

Lista 3 - Persistência com Spring Boot e Jackson

Prazo de entrega: 30/08/2024 até 23:59 - (Sábado)

Instruções de entrega: Você deve ter um repositório em seu github para a disciplina de Programação procedural chamado **desenvolvimento-backend**

Crie uma pasta dentro dele chamada Listas e dentro dela crie uma pasta chamada Lista3. Para cada exercício da lista deve ser criado um projeto spring boot (pode utilizar o [spring inicializ](#))

Ao terminar a lista, suba seus exercícios no github e envie o link no formulário abaixo.

Link para o formulário de envio: <https://forms.gle/Wrtuxo9ZvYdB5Un8>

MONITORIAS: Serão realizados quatro dias de monitorias para auxiliar com a resolução dos exercícios. As datas e horários das monitorias são:

Dia	Horários		
Segunda-Feira	19h até 21h		
Terça-Feira	19h até 21h	-	-
Quarta-Feira	09h até 11h	19h até 21h	-
Quinta-Feira	19h até 21h	-	-
Sexta-Feira	09h até 11h	19h até 21h	22h até 23:59

OBS: Se você não conseguir participar das monitorias nos dias e horários propostos, você pode solicitar uma monitoria em uma data e horário diferente, desde que seja comunicado com pelo menos um dia de antecedência 😊

Exercícios

1. Uma ONG de apoio a animais abandonados te contratou para realizar a construção de um sistema de adoção de pets.



O sistema em questão deve ser construído com Spring Boot, deve utilizar arquitetura MVC e a **biblioteca Jackson** para realizar a persistência dos Pets em um arquivo Json.

Você deve criar um programa em Spring boot que contenha uma classe chamada Pet

```
public class Pet {
    private int id;
    private String nome;
    private String especie;
    private boolean jaFoiAdotado;

    public Pet() {
    }

    public Pet(int id, String nome, String especie, boolean
jaFoiAdotado) {
        this.id = id;
        this.nome = nome;
        this.especie = especie;
        this.jaFoiAdotado = jaFoiAdotado;
    }
}
```

```

public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getEspecie() {
    return especie;
}

public void setEspecie(String especie) {
    this.especie = especie;
}

public String getNome() {
    return nome;
}

public void setNome(String nome) {
    this.nome = nome;
}

public boolean getJaFoiAdotado() {
    return jaFoiAdotado;
}

public void setJaFoiAdotado(boolean jaFoiAdotado) {
    this.jaFoiAdotado = jaFoiAdotado;
}
}

```

Requisitos:

- Criar uma classe PetRepository
- **Adicionar um File e um ObjectMapper ao PetRepository**
- O File deve manipular um arquivo tb_pets.json para obter e salvar os pets
- Criar os métodos getAll e save (ambos devem manipular o arquivo tb_pets.json)
- Criar a classe PetService com os métodos listarPets e cadastrarNovoPet

- Criar a classe PetController com os métodos POST e GET (cadastrar e listar pets, respectivamente)
- Ser capaz de cadastrar e listar os pets através da rota <http://localhost:8080/pets>

Exemplo de entrada:

POST <http://localhost:8080/app/pets>

body

```
{
  "id": 1,
  "nome": "Jake",
  "especie": "Cachorro",
  "jaFoiAdotado": false
}
```

Saída esperada:

- *Jake cadastrado com sucesso!*

Exemplo de entrada:

GET <http://localhost:8080/app/pets>

Saída esperada:

- Deve listar todos os pets. Suponha que existam 2 pets. Seriam listados nesse formato:

```
[
  {
    "id": 1,
    "nome": "Jake",
    "especie": "Cachorro",
    "jaFoiAdotado": false
  },
  {
    "id": 2,
    "nome": "Dowakhin",
    "especie": "Gato",
  }
]
```

```
    "jaFoiAdotado":true  
  }  
]
```