****

**Desenvolvimento de Aplicativos Web e Cloud Computing**

Nome:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Matrícula: \_\_\_\_\_\_\_\_\_

Curso: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Data: \_\_\_\_/\_\_\_\_/\_\_\_\_\_

Escreva as respostas abaixo

Questão 1:

a) Defina o papel de cada um e responda por que ao criarmos nosso controller fizemos

uma herança de controllerbase?

**Quando criamos um controller em uma API, é como se estivéssemos criando um guia que diz ao servidor o que fazer quando alguém pede alguma coisa, como buscar ou enviar informações. Agora, o ControllerBase é uma versão mais simples desse guia, feita só para APIs. Usamos ele porque ele tem exatamente o que precisamos para as APIs, sem partes extras que são usadas em sites que mostram páginas.**

b) Em sala, mostramos que ao abrir um navegador conseguimos chamar uma rota GET e

POST, explique por que só conseguimos executar a Rota GET no browser. **Quando você abre o navegador e digita um site, o navegador faz um pedido do tipo GET, que é como pedir para ver uma página ou pegar alguma informação.**

**Já o pedido do tipo POST é usado para mandar informações para o servidor, como quando você preenche um formulário. O navegador não faz isso sozinho; você precisa de algo a mais, como um botão de envio ou um programa especial, para fazer um pedido POST. É por isso que você só vê GET funcionando direto no navegador.**

Questão 2:

Dadas ações cite o verbo http correto para utilizarmos, considere ações no AVA do uniaraxá:

a) **Criar a aula** - POST  
b) **Remover um material** - DELETE  
c) **Listar as disciplinas disponíveis para o professor no semestre** - GET  
d) **Listar os alunos da disciplina** - GET  
e) **Atualizar um material** - PUT  
f) **Lançar uma nota** - POST

Questão 3: Em sala mostramos que ao fazer um return da action utilizamos um Status Http , por exemplo, Ok(), BadRequest(), NotFound() , entre outros. Especifique a diferença de cada um.

 **Ok()**: Significa que deu tudo certo. O servidor conseguiu processar o pedido e está devolvendo a resposta esperada (geralmente usado para retornos de sucesso, como status 200).

 **BadRequest()**: Significa que houve um problema com o que foi enviado para o servidor. O pedido estava errado ou faltando alguma coisa (é o status 400, que indica um erro do lado do cliente).

 **NotFound()**: Significa que o servidor não conseguiu encontrar o que foi pedido. Isso acontece quando a URL ou o recurso solicitado não existe (status 404).

Questão 4: A seguir , temos um exemplo de endpoint de API: https://minhaapi.com.br/api/vendas explique cada parte: • https:// • minhaapi.com.br • /api/vendas

**https://**: Isso é o protocolo que garante uma conexão segura entre o seu navegador (ou aplicativo) e o servidor. Ele criptografa os dados enviados, para que ninguém consiga interceptá-los.

**minhaapi.com.br**: Esse é o endereço do servidor onde a API está hospedada. É como o nome de um site que você visita, mas neste caso, é o endereço da API.

**/api/vendas**: Isso é o caminho dentro do servidor que aponta para um serviço específico. Aqui, está indicando que estamos acessando a parte da API que lida com vendas.