Universidade Federal do ABC (UFABC)
Campus Santo André
Av. dos Estados, 5001 – Bairro Bangu - CEP 09210-170
Santo André /SP - Brasil

Programação para Web
Q2/2018

Atividade 05: HTML, JSP e JSTL

1. Configurar a visão no projeto

- **Criar** a pasta /src/main/webapp/
  - Esta pasta deve ser no tipo "source folder"
- Dentro da pasta criada, **criar uma outra** pasta chamada WEB-INF
  - Esta pasta deve ser do tipo "folder"
- **Acrescentar** as seguintes linhas no arquivo application.properties, localizado na pasta /src/main/resources

```
spring.mvc.view.prefix: /WEB-INF/
spring.mvc.view.suffix: .jsp
```

- **Verificar** as dependências no arquivo build.gradle, conforme o código as instruções a seguir:

```
compile('org.springframework.boot:spring-boot-starter-data-jpa')
runtime('org.postgresql:postgresql')
compile('org.springframework.boot:spring-boot-starter-web')
compile('org.springframework.boot:spring-boot-devtools')
compile('javax.servlet:jstl')
runtime('org.springframework.boot:spring-boot-starter-tomcat')
compile('org.apache.tomcat.embed:tomcat-embed-jasper')
testCompile('org.springframework.boot:spring-boot-starter-test')
```

- **Executar** o comando "gradle eclipse" no terminal

2. Criar uma view no projeto

- Em seguida, dentro da pasta WEB-INF, **criar** um arquivo index.jsp com o seguinte código:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
      pageEncoding="ISO-8859-1"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="form"
uri="http://www.springframework.org/tags/form"%>

<!DOCTYPE html>
```

```
<html lang="pt-BR">
<head>
<meta charset="UTF-8" />
<title>Lista de Contas Corrente</title>
</head>
<body>

	<table border="1">

		<thead>
			<tr>
				<th><b>Número</b></th>
				<th><b>Agencia</b></th>
				<th><b>Descricao</b></th>
				<th><b>Ação</b></th>
			</tr>
		</thead>
		<tbody>
			<c:forEach items="${ccs}" var="cc">
				<tr>
					<td><c:out value="$
{cc.numero}"></c:out></td>
					<td><c:out value="$
{cc.agencia}"></c:out></td>
					<td><c:out value="$
{cc.descricao}"></c:out></td>

					<td><a href="/edit/${cc.id}">
							<button
type="submit">Editar</button>
					</a></td>
				</tr>
			</c:forEach>
		</tbody>
	</table>
</body>
</html>
```

- Na classe ContaCorrenteController, **criar** um método com o seguinte código:

```
@RequestMapping(value = { "/", "/index" })
@ResponseBody
public ModelAndView index() {
	ModelAndView mv = new ModelAndView("index");
	mv.addObject("ccs", ccDao.findAll());
	return mv;
}
```

- Executar o comando “gradle bootRun” no terminal e verificar a página no browser digitando “localhost:8080/”

3. Criar outras views

- **Criar** uma view que permita o cadastro de uma nova conta corrente, implementar a view e o controller.
- **Criar** uma view para editar uma conta corrente existente, implementar a view e o controller.

- **Implementar** a exclusão de contas corrente, implementar a view e o controller.

4. Desafio

- Assumindo que uma conta corrente pertence a um banco, realizar as seguintes instruções:
  - **Criar** uma entidade banco, com os seguintes atributos: nome, código, descrição e endereço.
  - **Criar** uma view para listar os bancos cadastrados e outra view para cadastrar um banco. **Implementar** a exclusão e alteração de banco também.
  - Em seguida, **associar** o banco a uma conta corrente por meio da associação @ManyToOne na conta corrente, ou seja, uma conta corrente tem apenas um banco e uma banco tem várias contas.
  - **Incluir** um elemento combobox na view de conta corrente para listas os bancos cadastrados.
  - Em seguida, **alterar** o cadastro de uma nova conta corrente para permitir o vínculo de um banco selecionado. **Possibilitar** também a edição de contas corrente cadastradas.