Node.js Parte 2: MVC, autenticação e autorização

- Esse curso capacita a:
  - Entender o padrão MVC usando Node.js;
  - Trabalhar com sessão de usuários;
  - Validar dados com Express Validator;
  - Implementar uma solução para autenticação e autorização;
  - Saber como lidar com erros da aplicação;

Aulas:

1. **Refinando a aplicação e tratamento de erros:**
   - Como incluir arquivos estáticos de CSS num projeto Node e apresentar uma aplicação mais agradável visualmente;
   - O que é o famoso erro 404 que se refere a recursos não encontrados e como tratá-lo;
   - O que são erros internos da aplicação e como trata-los passando ao usuário o status HTTP 500 e uma página condizente com o erro;

2. **Validação de dados:**

   Como a própria documentação diz, o Express Validator é um conjunto de *middlewares* que encapsulam diversos recursos da biblioteca **validator.js**, dentre os quais se encontra o recurso de validação de dados! Objeto de nossos estudos!

   Desse modo, quando fazemos algo como:

   ```
   const { check, validationResult } = require('express-validator/check');

   app.post('/user', [
           check('username').isEmail(),
           check('password').isLength({ min: 5 })
       ],
       function (req, res) {
           const errors = validationResult(req);
           if (!errors.isEmpty()) {
               return res.status(422).json({ errors: errors.array() });
           }

           // adição do usuário no BD.
       }
   );
   ```

   Na realidade, o que estamos fazendo ao passar um *array* como segundo parâmetro do método `post()` é adicionar um conjunto de validadores (todos eles *middlewares* que serão executados antes do *callback* da rota passado como terceiro parâmetro do `post()`), cada um deles retornando o que o *Express Validator* chama de **validation chain**! O *validation chain* nada mais é do que o encadeamento das validações, que nos possibilita, dentre outras coisas, definir qual a validação que será feita no campo e, inclusive, customizar a mensagem de erro da validação! Exatamente um recurso que veremos um pouco mais a frente, ainda nesse capítulo do curso! Portanto, sempre que fazemos `check('nomeDoCampo')`, obtemos um *validation chain* que podemos utilizar para configurar a validação, a mensagem de erro e muito mais!

   -

- O que são validações de dados e sua importância;
- O que é o Express Validator e como ele ajuda na escrita de um código mais organizado e semântico;
- Implementação de validações para o formulário de cadastro de livros;
- Apresentação de mensagens no template informando ao usuário sobre os possíveis erros de validação;

3. **MVC:**
    - O que é o padrão arquitetural MVC;
    - Como o MVC ajuda a organizar o projeto facilitando a separação de responsabilidades e aumentando o grau de manutenção do código;
    - Implementação na prática do padrão MVC no projeto;
    - Facilitadores para acesso centralizado à URLs de rotas;
    - Facilitadores para acesso centralizado aos templates criados;

4. **Autenticação de usuários:**
    - O que é autenticação;
    - O que é sessão;
    - O que são cookies;
    - Como implementar uma sessão autenticável em uma aplicação Node;
    - O que significa o termo injeção de dependências;
    - Como implementar injeção de dependências numa aplicação Node;

5. **Autorização de acesso:**
    - A autenticação diz respeito a verificação de identidade. Isso pode acontecer através de um login, token, impressão digital, RG, entre várias outras formas e combinações;
    - Na autorização, é verificada a permissão de acesso. No nosso contexto, queremos que apenas usuários autenticados tenham acesso, mas poderíamos criar permissões e papeis específicos que detalhem o acesso;
    - Resumo do dois cursos de Node:
      - Aplicação de bibliotecas front-end em projetos Node;
      - Validação de dados;
      - Tratamento de erros e apresentação de páginas coerentes e informativas ao usuário;
      - O padrão arquitetural MVC e boas práticas;
      - Autenticação;
      - Sessão e cookies;
      - Autorização;
        - O que é autorização;
        - Como implementar autorização numa aplicação Node;
        - Diferença entre autenticação e autorização;