Curso

Aplicações JAVA com SPRING BOOT

Prof. Msc. Antonio B. C. Sampaio Jr
ENGENHEIRO DE SOFTWARE & PROFESSOR

@abctreinamentos
@amazoncodebr

www.abctreinamentos.com.br
www.amazoncode.com.br

CONTEÚDO PROGRAMÁTICO

- UNIDADE 1 – INTRODUÇÃO
- UNIDADE 2 – FUNDAMENTOS DO SPRING BOOT
- UNIDADE 3 – PERSISTÊNCIA DE DADOS NO SPRING BOOT
- UNIDADE 4 – PROJETO WEB NO SPRING BOOT
- UNIDADE 5 – PROJETO REST API NO SPRING BOOT
- UNIDADE 6 – PROJETO REST API NO SPRING BOOT COM REACTJS

CONTEÚDO PROGRAMÁTICO

- UNIDADE 7 – PROJETO REST API NO SPRING BOOT COM THYMELEAF
- UNIDADE 8 – PROJETO REST API NO SPRING BOOT COM MICROSSERVIÇOS
- UNIDADE 9 – PROJETO FINAL
- EXTRAS

PROJETOS DO CURSO

- 1º Projeto Spring Boot – Impressão de Mensagens
- 2° Projeto Spring Boot – Impressão de Mensagens na WEB
- 3° Projeto Spring Boot – Aplicação Servidor Público
- 4° Projeto Spring Boot – Aplicação Servidor Público na WEB
- 5° Projeto Spring Boot – Aplicação Servidor Público no SGBD MYSQL
- 6° Projeto Spring Boot – Aplicação Servidor Público no MONGO DB

PROJETOS DO CURSO

- 7° Projeto Spring Boot – Aplicação Servidor Público WEB
- 8° Projeto Spring Boot – Aplicação Servidor Público REST API e MySQL
- 9° Projeto Spring Boot – Aplicação Servidor Público REST API com REACT
- 10° Projeto Spring Boot – Aplicação Servidor Público/Curso REST API – Monolítico
- 11° Projeto Spring Boot – Aplicação Servidor Público REST API - Microsserviços

PROJETOS
DO CURSO

- 12° Projeto Final Spring Boot – SISCAPACIT (Sistema de Capacitação de Servidores Públicos)

UNIDADE 9

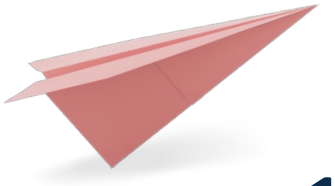
PROJETO FINAL

PROJETO FINAL

12° Projeto Final Spring Boot – SISCAPACIT
(Sistema de Capacitação de Servidores Públicos)

CONCLUSÃO

PROJETO FINAL



12° Projeto Final Spring Boot – SISCAPACIT (Sistema de Capacitação de Servidores Públicos)

SISCAPACIT

- **Mudanças Necessárias**
  - Alterar a camada de Dados:
    - Criar um novo schema (siscapacit_3) com uma nova modelagem.
    - Copiar o projeto 10 e fazer as alterações a seguir:

SISCAPACIT

- **Mudanças Necessárias**
  - (1) Alterar a camada Service: ServidorPublicoService e ServidorPublicoServiceImpl

```
@Service
public class ServidorPublicoServiceImpl implements ServidorPublicoService {
     ...
     @Override
     public void matricular(ServidorPublico servidor) {
          servidorRepository.save(servidor);
     }

     @Override
     public void desmatricular(ServidorPublico servidor) {
          servidorRepository.save(servidor);
     }
}
```

SISCAPACIT

- **Mudanças Necessárias**
  - (2) Alterar a Entidades: Curso.

```
@Entity
@Table(name="curso")
public class Curso implements Serializable
{
     ...
     @ManyToMany(mappedBy="cursos",fetch=FetchType.LAZY)
     @JsonIgnore
     private List<ServidorPublico> servidores;
     ...
}
```

SISCAPACIT

- **Mudanças Necessárias**
  - (3) Alterar a Entidade ServidorPublico.

```
@Entity
@Table(name=”servidorpublico")
public class ServidorPublico implements Serializable
{
     ...
     @ManyToMany(fetch = FetchType.EAGER)
     @JoinTable(name = "servidorpublicocurso",joinColumns = @JoinColumn(name = "matricula"),
     inverseJoinColumns = @JoinColumn(name = "idcurso"))
     private List<Curso> cursos;
     ...
}
```

SISCAPACIT

- **Mudanças Necessárias**
  - (4) Incluir a classe Error na camada Controller.
  - (5) Alterar a classe AppController para implementar a API Rest ServidorCurso.

```
@PostMapping("/matricularServidorCurso/{idcurso}")
public String matricular(@RequestBody ServidorPublico servidor,
                         @PathVariable("idcurso") long idcurso)
{
     Curso curso = cursoService.listByidCurso(idcurso).get();
     // Adicione o servidorPublicoCurso à lista cursos em ServidorPublico
     servidor.getCursos().add(curso);
     servidorService.matricular(servidor);
     throw new ResponseStatusException(HttpStatus.OK, "Servidor Matriculado no Curso");
}
```

SISCAPACIT

- **Mudanças Necessárias**

```
@PostMapping("/desmatricularServidorCurso/{idcurso}")
public String desmatricular(@RequestBody ServidorPublico servidor,
               @PathVariable long idcurso)
{
     Curso curso = cursoService.listByidCurso(idcurso).get();
     servidor.getCursos().remove(curso);
     servidorService.desmatricular(servidor);
     throw new ResponseStatusException(HttpStatus.OK, "Servidor Desmatriculado do Curso");
}
```

SISCAPACIT

- **Mudanças Necessárias**

```
@GetMapping("/listarCursosServidor/{matricula}")
public ResponseEntity<CursosServidor> listarCursosServidor(@PathVariable Long matricula) {
     Optional<ServidorPublico> servidorEncontrado =
                servidorService.listByMatricula(matricula);
     List<Curso> todosCursos = cursoService.listAll();
     List<Curso> cursosListados = new ArrayList<>();
     List<Curso> cursosNaoListados = new ArrayList<>();

     for (Curso curso : todosCursos) {
          List<ServidorPublico> servidores = curso.getServidores();
               if (servidores != null && servidores.stream().anyMatch
               (servidor -> servidor.getMatricula().equals(matricula))) {
                    cursosListados.add(curso);
               } else {
                    cursosNaoListados.add(curso);
               }
     }
     ...
```

SISCAPACIT

- **Mudanças Necessárias**

```
...
    // Retornar as duas listas no corpo da resposta
    Map<String, List<Curso>> cursosMap = new HashMap<>();
    cursosMap.put("Cursos Matriculados", cursosListados);
    cursosMap.put("Cursos Não Matriculados", cursosNaoListados);
    CursosServidor cursosservidor = new CursosServidor(cursosMap,
                                        servidorEncontrado.get());
    return new ResponseEntity<>(cursosservidor, HttpStatus.OK);
}
```

SISCAPACIT

- **Mudanças Necessárias**

```
@GetMapping("/listarServidoresCurso/{idcurso}")
public ResponseEntity<ServidoresCurso> listarServidoresCurso(@PathVariable Long idcurso) {
     Curso cursoEncontrado = cursoService.listByidCurso(idcurso).get();

     List<ServidorPublico> servidores = cursoEncontrado.getServidores();
     Map<String, List<ServidorPublico>> servidoresMap = new HashMap<>();
     servidoresMap.put("Servidores Públicos Matriculados", servidores);
     ServidoresCurso servidorescurso = new ServidoresCurso(servidoresMap, cursoEncontrado);
     return new ResponseEntity<>(servidorescurso, HttpStatus.OK);
}
```

SISCAPACIT

- **Mudanças Necessárias**
  - (6) Alterar a classe AppClient.

```
@GetMapping("/listagemCursosServidor/{matricula}")
public String getCursosServidores(@PathVariable("matricula") long matricula,
                                                    Model model) {
     String url = http://localhost:8080/listarCursosServidor/{matricula};
     ResponseEntity<CursosServidor> response = restTemplate.exchange(
     url, HttpMethod.GET, null,
     new ParameterizedTypeReference<CursosServidor>() {},matricula);
     CursosServidor cursosservidor = response.getBody();

     model.addAttribute("cursosservidor", cursosservidor);

     // Retorne a view Thymeleaf para renderização
     return "servidorpublico/cursosservidorpublico";
}
```

SISCAPACIT

- **Mudanças Necessárias**
  - (7) Alterar a classe AppClient.

```
@GetMapping("/matriculaServidorCurso/{matricula}/{idcurso}")
public String matricular(@PathVariable("matricula") long matricula,
                @PathVariable("idcurso") long idcurso, Model model)
{
     HttpHeaders headers = new HttpHeaders();
     headers.setContentType(MediaType.APPLICATION_JSON);
     String url = http://localhost:8080/listarServidor/{matricula};
     ResponseEntity<ServidorPublico> response = restTemplate.exchange(
     url, HttpMethod.GET, null,
     new ParameterizedTypeReference<ServidorPublico>() {},matricula);
     ServidorPublico servidorEncontrado = response.getBody();

     HttpEntity<ServidorPublico> requestEntity =
          new HttpEntity<>(servidorEncontrado, headers);
     ...
```

SISCAPACIT

- **Mudanças Necessárias**
  - (7) Alterar a classe AppClient.

```
...
String url2 = "http://localhost:8080/matricularServidorCurso/{idcurso}";
ResponseEntity<ServidorPublico> newresponse = restTemplate.exchange(
url2,HttpMethod.POST,requestEntity,
ServidorPublico.class,idcurso);

if (newresponse.getStatusCode() == HttpStatus.OK)
     return "redirect:/listagemCursosServidor/{matricula}";
else
 {
     model.addAttribute("mensagem", response.getStatusCode());
     return "erro/mensagem";
 }
}
```

SISCAPACIT

- **Mudanças Necessárias**
  - (8) Alterar a classe AppClient.

```
@GetMapping("/desmatriculaServidorCurso/{matricula}/{idcurso}")
public String desmatricular(@PathVariable("matricula") long matricula,
                    @PathVariable("idcurso") long idcurso, Model model)
{
     HttpHeaders headers = new HttpHeaders();
     headers.setContentType(MediaType.APPLICATION_JSON);
     String url = http://localhost:8080/listarServidor/{matricula};
     ResponseEntity<ServidorPublico> response = restTemplate.exchange(
     url, HttpMethod.GET, null,
     new ParameterizedTypeReference<ServidorPublico>() {},matricula);
     ServidorPublico servidorEncontrado = response.getBody();

     HttpEntity<ServidorPublico> requestEntity =
     new HttpEntity<>(servidorEncontrado, headers);
     ...
```

SISCAPACIT

- **Mudanças Necessárias**
  - (8) Alterar a classe AppClient.

```
...
String url2 = http://localhost:8080/desmatricularServidorCurso/{idcurso};
ResponseEntity<ServidorPublico> newresponse = restTemplate.exchange(
url,HttpMethod.POST,requestEntity,ServidorPublico.class,idcurso);

if (newresponse.getStatusCode() == HttpStatus.OK)
     return "redirect:/listagemCursosServidor/{matricula}";
else
 {
     model.addAttribute("mensagem", response.getStatusCode());
     return "erro/mensagem";
 }
}
```

SISCAPACIT

- **Mudanças Necessárias**
  - (9) Alterar a classe AppClient.
    - `@GetMapping("/listaServidor/{matricula}")` para `@GetMapping("/listagemServidor/{matricula}")`
    - `@GetMapping("/listaCurso/{idcurso}")` para `@GetMapping("/listagemCurso/{idcurso}")`

SISCAPACIT

- **Mudanças Necessárias**
  - (10) Alterar a classe AppClient.

```
@GetMapping("/listagemServidoresCurso/{idcurso}")
public String getServidoresCurso(@PathVariable("idcurso") long idcurso,
                                 Model model) {
     String url = http://localhost:8080/listarServidoresCurso/{idcurso};
     ResponseEntity<ServidoresCurso> response = restTemplate.exchange(
     url, HttpMethod.GET, null,
     new ParameterizedTypeReference<ServidoresCurso>() {},idcurso);
     ServidoresCurso servidorescurso = response.getBody();

     model.addAttribute("servidorescurso", servidorescurso);

     // Retorne a view Thymeleaf para renderização
     return "curso/servidorespublicoscurso";
}
```

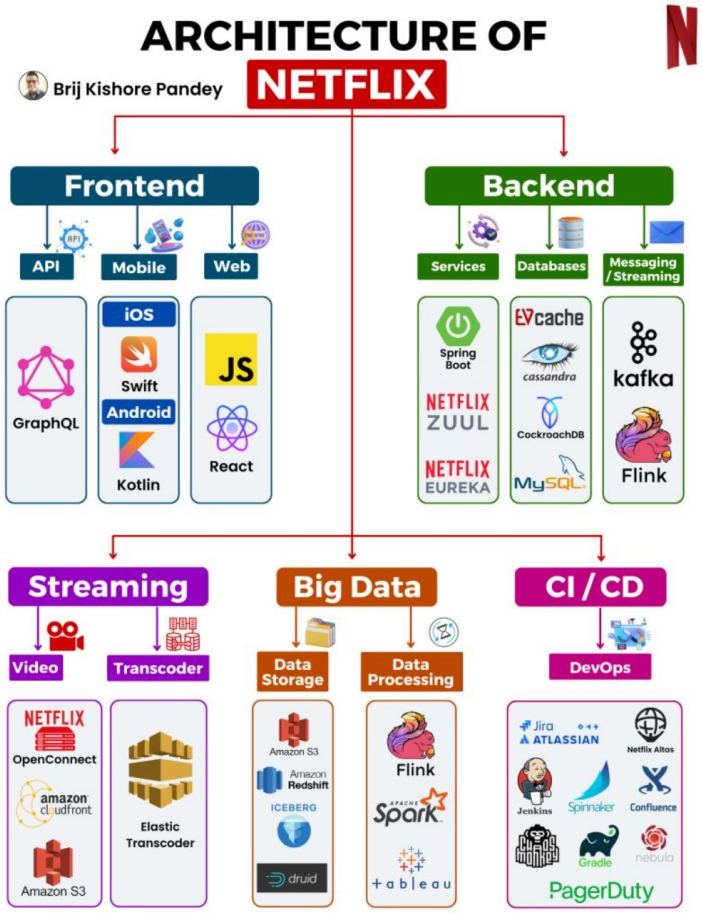
SISCAPACIT

- **Mudanças Necessárias**
  - (11) Criar um novo pacote aux com 02 records: CursosServidor e ServidoresCurso.

```
public record ServidoresCurso(Map<String, List<ServidorPublico>> servidores,
                              Curso curso) {

}

public record CursosServidor(Map<String, List<Curso>> cursos,
                              ServidorPublico servidor) {

}
```

CONCLUSÃO

TECNOLOGIAS NETFLIX



AGRADECIMENTO!

Obrigado pela confiança e por ter chegado até aqui!

Saiba que foram centenas de horas para preparar todo o material que você recebeu em formato PDF e MP4.

Tenho certeza que todo o conteúdo apresentado fará uma enorme diferença na sua Carreira Profissional!

Desejo Sucesso em Sua Caminhada!

Para saber mais dos nossos outros treinamentos e projetos, acesse os meus sites:

www.abctreinamentos.com.br
www.amazoncode.com.br

CEL/ZAP: (91) 98216-1883