Azure - Web Aplications

Rodrigo Machado

Esse artigo tem como objetivo apresentar as principais ferramentas do Azure. Vale ressaltar ainda que toda a documentação do azure está no site da Microsoft e que lá está cada uma das explicações para cada tipo de projetos. O objetivo desse artigo é fazer um resumo de cada um dos serviços que utilizo/utilizei, assim como formas de cria-los e consumi-los.

1 App Static Web

Assim como em outros serviços de hospedagem, o azure permite o deploy de arquivos estáticos para ficarem disponíveis na internet. Esse acesso pode ser dado de forma gratuita ou não, dependendo da disponibilidade desejada e o plano escolhido na criação da aplicação.

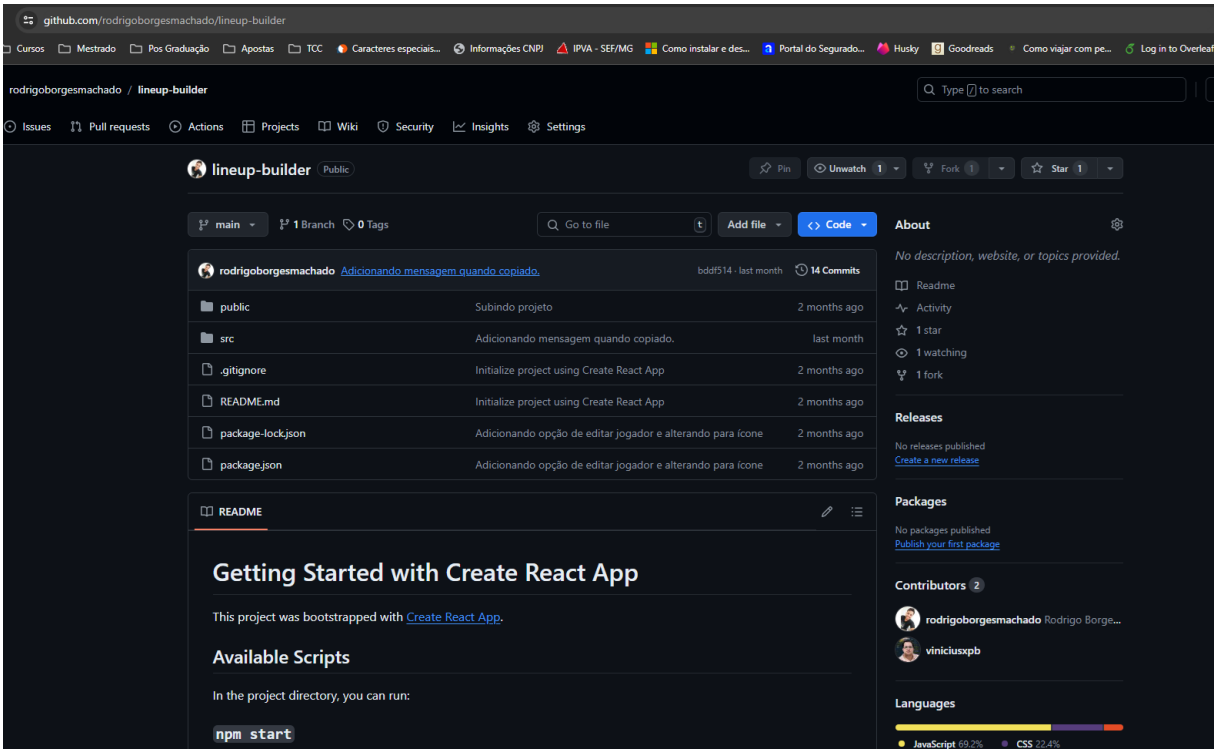
Primeiro vamos aos conceitos: arquivos estáticos são arquivos que não são alterados na comunicação entre servidor ou cliente, ou seja, da mesma forma que o arquivo está no servidor ele será transferido para o cliente. Arquivos html, css, javascripts, imagens, pdfs, mp3 e mp4 são exemplos de arquivos estáticos. Quando feito o deploy desses arquivos no azure é possível acessar os mesmos pela url que o próprio Azure disponibiliza, colocando o caminho do arquivo.

Como exemplo de um novo site utilizaremos um site desenvolvido em React. Para a criação do projeto no azure sugiro que o site já esteja comitado no git, assim você poderá configurar o CI/CD de forma automática com o Azure. Para a criação é necessário atribuir qual a organização e outras configurações, provavelmente você estará usando conta pessoal, então selecione sempre a primeira opção, assim ficará atribuído ao seu usuário.

Vamos ao passo a passo:

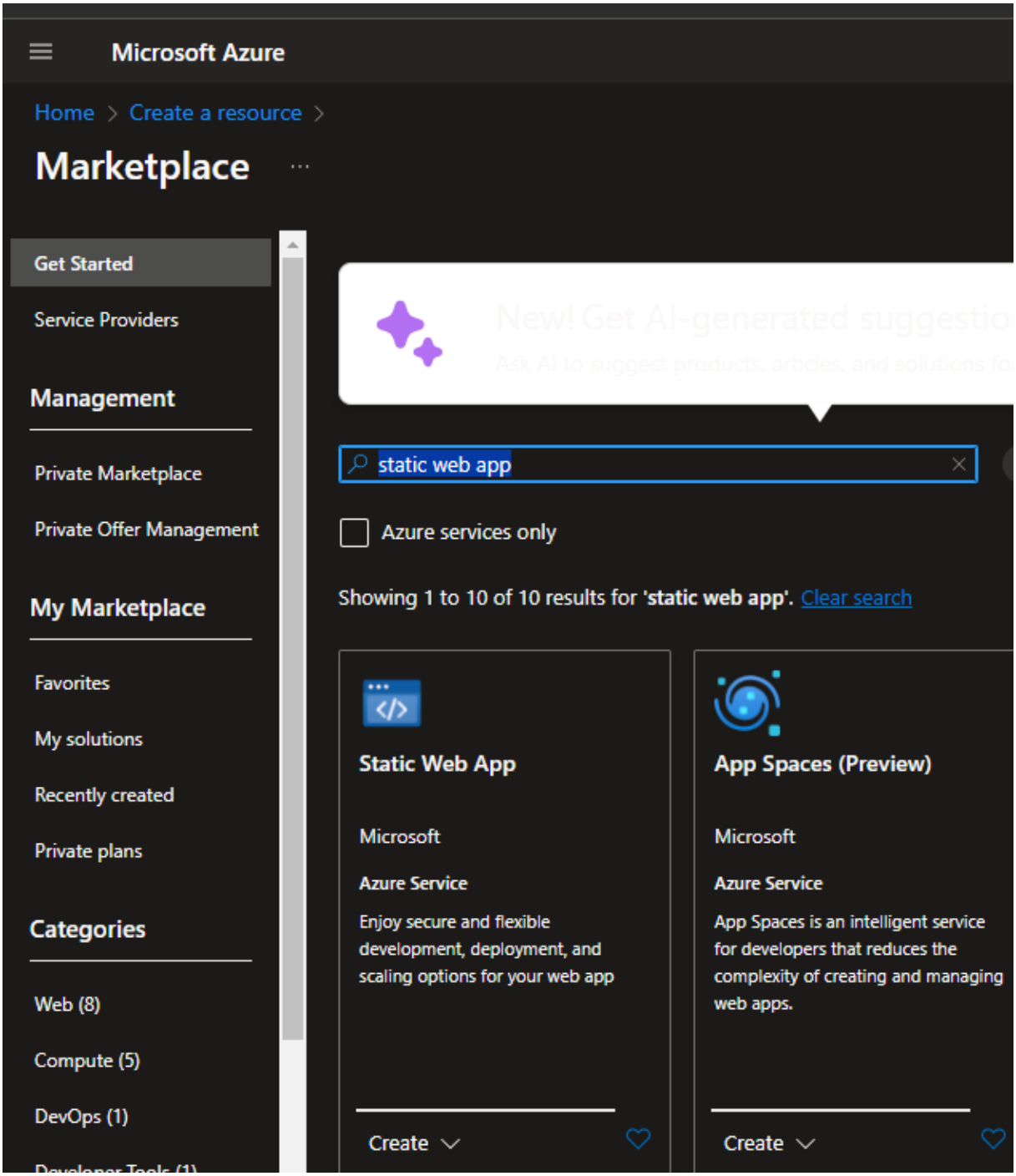
1.1 Criação do projeto e disponibilização no git hub

Antes da criação da app static no Azure vamos criar nosso projeto estático, aqui escolhi um projeto que já tenho e que disponibilizei a primeira versão no netlify, sendo assim o projeto já está no git. Crie seu projeto, faça o deploy para o git e vamos para a próxima etapa.



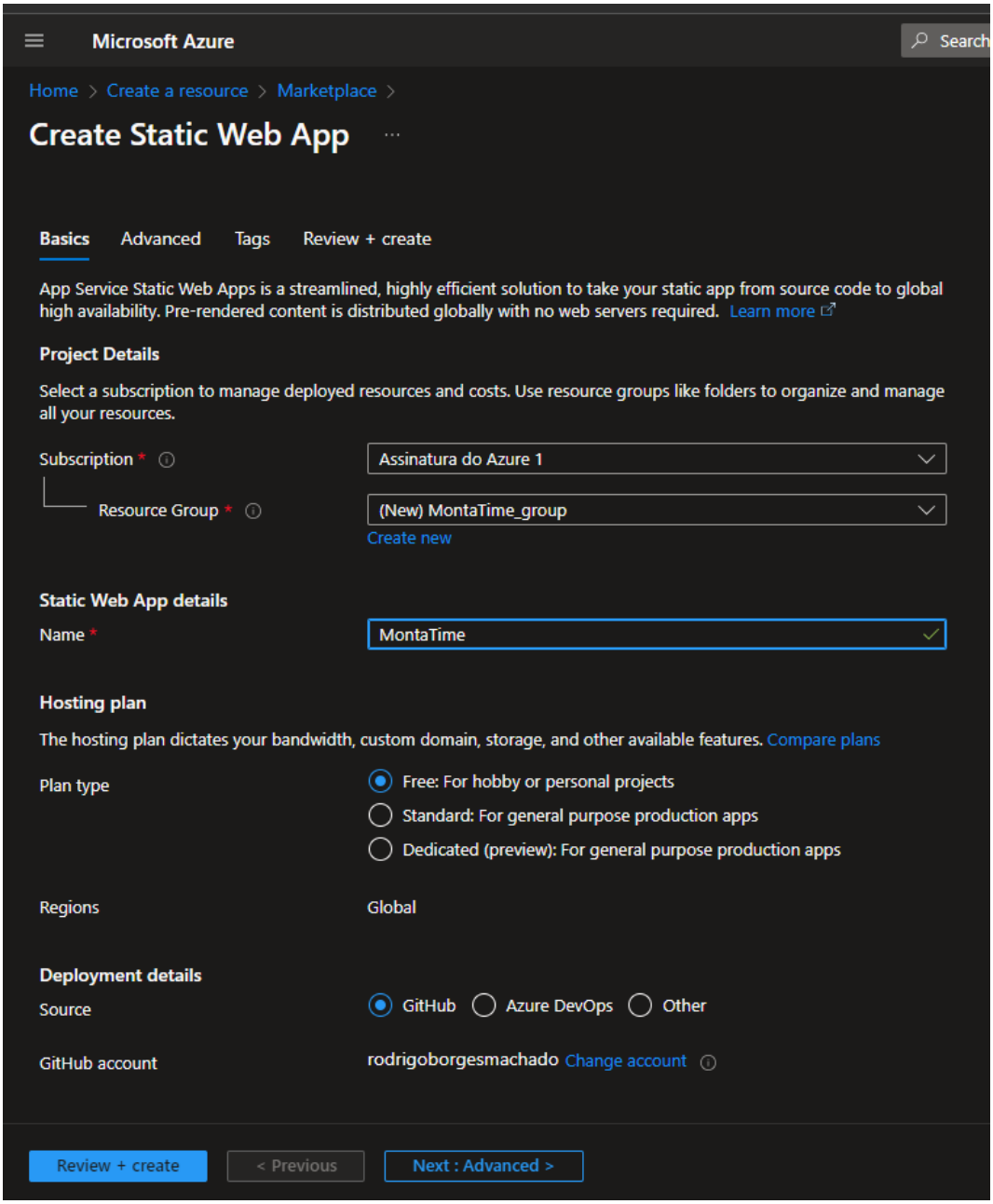
1.2 Criação do App Static no Azure

Para criação do projeto basta acessar o portal azure, ir em Create a Resource, filtrar por Static Web App e selecionar a opção Create.


Microsoft Azure
Home > Create a resource >
Marketplace
Get Started
Service Providers
Management
Private Marketplace
Private Offer Management
My Marketplace
Favorites
My solutions
Recently created
Private plans
Categories
Web (8)
Compute (5)
DevOps (1)
static web app
Azure services only
Showing 1 to 10 of 10 results for 'static web app'. Clear search
Static Web App
Microsoft
Azure Service
Enjoy secure and flexible development, deployment, and scaling options for your web app
Create
App Spaces (Preview)
Microsoft
Azure Service
App Spaces is an intelligent service for developers that reduces the complexity of creating and managing web apps.
Create

1.3 Configuração inicial

Selecione o nome do projeto e vincule o git com o azure, assim é possível selecionar qual o repositório. no formulário abaixo terá qual a assinatura do azure, assim como qual a organização, posteriormente terá a opção de selecionar qual o repositório. Essa configuração é bem simples, dê acesso ao git para o azure e publique a aplicação clicando em Review and Create.

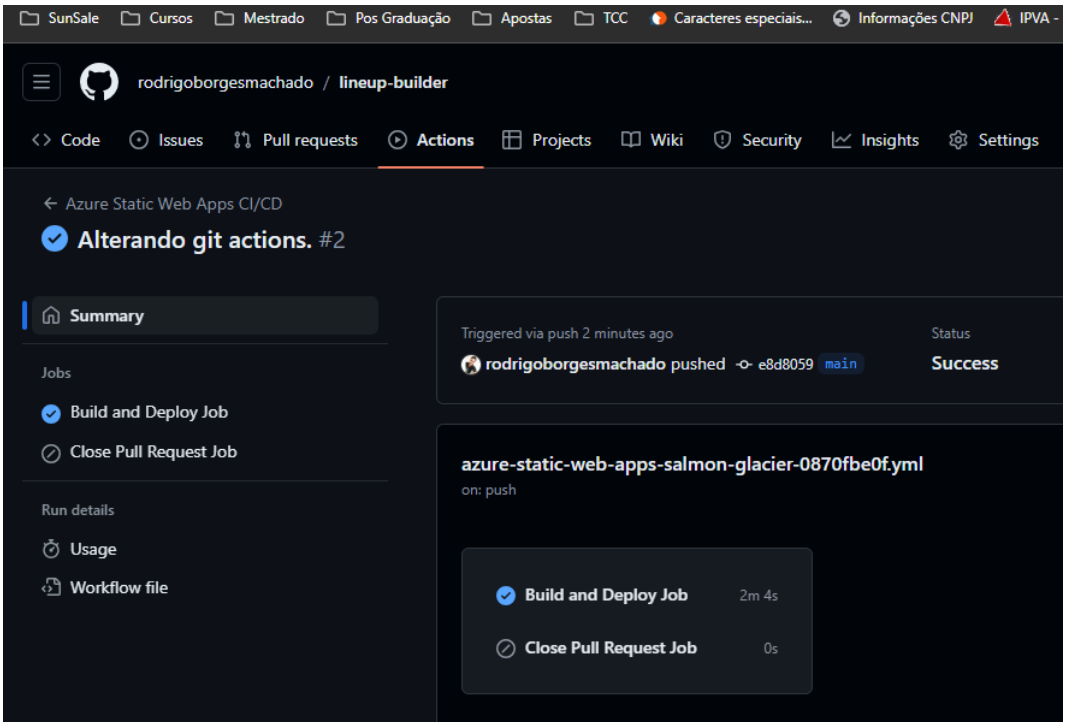


1.4 Configuração CI/CD

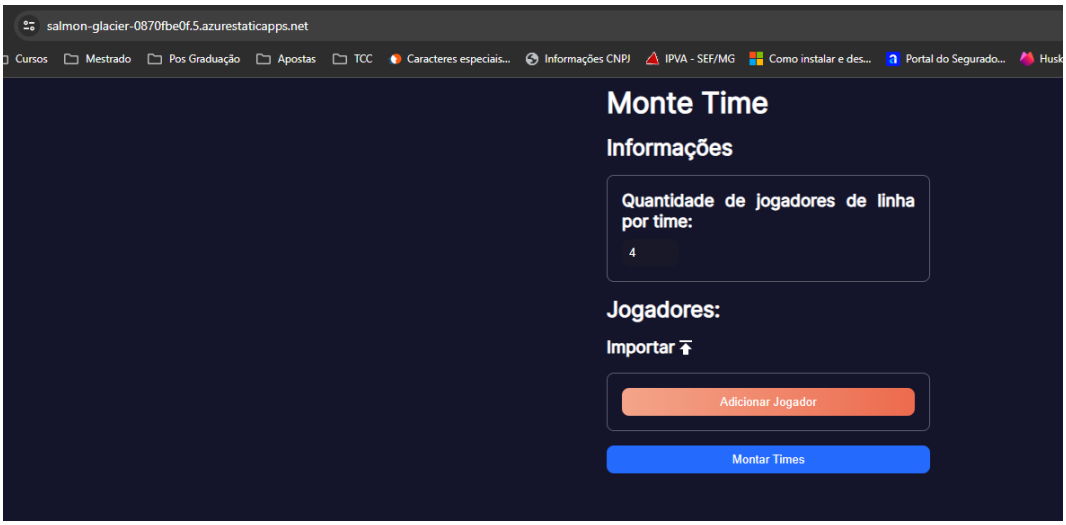
Assim que o recurso for criado ele irá acessar o repositório e adicionar um git actions no seu repositório, assim sempre que houver algum commit na branch escolhida será feito deploy de forma automática. Nesse processo é necessário alterar o arquivo de deploy do git actions para ele não validar o CI (se ele validar e houver qualquer warning o deploy não é feito). Para fazer isso é necessário alterar o arquivo em .githubworkflows.

```
12  jobs:
13    build_and_deploy_job:
14      if: github.event_name == 'push' || (gitH
15      runs-on: ubuntu-latest
16      name: Build and Deploy Job
17      env:
18          CI: false
19      steps:
20        - uses: actions/checkout@v3
21          with:
22            submodules: true
```

Depois de fazer essa alteração e comitar o projeto deve ser publicado de forma correta:



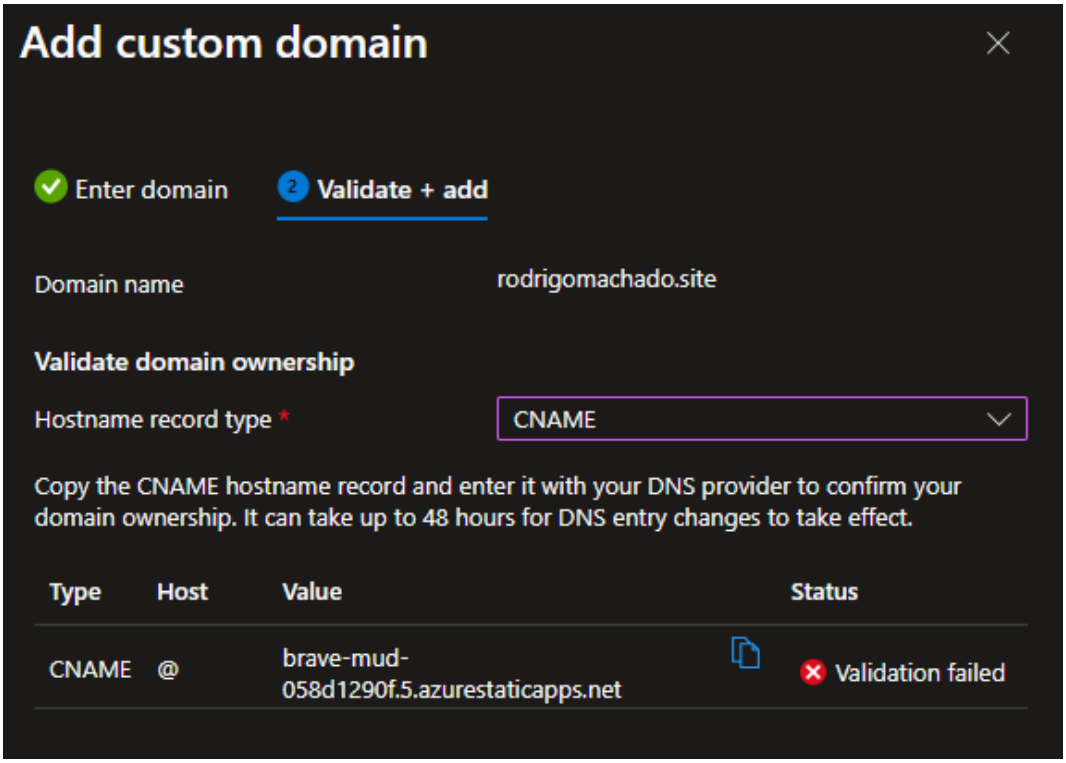
Após a publicação basta acessar conforme link disponibilizado pelo azure e pronto, seu site está hospedado no azure e disponível na internet.



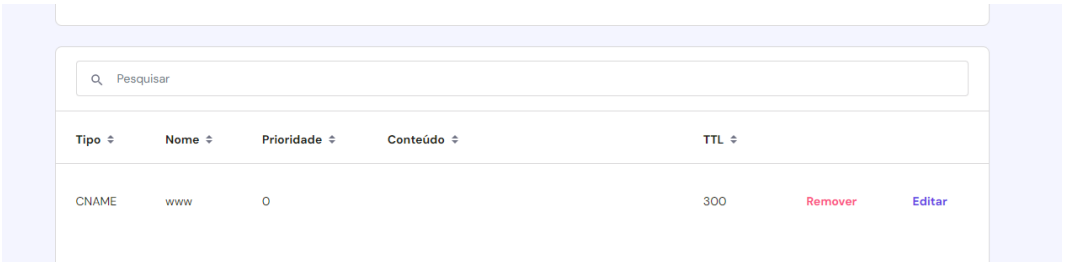
1.5 Configuração de Domínio

Após fazer a publicação dos arquivos estáticos você pode também configurar um nome de domínio para seu site. Para isso é necessário comprar um domínio e depois vincula-lo ao site. O próprio Azure possibilita venda de domínios, mas para isso é necessário ter configurado um dns no azure. Vamos utilizar a hostinder nesse exemplo para comprar o domínio e configurar o DNS.

Para adicionar o domínio vá em Custom domains, Add -¿ New Custom Domain on Other DNS. Após selecionar o nome clique em Next. Na próxima página você fara a configuração do DNS, para isso selecione type CNAME. Gere o código e tome cuidado para copiar as informações de forma correta nesse processo para fazer o apontamento de forma correta.



Para configurar o DNS basta selecionar o DNS e editar a linha que tem o CNAME e WWWW, ele deve apontar para achave que foi copiada no azure.



Após configurar basta selecionar Add, ele demorará certo tempo para configurar, mas assim que finalizar seu site estará hospedado no Azure e configurado com seu próprio custom domain.

1.6 Observações

O hosting no azure, assim como nos demais serviços para publicação, tem suas especificações em relação ao hosting da aplicação. No azure é necessário a presença do arquivo routes.json que configura onde as solicitações devem ir. Sem a configuração desse arquivo, as urls atuam como diretório de arquivo, utilizando o routes.json é possível definir onde enviar a informação

da requisição, como o exemplo a seguir:

2 Services Bus

Os services bus é um componente fundamental de uma arquitetura de commputação distribuída, provendo uma escalável infraestrutura de messageria para comunicação entre diferentes aplicaçãoes, serviçõe ou componentes do sistema. Essencialmente, um service bus atua como um comunicador intermediário, facilitando comunicações baseadas em mensagens assíncronas entre vários componentes. Seus principais conceitos estão: **mensageria**, **decoupling**, **escabilidade** e **padrões de mensagens**

- Comunicações Assíncronas: Services Bus suportam mensagens assíncronas, onde mensagens são enviadas e recebidas independentemente de cada uma. Essa natureza assíncrona permite sistemas lidar variando com o tempo de processamento de uma mensagem, instabilidade de conexão e alto volume de mensagens de forma efetiva.
- Decoupling: os services bus asseguram reabilidade na entrega de mensagens disponilizando funções como persistência de mensagens, mecanismos de retentativa e tratativas de erros. Mensagens são armazenadas enquanto elas não forem processadas, minimizando a perca de informações.
- Escabilidade: Services bus são desenhados para escalarem de forma horizontal para lidar com altos volumes e crescer de acordo com a necessidade. Eles conseguem distribuir mensagens entre vários processos ou instâncias para chegar em alto processamento e escabilidade.
- Padrões de Mensagens: Service bus suportam vários tipos de padrões de mensagem, incluindo filas (queues), publish/subscribe (topics), request/response, e event-driven messaging. Serão listados a seguir cada um deles

Para criar um Service bus é necessário criar um próprio recurso para ele no Azure. Criando um próprio recurso Service Bus você deverá também dar um nome ao namespace para comunicação, além de configurar o que será utilizado: Queues, Topics or Subscriptions.

Uma vez criado seu recurso é possível conectar no mesmo, é possível usar o Azure AD (Azure Active Directory), políticas de acesso e chaves de acesso para gerênciar autenticidade e autorização.

2.1 Services Bus Queues

Utilizar os services bus para consumir filas (queues) é uma boa opção pra processamento de grande massa de dados. Para entender melhor o processamento, é necessário primeiro entender o conceito da fila: **o primeiro que entra é o primeiro a ser atendido**.

O serviço funciona então de duas formas, em um ponto está alguém alimentando a fila (normalmente algum serviço web) e do outro lado alguém consumindo essa mesma fila (pode er qualquer tipo de serviço).

Para exemplificar vamos pegar o seguinte cenário:

1. precisamos processar um json de 2 gb que contém a informação de novos preços de todos os produtos de um ecommerce
2. a api que faz o processamento dos dados não pode ficar sem responder durante o processamento independente do quão grande for a requisição

Para conseguir atender às necessidades então o simples seria utilizar o serviço de fila, onde a API irá colocar uma nova tarefa e algum serviço escutará à essa fila. Nesse caso, é possível utilizar a **Standard Queue** do Azure, que dispnibiliza o serviço de messageria necessário, e um service bus escutando essa mesma fila.

A API então quando receber a informação pode envia-la direto para a fila do azure de forma assíncrona, assim o service bus quando escutar que há nova informação a ser processada pegará a mesma para processa-la. Dessa forma o processamento do json de 2gb ficará no lado do service bus, sem comprometer o processamento da api.

Ademais, as filas possibilitam ainda que seja criado o "Dead Letting" que é basicamente um fila à parte que armazena mensagens que foram expiradas. Dessa forma é possível verificar essas filas também para não perder informações e garantir que todas as mensagens foram processadas.

2.2 Service Bus Topics

Os topics proporcina uma espécie de mecanismo de publish/subscribe para receber e enviar mensagens para vários serviços ao mesmo tempo. São úteis em cenários onde multiplos componentes ou serviços precisam receber a mesma mensagem. Assim, aplicações webs podem fazer broadcast de notificações, atualizações e eventos para os subscribers interessados.

A grande diferença entre os Topics e as Queues é que os topoics todos os subscribers recebem a notificação, enquanto isso a fila apenas um dos serviços que estão escutando à aquela fica irá escuta-lo. Em restante aos outros aspéctos se mantém, toda a parte de segurança e garantia de recebimento e envio das mensagens, assim como o dead lettering.

2.3 Dead Letter Queue (DLQ)

A dead letter queue é basicamente uma fila à parte da queue ou topics onde ficam armazenados os itens que foram expirados e não foram processados. Há a possibilidade de colocar o tempo que quiser para processar uma mensagem, logo, pode haver expirações, essas mensagens expiradas iriam para essa fila. Ou mesmo mensagens que não foram processadas com sucesso, assim fica fácil de manter e ter controle de erros.

3 Azure Functions

As funções do azure é um serviço computacional oferecido pela Microsoft que permite o usuário executar código a partir de vários eventos sem a necessidade de uma infraestrutura pronta. Com o azure functions você cria o código que deseja que seja executado e configura qual o evento necessário para essa função executar, como um cron time. Além disso as functions permitem várias linguagens, tais como c#, javascript (Node.js), Python, PowerShell, Typescript.

As funções do azure são guiadas por eventos, isso abre a possibilidade para vários serviços do azure poderem rodar tais funcions, como Azure Storage, Azure Service Bus, Azure Event Hubs, Azure Cosmos DB, HTTP requests, timers etc. Existe a possibilidade ainda de criar eventos customizados com Azure Durable Functions ou Azure Event Grid.

Além disso as funções tem uma variedade de formas de deploy, incluindo docker (AZure Kubernetes Services). É possível fazer deploy das mesmas direto pelo Visual Studio (e o Code), Azure CLI, Azure DevOps ou usando CI/CD.

Ademais é possível fazer integração das funções com outros serviços do azure. É possível

por exemplo criar uma function que escute uma fila para executar, ou mesmo que execute a partir de uma chamada http com entrada possível.