07

Consolidando o seu conhecimento

Segue o nosso último exercício desse aula!

1) De volta no nosso primeiro projeto `gerenciador-de-cursos` crie uma nova classe `src\Controller\CursosEmJson` :

```
<?php

namespace Alura\Cursos\Controller;

use Alura\Cursos\Entity\Curso;
use Doctrine\ORM\EntityManagerInterface;
use Nyholm\Psr7\Response;
use Psr\Http\Message\ResponseInterface;
use Psr\Http\Message\ServerRequestInterface;
use Psr\Http\Server\RequestHandlerInterface;

class CursosEmJson implements RequestHandlerInterface
{
    /**
     * @var \Doctrine\Common\Persistence\ObjectRepository
     */
    private $repositorioDeCursos;

    public function __construct(EntityManagerInterface $entityManager)
    {
        $this->repositorioDeCursos = $entityManager
            ->getRepository(Curso::class);
    }

    public function handle(ServerRequestInterface $request): ResponseInterface
    {
        $cursos = $this->repositorioDeCursos->findAll();
        return new Response(
            200,
            ['Content-Type' => 'application/json'],
            json_encode($cursos)
        );
    }
}
```

2) Também já crie a classe `CursosEmXml` na mesma pasta `src\Controller` :

```
<?php

namespace Alura\Cursos\Controller;

use Alura\Cursos\Entity\Curso;
use Doctrine\ORM\EntityManagerInterface;
use Nyholm\Psr7\Response;
use Psr\Http\Message\ResponseInterface;
use Psr\Http\Message\ServerRequestInterface;
```

```
use Psr\Http\Server\RequestHandlerInterface;

class CursosEmXml implements RequestHandlerInterface
{
    /**
     * @var \Doctrine\Common\Persistence\ObjectRepository
     */
    private $repositorioDeCursos;

    public function __construct(EntityManagerInterface $entityManager)
    {
        $this->repositorioDeCursos = $entityManager
            ->getRepository(Curso::class);
    }
    public function handle(ServerRequestInterface $request): ResponseInterface
    {
        /** @var Curso[] $cursos */
        $cursos = $this->repositorioDeCursos->findAll();
        $cursosEmXml = new \SimpleXMLElement('<cursos/>');

        foreach ($cursos as $curso) {
            $cursoEmXml = $cursosEmXml->addChild('curso');
            $cursoEmXml->addChild('id', $curso->getId());
            $cursoEmXml->addChild('descricao', $curso->getDescricao());
        }

        return new Response(
            200,
            ['Content-Type' => 'application/xml'],
            $cursosEmXml->asXML()
        );
    }
}
```

3) Abra a classe `src\Entity\Curso` e implemente a interface:

```
class Curso implements \JsonSerializable
```

4) Ainda na classe `src\Entity\Curso` adicione um novo método:

```
public function jsonSerialize()
{
        return [
            'id' => $this->id,
            'descricao' => $this->descricao
        ];
}
```

5) Para definir as rotas abra o arquivo `config\routes.php` e adicione duas novas rotas para chamar `CursosEmJson` e `CursosEmXml` respectivamente:

```
<?php
```

```
use Alura\Cursos\Controller\{CursosEmJson,
    CursosEmXml,
    Deslogar,
    Exclusao,
    FormularioEdicao,
    FormularioInsercao,
    FormularioLogin,
    ListarCursos,
    Persistencia,
    RealizarLogin};

return [
    '/listar-cursos' => ListarCursos::class,
    '/novo-curso' => FormularioInsercao::class,
    '/salvar-curso' => Persistencia::class,
    '/excluir-curso' => Exclusao::class,
    '/alterar-curso' => FormularioEdicao::class,
    '/login' => FormularioLogin::class,
    '/realiza-login' => RealizarLogin::class,
    '/logout' => Deslogar::class,
    '/buscarCursosEmJson' => CursosEmJson::class,
    '/buscarCursosEmXml' => CursosEmXml::class
];
```

6) Com o servidor PHP rodando teste os URLs:

```
http://localhost:8080/buscarCursosEmJson
http://localhost:8080/buscarCursosEmXml
```