Perguntas Prova 1 – LIP1 – Spring Boot, Spring MVC

1 – Qual é o principal framework utilizado para desenvolvimento de aplicações web em Java?

a) Apache Struts
b) Click
c) JSF
d) Spring MVC
e) Play

2 – Framework Java que permite acelerar e facilitar a configuração inicial de um novo projeto fazendo a configuração do projeto para o desenvolvedor utilizando configurações padrão comumente utilizadas no mercado. Esse framework incorpora o próprio servidor web ficando responsável por disparar e interromper sua execução quando necessário.

a) Apache Tomcat
b) Controller
c) Spring Boot
d) Spring Data
e) Thymeleaf

3 – Componente que permite produzir páginas html mesclando arquivos de páginas html semiacabadas com dados provenientes de outras fontes.

a) Apache Tomcat
b) Controller
c) Spring Boot
d) Spring Data
e) Thymeleaf

4 – Quais anotações devemos utilizar antes da definição de uma classe para que a classe passe a responder requisições HTTP?

a) @Controller e @ControllerRest
b) @ControllerHttp e @RestControllerHttp
c) @HttpController e @ControllerRest
d) @Controller e @RestController
e) @WebController e @Controller

5 – Considerando que nosso servidor está rodando na máquina local e que nossa aplicação – construída utilizando os frameworks Spring Boot e Spring MVC – esteja escutando a porta 8080, qual anotação devemos utilizar para indicar que um método deverá responder uma requisição HTTP do tipo GET para a url “http://localhost:8080/”?

a) @GetMapping(“/list”)
b) @GetMapping(“/produto”)
c) @PostMapping(“/produto/novo”)
d) @RequestMapping(value = "/produto", method = RequestMethod.GET)
e) @GetMapping(“/”)

6 – Considerando que nosso servidor está rodando na máquina local e que nossa aplicação – construída utilizando os frameworks Spring Boot e Spring MVC – esteja escutando a porta 8080, qual anotação devemos utilizar para indicar que um método deverá responder uma requisição HTTP do tipo POST para a url “http://localhost:8080/produto/novo”?

a) @GetMapping(“/list”)
b) @GetMapping(“/produto”)
c) @PostMapping(“/produto/novo”)
d) @RequestMapping(value = "/produto", method = RequestMethod.GET)
e) @GetMapping(“/”)

7 – Considerando que nosso servidor está rodando na máquina local e que nossa aplicação – construída utilizando os frameworks Spring Boot e Spring MVC – esteja escutando a porta 8080, qual anotação devemos utilizar para indicar que um método deverá responder uma requisição HTTP do tipo GET para a url “http://localhost:8080/produto/list”?

a) @GetMapping(“/list”)
b) @GetMapping(“/produto”)
c) @PostMapping(“/produto/novo”)
d) @RequestMapping(value = "/produto/list", method = RequestMethod.GET)
e) @GetMapping(“/”)

8 – Considerando que nosso servidor está rodando na máquina local e que nossa aplicação – construída utilizando os frameworks Spring Boot e Spring MVC – esteja escutando a porta 8080, o que devemos incluir no método abaixo para que uma requisição HTTP do tipo GET para a url “http://localhost:8080/produto/3” seja processada por este método e que o número “3” do final da url seja carregado no parâmetro “id” do método?

```
public String detalhe(int id) {
	return "detalhe-produto";
}
```

a) É preciso anotar o método detalhe com a anotação @GetMapping(“/produto/id”)
b) É preciso anotar o método detalhe com a anotação @GetMapping(“/produto/id”) e o parâmetro id com a anotação @RequestParam(“{id}”)
c) É preciso anotar o método detalhe com a anotação @GetMapping(“/produto/{id}”) e o parâmetro id com a anotação @Param(“id”)
d) É preciso anotar o parâmetro id com a anotação @RequestParam(“{id}”)
e) É preciso anotar o método detalhe com a anotação @GetMapping(“/produto/{id}”) e o parâmetro id com a anotação @PathVariable(“id”)

9 – Qual arquivo devemos alterar e qual alteração devemos fazer para que nossa aplicação, em vez de escutar a porta 8080 que é o padrão adotado pelo Spring Boot no momento de configurar o Spring MVC, passe a escutar a porta 8081?

a) Devemos alterar o arquivo pom.xml adicionando o trecho

```
<server>
	<port>8081</port>
</server>
```

b) Devemos alterar o arquivo master-template.html adicionando

```
<server>
	<port>8081</port>
</server>
```

c) Devemos alterar o arquivo application.properties adicionando

```
<server>
	<port>8081</port>
</server>
```

d) Devemos alterar o arquivo pom.xml adicionando

```
server.port=8081
```

e) Devemos alterar o arquivo application.properties adicionando

```
server.port=8081
```

10 – Considerando que nossa aplicação possui o Thymeleaf adequadamente configurado qual arquivo de template será utilizado para gerar a página de resposta quando o método abaixo for chamado para processar uma requisição HTTP?

```
...
public String detalhe(int id) {
	return "detalhe-produto";
}
...
```

a) detalhe-produto.html
b) detalhe-produto
c) detalhe.html
d) detalhe-produto.xml
e) detalhe-produto.txt

11 – O método a seguir deverá processar requisições do tipo GET para a url's como "http://localhost:8080/produto/3", e está presente em uma classe que foi marcada para desempenhar o papel de controlador e cujo método detalhe deverá apresentar uma página para o usuário com alguns dados contidos nela.

```
@GetMapping("/produto/{id}")
public String detalhe(
        @PathVariable("id") int id,
        Model model) {

    Produto p = buscarProdutoPeloId(id);

    model.addAttribute("produto", p);

    return "detalhe-produto";
}
```

A classe Model, recebida como parâmetro do método detalhe, permite que o método detalhe informe parâmetros, isto é, valores associados a nomes, que poderão ser utilizados no processo de geração da página apresentada ao usuário mediante o processo de mesclagem desses parâmetros com os templates.

Qual outra classe nos permite definir parâmetros para serem mesclados com os templates?

a) ModelAndView
b) View
c) Properties
d) String
e) HashMap

12 – Considere que o texto a seguir é o conteúdo de um arquivo de um arquivo de template

```
<!DOCTYPE html>
<html>
      <head>
            <title>Um template qualquer</title>
            <meta charset="utf-8">
      </head>
      <body>
            <span></span>

      </body>
</html>
```

Considere também o método **mostrarMensagem** abaixo

```
...
public String mostrarMensagem(Model model) {
      model.addAttribute("mensagem", "Uma mensagem qualquer");
      ...
}
...
```

Como devemos reescrever a tag para que o seu conteúdo seja preenchido com o valor do atributo "mensagem" que foi adicionado no método **mostrarMensagem**?

a) ${mensagem}
b) th:="{mensagem}"

c) th:text=”{mensagem}”
d) th:text=”${mensagem}”
e)

Para as questões 13 e 14 a seguir, considere que temos uma classe produto como a seguinte em nosso projeto:

```
package com.alessandro.app1.model;

public class Produto {
	private int id;
	private String nome;
	private String descricao;
	private double preco;
	public Produto() {
		super();
	}
	public Produto(int id, String nome, String descricao, double preco) {
		super();
		this.id = id;
		this.nome = nome;
		this.descricao = descricao;
		this.preco = preco;
	}
	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getNome() {
		return nome;
	}
	public void setNome(String nome) {
		this.nome = nome;
	}
	public String getDescricao() {
		return descricao;
	}
	public void setDescricao(String descricao) {
		this.descricao = descricao;
	}
	public double getPreco() {
		return preco;
	}
	public void setPreco(double preco) {
		this.preco = preco;
	}
}
```

13 – Além da classe Produto definida anteriormente, considere também o método **novo2()** abaixo, presente em uma classe definida como controladora

```
	@PostMapping("/produto/new")
	public String novo2(
			Produto produto,
			Model model) {

		model.addAttribute("produto", produto);
```

```
        return "novo-produto-criado2";
    }
```

Por fim, considere o template a seguir:

```
<!DOCTYPE html>
<html>
    <head>
        ...
    </head>
    <body>
        <h1>Novo Produto</h1>
        <form action="..." method="post">
            <span for="id">ID:</span>
            <input type="text" name="id"> <br>

            <span for="nome">Nome:</span>
            <input type="text" name="nome"> <br>

            <span for="descricao">Descrição:</span>
            <input type="text" name="descricao"> <br>

            <span for="preco">Preço:</span>
            <input type="text" name="preco"> <br>

            <input type="submit" value="Enviar">
        </form>
    </body>
</html>
```

Qual valor deve estar presente na propriedade “action” da tag form para que o método **novo2()** seja chamado?

a) /produto/novo2
b) /produto/novo
c) /produto
d) novo/produto
e) /produto/new

14 – Considerando a classe **Produto** definida anteriormente, considerando também o método abaixo:

```
    @PostMapping("/produto/new")
    public String novo2(
                Produto produto,
                Model model) {

        model.addAttribute("produto", produto);

        return "novo-produto-criado2";
    }
```

Qual notação devemos utilizar para que o conteúdo do atributo **descricao** de Produto seja utilizado para preencher a tag no template abaixo:

```
<!DOCTYPE html>
<html>
    <head>
        ...
    </head>
    <body>
        <span ...></span>
    </body>
</html>
```

a) th="${produto.descricao}"
b) text="${produto.descricao}"
c) th:text="${produto_descricao}"
d) th:text="${produto.descricao}"
e) text="${produto.descricao}"

Respostas:

1d;2c;3e;4d;5e;6c;7d;8e;9e;10a;11a;12e;13e;14d;