Desenvolvimento de Software para WEB

Aula 20 - Spring Data

Professor: Anderson Almada

pom.xml

```
<dependency>
	<groupId>org.springframework.boot</groupId>
	<artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<dependency>
	<groupId>org.postgresql</groupId>
	<artifactId>postgresql</artifactId>
</dependency>
```

application.properties

```
## PostgreSQL
spring.datasource.url=jdbc:postgresql://localhost:5432/projeto
spring.datasource.username=postgres
spring.datasource.password=postgres
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.temp.use_jdbc_metadata_defaults=false
```

uploads.properties

folder=/var/www/vue/vue-spring/uploads/

Classe auxiliar -> Upload

- Package: util
    - https://pastebin.com/9d8i0PuT

Classe model -> Curso

- Package: model
  - @Entity
  - @Id
  - @GeneratedValue
  - @Column(columnDefinition="text")

    - https://pastebin.com/a9WtRRcf

Classe repository -> CursoRepository

- Package: repository

```
public interface CursoRepository extends JpaRepository<Curso, Integer> {

}
```

(Opcional) CursoRepository

- Package: repository

```
public interface CursoRepository extends JpaRepository<Curso, Integer>{
	@Query("from Curso where nome = ?1")
	Curso findByNome(String nome);
}
```

Classe service - CursoService

- Package: controller

```
@RestController
@RequestMapping(path = "/api/cursos")
@CrossOrigin
public class CursoService {

    @Autowired
    private CursoRepository cursos;
```

GET ALL

```
@RequestMapping(method = RequestMethod.GET)
public ResponseEntity<List<Curso>> getCursos() {
        return new ResponseEntity<List<Curso>>(cursos.findAll(),HttpStatus.OK);
}
```

(Opcional) GET ALL - ORDER BY ID ASC

```
@RequestMapping(method = RequestMethod.GET)
public ResponseEntity<List<Curso>> getCursos() {
        return new ResponseEntity<List<Curso>>(
                cursos.findAll(new Sort(Sort.Direction.ASC, "id")), HttpStatus.OK);
}
```

GET ID

```
@RequestMapping(value = "{id}", method = RequestMethod.GET)
public ResponseEntity<Curso> getCurso(@PathVariable("id") Integer id) {
	Optional<Curso> curso = cursos.findById(id);
	if (curso.isPresent()) {
		return new ResponseEntity<Curso>(curso.get(), HttpStatus.OK);
	} else {
		return new ResponseEntity<>(HttpStatus.NOT_FOUND);
	}
}
```

(Opcional) GET CURSO BY NOME

```
@RequestMapping(value = "/search", method = RequestMethod.GET)
public ResponseEntity<Curso> getCurso(@RequestParam("nome") String nome) {
	Curso curso = cursos.findByNome(nome);
	if (curso != null) {
		return new ResponseEntity<Curso>(curso, HttpStatus.OK);
	} else {
		return new ResponseEntity<>(HttpStatus.NOT_FOUND);
	}
}
```

POST

```
@RequestMapping(method = RequestMethod.POST)
public ResponseEntity<Curso> addCurso(String nome, String duracao, MultipartFile image) {
	if (nome == null || duracao == null || nome.equals("null") || duracao.equals("null") || image
== null) {
		return new ResponseEntity<>(HttpStatus.NOT_FOUND);
	}
	Curso curso = new Curso(null, nome, duracao);
	Curso cursoAux = cursos.save(curso);
	try {
		Upload.uploadFile(image.getInputStream(), cursoAux.getId());
	} catch (IOException e) {  }
	return new ResponseEntity<Curso>(curso, HttpStatus.OK);
}
```

PUT

```
@RequestMapping(value = "{id}", method = RequestMethod.PUT)
public ResponseEntity<Curso> atualizarCurso(@PathVariable("id") int id, String nome, String
duracao, MultipartFile image) {
	if (nome == null || duracao == null || nome.equals("null") || duracao.equals("null")) {
		return new ResponseEntity<>(HttpStatus.NOT_FOUND);
	}
	Optional<Curso> curso = cursos.findById(id);

	if (curso.isPresent()) {
		curso.get().setNome(nome);
		curso.get().setDuracao(duracao);
		cursos.save(curso.get());
```

PUT

```
        try {
                if (image != null) {
                        Upload.uploadFile(image.getInputStream(), id);
                }
        } catch (IOException e) { }

        return new ResponseEntity<Curso>(curso.get(), HttpStatus.OK);
        } else {
                return new ResponseEntity<>(HttpStatus.NOT_FOUND);
        }
}
```

DELETE

```
@RequestMapping(value = "{id}", method = RequestMethod.DELETE)
public ResponseEntity<?> deletarCurso(@PathVariable("id") int id) {
	if(cursos.existsById(id)) {
		cursos.deleteById(id);
		return new ResponseEntity<>(HttpStatus.NO_CONTENT);
	} else {
		return new ResponseEntity<>(HttpStatus.NOT_FOUND);
		}
}
```

CursoService

- https://pastebin.com/d9gUxfFY

Links importantes

- https://grokonez.com/spring-framework/spring-data/spring-boot-vue-example-spring-data-jpa-rest-postgresql-crud-example
- https://www.mkyong.com/spring-boot/spring-boot-spring-data-jpa-postgresql/
- https://docs.spring.io/spring-data/jpa/docs/current/reference/html/
- https://blog.algaworks.com/spring-data-jpa/

Dúvidas??

E-mail: almada@crateus.ufc.br