



EducaCiência FastCode

Fala Galera,

- Artigo: 31/2020 Data: Outubro/2020
- Público Alvo: Desenvolvedores – Iniciantes
- Tecnologia: Java
- Tema: Spring Boot Método Select + documentação Swagger
- Link: <https://github.com/perucello/DevFP>

Neste artigo, daremos abordaremos Spring Boot e iremos mapear o CRUD com repositório CRUD Repository.

Iniciaremos uma série de 4 artigos para concluir nosso propósito , sendo este 31/2020 o primeiro.

Para este ambiente , criaremos nosso Banco de Dados MySql e as persistências serão via código.

Nosso ambiente consiste em:

- ⇒ Banco de Dados MySql

```
1  
2 • create database EducaSpring;  
3 • use EducaSpring;  
4  
5
```

Output

Action Output

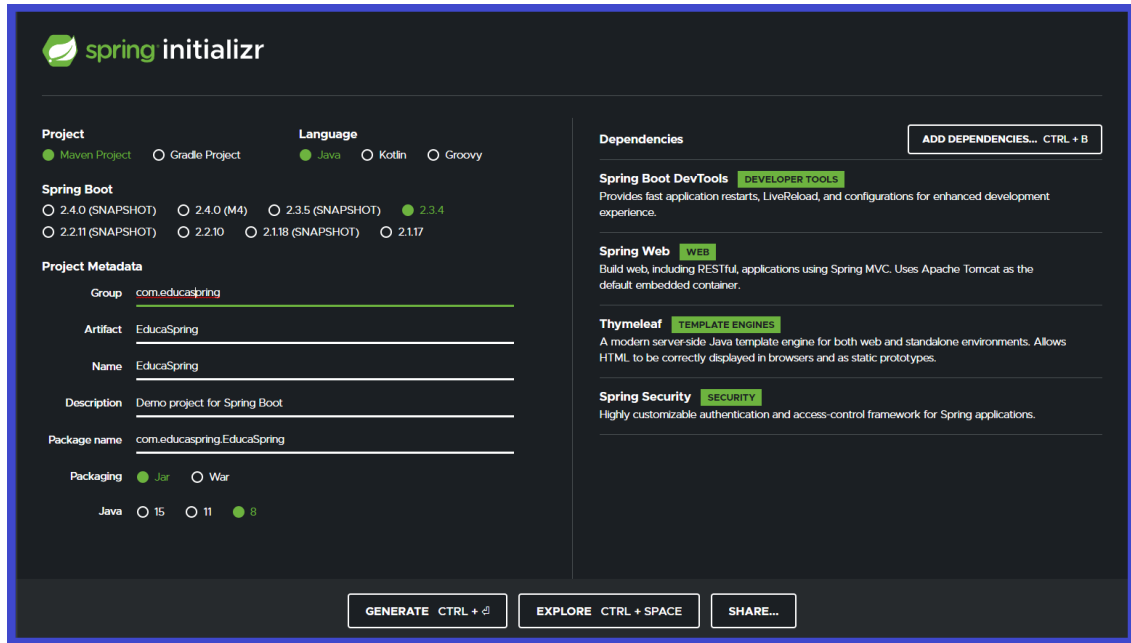
| # | Time | Action |
|-----|----------|-----------------------------|
| ✓ 1 | 13:20:44 | create database EducaSpring |
| ✓ 2 | 13:20:49 | use EducaSpring |

Para isso, utilizaremos de uma ferramenta para criarmos nosso projeto que é o Spring.io

- ⇒ <https://start.spring.io/>



Ao acessar a página , definiremos nosso Projeto como proposto:



Project

☒ Maven Project ☐ Gradle Project

Language

☒ Java ☐ Kotlin ☐ Groovy

Spring Boot

☐ 2.4.0 (SNAPSHOT) ☐ 2.4.0 (M4) ☐ 2.3.5 (SNAPSHOT) ☒ 2.3.4

☐ 2.2.11 (SNAPSHOT) ☐ 2.2.10 ☐ 2.1.18 (SNAPSHOT) ☐ 2.1.17

Project Metadata

Group:

Artifact:

Name:

Description:

Package name:

Packaging: ☒ Jar ☐ War

Java: ☐ 15 ☐ 11 ☒ 8

Dependencies

Spring Boot DevTools **DEVELOPER TOOLS**
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

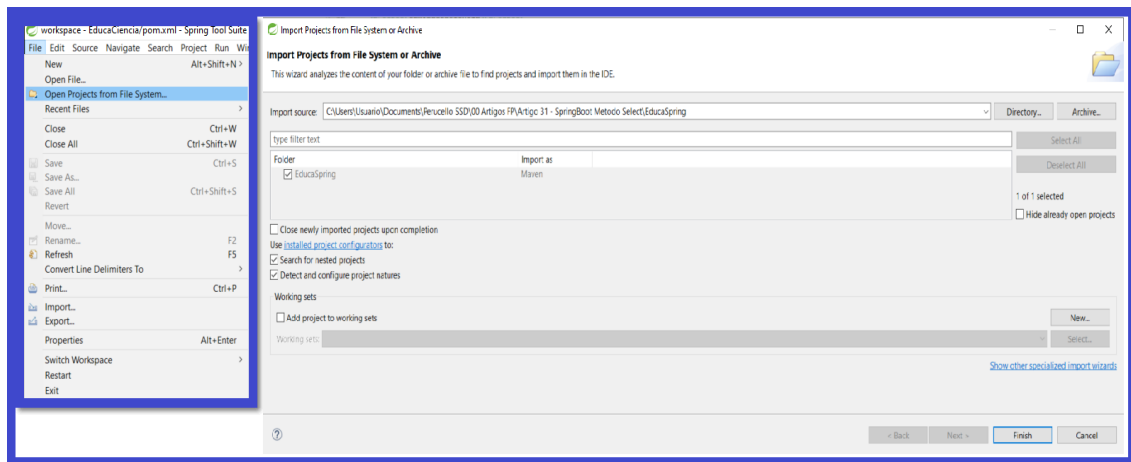
Spring Web **WEB**
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Thymeleaf **TEMPLATE ENGINES**
A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.

Spring Security **SECURITY**
Highly customizable authentication and access-control framework for Spring applications.

Buttons: GENERATE CTRL + G, EXPLORE CTRL + SPACE, SHARE...

Feito isso, basta baixarmos nosso projeto e abri-lo no STS.



Import Projects from File System or Archive

This wizard analyzes the content of your folder or archive file to find projects and import them in the IDE.

Import source:

Directory: Archive:

Type filter text:

Folder: ☒ EducaSpring

Import as:

☐ Close newly imported projects upon completion

Use [installed project configurations](#) too:

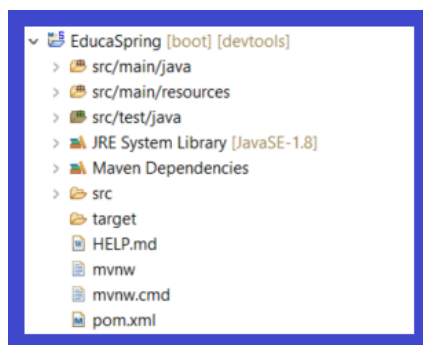
☒ Search for nested projects

☒ Detect and configure project natures

Working sets:

☐ Add project to working sets

Working sets:



Com o projeto aberto, vamos adicionar a dependência do MySQL no nosso arquivo pom.xml

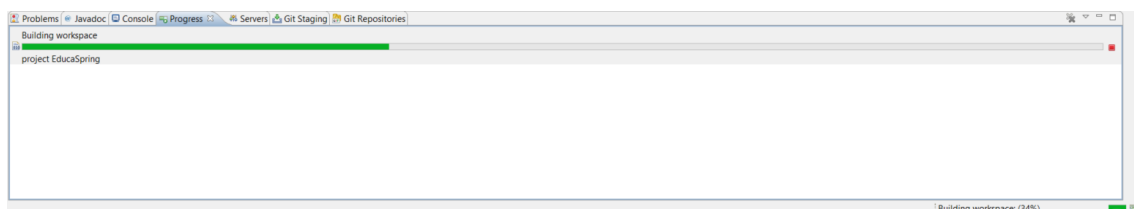


```

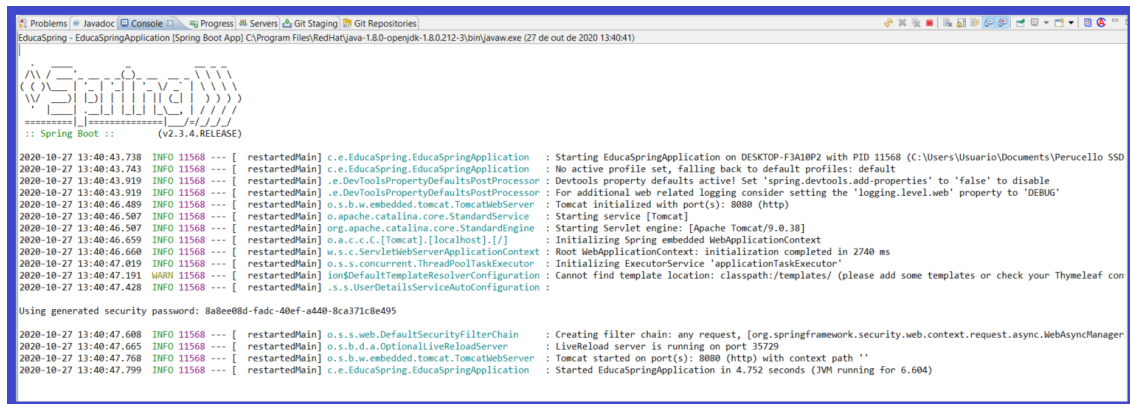
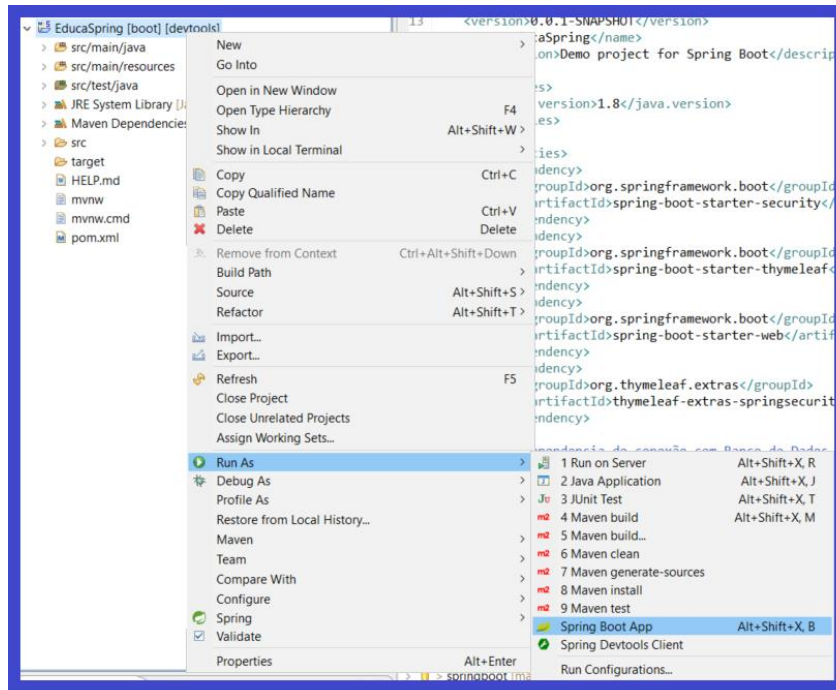
5<parent>
6  <groupId>org.springframework.boot</groupId>
7  <artifactId>spring-boot-starter-parent</artifactId>
8  <version>2.3.4.RELEASE</version>
9  <relativePath/> <!-- lookup parent from repository -->
10</parent>
11<groupId>com.educaspring</groupId>
12<artifactId>EducaSpring</artifactId>
13<version>0.0.1-SNAPSHOT</version>
14<name>EducaSpring</name>
15<description>Demo project for Spring Boot</description>
16
17<properties>
18  <java.version>1.8</java.version>
19</properties>
20
21<dependencies>
22  <dependency>
23    <groupId>org.springframework.boot</groupId>
24    <artifactId>spring-boot-starter-security</artifactId>
25  </dependency>
26  <dependency>
27    <groupId>org.springframework.boot</groupId>
28    <artifactId>spring-boot-starter-thymeleaf</artifactId>
29  </dependency>
30  <dependency>
31    <groupId>org.springframework.boot</groupId>
32    <artifactId>spring-boot-starter-web</artifactId>
33  </dependency>
34  <dependency>
35    <groupId>org.thymeleaf.extras</groupId>
36    <artifactId>thymeleaf-extras-springsecurity5</artifactId>
37  </dependency>
38
39  <!-- Dependência de conexão com Banco de Dados Mysql -->
40  <dependency>
41    <groupId>mysql</groupId>
42    <artifactId>mysql-connector-java</artifactId>
43  </dependency>

```

Feito isso podemos salvar nosso Projeto, um dos pontos favoráveis do STS é que ao salvarmos o projeto, as dependências que foram recentemente incluídas já são baixadas automaticamente, caso esteja com outra IDE, se faz necessário ir em **Run As\Maven Install**

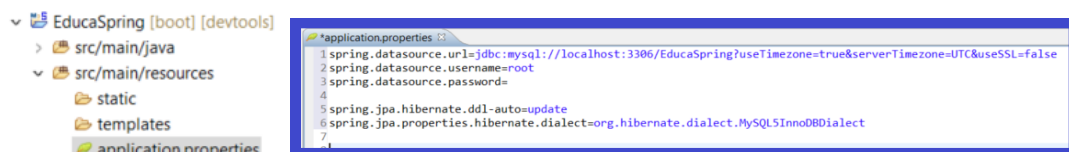


Para testarmos nossa aplicação, se está subindo corretamente nosso projeto, vamos em:



Agora que nossa aplicação carregou corretamente, vamos preparar nosso ambiente. Como vimos anteriormente, nós criamos o Banco de Dados no MySQL e já trouxemos a dependência no nosso arquivo pom.xml, no entanto, vamos agora preparar nossa conexão.

A classe responsável por receber estas informações é application.properties, então vamos lá.





Podemos salvar estas alterações, com o projeto em execução, ele irá reiniciar e atualizar de maneira automática.

```
Problems | Javadoc | Console | Progress | Servers | Git Staging | Git Repositories
EducaSpring - EducaSpringApplication [Spring Boot App] C:\Program Files\RedHat\java-1.8.0-openjdk-1.8.0.212-3\bin\javaw.exe (27 de out de 2020 13:40:41)
:: Spring Boot :: (v2.3.4.RELEASE)

2020-10-27 13:40:43.738 INFO 11568 --- [ restartedMain] c.e.EducaSpring.EducaSpringApplication : Starting EducaSpringApplication on DESKTOP-F3A10P2 with PID 11568 (C:\Users\Usuario\Documents\Perucello SSD
2020-10-27 13:40:43.743 INFO 11568 --- [ restartedMain] c.e.EducaSpring.EducaSpringApplication : No active profile set, falling back to default profiles: default
2020-10-27 13:40:43.919 INFO 11568 --- [ restartedMain] e.DevToolsPropertyDefaultsPostProcessor : DevTools property defaults active! Set 'spring.devtools.add-properties' to 'false' to disable
2020-10-27 13:40:43.919 INFO 11568 --- [ restartedMain] e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging consider setting the 'logging.level.web' property to 'DEBUG'
2020-10-27 13:40:46.489 INFO 11568 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2020-10-27 13:40:46.507 INFO 11568 --- [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-10-27 13:40:46.507 INFO 11568 --- [ restartedMain] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-10-27 13:40:46.659 INFO 11568 --- [ restartedMain] o.a.c.c.c.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2020-10-27 13:40:46.660 INFO 11568 --- [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 2740 ms
2020-10-27 13:40:47.019 INFO 11568 --- [ restartedMain] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-10-27 13:40:47.191 WARN 11568 --- [ restartedMain] ionDefaultTemplateResolverConfiguration : Cannot find template location: classpath:/templates/ (please add some templates or check your Thymeleaf con
2020-10-27 13:40:47.428 INFO 11568 --- [ restartedMain] .s.s.UserDetailsServiceAutoConfiguration :

Using generated security password: 8a8ee08d-fadc-40ef-a440-8ca371c8e495

2020-10-27 13:40:47.608 INFO 11568 --- [ restartedMain] o.s.s.web.DefaultSecurityFilterChain : Creating filter chain: any request, [org.springframework.security.web.context.request.async.WebAsyncManager
2020-10-27 13:40:47.665 INFO 11568 --- [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2020-10-27 13:40:47.768 INFO 11568 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2020-10-27 13:40:47.799 INFO 11568 --- [ restartedMain] c.e.EducaSpring.EducaSpringApplication : Started EducaSpringApplication in 4.752 seconds (JVM running for 6.604)
2020-10-27 13:45:54.093 INFO 11568 --- [ Thread-7] o.s.s.concurrent.ThreadPoolTaskExecutor : Shutting down ExecutorService 'applicationTaskExecutor'

2020-10-27 13:45:54.275 INFO 11568 --- [ restartedMain] c.e.EducaSpring.EducaSpringApplication : Starting EducaSpringApplication on DESKTOP-F3A10P2 with PID 11568 (C:\Users\Usuario\Documents\Perucello SSD
2020-10-27 13:45:54.276 INFO 11568 --- [ restartedMain] c.e.EducaSpring.EducaSpringApplication : No active profile set, falling back to default profiles: default
2020-10-27 13:45:54.724 INFO 11568 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080 (http)
2020-10-27 13:45:54.725 INFO 11568 --- [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2020-10-27 13:45:54.725 INFO 11568 --- [ restartedMain] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat/9.0.38]
2020-10-27 13:45:54.742 INFO 11568 --- [ restartedMain] o.a.c.c.c.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2020-10-27 13:45:54.742 INFO 11568 --- [ restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 461 ms
2020-10-27 13:45:54.823 INFO 11568 --- [ restartedMain] o.s.s.concurrent.ThreadPoolTaskExecutor : Initializing ExecutorService 'applicationTaskExecutor'
2020-10-27 13:45:54.871 INFO 11568 --- [ restartedMain] ionDefaultTemplateResolverConfiguration : Cannot find template location: classpath:/templates/ (please add some templates or check your Thymeleaf con
2020-10-27 13:45:54.908 INFO 11568 --- [ restartedMain] .s.s.UserDetailsServiceAutoConfiguration :

Using generated security password: a5b148c3-243b-4594-86fc-7b540c015a44

2020-10-27 13:45:54.916 INFO 11568 --- [ restartedMain] o.s.s.web.DefaultSecurityFilterChain : Creating filter chain: any request, [org.springframework.security.web.context.request.async.WebAsyncManager
2020-10-27 13:45:54.933 INFO 11568 --- [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2020-10-27 13:45:54.953 INFO 11568 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2020-10-27 13:45:54.976 INFO 11568 --- [ restartedMain] c.e.EducaSpring.EducaSpringApplication : Started EducaSpringApplication in 0.755 seconds (JVM running for 313.782)
2020-10-27 13:45:54.981 INFO 11568 --- [ restartedMain] .ConditionEvaluationDeltaLoggingListener : Condition evaluation unchanged
```

Agora , com a conexão preparada junto ao nosso Banco de Dados, precisamos partir para nossa arquitetura criando nossos pacotes que irão compor nosso projeto.

Sendo assim, criaremos os seguintes pacotes:



No Pacote **com.educaspring.EducaSpring.model.entities** criaremos nossa classe que receberá os dados da nossa tabela que criaremos no Banco de Dados.

Para isso, adicionaremos mais uma dependência em nosso arquivo pom.xml e salvar na sequência:




```

44
45 <!-- Dependencia de data JPA -->
46 <dependency>
47   <groupId>org.springframework.boot</groupId>
48   <artifactId>spring-boot-starter-data-jpa</artifactId>
49 </dependency>
50

```

Agora, iremos criar uma tabela no nosso banco de dados, para isso, iremos utilizar da anotação Entity.

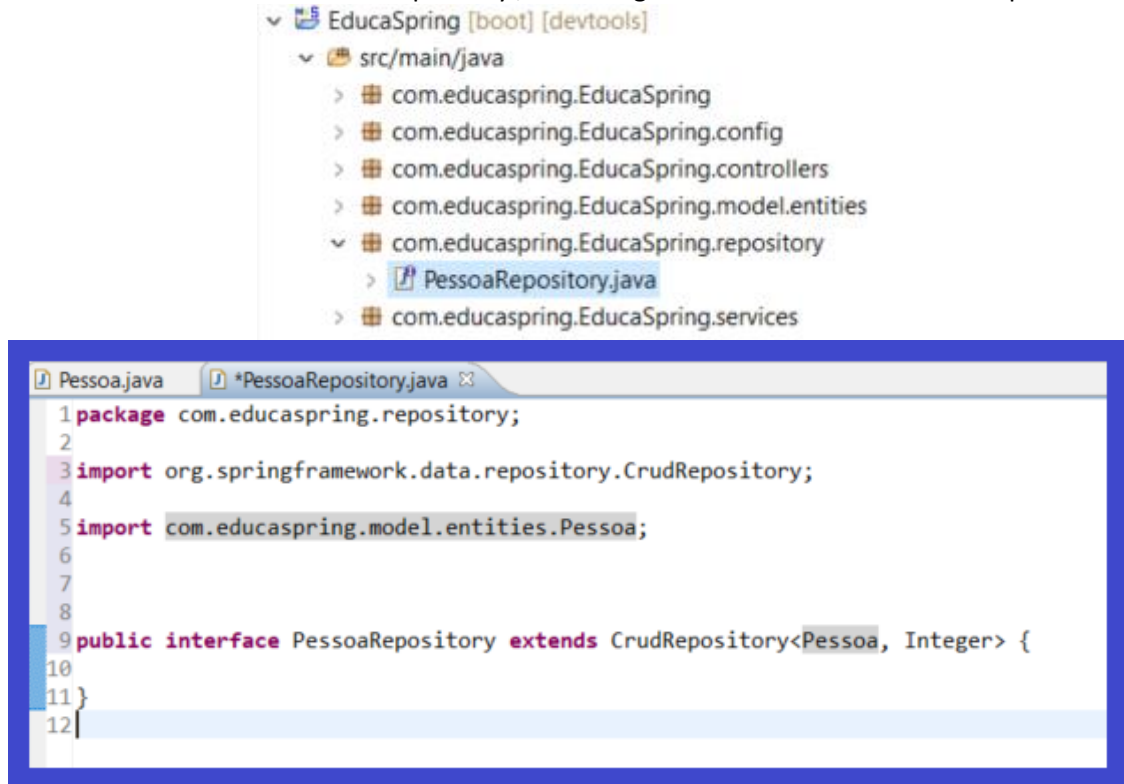
 EducaSpring [boot] [devtools]
 src/main/java
 > com.educaspring.EducaSpring
 > com.educaspring.EducaSpring.config
 > com.educaspring.EducaSpring.controllers
 > com.educaspring.EducaSpring.model.entities
 > Pessoa.java
 > com.educaspring.EducaSpring.repository
 > com.educaspring.EducaSpring.services

```

Pessoa.java
1 package com.educaspring.model.entities;
2
3 import javax.persistence.Entity;
4 import javax.persistence.GeneratedValue;
5 import javax.persistence.GenerationType;
6 import javax.persistence.Id;
7
8 import org.springframework.lang.NonNull;
9
10 @Entity
11 public class Pessoa {
12     @Id
13     @GeneratedValue(strategy = GenerationType.IDENTITY)
14     private int id;
15
16     @NonNull
17     private String nome;
18
19     private String email;
20
21     /**
22      *
23      */
24     public Pessoa() {
25         super();
26     }
27
28     /**
29      * @param id
30      * @param nome
31      * @param email
32      */
33     public Pessoa(int id, String nome, String email) {
34         super();
35         this.id = id;
36         this.nome = nome;
37         this.email = email;
38     }
39
40     public int getId() {
41         return id;
42     }
43
44     public void setId(int id) {
45         this.id = id;
46     }
47
48     public String getNome() {
49         return nome;
50     }
51
52     public void setNome(String nome) {
53         this.nome = nome;
54     }
55
56     public String getEmail() {
57         return email;
58     }
59
60     public void setEmail(String email) {
61         this.email = email;
62     }
63
64     @Override
65     public int hashCode() {
66         final int prime = 31;
67         int result = 1;
68         result = prime * result + ((email == null) ? 0 : email.hashCode());
69         result = prime * result + id;
70         result = prime * result + ((nome == null) ? 0 : nome.hashCode());
71         return result;
72     }
73
74     @Override
75     public boolean equals(Object obj) {
76         if (this == obj)
77             return true;
78         if (obj == null)
79             return false;
80         if (getClass() != obj.getClass())
81             return false;
82         Pessoa other = (Pessoa) obj;
83         if (email == null) {
84             if (other.email != null)
85                 return false;
86         } else if (!email.equals(other.email))
87             return false;
88         if (id != other.id)
89             return false;
90         if (nome == null) {
91             if (other.nome != null)
92                 return false;
93         } else if (!nome.equals(other.nome))
94             return false;
95         return true;
96     }
97
98 }
99
100
101

```

Como trabalharemos com CrudRepository, iremos agora criar nossa interface do Repositório.



The screenshot shows the project structure of 'EducaSpring' in the 'src/main/java' directory. The package 'com.educaspring.repository' is expanded, showing the 'PessoaRepository.java' file. Below, the code for 'PessoaRepository.java' is displayed:

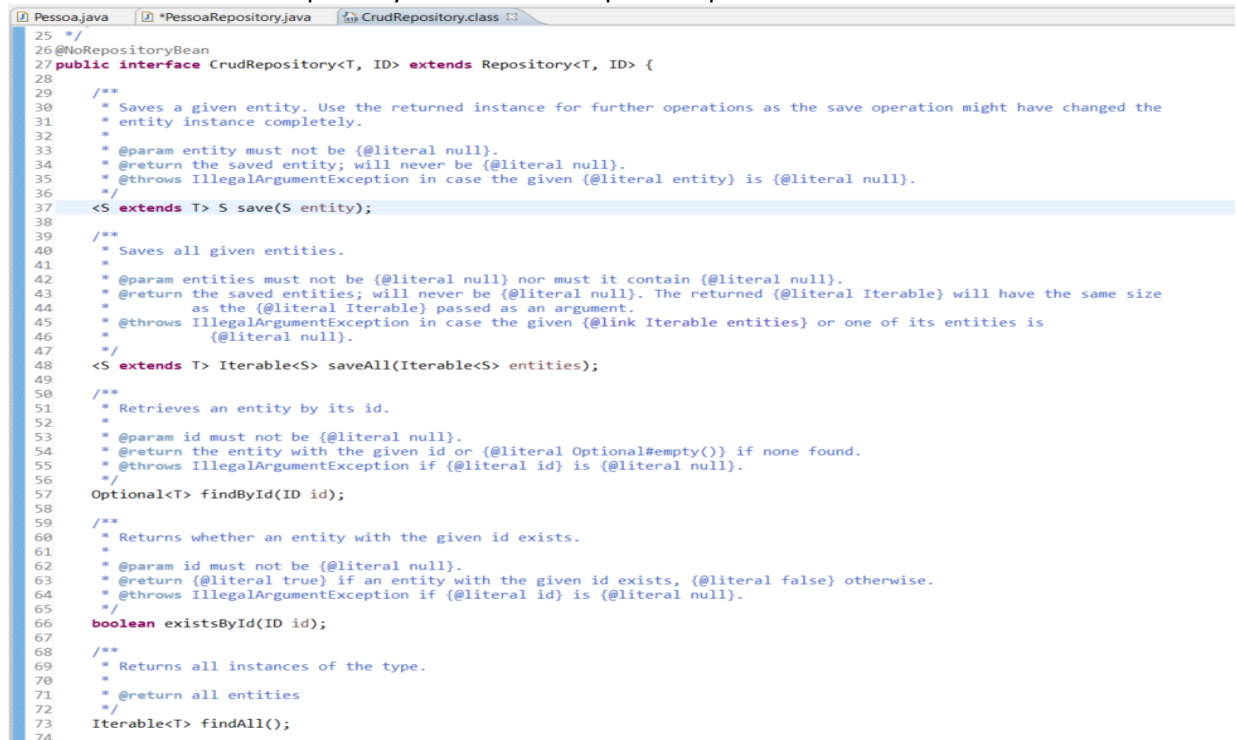
```

1 package com.educaspring.repository;
2
3 import org.springframework.data.repository.CrudRepository;
4
5 import com.educaspring.model.entities.Pessoa;
6
7
8
9 public interface PessoaRepository extends CrudRepository<Pessoa, Integer> {
10
11 }
12

```

Dica:

Saber mais sobre CrudRepository basta dar um duplo click para saber mais sobre este recurso.



The screenshot shows the 'CrudRepository.class' file in the IDE. The code for 'CrudRepository' is displayed:

```

25 /**
26 * @NoRepositoryBean
27 * public interface CrudRepository<T, ID> extends Repository<T, ID> {
28
29 * /**
30 * * Saves a given entity. Use the returned instance for further operations as the save operation might have changed the
31 * * entity instance completely.
32 * * @param entity must not be {@literal null}.
33 * * @return the saved entity; will never be {@literal null}.
34 * * @throws IllegalArgumentException in case the given {@literal entity} is {@literal null}.
35 * */
36 * <S extends T> S save(S entity);
37
38 * /**
39 * * Saves all given entities.
40 * * @param entities must not be {@literal null} nor must it contain {@literal null}.
41 * * @return the saved entities; will never be {@literal null}. The returned {@literal Iterable} will have the same size
42 * * as the {@literal Iterable} passed as an argument.
43 * * @throws IllegalArgumentException in case the given {@literal Iterable} or one of its entities is
44 * * {@literal null}.
45 * */
46 * <S extends T> Iterable<S> saveAll(Iterable<S> entities);
47
48 * /**
49 * * Retrieves an entity by its id.
50 * * @param id must not be {@literal null}.
51 * * @return the entity with the given id or {@literal Optional#empty()} if none found.
52 * * @throws IllegalArgumentException if {@literal id} is {@literal null}.
53 * */
54 * Optional<T> findById(ID id);
55
56 * /**
57 * * Returns whether an entity with the given id exists.
58 * * @param id must not be {@literal null}.
59 * * @return {@literal true} if an entity with the given id exists, {@literal false} otherwise.
60 * * @throws IllegalArgumentException if {@literal id} is {@literal null}.
61 * */
62 * boolean existsById(ID id);
63
64 * /**
65 * * Returns all instances of the type.
66 * * @return all entities
67 * */
68 * Iterable<T> findAll();
69
70
71
72
73
74

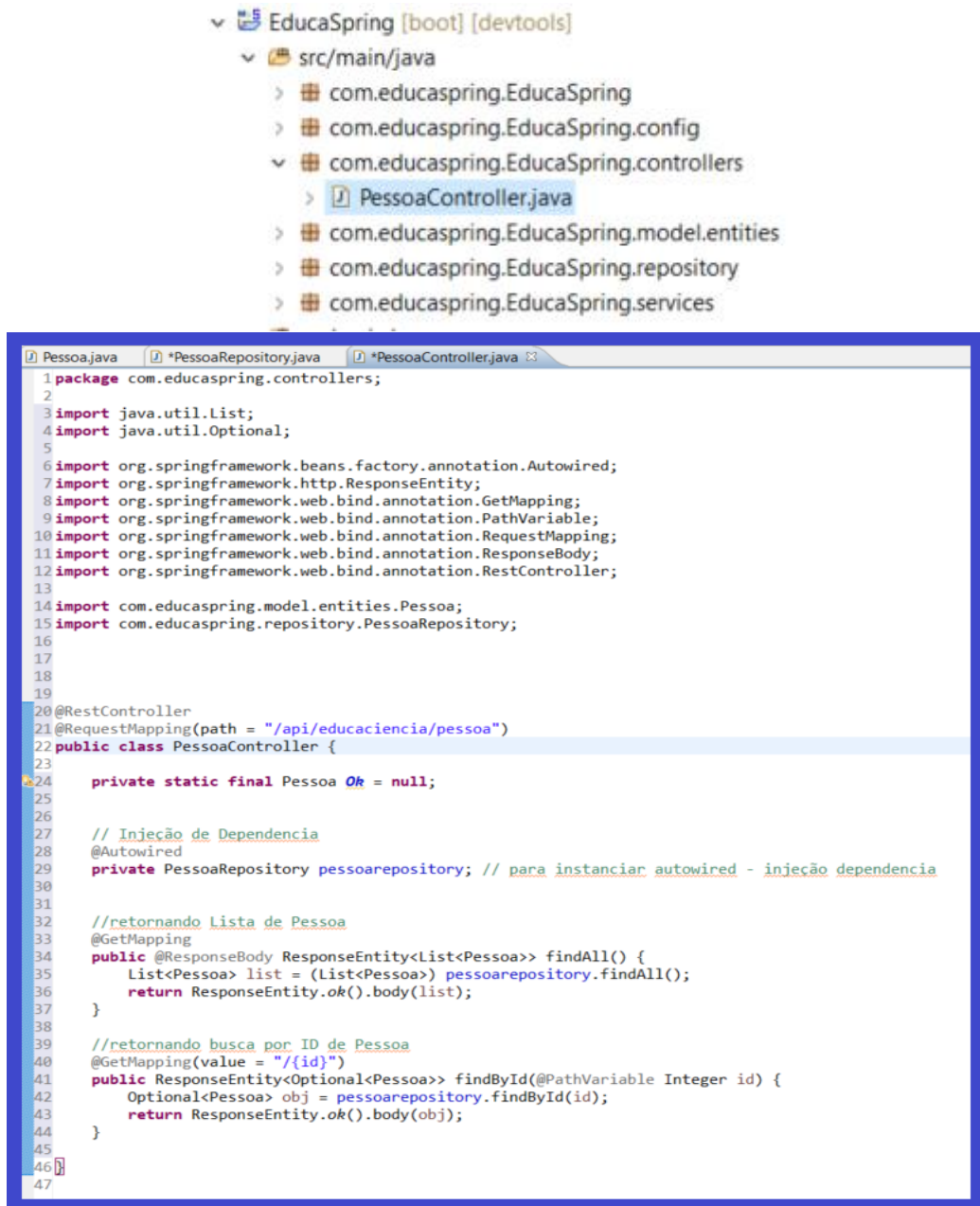
```

Agora que já temos nossa entidade do Banco de Dados e nossa classe pessoa, vamos preparar nosso Controller.

Para isso, iremos criar mais uma classe porém agora no pacote Controller.

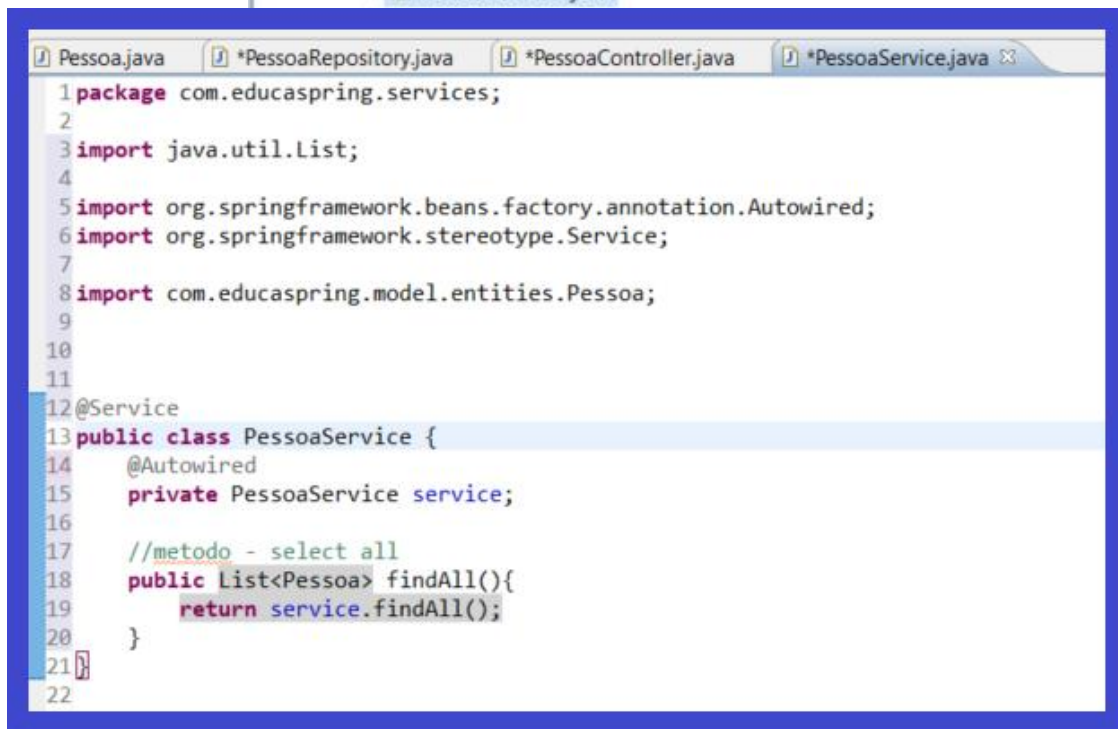
Neste momento iremos criar também nossos Endpoints que serão acessados e manipulados além de nosso método que irá interagir com o Banco de Dados.

Neste artigo, iremos criar o Select (All e for Id) do nosso CRUD onde nos demais artigos, iremos abordar os demais métodos.



```
1 package com.educaspring.controllers;
2
3 import java.util.List;
4 import java.util.Optional;
5
6 import org.springframework.beans.factory.annotation.Autowired;
7 import org.springframework.http.ResponseEntity;
8 import org.springframework.web.bind.annotation.GetMapping;
9 import org.springframework.web.bind.annotation.PathVariable;
10 import org.springframework.web.bind.annotation.RequestMapping;
11 import org.springframework.web.bind.annotation.ResponseBody;
12 import org.springframework.web.bind.annotation.RestController;
13
14 import com.educaspring.model.entities.Pessoa;
15 import com.educaspring.repository.PessoaRepository;
16
17
18
19
20 @RestController
21 @RequestMapping(path = "/api/educaciencia/pessoa")
22 public class PessoaController {
23
24     private static final Pessoa Ok = null;
25
26     // Injeção de Dependência
27     @Autowired
28     private PessoaRepository pessoarepository; // para instanciar autowired - injeção dependência
29
30
31     //retornando Lista de Pessoa
32     @GetMapping
33     public @ResponseBody ResponseEntity<List<Pessoa>> findAll() {
34         List<Pessoa> list = (List<Pessoa>) pessoarepository.findAll();
35         return ResponseEntity.ok().body(list);
36     }
37
38     //retornando busca por ID de Pessoa
39     @GetMapping(value =("/{id}")
40     public ResponseEntity<Optional<Pessoa>> findById(@PathVariable Integer id) {
41         Optional<Pessoa> obj = pessoarepository.findById(id);
42         return ResponseEntity.ok().body(obj);
43     }
44
45
46
47
```


Feito isso, precisamos criar nosso serviço, então vamos criar mais esta classe.



Agora, iremos criar nossa Config para que possamos acessá-lo.

Vamos criar nossa classe **AuthenticationEntryPoint**.

Iremos configurar para que o acesso seja em Basic Auth , então vamos criar nossa classe e configurá-la.

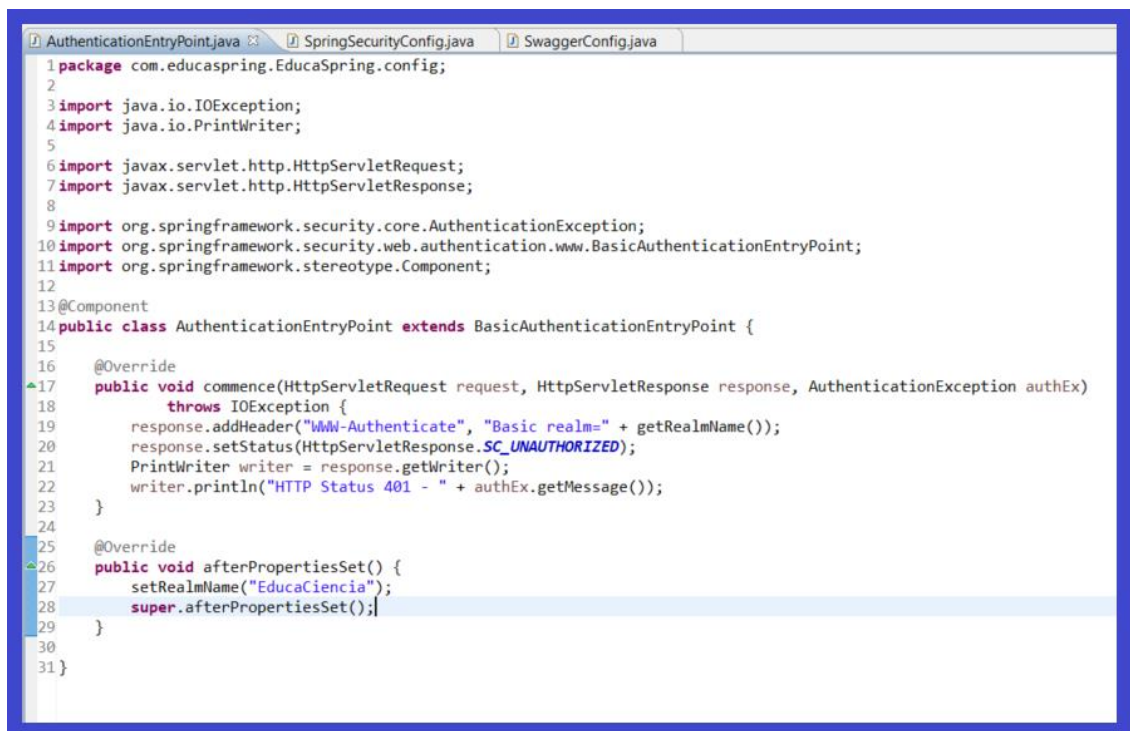
Para sermos um pouco mais performáticos, iremos já criar a documentação Swagger cujo qual é muito requisitada nos dias de hoje nas empresas.



Precisamos importar algumas dependências, vamos lá !

```
<!-- https://mvnrepository.com/artifact/io.springfox/springfox-swagger2 -->
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger2</artifactId>
  <version>2.9.2</version>
</dependency>

<!-- https://mvnrepository.com/artifact/io.springfox/springfox-swagger-ui -->
<dependency>
  <groupId>io.springfox</groupId>
  <artifactId>springfox-swagger-ui</artifactId>
  <version>2.9.2</version>
</dependency>
```



```
1 package com.educaspring.EducaSpring.config;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5
6 import javax.servlet.http.HttpServletRequest;
7 import javax.servlet.http.HttpServletResponse;
8
9 import org.springframework.security.core.AuthenticationException;
10 import org.springframework.security.web.authentication.www.BasicAuthenticationEntryPoint;
11 import org.springframework.stereotype.Component;
12
13 @Component
14 public class AuthenticationEntryPoint extends BasicAuthenticationEntryPoint {
15
16     @Override
17     public void commence(HttpServletRequest request, HttpServletResponse response, AuthenticationException authEx)
18         throws IOException {
19         response.addHeader("WWW-Authenticate", "Basic realm=" + getRealmName());
20         response.setStatus(HttpServletResponse.SC_UNAUTHORIZED);
21         PrintWriter writer = response.getWriter();
22         writer.println("HTTP Status 401 - " + authEx.getMessage());
23     }
24
25     @Override
26     public void afterPropertiesSet() {
27         setRealmName("EducaCiencia");
28         super.afterPropertiesSet();
29     }
30
31 }
```

```

1 package com.educaspring.EducaSpring.config;
2
3
4 import org.springframework.beans.factory.annotation.Autowired;
5 import org.springframework.context.annotation.Configuration;
6 import org.springframework.security.config.annotation.web.builders.HttpSecurity;
7 import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
8 import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
9 import org.springframework.security.web.AuthenticationEntryPoint;
10
11 @Configuration
12 @EnableWebSecurity
13 public class SpringSecurityConfig extends WebSecurityConfigurerAdapter {
14
15     @Autowired
16     private AuthenticationEntryPoint authEntryPoint;
17
18     private static final String[] AUTH_WHITELIST = { "/v1/api-docs-educ", "/swagger-resources/**", "/configuration/ui",
19     "/configuration/security", "/swagger-ui.html**", "/webjars/**", "/actuator/**", "/version/**" };
20
21     @Override
22     protected void configure(HttpSecurity http) throws Exception {
23         http.csrf().disable().authorizeRequests().antMatchers(AUTH_WHITELIST).permitAll()
24         // .anyRequest().authenticated()
25         .and().httpBasic().authenticationEntryPoint(authEntryPoint);
26     }
27
28 }
29

```

```

1 package com.educaspring.EducaSpring.config;
2
3 import java.util.HashSet;
4 import java.util.Set;
5
6 import org.springframework.context.annotation.Bean;
7 import org.springframework.context.annotation.Configuration;
8 import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
9 import org.springframework.web.servlet.config.annotation.WebMvcConfigurationSupport;
10
11 import springfox.documentation.builders.ApiInfoBuilder;
12 import springfox.documentation.builders.RequestHandlerSelectors;
13 import springfox.documentation.service.ApiInfo;
14 import springfox.documentation.spi.DocumentationType;
15 import springfox.documentation.spring.web.plugins.Docket;
16 import springfox.documentation.swagger2.annotations.EnableSwagger2;
17
18 @Configuration
19 @EnableSwagger2
20 public class SwaggerConfig extends WebMvcConfigurationSupport {
21
22     @Bean
23     public Docket api() {
24         return new Docket(DocumentationType.SWAGGER_2).select()
25         .apis(RequestHandlerSelectors.basePackage("com.educaspring.EducaSpring.controllers"))
26         .apiInfo(metaData()).protocols(protocols()).host("localhost:8080");
27     }
28
29     private Set<String> protocols() {
30         Set<String> protocols = new HashSet<>(1);
31         protocols.add("http");
32         return protocols;
33     }
34
35     private ApiInfo metaData() {
36         return new ApiInfoBuilder().title("EducaCiência API")
37         .description("Applications responsible to configure devices and expose current setup configuration.")
38         .version("v1")
39         .license("Copyrights 2020 - EducaCiência FastCode - Todos os Direitos Reservados")
40         .licenseUrl("https://www.evoluasumare.com.br").build();
41     }
42
43     @Override
44     protected void addResourceHandlers(ResourceHandlerRegistry registry) {
45         registry.addResourceHandler("/swagger-ui.html").addResourceLocations("classpath:/META-INF/resources/");
46         registry.addResourceHandler("/webjars/**").addResourceLocations("classpath:/META-INF/resources/webjars/");
47     }
48
49 }

```



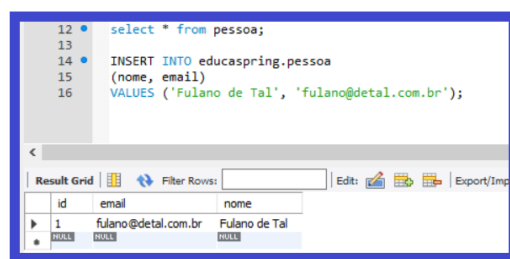
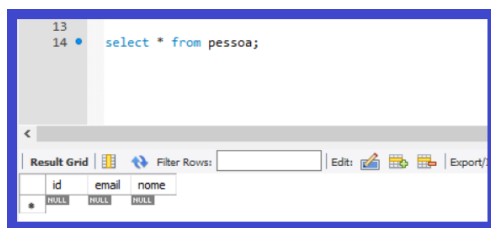
Vamos salvar nosso Projeto para que as alterações entrem em vigor.

```
Using generated security password: 9b117a39-90e3-459c-ad40-7e38e6a4c34d
2020-10-27 14:25:59.091 INFO 11568 --- [ restartedMain] o.s.s.web.DefaultSecurityFilterChain : Creating filter chain: any request, [org.springframework.security.web.context.request.async.WebAsyncManager
2020-10-27 14:25:59.110 INFO 11568 --- [ restartedMain] o.s.b.d.a.OptionalLiveReloadServer : LiveReload server is running on port 35729
2020-10-27 14:25:59.127 INFO 11568 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2020-10-27 14:25:59.139 INFO 11568 --- [ restartedMain] c.e.EducaSpring.EducaSpringApplication : Started EducaSpringApplication in 0.652 seconds (JVM running for 2717.945)
2020-10-27 14:25:59.143 INFO 11568 --- [ restartedMain] .ConditionEvaluationDeltaLoggingListener : Condition evaluation unchanged
```

Pronto, feito isso, basta salvar para que nosso Projeto seja atualizado.

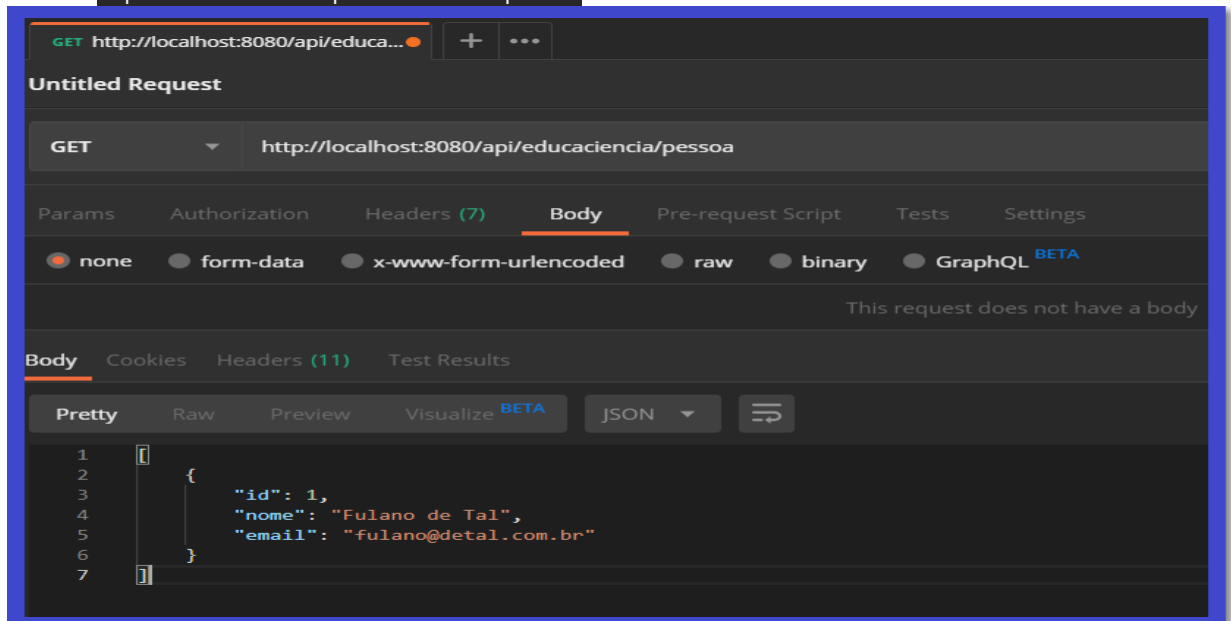
Para que possamos testá-lo, necessitaremos fazer alguns Insert manual no nosso Banco de Dados pois ainda não temos criado o método Inserir, este será o tema do nosso próximo artigo, porem , para este artigo vamos criar manualmente os dados.

Não precisamos criar nossa tabela, ela foi criada pelo Spring Boot na persistência dos dados.

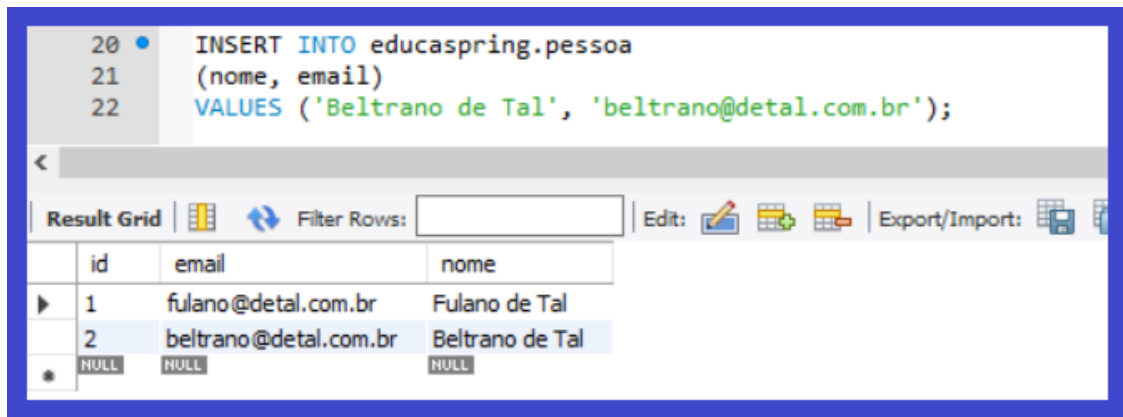


Feito este processo, e como temos nosso Swagger já configurado também, podemos abrir nosso Postman e executarmos um GET para testa-lo.

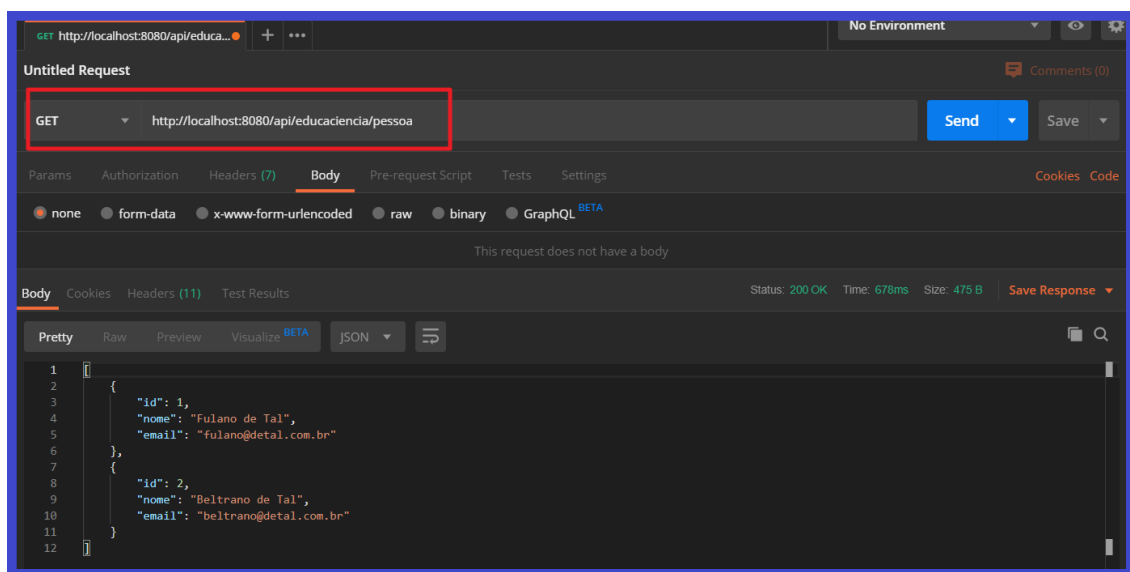
GET => <http://localhost:8080/api/educaciencia/pessoa>



Vamos inserir mais um registro no nosso Banco de Dados como mostramos a seguir:

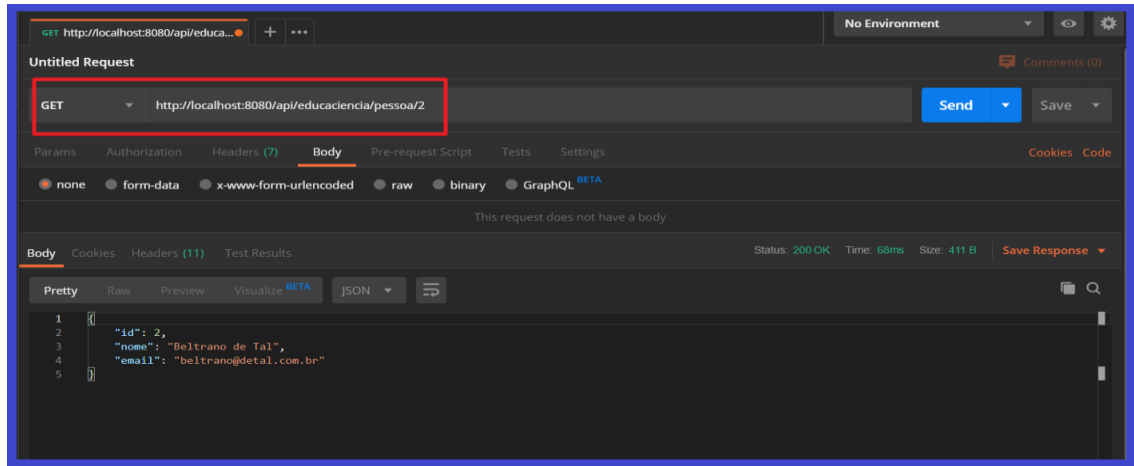


Vamos novamente testar nosso retorno no Postman

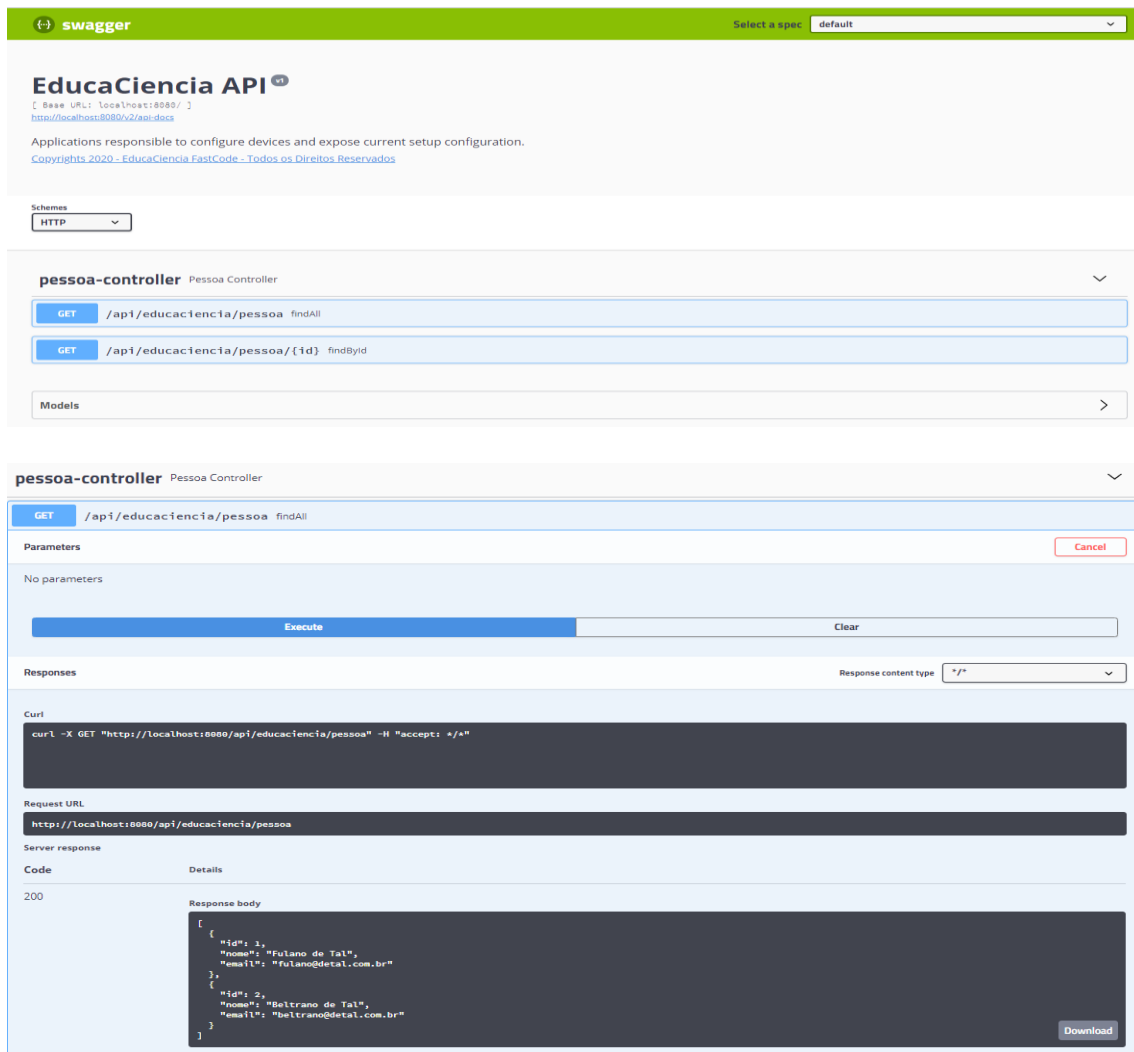




Para validar, iremos testar nosso retorno passando como parâmetro o ID = 2



Nota-se que os códigos funcionaram corretamente, vamos testar nossa documentação Swagger





GET /api/educaciencia/pessoa/{id} findById

Parameters Cancel

| Name | Description |
|--|--------------------------------------|
| id required integer (5int32) (path) | id <input type="text" value="2"/> |

Execute Clear

Responses Response content type */*

Curl

```
curl -X GET "http://localhost:8080/api/educaciencia/pessoa/2" -H "accept: */*"
```

Request URL

```
http://localhost:8080/api/educaciencia/pessoa/2
```

Server response

| Code | Details |
|------|--|
| 200 | <p>Response body</p> <pre>{ "id": 2, "nome": "Beltrano de Tal", "email": "beltrano@detal.com.br" }</pre> Download |

Vimos no entanto que nosso código funcionou como esperado, saliento que o artigo 31/2020 é o início da nossa sequência de Spring Boot, fiquem ligados nos sequenciais onde os dados são baseados no curso de Java que ministro na Escola Evolua – Ensino Profissionalizante e como proposito de ajuda à comunidade, estamos trazendo parte da didática em forma de artigo comunitário e assim podemos contribuir com a comunidade Tecnológica como um todo.

Agradeço imensamente a Diretoria da Escola Evolua de Sumaré.

Os códigos estarão disponíveis no Git.

Até mais !

Espero ter ajudado !

