

Avaliação Técnica

1. Objetivo

O objetivo principal desta avaliação é determinar o nível de conhecimento do candidato sobre a implementação de soluções com base na demanda estipulada.

Tenha atenção aos detalhes solicitados, garantindo que todos os requisitos solicitados sejam atendidos de forma adequada.

2. Requisitos

Criar uma API REST em que tem o objetivo de gerenciar e cadastrar um produto.

A) Características do produto:

- Id
- Descrição
- Categoria
- Dimensões
- Código
- Referência
- Saldo de Estoque
- Preço
- Ativo

B) Deverá ter categoria:

- Id
- Descrição
- Ativo

C) Deverá possuir operações que contemplem os seguintes métodos de HTTP

- GET
- POST
- PUT
- DELETE
- PATCH

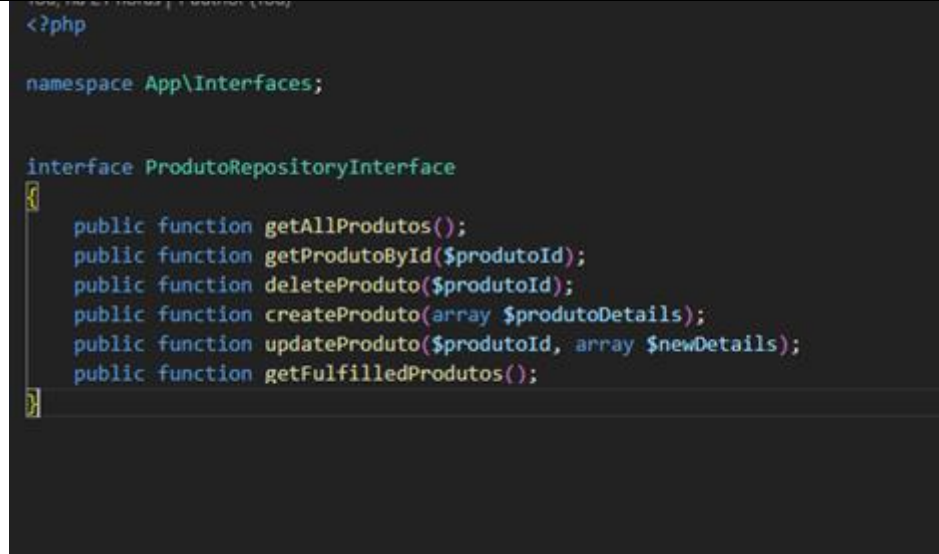
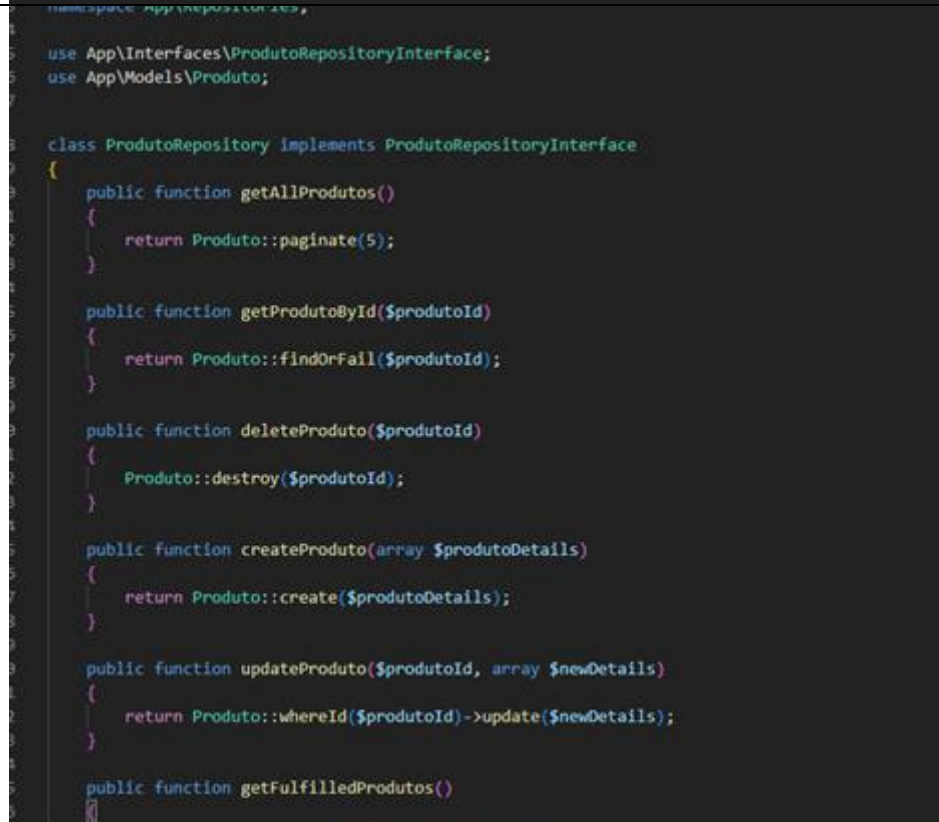
D) O serviço responsável por listar os produtos e categorias deverá ter paginação implementada

E) Utilize o Swagger para documentar a sua API

F) Deverá salvar os dados em um banco de dados MySQL.

G) O projeto deverá implementar repository pattern

- H) Utilizar técnicas para apresentar um código limpo, portanto mostre pelo menos 3 exemplos no seu código que você considera estar escrevendo código de forma limpa e organizada e justifique ao lado por qual motivo tomou esta decisão. Para isso você deve fazer um print da parte do código como o exemplo abaixo:

Imagem	Justificativa
 <pre><?php namespace App\Interfaces; interface ProdutoRepositoryInterface { public function getAllProdutos(); public function getProdutoById(\$produtoId); public function deleteProduto(\$produtoId); public function createProduto(array \$produtoDetails); public function updateProduto(\$produtoId, array \$newDetails); public function getFulfilledProdutos(); }</pre>	Criação de Interfaces para manipulação
 <pre>namespace App\Repositories; use App\Interfaces\ProdutoRepositoryInterface; use App\Models\Produto; class ProdutoRepository implements ProdutoRepositoryInterface { public function getAllProdutos() { return Produto::paginate(5); } public function getProdutoById(\$produtoId) { return Produto::findOrFail(\$produtoId); } public function deleteProduto(\$produtoId) { Produto::destroy(\$produtoId); } public function createProduto(array \$produtoDetails) { return Produto::create(\$produtoDetails); } public function updateProduto(\$produtoId, array \$newDetails) { return Produto::whereId(\$produtoId)->update(\$newDetails); } public function getFulfilledProdutos() { } }</pre>	Criação de Repositorios para manipulação dos dados

```

/**
 * @OA\Post(
 *   path="/produtos",
 *   tags={"Products"},
 *   security={{"bearerAuth":{}}},
 *   @OA\Response(
 *     response="200",
 *     description="Store new product"
 *   ),
 *
 *   @OA\RequestBody(
 *     required=true,
 *     @OA\JsonContent(
 *       required={"descricao","dimensoes","codigo","referencia","saldo_estoque","preco","ca
 *       @OA\Property(property="descricao", type="string"),
 *       @OA\Property(property="dimensoes", type="string"),
 *       @OA\Property(property="codigo", type="string"),
 *       @OA\Property(property="referencia", type="string"),
 *       @OA\Property(property="saldo_estoque", type="integer"),
 *       @OA\Property(property="preco", type="float"),
 *       @OA\Property(property="categoria_id", type="integer"),
 *     ),
 *   ),
 * )
 */
public function store(Request $request): JsonResponse
{
    $produtoDetails = $request->only([
        'descricao',
        'dimensoes',
        'codigo',
        'referencia',
        'saldo_estoque',
    ]);
}

```

Annotations nos controllers

- I) O código deve ter pelo menos 80% de cobertura de testes unitários
- J) Faça um breve resumo dos motivos que te levaram a criar o projeto com a arquitetura escolhida, por exemplo, implementações de interfaces, separações de camadas, separações de pastas, criação das tabelas do banco de dados etc...

Laravel é um framework completo que fornece ferramentas para construção de uma API de forma fácil, rápida e confiável. Com a capacidade de migrations e gerenciamento da base de dados. As pastas gerenciadas com esse framework é intuitiva.

3. Bonus 1

Caso queira entregar um plus, sugiro a implementação de Autenticação e Autorização (JWT) utilizando Identity nos endpoints criados, estabelecendo pelo menos uma role específica para cada método HTTP.

4. Bonus 2

Implementar a Autenticação e Autorização (JWT) a partir do próprio Swagger.

5. Onde disponibilizar o código fonte?

Você deverá criar um repositório em um controle de versão de sua preferência, seguindo alguns requisitos:

- Deverá ser privado
- Deverá dar acesso para lsilva@alter-solutions.com

6. Dicas

- ✓ Apesar de ser um projeto muito simples, pense em uma arquitetura de um projeto maior, desta forma podemos avaliar a sua capacidade de criar projetos escaláveis e com código reutilizável
- ✓ Lembre que em projetos um pouco maiores teremos a necessidade de mais ambientes, como Development, Production etc...
- ✓ Caso ache necessário você pode utilizar pacotes nuget de sua preferência para implementar suas funcionalidades
- ✓ **Você terá 48 horas para entregar este projeto**