Moralar

Manual de instalação para Api Moralar

Tabela de conteúdo

1. detalhes do projeto

Api Moralar: É uma Api feita em **.NET8** onde oferecemos serviços privados para nossos aplicativos web e móveis da Moralar.

Nessa API, lidamos com os padrões de uma API de repouso com toda a segurança necessária do JWT e toda a segurança necessária oferecida pelas versões mais recentes do .NET.

Os desenvolvedores front-end e back-end do Moralar poderão consultar nosso Moralar **Swagger** para facilitar o consumo de nossos serviços https. Mais adiante, mostraremos como acessar essa excelente ferramenta.

MongoDB: a API do Moralar usa o MongoDB para chamar dados não relacionados.

1.1 Verificação da raiz do projeto

a. Opção A:
   - Clone o projeto que já foi passado para o repositório, acesse a ramificação "Development" e entre na raiz do projeto, onde a pasta se chama ***moralar-api***.

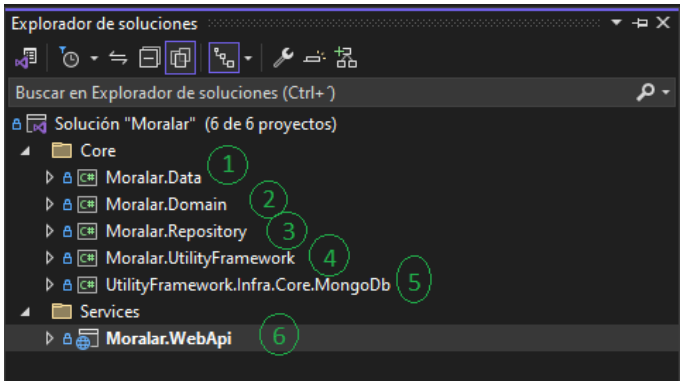
   Opção B:

   - Acesse a pasta inputs e entre na pasta ***moralar-api,*** dentro dessa pasta estará a raiz do projeto.

b. Depois de sabermos qual é a raiz do projeto, precisamos verificar se o arquivo README.md está dentro dessa pasta, para sabermos que é a raiz do projeto, o que é essencial ter em mente ao executar o projeto localmente.

1.2 Arquitetura básica do projeto

Nesta seção, discutiremos como o projeto é montado para facilitar a compilação e a implantação da API.



1. **Dados:** camada de modelo que contém as entidades.
2. **Domínio:** contém serviços e classes reutilizáveis que usamos como modelos, mas essas classes servem para mapear de entidades a objetos simples.
3. **Repositório:** camada que funciona como um mapeador de tabelas para facilitar a conexão com essa tabela por meio de um repositório e acessar os dados.
4. **UtilityFramework:** é uma biblioteca interna que oferece um número infinito de funcionalidades reutilizáveis de todos os tipos.
5. **MongoDb:** usado para se conectar ao banco de dados e conter configurações de conexão personalizadas.
6. **WebApi:** essa é a camada principal do projeto, sendo o controlador que fornece as informações ao consumidor do serviço. Ela também é responsável pelo gerenciamento da segurança da API, controla quais funções podem acessar qual serviço e muito mais.

2. Como executar o programa localmente

Nesta seção, falaremos sobre como executar nosso aplicativo angular em um computador Windows local, levando em conta que ele deve estar em conformidade com o conhecimento anterior (guia básico).

2.1 Requisitos antes da instalação

Você precisa ter essas ferramentas ou programas instalados ou configurados para poder compilar e implementar corretamente.

a. **Visual Studio:** qualquer versão compatível com download do .net8.
b. **Instalador do Visual Studio:** o Visual Studio deve estar ativado para aceitar o manuseio da API da Web (guia básico).
c. Tenha em mãos as credenciais do SMTP para o qual você deseja enviar e-mails, como o host, a porta e, se ele lidar com ssl, o e-mail e a senha (guia do gmail).
d. A chave de API do Google Maps deve ser configurada primeiro e a biblioteca "Places" deve ser ativada no gerenciador de API do Google Maps (guia).

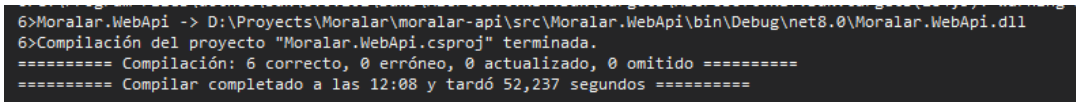
2.2 Conhecimento prévio

1. Ter conhecimentos básicos de manuseio e compilação do Visual Studio para aplicativos Web (guia básico).
2. Ter conhecimento básico de como fazer publicações no Visual Studio (guia básico).
3. Ter conhecimento básico sobre como implantar um WebApp no Azure (guia).
4. Ter permissões para acessar o aplicativo Web do Azure e poder fazer algumas configurações que veremos mais tarde (Portal do Azure).
5. Saber como reconhecer erros de console do tipo CORS, para poder acessar o módulo cors no portal do Azure e adicionar o domínio que apresentou os erros cors.
6. Tenha permissões para acessar o cluster do mongoDb para que possa obter a connectionString e também conceder permissões CORS à nossa API.

3. Compilação do projeto

3.1 Compilar no modo de produção.

- Etapa 1: No Visual Studio, execute as seguintes ações na guia Compile (Compilar):
    - Solução limpa
    - Recompilar a solução
    - Compilar a solução
- Etapa final: verifique se, ao final da compilação, a saída mostra que tudo foi compilado corretamente.



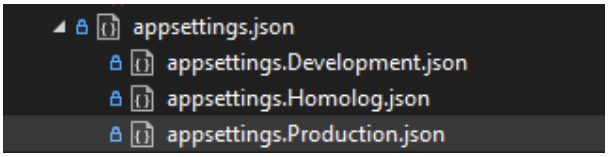
4. Como executar o programa localmente

4.1 Definição de variáveis de ambiente

É crucial que, antes de compilar e implantar, seja necessário verificar se as variáveis de ambiente estão definidas corretamente para que tudo funcione adequadamente.

- Etapa 1: Digite a raiz do projeto ***moralar-api***
- Etapa 2: Vá para o pacote chamado **Moralar.WeApi** e procure o arquivo chamado ***appsettings.json***, onde você encontrará algumas variáveis de ambiente que precisamos modificar.

> **Observação:** explicaremos apenas o arquivo ***appsettings.json***, mas você deve verificar que esse arquivo tem subarquivos que, dependendo do ambiente que você gerencia, precisarão ser modificados em ambos.



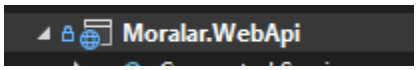
- Etapa 3: Vá até o arquivo e substitua todas as variáveis entre colchetes pelo valor real.
    - ConfigDev (todas as variáveis dentro dessa seção)
    - Config (Todas as variáveis dentro dessa seção)
    - DATABASE (Todas as variáveis desta seção)

- Etapa 4: Certifique-se de que nenhuma variável não esteja configurada e salve as alterações corretamente.
- Etapa 5: Localize o arquivo chamado ***Config.json*** dentro da pasta ***Settings***, onde você encontrará algumas variáveis de ambiente que precisam ser modificadas.
- Etapa 6: Vá até o arquivo e substitua todas as variáveis entre colchetes pelo valor real.
    - googleMapsKey
    - supportEmail
    - SMTP (todas as variáveis dessa seção)
- Etapa final: certifique-se de que nenhuma variável não esteja configurada e salve as alterações corretamente.

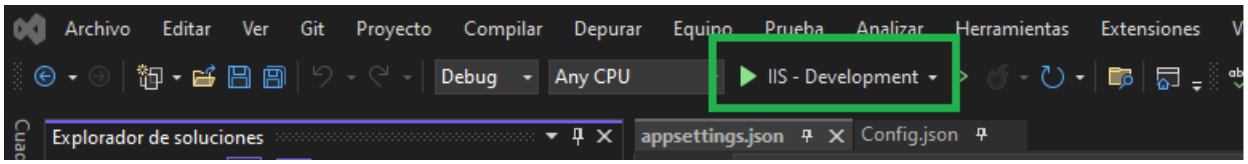
4.2 Execução do aplicativo localmente

É crucial que, antes de compilar e implantar, seja necessário verificar se as variáveis de ambiente estão definidas corretamente para que tudo funcione adequadamente.

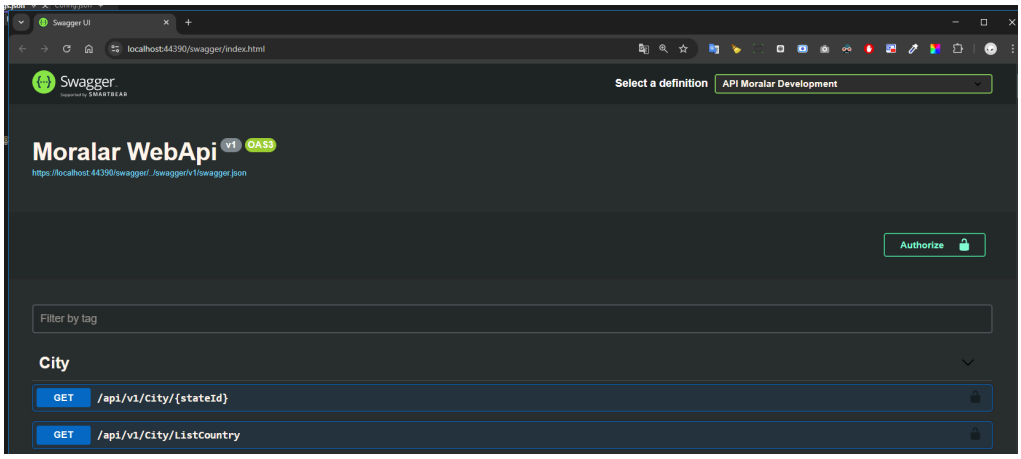
- Etapa 1: Digite a raiz do projeto ***moralar-api***.
- Etapa 2: clique com o botão direito do mouse no pacote **Moralar.WebApi** e certifique-se de que o pacote seja o projeto de inicialização.



- Etapa 3: Compile o projeto, caso ainda não o tenha feito.
- Etapa 4: Executar o perfil **IIS -Development**



- Etapa final: Você verá que agora está no modo depurador e poderá testar suas alterações no computador local.



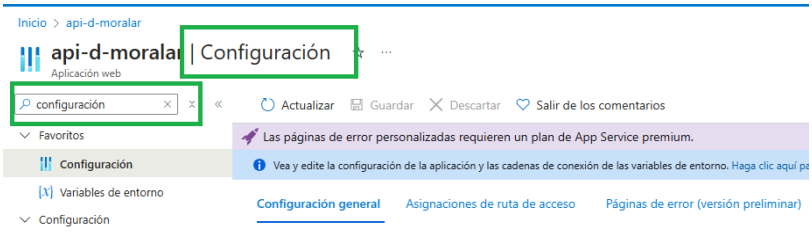
5. Implementação da API

5.1 Configuração do aplicativo Web no Azure

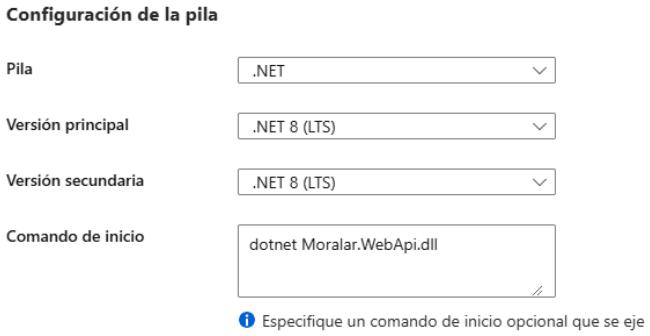
Antes da implantação, queremos tornar o ambiente do Azure adequado para o funcionamento da API Moralar. Você só precisa fazer essa configuração uma vez. Siga estas instruções para que os aplicativos da Web e móveis possam acessar nossa API sem restrições.

A. Verifique a configuração do aplicativo da Web

- Etapa 1: Vá para a seção **Configuração -> Configurações gerais** no Azure.



- Etapa 2: verifique se as configurações a seguir são semelhantes:
  - Configuração da pilha:

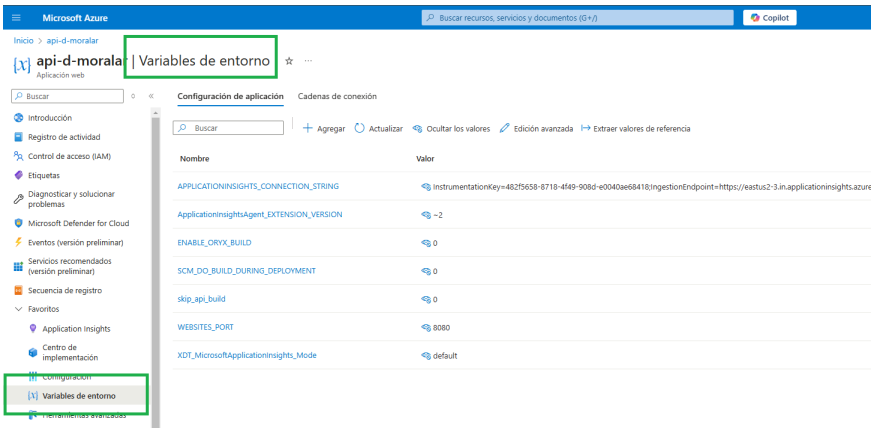


- Configuração da plataforma:

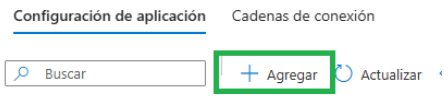
- Etapa final: verifique se a configuração foi salva corretamente.

B. Configurar variáveis de ambiente no Azure

- Etapa 1: Entre na seção **Environment Variables (Variáveis de ambiente)** no Azure.



- Etapa 2: Adicione essas opções se você não as tiver:

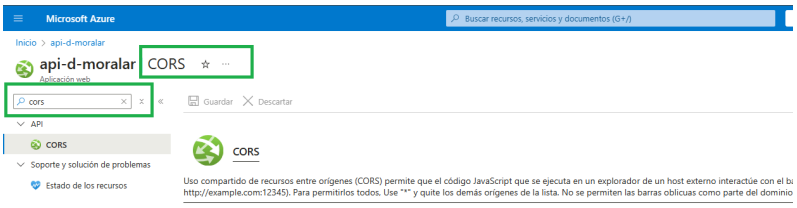


  - ENABLE_ORYX_BUILD: 0
  - SCM_DO_BUILD_DURING_DEPLOYMENT: 0
  - skip_api_build: 0
  - PORTA_DO_SITE: 8080

- Etapa final: verifique se a configuração foi salva.

C. Configurar o Cors

- Etapa 1: Faça login na seção **CORS** no Azure.



- Etapa 2: vá para a seção **Allowed Sources (Fontes permitidas)** e adicione os seguintes valores e substitua as variáveis entre colchetes pelos valores reais:
    - http://localhost:4200
    - http://localhost
    - *
    - {url do site administrativo da moralar}
    - {ip do cluster mongoDb onde o banco de dados moralar está localizado}
    - {https://{user-cluster-mongo-db}:{password}@{host-of-closter} exemplo: "https://nicolasviviescas:AZdKh1p4U9xyudm5@cluster-sinapsis.eeguq.mongodb.net".

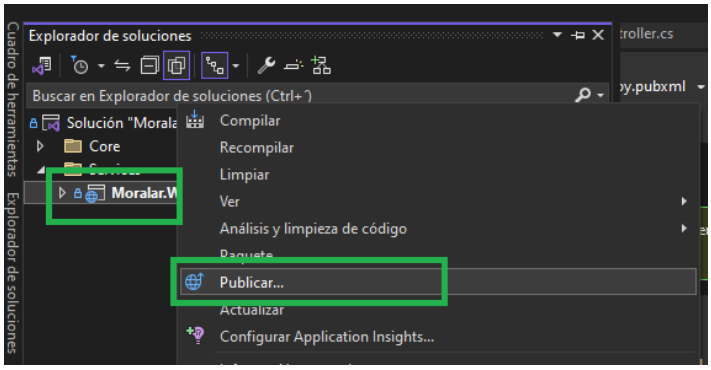
  Observação: o mais importante é o asterisco porque o cluster mudará os ip's e não queremos que a conexão com o banco de dados a partir da API falhe.

- Etapa final: certifique-se de salvar os valores corretamente.

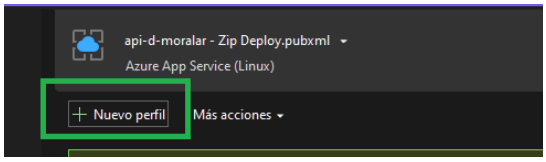
5.2 Implantar na nuvem do Azure

Depois de compilarmos nosso aplicativo e configurarmos as variáveis de ambiente em nosso projeto, podemos implementar nossas alterações na nuvem.

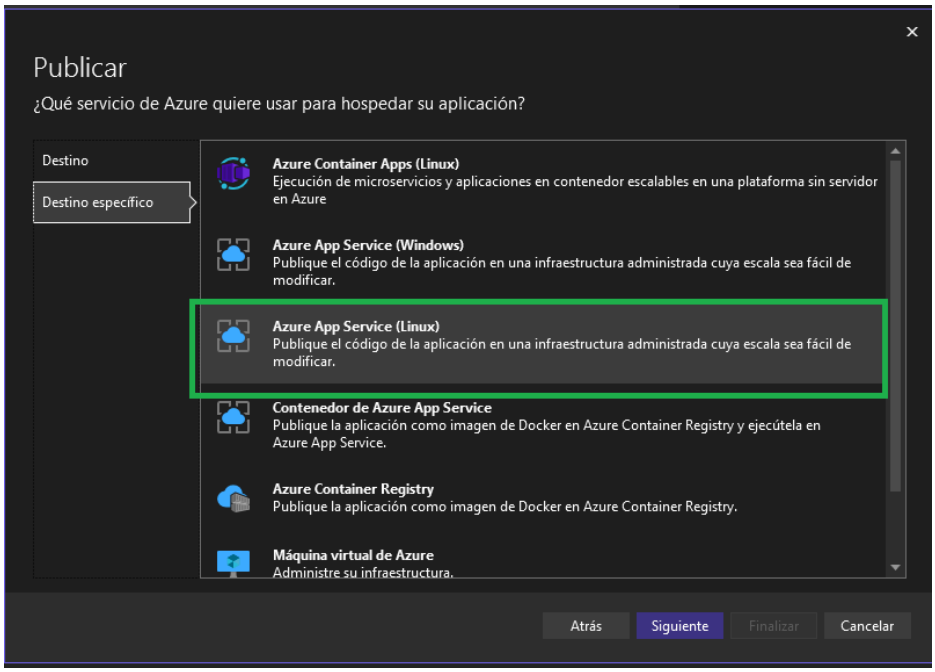
- Etapa 1: esteja dentro da raiz do projeto no Visual Studio e, em seguida, clique com o botão direito do mouse no pacote **Moralar.WebApi ->** Publish.



- Etapa 2: Vamos adicionar um novo perfil, caso não tenhamos configurado onde implantar a partir da publicação.

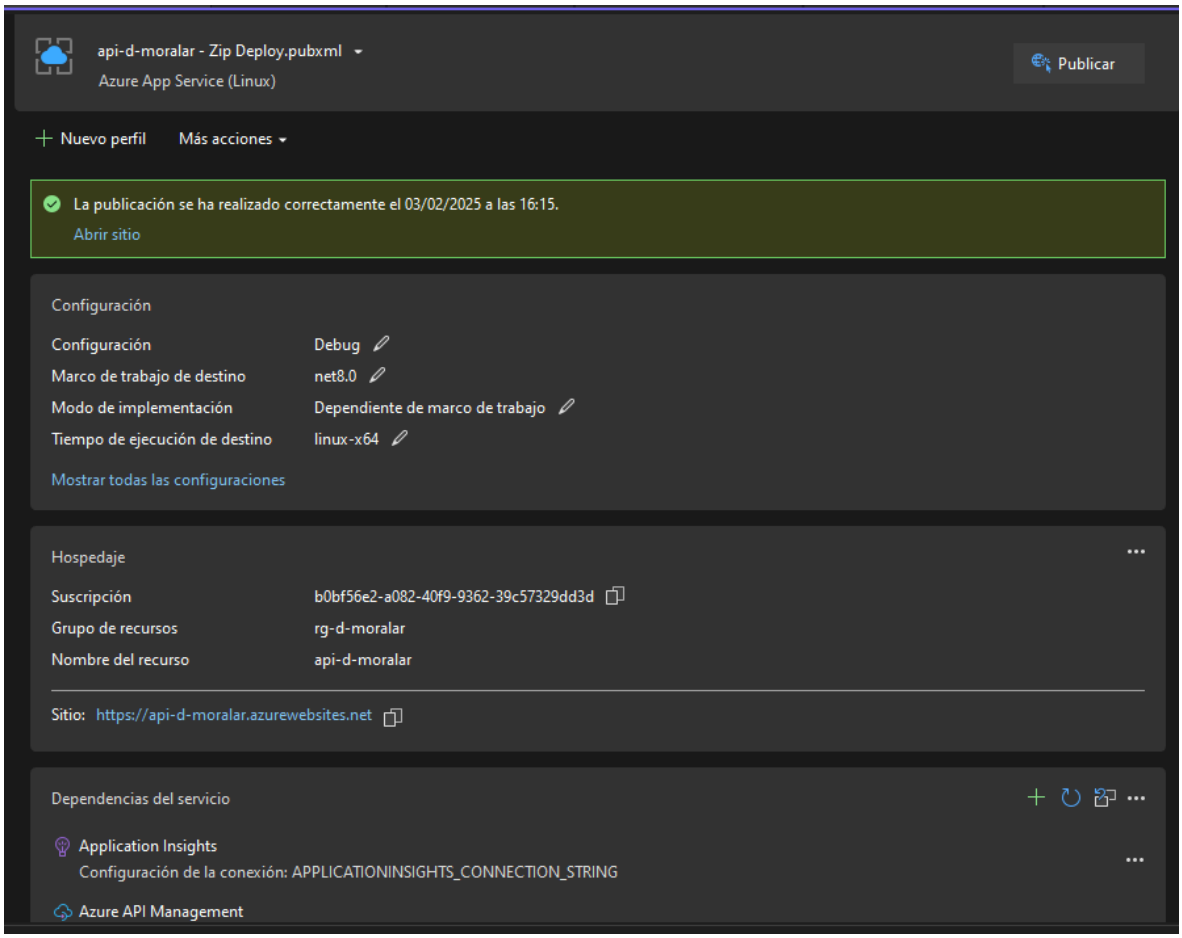


- Etapa 3: selecione a opção azure e, em seguida, next (próximo) e escolha essa opção, dependendo do fato de o servidor de serviço de aplicativo ser Linux ou Windows:

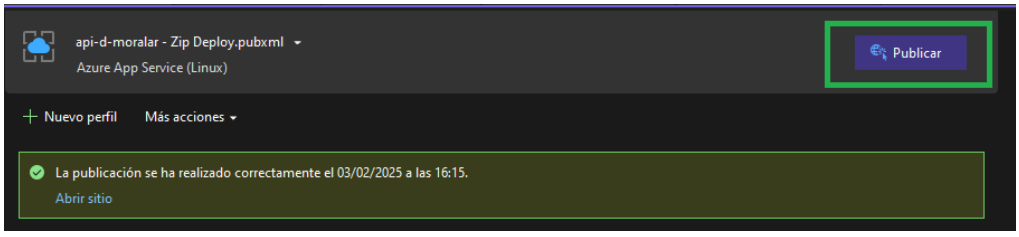


- Etapa 4: Faça login com suas credenciais do Azure e selecione o serviço de aplicativo no qual deseja implantar.

- Etapa 5: Conclua a configuração e você estará pronto para publicar. A aparência será semelhante a esta.



- Etapa 6: Clique no botão publicar e, depois de concluído, será aberta uma janela pop-up do site.



- Etapa final: depois que ele o levar a essa página, adicionamos o caminho "swagger/index.html", algo como "{url of the api}/swagger/index.html" e, se você vir que usamos o swagger, tudo correu muito bem.