

## ChatGPT

Preciso criar a documentação de um projeto de um aplicativo sobre os chefes do jogo Megaman da Nintendo. # Contexto: Esse projeto é um projeto de uma api feito em dotnet para listas os dados dos bosses do jogo Megaman. O objetivo principal é ter um back-end que fornece json no formato abaixo: "" { Id =1, Code = "DLN/DRN-003", Name = "Cutman", HP = 150, Picture = "https://vignette.wikia.nocookie.net/megaman/images/2/22/Cutman.png" } ""

# Especificação do projeto: "" <Project Sdk="Microsoft.NET.Sdk.Web">  
<PropertyGroup> <TargetFramework>netcoreapp3.1</TargetFramework>  
</PropertyGroup> <ItemGroup> <PackageReference  
Include="Microsoft.EntityFrameworkCore" Version="3.1.8" />  
<PackageReference Include="Microsoft.EntityFrameworkCore.Design"  
Version="3.1.8"> <IncludeAssets>runtime; build; native; contentfiles;  
analyzers; buildtransitive</IncludeAssets>  
<PrivateAssets>all</PrivateAssets> </PackageReference>  
<PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer"  
Version="3.1.8" /> <PackageReference Include="Newtonsoft.Json"  
Version="12.0.2" /> </ItemGroup> </Project> "" Os endpoints do projeto são: "" namespace Megaman.Controllers { //api/v1/robots [ApiController]  
[Route("api/v1/robots")] public class RobotsController : ControllerBase {  
private readonly IRobotServices \_services; public  
RobotsController(IRobotServices services) { \_services = services; } //GET  
api/robots [HttpGet] public ActionResult<IEnumerable<RobotReadDTO>>

```
GetAllRobots() { var robotItems = _services.SearchAll(); return  
Ok(robotItems); } //GET api/v1/robots/{id} [HttpGet] [Route("{id:int}")] public  
object GetCommandById([FromRoute]int id) { var robot =  
_services.SearchById(id); if(robot != null) return Ok(robot); return NotFound(  
new { message = "Nenhum robo encontrado" } ); } //POST api/v1/robots  
[HttpPost] public ActionResult RobotSend(){ return Ok(); } } "" # REGRAS: -  
Sempre que citar alguma dependência do projeto, deixe ele como hyperlink  
para a página oficial daquela dependência. - Organize as dependências em  
uma sessão em formato de tabela. - Crie uma estrutura do projeto com  
base na árvore de pasta e arquivos abaixo /.vs /.vscode /bin /Controllers  
/Database /middlewares /Models /obj /Properties /Services  
appsettings.Development.json appsettings.json global.json  
MegamanApi.csproj MegamanApi.sln Program.cs Startup.cs # Extras -  
Adicione uma sessão de endpoints em forma de tabela - Inclua uma sessão  
com as técnicas utilizadas - Inclua um sessão com as dependências  
utilizadas
```

Atualizar memória?

Aqui está a documentação inicial do seu projeto de API sobre os chefes do jogo Megaman. Você pode revisar e sugerir ajustes conforme necessário.

---

## Documentação da API - Megaman Bosses

---

### Visão Geral

---

Esta API fornece informações sobre os chefes (bosses) da franquia Megaman. Os dados são retornados no formato JSON e incluem detalhes como código, nome, HP e imagem.

### Formato de Resposta

---

json

```
{
  "Id": 1,
  "Code": "DLN/DRN-003",
  "Name": "Cutman",
  "HP": 150,
  "Picture":
  "https://vignette.wikia.nocookie.net/megaman/images/2/22/Cutman.png"
}
```

## Estrutura do Projeto

---

bash

```
/.vs
/.vscode
/bin
/Controllers
/Database
/middlewares
/Models
/obj
/Properties
/Services
appsettings.Development.json
appsettings.json
global.json
MegamanApi.csproj
MegamanApi.sln
Program.cs
Startup.cs
```

## Dependências

---

Pacote	Versão	Descrição
<u>Microsoft.EntityFrameworkCore</u>	3.1.8	ORM para acesso ao banco de dados
<u>Microsoft.EntityFrameworkCore.Design</u>	3.1.8	Ferramentas de design para Entity Framework Core
<u>Microsoft.EntityFrameworkCore.SqlServer</u>	3.1.8	Provedor SQL Server para Entity Framework Core
<u>Newtonsoft.Json</u>	12.0.2	Biblioteca para manipulação de JSON

## Endpoints

Método	Endpoint	Descrição
GET	/api/v1/robots	Retorna todos os robôs disponíveis
GET	/api/v1/robots/{id}	Retorna os detalhes de um robô específico
POST	/api/v1/robots	Adiciona um novo robô à base de dados

## Técnicas Utilizadas

- **ASP.NET Core 3.1** para construção da API
- **Entity Framework Core** para mapeamento objeto-relacional (ORM)
- **Injeção de Dependência** para gerenciamento de serviços
- **Newtonsoft.Json** para serialização e manipulação de JSON

Se precisar de ajustes ou adicionar mais detalhes, me avise!

O ChatGPT pode cometer erros. Considere verificar informações importantes.