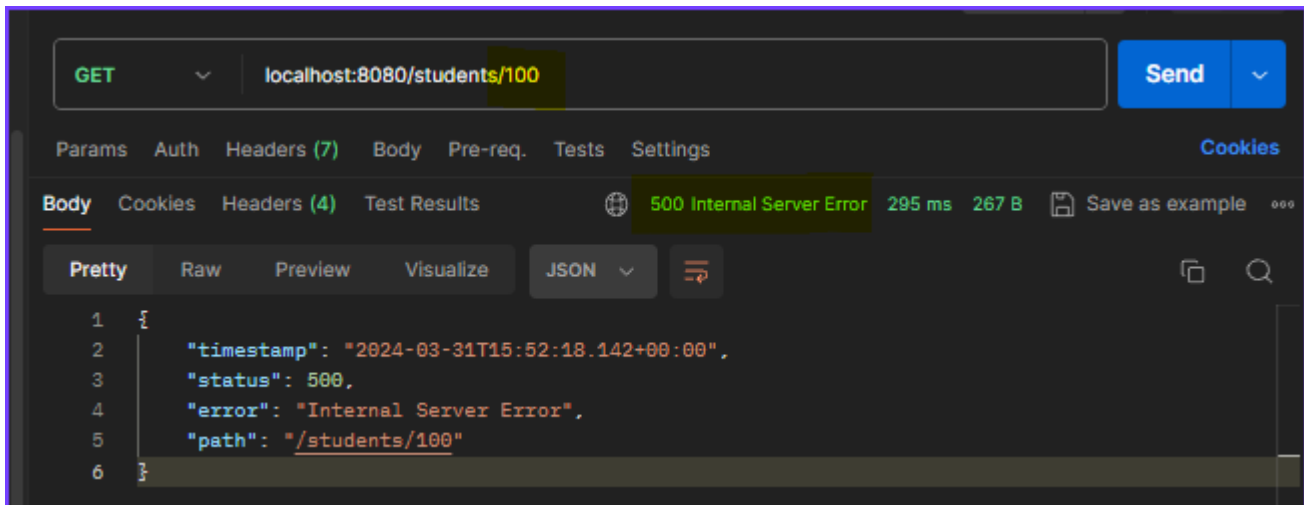
### Rodar o projeto



```
Boot Dashboard X
Type tags, projects, or working set names to match (inc)
local
  apirestful [:8080]

apirestful - ApirestfulApplication [Spring Boot App] C:\Program Files\Java\jdk-20\bin\javaw.exe (31 de mar. de 2024 12:51:56)
2024-03-31T11:56:07.930-03:00 INFO 13592 --- [apirestful] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet
2024-03-31T11:56:08.864-03:00 INFO 13592 --- [apirestful] [nio-8080-exec-1] o.a.c.c.C.[.][.][dispatcherServlet]
2024-03-31T11:56:08.909-03:00 INFO 13592 --- [apirestful] [nio-8080-exec-1] o.a.c.c.C.[.][.][dispatcherServlet]
2024-03-31T11:56:09.543-03:00 INFO 13592 --- [apirestful] [nio-8080-exec-1] o.a.c.c.C.[.][.][dispatcherServlet]
2024-03-31T11:56:09.552-03:00 INFO 13592 --- [apirestful] [nio-8080-exec-1] o.a.c.c.C.[.][.][dispatcherServlet]
```

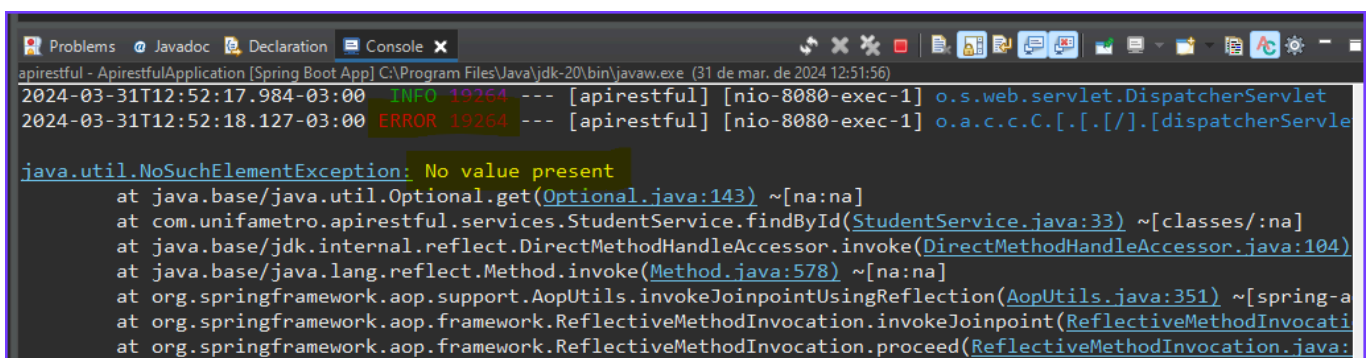
### Simular erro no Postman



```
GET localhost:8080/students/100 Send
Params Auth Headers (7) Body Pre-req. Tests Settings Cookies
Body Cookies Headers (4) Test Results 500 Internal Server Error 295 ms 267 B Save as example
Pretty Raw Preview Visualize JSON
1 {
2   "timestamp": "2024-03-31T15:52:18.142+00:00",
3   "status": 500,
4   "error": "Internal Server Error",
5   "path": "/students/100"
6 }
```

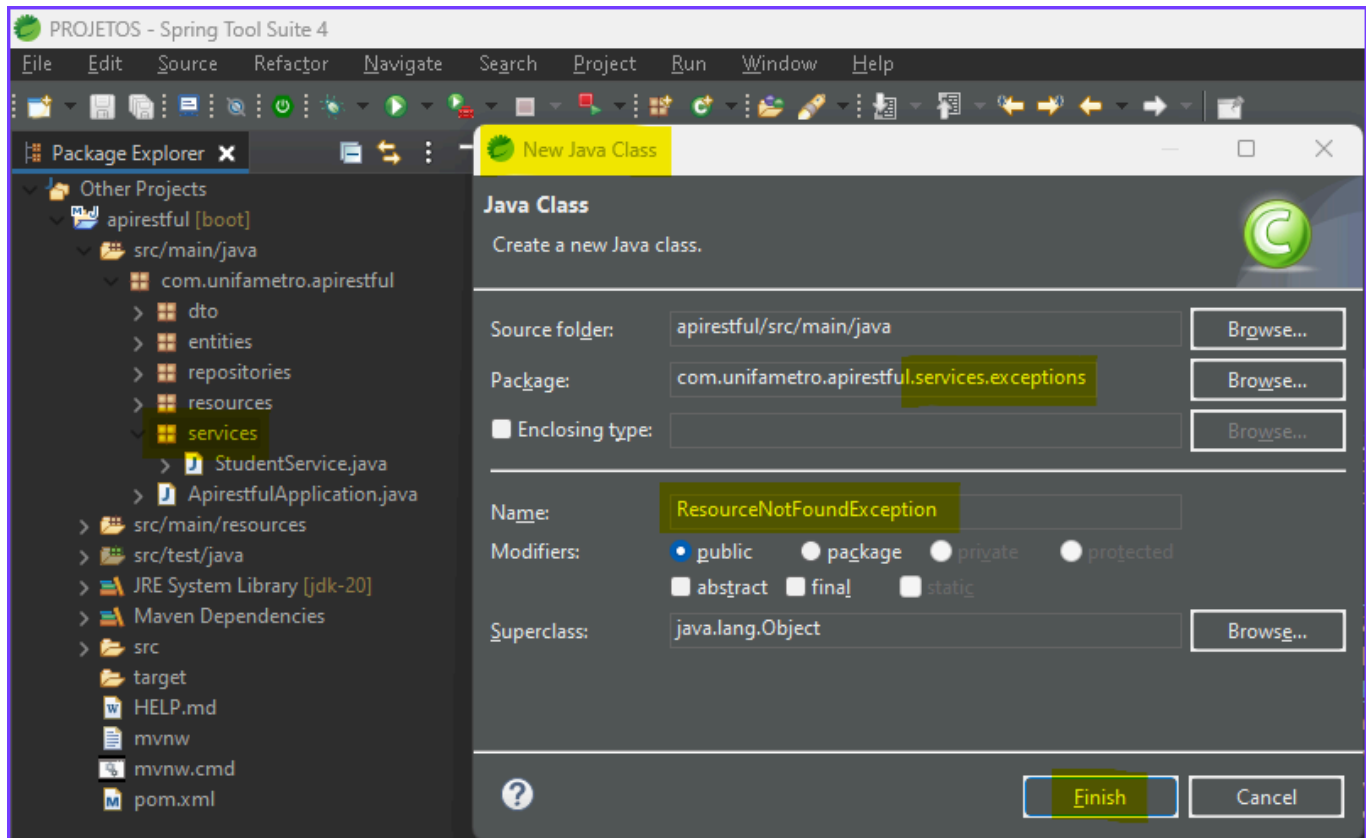
NOTA: Quando realizamos uma busca por um ID que não existe, é retornado o erro 500. Iremos tratá-lo a seguir.

### Verificar o erro no console



```
Problems Javadoc Declaration Console X
apirestful - ApirestfulApplication [Spring Boot App] C:\Program Files\Java\jdk-20\bin\javaw.exe (31 de mar. de 2024 12:51:56)
2024-03-31T12:52:17.984-03:00 INFO 19264 --- [apirestful] [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet
2024-03-31T12:52:18.127-03:00 ERROR 19264 --- [apirestful] [nio-8080-exec-1] o.a.c.c.C.[.][.][dispatcherServlet]
java.util.NoSuchElementException: No value present
    at java.base/java.util.Optional.get(Optional.java:143) ~[na:na]
    at com.unifametro.apirestful.services.StudentService.findById(StudentService.java:33) ~[classes/:na]
    at java.base/jdk.internal.reflect.DirectMethodHandleAccessor.invoke(DirectMethodHandleAccessor.java:104)
    at java.base/java.lang.reflect.Method.invoke(Method.java:578) ~[na:na]
    at org.springframework.aop.support.AopUtils.invokeJoinpointUsingReflection(AopUtils.java:351) ~[spring-aop-6.0.10.jar:6.0.10]
    at org.springframework.aop.framework.ReflectiveMethodInvocation.invokeJoinpoint(ReflectiveMethodInvocation.java:191)
    at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:137)
```

## Criar a estrutura de exceções no service e uma exceção personalizada



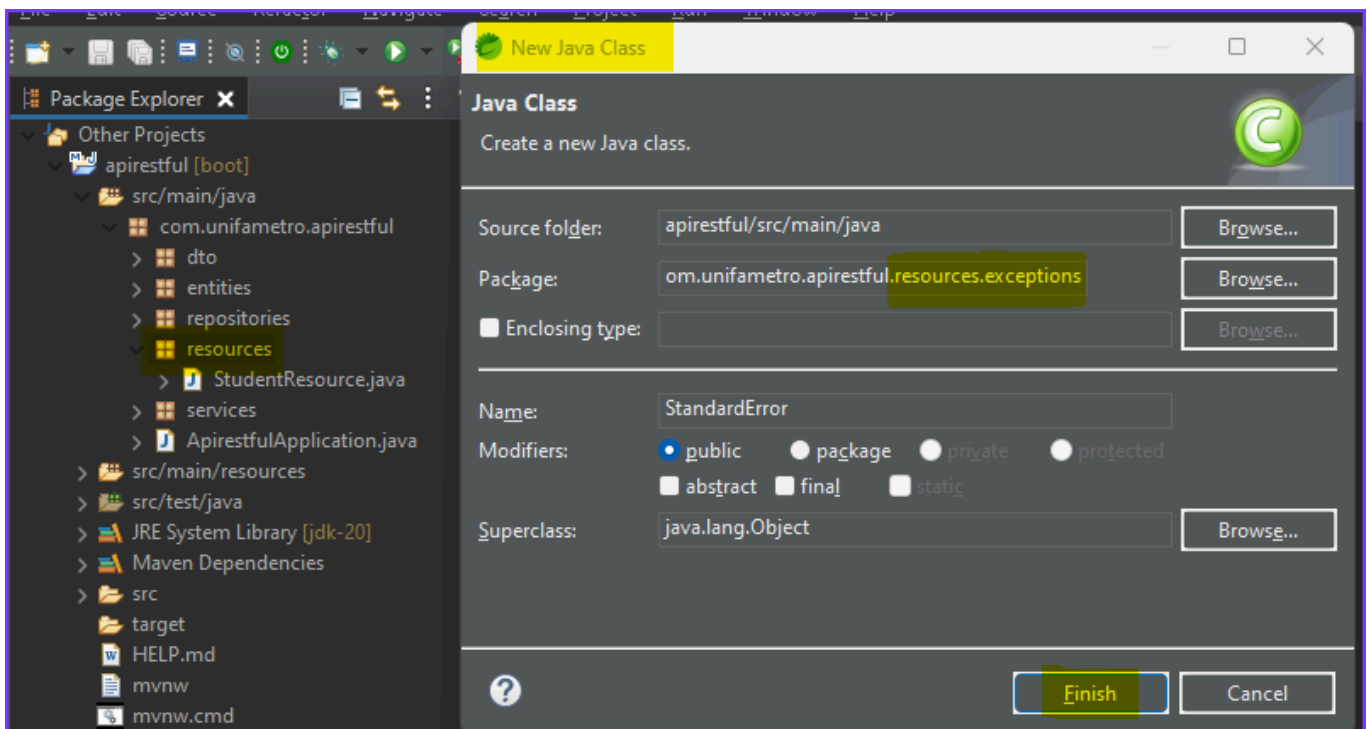
## Implementar o ResourceNotFoundException

```
ResourceNotFoundException.java
1 package com.unifametro.apirestful.services.exceptions;
2
3 public class ResourceNotFoundException extends RuntimeException{
4     private static final long serialVersionUID = 1L;
5
6     public ResourceNotFoundException(String msg) {
7         super(msg);
8     }
9 }
10
```

## Implementar a exceção no findById, do StudentService

```
*StudentService.java x
29
30 @Transactional(readonly = true)
31 public StudentDTO findById(Long id) {
32
33     Optional<Student> obj = repository.findById(id);
34     Student entity = obj.orElseThrow(() -> new ResourceNotFoundException("Entity not found"));
35
36     return new StudentDTO(entity);
37 }
```

## Criar a estrutura de exceções no resource e uma classe personalizada



## Implementar o Serializable e os atributos do StandardError

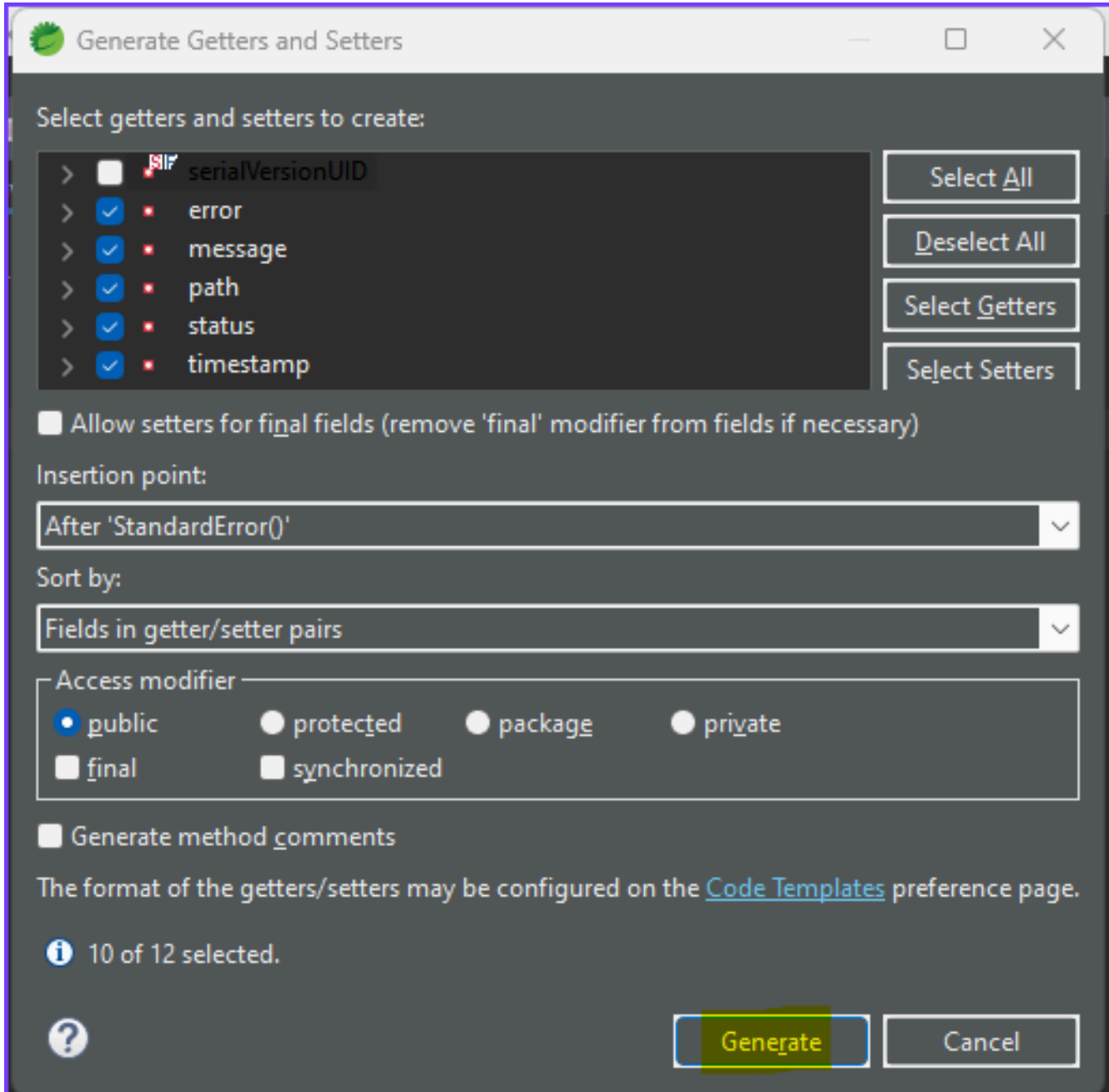
```
StandardError.java X
1 package com.unifametro.apirestful.resources.exceptions;
2
3 import java.io.Serializable;
4 import java.time.Instant;
5
6 public class StandardError implements Serializable{
7     private static final long serialVersionUID = 1L;
8
9     private Instant timestamp;
10    private Integer status;
11    private String error;
12    private String message;
13    private String path;
14
```

NOTA: Devemos definir os mesmos atributos mostrados no teste de erro do postman.

## Implementar um construtor vazio

```
*StandardError.java X
14
15 public StandardError() {
16
17 }
18
```

## Implementar os Getters and Setters



Generate Getters and Setters

Select getters and setters to create:

- > ☐ serialVersionUID
- > ☒ error
- > ☒ message
- > ☒ path
- > ☒ status
- > ☒ timestamp

Select All  
Deselect All  
Select Getters  
Select Setters

☐ Allow setters for final fields (remove 'final' modifier from fields if necessary)

Insertion point:  
After 'StandardError()'

Sort by:  
Fields in getter/setter pairs

Access modifier

☒ public ☐ protected ☐ package ☐ private

☐ final ☐ synchronized

☐ Generate method comments

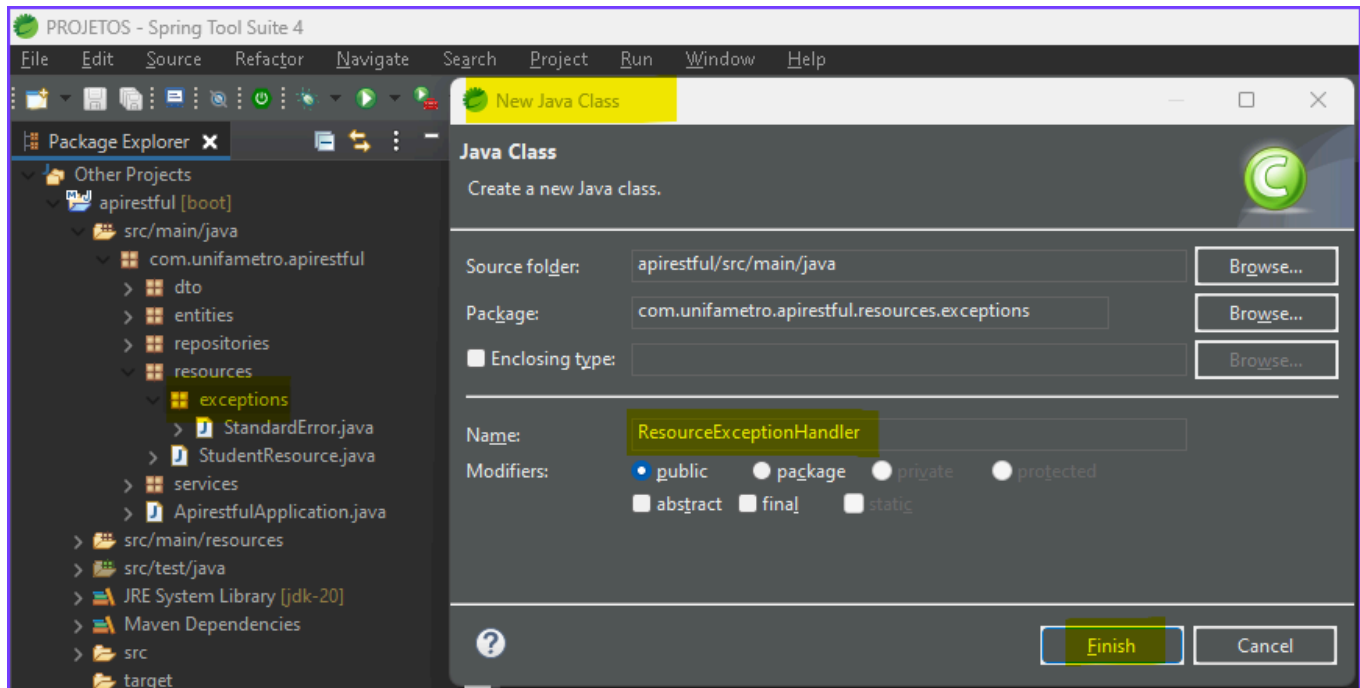
The format of the getters/setters may be configured on the [Code Templates](#) preference page.

**i** 10 of 12 selected.

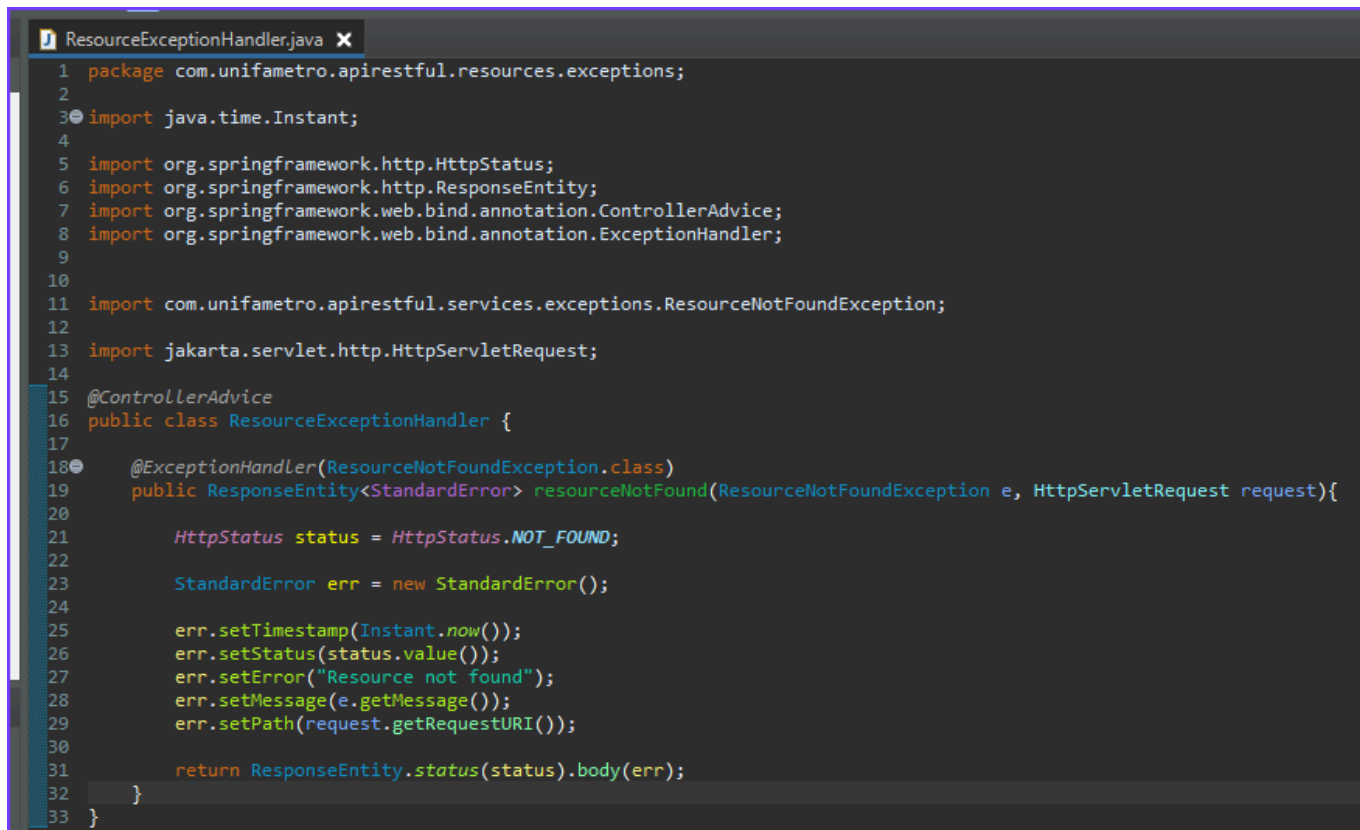
? Generate Cancel

```
StandardError.java x
18
19 public Instant getTimestamp() {
20     return timestamp;
21 }
22
23 public void setTimestamp(Instant timestamp) {
24     this.timestamp = timestamp;
25 }
26
27 public Integer getStatus() {
28     return status;
29 }
30
31 public void setStatus(Integer status) {
32     this.status = status;
33 }
34
35 public String getError() {
36     return error;
37 }
38
39 public void setError(String error) {
40     this.error = error;
41 }
42
43
44 public String getMessage() {
45     return message;
46 }
47
48 public void setMessage(String message) {
49     this.message = message;
50 }
51
52 public String getPath() {
53     return path;
54 }
55
56 public void setPath(String path) {
57     this.path = path;
58 }
59 }
```

## Criar um controller advice para manipular a exceção

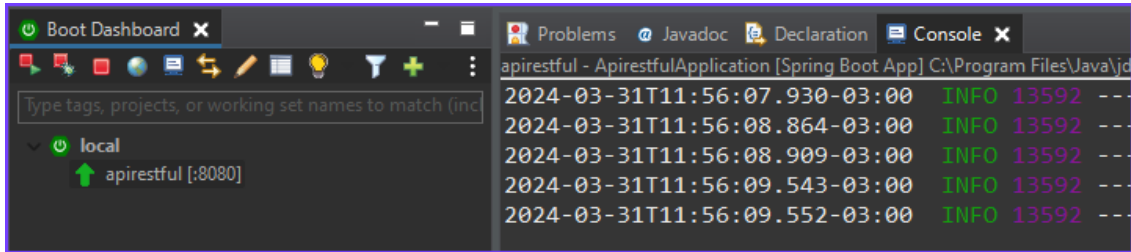


## Implementar o ResourceExceptionHandler





## Rodar o projeto

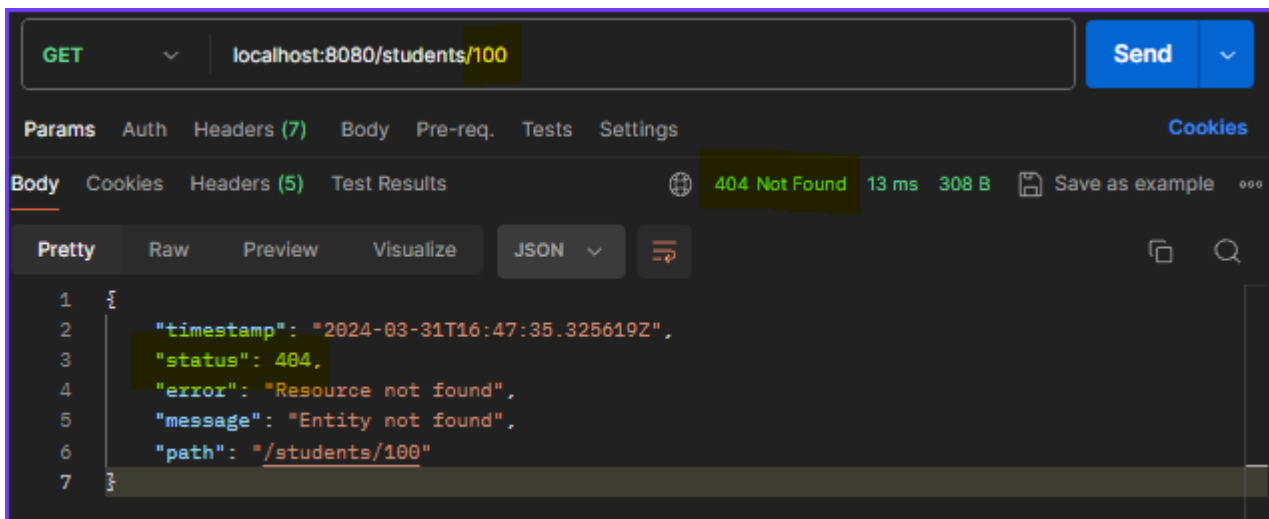


The screenshot shows the Spring Boot IDE interface. The left sidebar displays the 'local' environment with the 'apirestful' application running on port 8080. The main console window shows the following log output:

```
2024-03-31T11:56:07.930-03:00 INFO 13592 ---
2024-03-31T11:56:08.864-03:00 INFO 13592 ---
2024-03-31T11:56:08.909-03:00 INFO 13592 ---
2024-03-31T11:56:09.543-03:00 INFO 13592 ---
2024-03-31T11:56:09.552-03:00 INFO 13592 ---
```

## TESTES

### Testar no Postman



The screenshot shows the Postman interface. A GET request is sent to `localhost:8080/students/100`. The response is a `404 Not Found` status with a response time of 13 ms and a body size of 308 B. The response body is displayed in JSON format:

```
{
  "timestamp": "2024-03-31T16:47:35.325619Z",
  "status": 404,
  "error": "Resource not found",
  "message": "Entity not found",
  "path": "/students/100"
}
```


## Github-10

- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Implemented exceptions”
  - git push


```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful
auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Implemented exceptions"
[main 4e168e0] Implemented exceptions
4 files changed, 124 insertions(+), 1 deletion(-)
create mode 100644 backend/src/main/java/com/unifametro/apirestful/resources/exceptions/ResourceExceptionHandler.java
create mode 100644 backend/src/main/java/com/unifametro/apirestful/resources/exceptions/StandardError.java
create mode 100644 backend/src/main/java/com/unifametro/apirestful/services/exceptions/ResourceNotFoundException.java

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 28, done.
Counting objects: 100% (28/28), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (17/17), 2.11 KiB | 2.11 MiB/s, done.
Total 17 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
743eeca..4e168e0 main -> main
```

 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags  Add file Code

 **auriceliof** Implemented exceptions 4e168e0 · now 19 Commits

backend	Implemented exceptions	now
README.md	Update README.md	yesterday

## PAGINAÇÃO

### AJUSTAR O FIND\_ALL PARA BUSCA PAGINADA

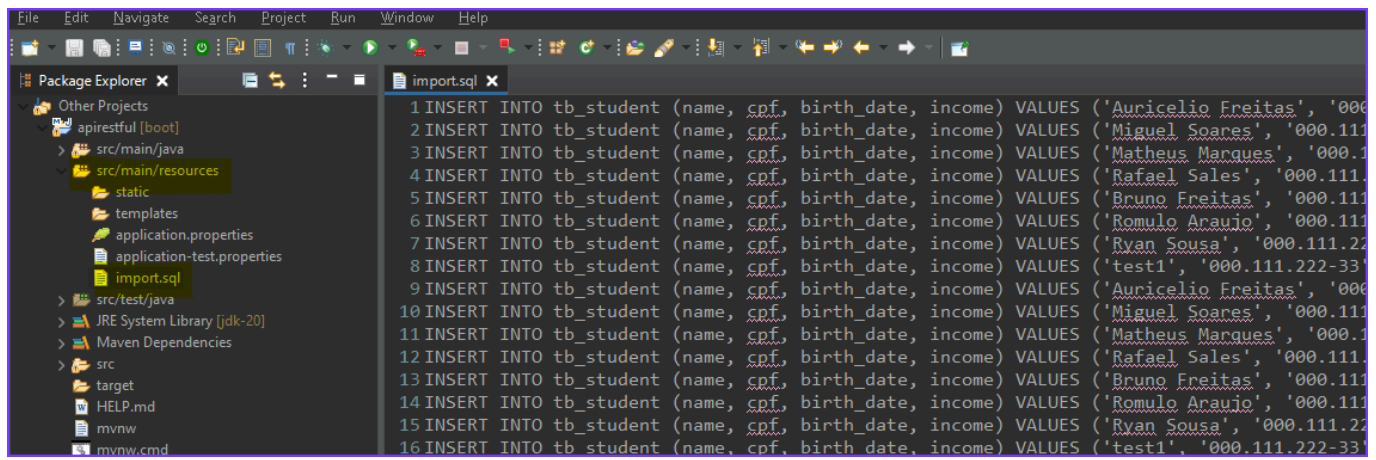
Implementar a busca paginada, no StudentResource

```
StudentResource.java x
23
24 @GetMapping
25 public ResponseEntity<Page<StudentDTO>> findAll(Pageable pageable){
26
27     Page<StudentDTO> list = service.findAllPaged(pageable);
28
29     return ResponseEntity.ok().body(list);
30 }
31
```

Ajustar a busca paginada, no StudentService

```
StudentService.java x
21
22 @Transactional(readonly = true)
23 public Page<StudentDTO> findAllPaged(Pageable pageable){
24
25     Page<Student> list = repository.findAll(pageable);
26
27     return list.map(x -> new StudentDTO(x));
28 }
29
```

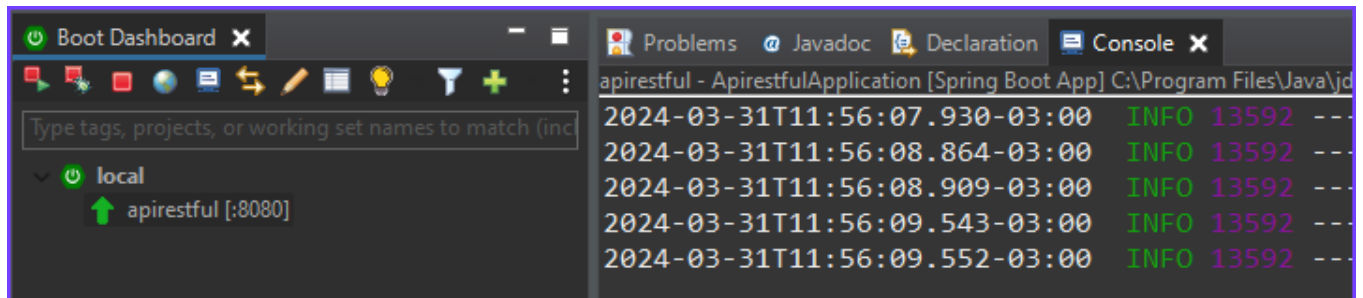
## Expandir o seed do banco para teste



```
1 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Auricelio Freitas', '000.111.222-33')
2 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Miguel Soares', '000.111.222-33')
3 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Matheus Marques', '000.111.222-33')
4 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Rafael Sales', '000.111.222-33')
5 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Bruno Freitas', '000.111.222-33')
6 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Romulo Araujo', '000.111.222-33')
7 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Ryan Sousa', '000.111.222-33')
8 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('test1', '000.111.222-33')
9 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Auricelio Freitas', '000.111.222-33')
10 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Miguel Soares', '000.111.222-33')
11 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Matheus Marques', '000.111.222-33')
12 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Rafael Sales', '000.111.222-33')
13 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Bruno Freitas', '000.111.222-33')
14 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Romulo Araujo', '000.111.222-33')
15 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('Ryan Sousa', '000.111.222-33')
16 INSERT INTO tb_student (name, cpf, birth_date, income) VALUES ('test1', '000.111.222-33')
```

*NOTA: Neste caso, apenas replicar os existentes (copiar, colar).*

## Rodar o projeto

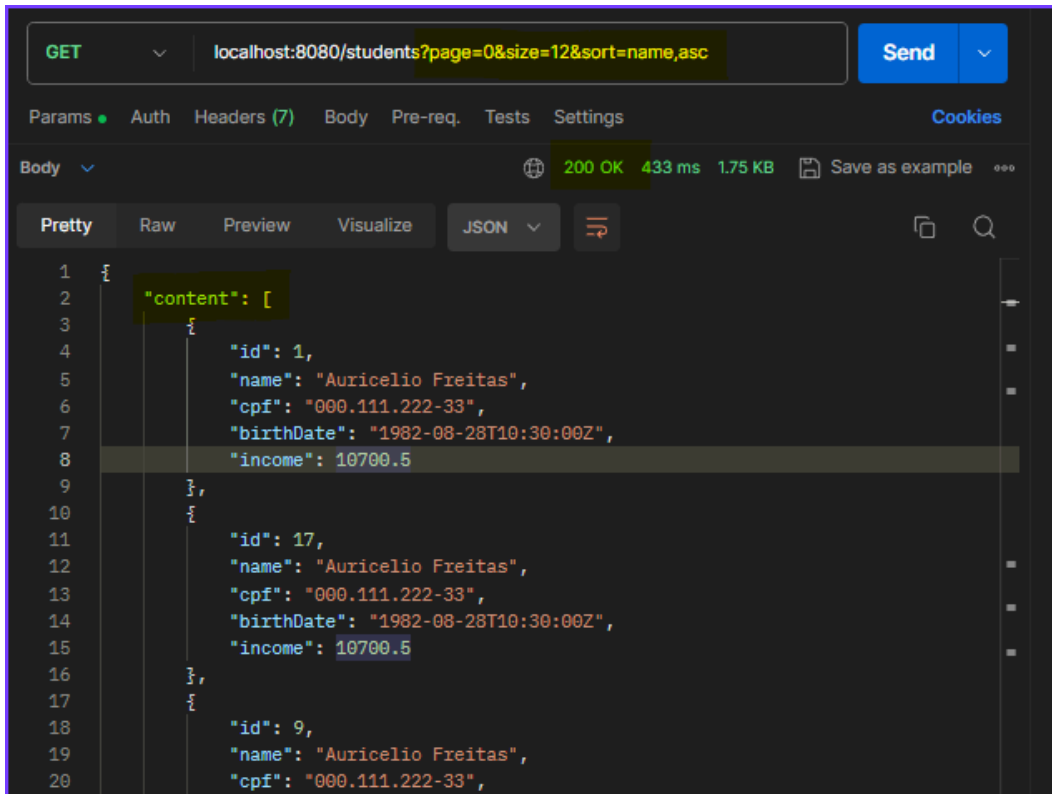


```
2024-03-31T11:56:07.930-03:00 INFO 13592 --
2024-03-31T11:56:08.864-03:00 INFO 13592 --
2024-03-31T11:56:08.909-03:00 INFO 13592 --
2024-03-31T11:56:09.543-03:00 INFO 13592 --
2024-03-31T11:56:09.552-03:00 INFO 13592 --
```

## TESTES

### Testar no Postman

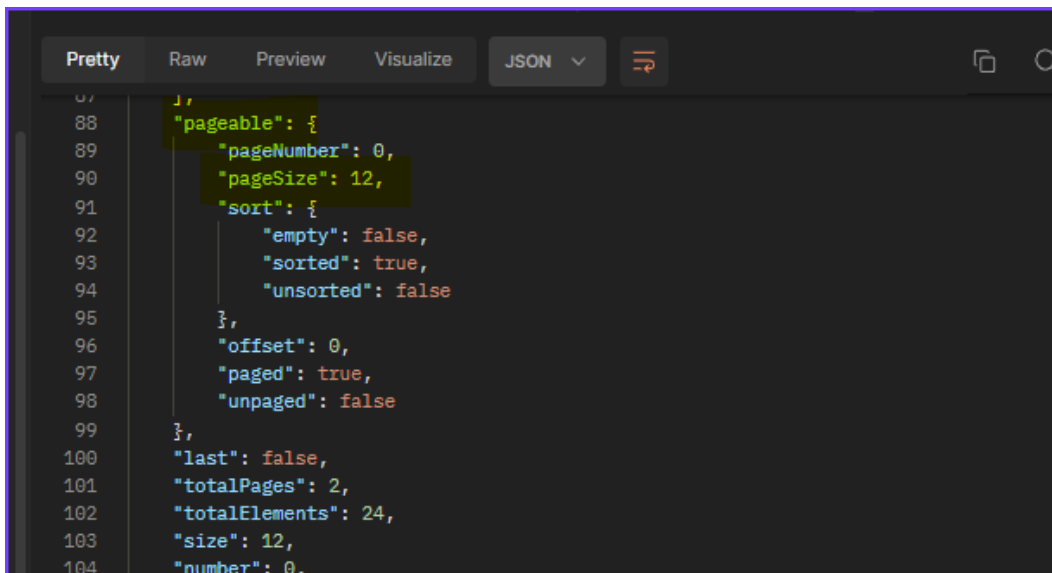
- *GET: localhost:8080/students?page=0&size=12&sort=name,asc*



GET localhost:8080/students?page=0&size=12&sort=name,asc

200 OK 433 ms 1.75 KB

```
1 {
2   "content": [
3     {
4       "id": 1,
5       "name": "Auricelio Freitas",
6       "cpf": "000.111.222-33",
7       "birthDate": "1982-08-28T10:30:00Z",
8       "income": 10700.5
9     },
10    {
11      "id": 17,
12      "name": "Auricelio Freitas",
13      "cpf": "000.111.222-33",
14      "birthDate": "1982-08-28T10:30:00Z",
15      "income": 10700.5
16    },
17    {
18      "id": 9,
19      "name": "Auricelio Freitas",
20      "cpf": "000.111.222-33",
```



```
87 },
88 "pageable": {
89   "pageNumber": 0,
90   "pageSize": 12,
91   "sort": {
92     "empty": false,
93     "sorted": true,
94     "unsorted": false
95   },
96   "offset": 0,
97   "paged": true,
98   "unpaged": false
99 },
100 "last": false,
101 "totalPages": 2,
102 "totalElements": 24,
103 "size": 12,
104 "number": 0,
```

## Github-11


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Implemented FindAllPaged”
  - git push

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .



auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Implemented FindAllPaged"
[main 16c78ef] Implemented FindAllPaged
3 files changed, 25 insertions(+), 7 deletions(-)

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 8 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (15/15), 1.54 KiB | 1.54 MiB/s, done.
Total 15 (delta 5), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (5/5), completed with 5 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
4e168e0..16c78ef  main -> main
```

 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags  Add file Code

 **auriceliof** Implemented FindAllPaged 16c78ef · now 20 Commits

 backend	Implemented FindAllPaged	now
 README.md	Update README.md	yesterday

## ENDPOINT - INSERT

### INSERIR NOVO ALUNO COM POST

Implementar o insert, no StudentResource

```
StudentResource.java x
39
40 @PostMapping
41 public ResponseEntity<StudentDTO> insert(@RequestBody StudentDTO dto){
42
43     dto = service.insert(dto);
44
45     return ResponseEntity.ok().body(dto);
46 }
47
```

Implementar a metodologia REST ao método

URI uri = ServletUriComponentsBuilder.fromCurrentRequest().path("/{id}").buildAndExpand(dto.getId()).toUri();

```
StudentResource.java x
42
43 @PostMapping
44 public ResponseEntity<StudentDTO> insert(@RequestBody StudentDTO dto){
45
46     dto = service.insert(dto);
47
48     URI uri = ServletUriComponentsBuilder.fromCurrentRequest().path("/{id}").buildAndExpand(dto.getId()).toUri();
49
50     return ResponseEntity.created(uri).body(dto);
51 }
52
```

OBS: Importar o URI do "java.net.URI"

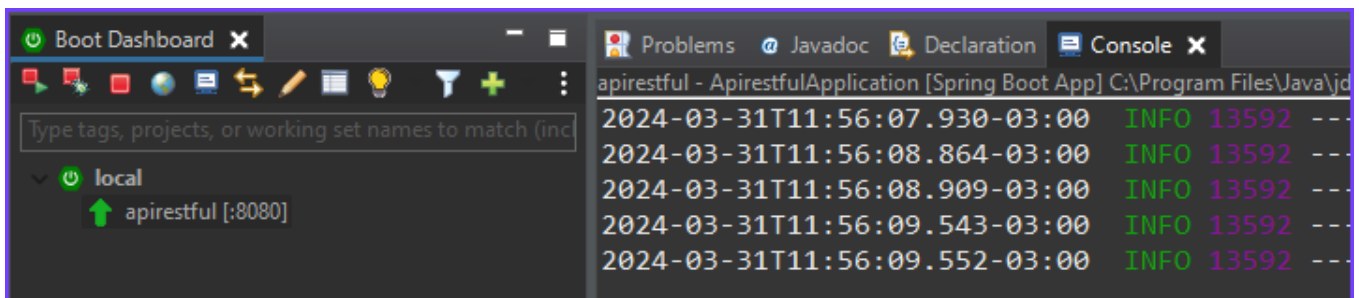
NOTA: Implantamos o caminho no Header da requisição e corrigimos o retorno de 200 (padrão) para 201 (recomendação REST), com o created.

Implementar o método insert convertendo o DTO para uma entidade, no StudentService

```
StudentService.java x
38
39 @Transactional
40 public StudentDTO insert(StudentDTO dto) {
41
42     Student entity = new Student();
43
44     entity.setName(dto.getName());
45     entity.setCpf(dto.getCpf());
46     entity.setBirthDate(dto.getBirthDate());
47     entity.setIncome(dto.getIncome());
48
49     entity = repository.save(entity);
50
51     return new StudentDTO(entity);
52 }
53 }
54
```

NOTA: Não colocar o atributo ID, pois é o banco que irá autoincrementar.

### Rodar o projeto



The screenshot shows the Spring Boot IDE interface. On the left, the 'Boot Dashboard' tab is active, displaying a list of running applications: 'local' and 'apirestful [:8080]'. The 'apirestful' application is highlighted with a green arrow. On the right, the 'Console' tab is active, showing the application's output. The output consists of five log entries, each starting with a timestamp and the word 'INFO', followed by the application's name and port number. The log entries are:

Timestamp	Log Level	Application Name	Port
2024-03-31T11:56:07.930-03:00	INFO	13592	--
2024-03-31T11:56:08.864-03:00	INFO	13592	--
2024-03-31T11:56:08.909-03:00	INFO	13592	--
2024-03-31T11:56:09.543-03:00	INFO	13592	--
2024-03-31T11:56:09.552-03:00	INFO	13592	--



## TESTES

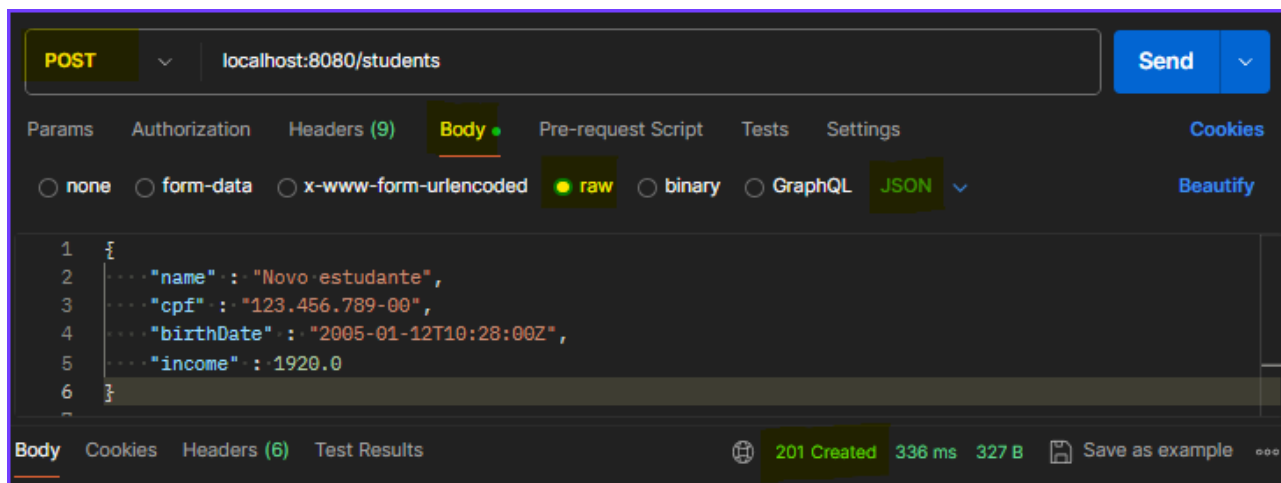
### Testar no Postman

- *POST: localhost:8080/students*

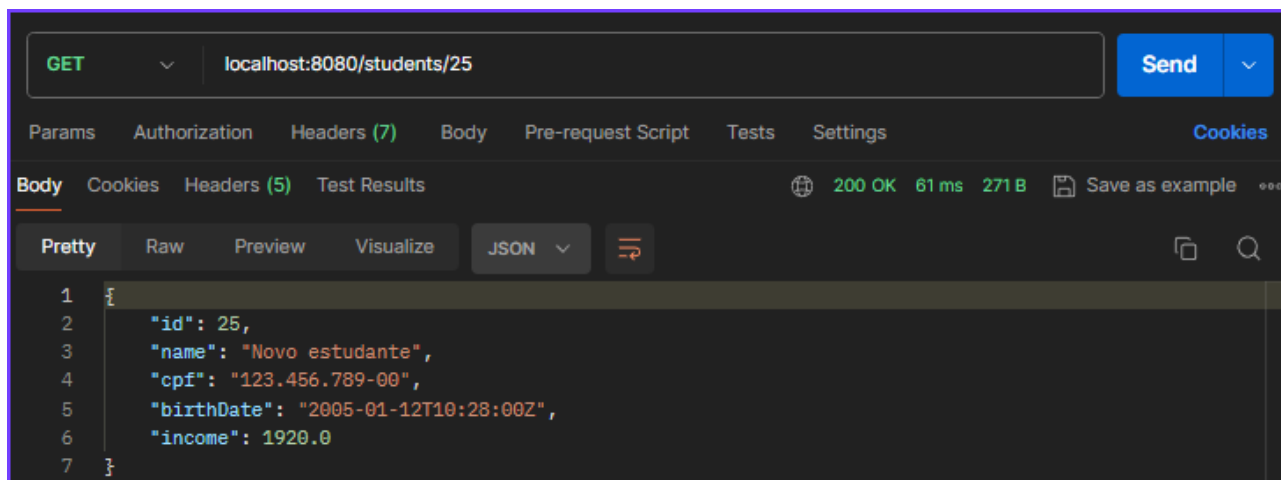
- *Body:*

```
{  
  "name": "Novo aluno",  
  "cpf": "123.456.789-00",  
  "birthDate": "2005-01-12T10:28:00Z",  
  "income": 1920.0  
}
```

### Inserir



### Buscar por Id



## Github-12


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Implemented Insert”
  - git push

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Implemented Insert"
[main 36849ac] Implemented Insert
2 files changed, 29 insertions(+), 1 deletion(-)

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (13/13), 1.35 KiB | 1.35 MiB/s, done.
Total 13 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
16c78ef..36849ac main -> main
```


 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags  Add file Code

 **auriceliof** Implemented Insert 36849ac · now 21 Commits

 backend

 Implemented Insert now

 README.md

 Update README.md yesterday

## ENDPOINT - UPDATE

### ATUALIZAR ALUNO COM PUT

Implementar o update, no StudentResource

```
StudentResource.java x
53
54 @PutMapping(value =("/{id}")
55 public ResponseEntity<StudentDTO> update(@PathVariable Long id, @RequestBody StudentDTO dto){
56
57     dto = service.update(id, dto);
58
59     return ResponseEntity.ok().body(dto);
60 }
61
```

Implementar o método update, no StudentService

```
*StudentService.java x
54
55 @Transactional
56 public StudentDTO update(Long id, StudentDTO dto) {
57
58     Student entity = repository.getReferenceById(id);
59
60     entity.setName(dto.getName());
61     entity.setCpf(dto.getCpf());
62     entity.setBirthDate(dto.getBirthDate());
63     entity.setIncome(dto.getIncome());
64
65     entity = repository.save(entity);
66
67     return new StudentDTO(entity);
68 }
```

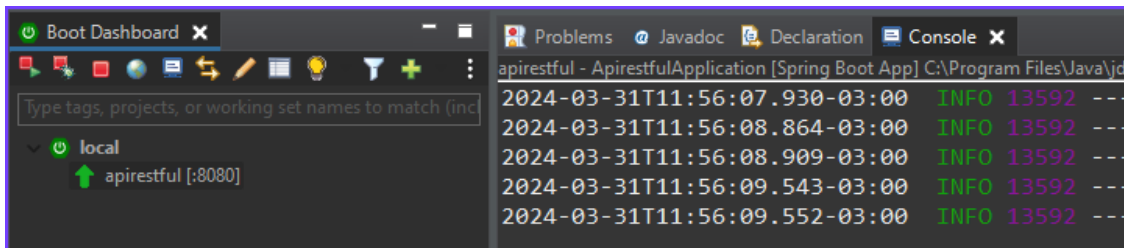
## TRATAMENTO DE ERRO

### Implementar o tratamento para ID Não encontrado

```
StudentService.java x
54
55 @Transactional
56 public StudentDTO update(Long id, StudentDTO dto) {
57     try {
58         Student entity = repository.getReferenceById(id);
59
60         entity.setName(dto.getName());
61         entity.setCpf(dto.getCpf());
62         entity.setBirthDate(dto.getBirthDate());
63         entity.setIncome(dto.getIncome());
64
65         entity = repository.save(entity);
66
67         return new StudentDTO(entity);
68     }
69     catch (ResourceNotFoundException e) {
70         throw new ResourceNotFoundException("ID not found" + id);
71     }
72 }
73
```

*NOTA: Devemos colocar o método update num bloco "Try Catch", pois ao atualizar um ID, este pode não existir.*

### Rodar o projeto



The screenshot shows the IDE interface with the 'Console' tab active. The console displays the following log messages:

```
apirestful - ApirestfulApplication [Spring Boot App] C:\Program Files\Java\jdk
2024-03-31T11:56:07.930-03:00 INFO 13592 ---
2024-03-31T11:56:08.864-03:00 INFO 13592 ---
2024-03-31T11:56:08.909-03:00 INFO 13592 ---
2024-03-31T11:56:09.543-03:00 INFO 13592 ---
2024-03-31T11:56:09.552-03:00 INFO 13592 ---
```

## TESTES

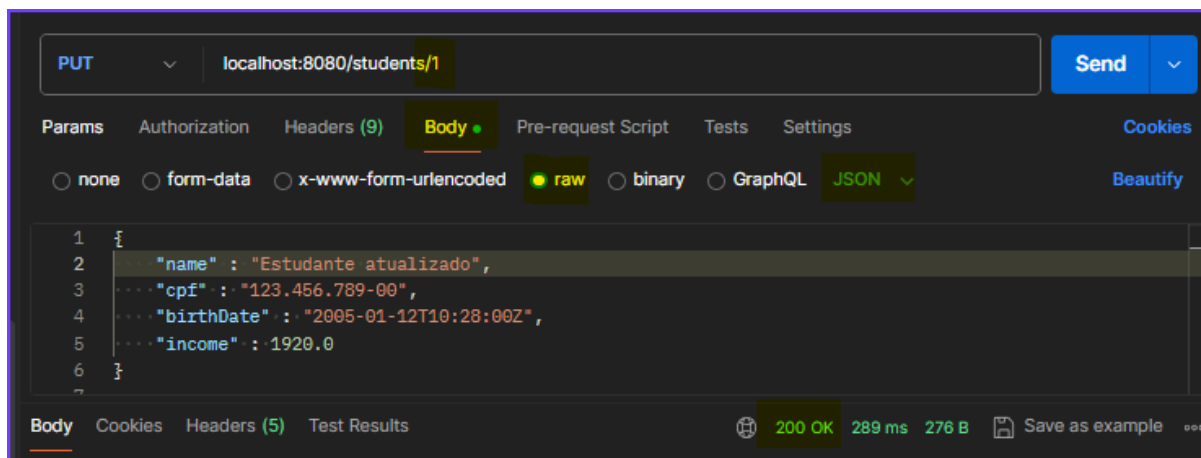
### Testar no Postman

- *PUT: localhost:8080/students/1*

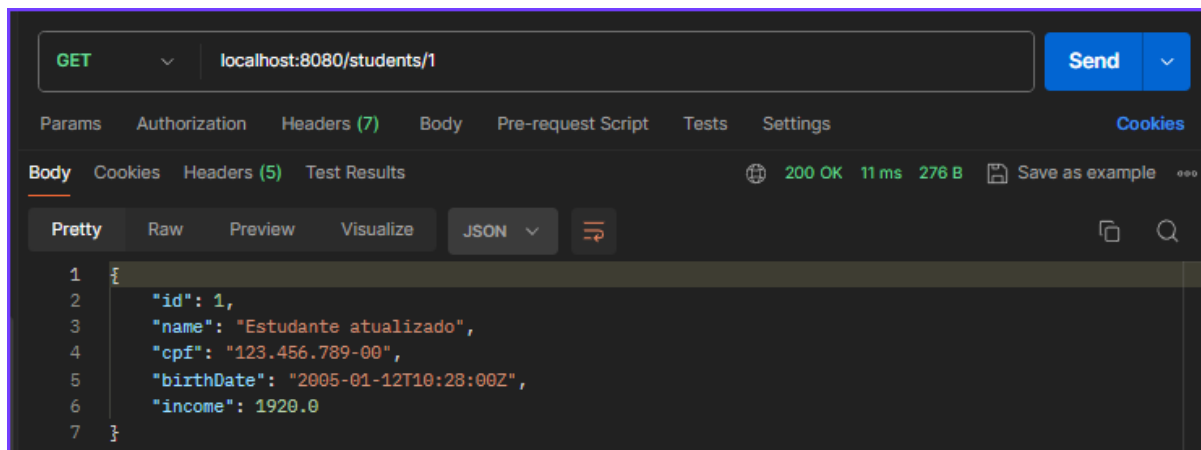
- *Body:*

```
{  
  "name" : "Aluno atualizado",  
  "cpf" : "123.456.789-00",  
  "birthDate" : "2005-01-12T10:28:00Z",  
  "income" : 1920.0  
}
```

### Update



### Busca por ID



## Github-13


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Implemented Update”
  - git push

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Implemented Update"
[main 9f798fe] Implemented Update
 2 files changed, 36 insertions(+), 1 deletion(-)

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (13/13), 1.25 KiB | 1.25 MiB/s, done.
Total 13 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
 36849ac..9f798fe  main -> main
```

 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags

Go to file t Add file <> Code

 auriceliof Implemented Update 9f798fe · now 22 Commits

backend	Implemented Update	now
README.md	Update README.md	yesterday

## ENDPOINT - DELETE

### DELETAR UM ALUNO COM DELETE

Implementar o update, no StudentResource

```
StudentResource.java x
62
63 @DeleteMapping(value =("/{id}")
64 public ResponseEntity<StudentDTO> delete(@PathVariable Long id){
65
66     service.delete(id);
67
68     return ResponseEntity.noContent().build();
69 }
70 }
```

Implementar o método update, no StudentService

```
*StudentService.java x
73
74 public void delete(Long id) {
75
76     repository.deleteById(id);
77 }
78 }
79 }
```

NOTA: No método delete não colocamos o `@Transactional`.

## TRATAMENTO DE ERRO DO DELETE

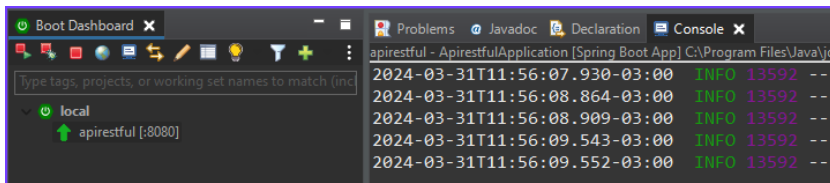
### Implementar o tratamento para ID Não encontrado

```
*StudentService.java x
74
75 public void delete(Long id) {
76     try {
77         repository.deleteById(id);
78     }
79     catch (EmptyResultDataAccessException e) {
80         throw new ResourceNotFoundException("ID not found" + id);
81     }
82 }
83 }
```

*NOTA: Devemos colocar o método delete num bloco "Try Catch", pois ao deletar um ID, este pode não existir.*

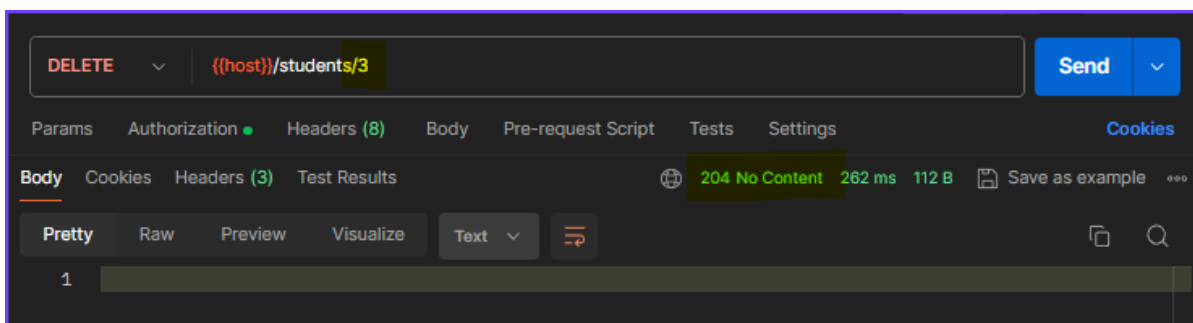
*OBS: Neste projeto não iremos abordar o tratamento de erro para INTEGRIDADE REFERENCIAL.*

### Rodar o projeto



## TESTES

### Testar no Postman





## Github-14


- “Git bash here” no diretório do projeto
  - git add backend
  - git commit -m “Implemented Update”
  - git push

```
MINGW64:/c/PROJETOS/UNIFAMETRO/APIRestful


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git add .


auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git commit -m "Implemented Delete"
[main 41cb7d1] Implemented Delete
2 files changed, 18 insertions(+), 2 deletions(-)

auric@NOTE-SAMSUNG MINGW64 /c/PROJETOS/UNIFAMETRO/APIRestful (main)
$ git push
Enumerating objects: 25, done.
Counting objects: 100% (25/25), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (13/13), 1.15 KiB | 1.15 MiB/s, done.
Total 13 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To https://github.com/auriceliof/unifametro-afdpw-APIRestful.git
9f798fe..41cb7d1  main -> main
```


 **unifametro-afdpw-APIRestful** Public Pin Unwatch 1

main 1 Branch 0 Tags  Add file Code

 **auriceliof** Implemented Delete 41cb7d1 · 1 minute ago 23 Commits

 backend

 Implemented Delete 1 minute ago

 README.md

 Update README.md yesterday