## **Teste técnico - Desenvolvedor Backend**

Você recebeu a tarefa de criar uma aplicação que permita pesquisar repositórios públicos no GitHub, marcar os favoritos e organizar os resultados conforme critérios de relevância.

Seu objetivo é construir essa aplicação com foco em arquitetura limpa e boas práticas de desenvolvimento, organizando bem as camadas e separando as responsabilidades do sistema.

A aplicação já possui um esqueleto (backend + Frontend), mas de forma proposital está simples e acoplada. Cabe a você aplicar as melhorias de estrutura, implementar os comportamentos solicitados, integrar com o Github e aplicar sua lógica de negócio onde for necessário.

Você terá liberdade para propor a organização da solução, contanto que atenda aos requisitos técnicos descritos abaixo.

## **Desafio Técnico – Repositórios do GitHub**

### **Objetivo**

Este desafio tem como objetivo avaliar sua capacidade de:

* Escrever código limpo, modular e de fácil manutenção;
* Aplicar boas práticas de arquitetura e desenvolvimento back-end;
* Projetar uma solução organizada com foco em lógica de negócio;
* Resolver um problema com liberdade para propor estrutura e soluções.

### **Descrição do Projeto**

Você deverá desenvolver uma aplicação web que permita:

* Buscar repositórios públicos do GitHub utilizando o **nome (ou parte do nome)** do repositório como filtro;
* Marcar repositórios como favoritos durante a execução da aplicação;
* Exibir uma lista de repositórios ordenada por **relevância**, conforme lógica de negócio definida por você.

### **Funcionalidades Obrigatórias**

#### **1. 🔍 Buscar repositórios públicos por nome**

* Deve ser possível pesquisar por qualquer repositório público no GitHub.
* A chamada à API do GitHub deve ser implementada por você.
  + API aberta: [https://api.github.com](https://api.github.com/search/repositories?q={nome)

#### **2. ⭐ Marcar repositórios como favoritos**

* Deve existir uma funcionalidade para favoritar/desfavoritar repositórios.
* Os favoritos devem ser mantidos **em tempo de execução**, sem persistência em banco de dados.

#### **3. 🔥 Listar repositórios por relevância**

* Crie um endpoint que retorne os repositórios ordenados por **relevância**.
* Você deverá definir a lógica de cálculo da relevância com base em dados como:
  + Estrelas (stargazers\_count)
  + Forks (forks\_count)
  + Watchers (watchers\_count)
* Explique sua escolha de lógica via comentários no código.

### **Requisitos Técnicos**

#### **Backend**

* Utilize **ASP.NET Core (WebAPI)**.
* Organize o projeto utilizando boas práticas de arquitetura:
  + Sugerimos separação por camadas: Domain, Application, Infrastructure, API.
* Utilize **DTOs** e modelos tipados para entrada e saída.
* Separe regras de negócio dos controllers.
* Implemente testes unitários para a lógica de relevância.

#### **Frontend**

* Utilize **Angular** apenas como apoio para exibir as informações da API.
* Não será avaliado design, apenas:
  + Separação da lógica de apresentação e consumo de API;
  + Uso de **interfaces tipadas** para respostas;
  + Código organizado.

### **Avaliação**

Serão considerados os seguintes critérios:

|  |  |
| --- | --- |
| **Critério** | **Peso** |
| Organização de código e arquitetura | 🟢 Alto |
| Clareza e manutenibilidade do código | 🟢 Alto |
| Separação de responsabilidades | 🟢 Alto |
| Aplicação de boas práticas (SOLID, DDD) | 🟡 Médio |
| Implementação da lógica de relevância | 🟢 Alto |
| Testes unitários | 🟡 Médio |
| Documentação mínima no código | 🟡 Médio |
| Funcionalidades funcionando corretamente | 🟢 Obrigatório |