13

Consolidando seu conhecimento

Chegou a hora do exercício do capítulo. Para fazê-lo, é importante que você tenha feito o último exercício do capítulo anterior ou tenha baixado o stage com o código completo do capítulo anterior no texto explicativo deste capítulo.

Questão

Como sempre, para não fazer spoilers, passarei as linhas gerais em uma ordem cronológica válida, para que você implemente o que viu neste capítulo em nosso projeto.

1 - Configuramos a rota `v1/fotos` através do Express disponibilizando uma lista de fotos para nossa aplicação Angular. Muito bem, mas durante o treinamento de Angular, requisitávamos outros recursos do servidor, como uma lista de grupos para popular a combobox do cadastro de fotos. Altere `alurapic/config/express.js` e disponibilize uma lista de grupos para a URL `/v1/grupos`. Para adiantar, segue a lista de grupos para você usar o famoso CONTROL + C / CONTROL + V:

```
var grupos = [
    { _id: 1, nome: 'esporte' },
    { _id: 2, nome: 'lugares' },
    { _id: 3, nome: 'animais' }
];
```

2 - Que tal separarmos melhor as responsabilidades do nosso código? Crie os módulos que armazenarão nossas rotas `alurapic/app/routes/foto.js`, `alurapic/app/routes/grupo.js` e os arquivos que armazenarão nossas API's `alurapic/app/api/foto.js` e `alurapic/app/api/grupo.js`.

3 - Depois de separar o código nesses arquivos, você precisará fazer com que o `alurapic/config/express.js` seja capaz de carregá-los e você usará o módulo `consign` para isso. Baixe-o através do npm, configurando-o em `alurapic/config/express.js`.

4 - No final, tudo deve continuar funcionando. Sendo assim, acesse sua aplicação e verifique se o servidor sobe e que sua aplicação Angular ainda continua exibindo uma lista de fotos, inclusive se já preenche a combobox de grupos na tela de cadastro de fotos.

Responda

INSERIR CÓDIGO		FORMATAÇÃO