**LISTA 2 – Desenvolvimento Web**

**Nome: Náthan Luciano de Oliveira e Silva RA:043412**

**1)** Validar dados em uma API é fundamental para garantir a integridade, segurança e eficácia da comunicação entre sistemas e serviços.

Integridade dos Dados: Garante que os dados estejam corretos e em conformidade, evitando erros e inconsistências.

Segurança: Protege contra ataques e vulnerabilidades, como injeção de SQL e XSS.

Experiência do Usuário: Melhora a usabilidade da API, identificando erros de forma clara e permitindo correções adequadas.

Melhorar a Experiência do Usuário: A validação de dados também desempenha um papel crucial na melhoria da experiência do usuário. Quando os dados são validados corretamente, os erros são identificados e tratados de maneira adequada, em vez de resultarem em falhas inesperadas ou respostas de erro genéricas.

**2) - a) Cenário 1: Cadastro de Cliente:**

Nome: Verificar se o campo não está em branco e se contém apenas caracteres válidos (alfabéticos).

CPF: Validar a estrutura do CPF (formato e dígitos verificadores) e verificar se ele já não está cadastrado.

Data de Nascimento: Garantir que a data seja válida e que o cliente tenha uma idade adequada (por exemplo, maior de 18 anos).

Email: Verificar se o email tem um formato válido e se não está duplicado.

Telefone: Validar o formato do número de telefone (por exemplo, seguindo o padrão nacional) e garantir que seja único.

Senha: Aplicar políticas de segurança de senha, como comprimento mínimo, uso de letras maiúsculas e minúsculas, números e caracteres especiais.

CEP, Rua, Bairro, Cidade e Estado: Verificar se os campos não estão em branco e se correspondem aos dados reais obtidos a partir do CEP.

Cep: Validar o formato do CEP (por exemplo, seguindo o padrão nacional).

**b) Cenário 2: Cadastro de Disciplina:**

Nome: Verificar se o campo não está em branco e se contém apenas caracteres válidos (alfabéticos).

Carga Horária: Garantir que seja um número positivo e realista em relação ao contexto da disciplina.

Objetivo: Verificar se o campo não está em branco e se não excede um comprimento máximo.

Ementa: Verificar se o campo não está em branco e se não excede um comprimento máximo.

Semestre e Ano: Validar se são números inteiros positivos e se fazem sentido em relação ao calendário acadêmico.

Nome do Professor: Verificar se o campo não está em branco e se corresponde a um professor cadastrado no sistema.

**3)** Vantagens:

Controle Total: Você tem total controle sobre a lógica de validação, o que permite implementar regras de negócios complexas e personalizadas.

Flexibilidade: É possível combinar validações personalizadas com validações padrão, permitindo uma abordagem mais flexível.

Independência da Plataforma: Não está vinculado a um framework ou biblioteca específicos, tornando-o independente da plataforma.

Desvantagens:

Código Verboso: A implementação de validações usando "if/else" pode levar a um código extenso e verboso, tornando-o difícil de manter e aumentando o risco de erros.

Repetição de Código: As validações precisam ser repetidas em várias partes do código, o que pode levar a inconsistências e duplicações.

Dificuldade na Manutenção: À medida que a complexidade das regras de validação aumenta, a manutenção se torna mais desafiadora, tornando o código propenso a erros.

Validação com "Data Annotations":

Vantagens:

Declarativo: As regras de validação são declaradas diretamente nas propriedades do objeto usando atributos, o que torna o código mais claro e legível.

Menos Código: Requer menos código em comparação com implementações manuais, o que reduz a complexidade e a probabilidade de erros de codificação.

Integração com Frameworks: Muitos frameworks, como ASP.NET MVC, fazem uso extensivo de "Data Annotations" para automatizar a validação, tornando a integração com esses frameworks mais fácil.

Desvantagens:

Limitação de Expressividade: Em alguns casos, as "Data Annotations" podem não ser suficientemente expressivas para representar regras de validação complexas.

Acoplamento com o Modelo de Dados: As "Data Annotations" podem criar um acoplamento entre o modelo de dados e a lógica de validação, o que pode não ser desejável em alguns cenários.

Limitações na Validação Customizada: Para validações altamente personalizadas, pode ser necessário recorrer a abordagens adicionais, tornando o código menos uniforme.

4) “ele automaticamente já validava e não caía no debug da action," isso significa que o framework ou a estrutura que você está usando para desenvolver o aplicativo web está projetado para realizar a validação automaticamente antes de chegar à ação do controlador. Isso ajuda a manter a integridade dos dados e a melhorar a segurança do aplicativo, pois impede que dados inválidos entrem no sistema e causem problemas mais tarde.