Laboratório 8

Sistemas Distribuídos
Prof Adailton de Jesus Cerqueira Junior

Neste laboratório, daremos continuidade à criação de uma aplicação simples utilizando NodeJS com o framework Express. Agora, vamos aplicar um padrão arquitetural para separar responsabilidades dentro de uma aplicação, promovendo uma estrutura mais organizada, modular e de fácil manutenção.

Servidor:

1 – Crie uma pasta chamada lab08 e dentro desta pasta vamos copiar os arquivos criados no laboratório 7.

NOTA: *Você pode fazer esse laboratório direto na pasta do lab07, mas aconselho a criação de um novo diretório para manter o histórico dos laboratórios.*

2 – Dentro da pasta lab08, vamos criar uma pasta chamada controllers.

3 – A pasta controllers será responsável por armazenar nossas classes que irão intermediar a comunicação entre os modelos e as requisições do cliente. Basicamente estamos removendo uma responsabilidade que antes estava com o arquivo de rotas e agora será separado pelo contexto de cada recurso.

4 - Na pasta controllers iremos criar duas classes: JogoController e EmpresaController.

Class JogoController:

1 - A classe JogoController será responsável por ter nossas regras de negócio relacionadas ao recurso Jogo.

```
const JogoDAO = require('../daos/JogoDAO');

class JogoController {

}

module.exports = new JogoController;
```

2 - Dentro da classe JogoController, vamos transferir as responsabilidades que estavam nas rotas para funções específicas nessa classe.

```
index(req, res) {
    JogoDAO.findAll(req.query.categoria, (err, jogos) => {
        if (err) return res.status(500).json({ error: err.message });
        res.json(jogos);
    });
}
```

```
show(req, res) {
    const id = req.params.id;
    JogoDAO.findById(id, (err, jogo) => {
        if (err) return res.status(500).json({ error: err.message });
        if (jogo) {
            res.json(jogo);
        } else {
            res.status(404).json('Jogo não encontrado.');
        }
    });
}
```

```
create(req, res) {
    const { nome, categoria, ano, fkEmpresa } = req.body;
    if (!nome && !categoria && !ano && !fkEmpresa) return res.status(400)
        .json({ error: "Campos nome, categoria e ano são obrigatórios" });

    JogoDAO.create(nome, categoria, ano, fkEmpresa, (err, jogo) => {
        if (err) return res.status(500).json({ error: err.message });
        res.status(201).json(jogo);
    });
}
```

```
update(req, res) {
    const { nome, categoria, ano } = req.body;
    const id = req.params.id;

    JogoDAO.update(id, nome, categoria, ano, (err, jogo) => {
        if (err) return res.status(500).json({ error: err.message });
        if (!jogo) return res.status(404).json({ error: "Jogo não encontrado" });
        res.json({ message: "Jogo editado com sucesso." });
    });
}
```

```
delete(req, res) {
    const id = req.params.id;

    JogoDAO.delete(id, (err, jogo) => {
        if (err) return res.status(500).json({ error: err.message });
        if (!jogo) return res.status(404).json({ error: "Jogo não encontrado." });
        res.json({ message: "Jogo removido com sucesso." });
    });
}
```

NOTA: *Dedique um tempo para analisar o que os métodos estão fazendo.*

Rotas:

1 – Precisaremos realizar algumas modificações no arquivo **index.js** e nas rotas existentes para utilizar o controller.

2 – Vamos ajustar o início do arquivo **index.js**, incluindo informações, como a importação dos controllers.

```
const express = require('express');
const JogoController = require('./controllers/JogoController');
const EmpresaController = require('./controllers/EmpresaController');

const app = express();
const APP_PORT = process.env.APP_PORT || 3000;
```

3 – Agora, vamos atualizar as rotas para utilizar a classe controller, conforme apresentado a seguir.

```
app.get('/', (req, res) => res.send('API Version 1.2.0 on-line!'));

app.get('/jogos', JogoController.index);
app.get('/jogos/:id', JogoController.show);
app.post('/jogos', JogoController.create);
app.put('/jogos/:id', JogoController.update);
app.delete('/jogos/:id', JogoController.delete);
```

NOTA: *Compare a implementação das rotas do laboratório 7 com essa nova implementação. Note que a responsabilidade foi migrada para classe controller.*

4 – Aplique a mesma solução para o recurso empresa.