07

Consolidando o seu conhecimento

Chegou a hora de você seguir todos os passos realizados por mim durante esta aula:

- Na classe `Medico`, deixe os seus atributos privados e implemente os seus *getters* e *setters* (exceto o `setId`)
  - Tipe os seus *getters* e *setters*
  - Faça com que ela implemente a classe `JsonSerializable`. Logo, implemente o método `jsonSerialize`, fazendo com que ele retorne um array associativo, com as propriedades da classe:

```
public function jsonSerialize()
{
    return [
        'id' => $this->getId(),
        'crm' => $this->getCrm(),
        'nome' => $this->getNome(),
        'especialidadeId' => $this->getEspecialidade()->getId()
    ];
}
```

- Implemente o construtor da classe `MedicoFactory`, injetando um `EspecialidadeRepository` nele e atribuindo-o a um atributo privado da classe
  - Dentro do método `criarMedico`, corrija o código para utilizar os métodos de acesso aos atributos da classe `Medico`
  - Além disso, busque a especialidade, através do seu `id` que está nos dados em JSON, dentro de um repositório de especialidades
  - Por fim, atribua a especialidade ao médico
- Na classe `MedicosController`, corrija o código para utilizar os métodos de acesso aos atributos da classe `Medico`
- No construtor da classe `EspecialidadesController`, injete um `EspecialidadeRepository` nele e atribua-o a um atributo privado da classe
  - Além disso, crie e implemente o método `buscarTodas()`, que mapeia a rota `/especialidades` e só aceita requisições do tipo GET
  - Esse método irá buscar as especialidades dentro do repositório de especialidades e retorná-las em um `JsonResponse`
  - Crie e implemente também o método `buscarUma`, que recebe o `id` por parâmetro, mapeia a rota `/especialidades/{id}` e só aceita requisições do tipo GET
  - Esse método irá buscar uma especialidade, através do seu `id` recebido por parâmetro, dentro do repositório de especialidades e retorná-la em um `JsonResponse`
  - Crie e implemente o método `atualiza`, que recebe o `id` (inteiro) e o `Request` por parâmetro, mapeia a rota `/especialidades/{id}` e só aceita requisições do tipo PUT
  - Esse método irá buscar uma especialidade, através do seu `id` recebido por parâmetro, dentro do repositório de especialidades. Em seguida, irá atribuir a `descricao` dos dados em JSON à `descricao` da especialidade buscada anteriormente
  - Por fim, o método irá mandar as alterações para o banco de dados e retornar a especialidade em um `JsonResponse`
  - Crie e implemente o método `remove`, que recebe o `id` (inteiro) por parâmetro, mapeia a rota `/especialidades/{id}` e só aceita requisições do tipo DELETE
  - Esse método irá buscar a especialidade no seu repositório, através do `id` recebido por parâmetro, removê-la, e mandar as alterações para o banco de dados. Ao final, esse método retornará um `Response` sem conteúdo

Caso já tenha feito, excelente. Se ainda não, é importante que você execute o que foi visto nos vídeos para poder continuar com os próximos cursos que tenham este como pré-requisito.