# Respostas das Questões – Aula 10

1) Identificar as partes principais de um CRUD em Node.js e Express. Identifique e explique cada uma das seguintes partes: Configuração do servidor, Rota para listar os itens, Rota para adicionar um item, Conexão com o banco de dados.

R: No CRUD a gente tem quatro partes principais:

* configuração do servidor, que é onde o sistema começa a rodar;
* rota para listar os itens, que mostra tudo o que tem salvo;
* rota para adicionar um item, que serve para cadastrar coisas novas;
* conexão com o banco de dados, que é o que faz o sistema guardar e buscar informações.

2) O que é um servidor e qual sua função no back-end de uma aplicação?

R: O servidor é como o coração do sistema, ele recebe os pedidos do usuário, faz o que precisa e devolve a resposta. No back-end, ele cuida de processar as informações e conversar com o banco de dados.

3) Qual a função do Node.js no desenvolvimento back-end?

R: O Node.js serve pra rodar JavaScript no servidor. Ele é usado pra criar o back-end, fazer o sistema funcionar e responder as requisições dos usuários.

4) Como funcionam as rotas no Express.js e como são definidas?

R: As rotas são os caminhos que o servidor segue pra saber o que fazer quando o usuário faz um pedido. Cada rota tem uma função, tipo mostrar, adicionar ou apagar algo.

5) O que são controladores (controllers) no contexto do back-end e como eles organizam a lógica de negócio?

R: Os controladores são como as partes do sistema que decidem o que acontece quando alguém faz um pedido. Eles deixam o código mais organizado, separando as funções de cada parte do sistema.

6) O que é uma requisição HTTP e quais são suas partes principais?

R: Uma requisição HTTP é quando o cliente pede algo pro servidor. Ela tem o tipo do pedido (como GET ou POST), o endereço que quer acessar e, às vezes, os dados que quer enviar.

7) O que são middlewares e como eles podem ser aplicados em rotas específicas no Express?

R: Middlewares são funções que rodam antes de chegar na rota principal. Eles servem pra coisas como checar login, validar dados ou mostrar mensagens de erro.

8) Como o Mongoose é utilizado para MongoDB?

R: O Mongoose ajuda o Node.js a conversar com o banco de dados MongoDB. Ele organiza os dados e facilita pra gente salvar, buscar e atualizar informações.

9) O que é validação de dados no back-end e como ela pode ser feita?

R: Validação de dados é conferir se o que o usuário mandou está certo antes de salvar. Dá pra fazer isso com regras no código ou com ferramentas que ajudam a verificar os dados.

10) O que é validação de dados no back-end e como ela pode ser feita com Mongoose?

R: No Mongoose dá pra colocar regras dentro do modelo, tipo campos obrigatórios ou tipo de dado. Assim ele já verifica tudo antes de salvar no banco.

11) Como o Express.js é utilizado para criar rotas que permitem realizar as operações CRUD no banco de dados MongoDB?

R: O Express cria as rotas que fazem o CRUD funcionar. Cada rota faz uma coisa: uma adiciona, outra mostra, outra atualiza e outra apaga informações do banco.

12) Qual é a função do modelo Book.js (usando Mongoose) no sistema, e como ele define a estrutura de um documento no MongoDB?

R: O modelo Book.js define como os dados de um livro vão ser salvos no banco. Ele mostra quais informações cada livro precisa ter, tipo título e autor.

13) Como a função findByIdAndUpdate é utilizada na rota de atualização (PUT) para modificar os dados de um livro?

R: Essa função serve pra achar um item no banco pelo ID e mudar as informações dele. É o que o sistema usa pra atualizar algo já existente.

14) Por que utilizamos try...catch nas operações CRUD e como tratamos erros em requisições que falham?

R: A gente usa try...catch pra evitar que o sistema pare quando dá erro. Assim, se algo der errado, dá pra mostrar uma mensagem pro usuário sem travar tudo.

15) Como o método find() no Mongoose funciona para retornar a lista de todos os livros na rota GET, e como ele interage com o MongoDB?

R: O método find busca todos os itens dentro de uma coleção do banco e devolve a lista. É o jeito mais fácil de mostrar tudo que tem salvo.

16) O que o comando npm init -y faz ao criar um novo projeto Node.js?

R: Esse comando cria o arquivo principal do projeto (package.json) com as informações básicas pra começar o projeto.

17) O que o método app.get() faz na API Express?

R: O app.get cria uma rota pra mostrar informações. Ele serve pra quando o usuário quer ver dados do servidor.

18) Qual a função da linha app.listen(port, () => {...}); no código da API?

R: Essa linha é o que faz o servidor começar a rodar. Ela define a porta e mostra que o sistema tá funcionando.

19) Qual é o papel do require('express') no início do código?

R: Ele importa o Express pro projeto, pra gente poder usar ele e criar as rotas do servidor.

20) Como você pode testar se a API está funcionando corretamente sem usar o navegador?

R: Dá pra testar usando programas como Insomnia ou Postman, que mandam pedidos pro servidor e mostram as respostas.