## Dapr Service Invocation: Mikro Servisler Arası İletişim Kolaylığı

## Bildiğiniz üzere mikro servis mimarileri, uygulamaları küçük, bağımsız servislere bölmeyi ve her bir servisi ayrı bir uygulama olarak çalıştırmayı benimsemektedir. Ancak, bu da mikro servisler arasında iletişim ihtiyacını beraberinde getirir. Bu iletişimi sağlayabilmek için çeşitli zorluklarla ile karşılaşıyoruz. Dapr (Distributed Application Runtime), bu sorunu çözmek için geliştirilmiş bir çerçevedir ve bu makalede, Dapr‘ın yapı taşlarından biri olan Service Invocation hakkında konuşacağız.

## Nedir?

## Service Invocation, mikro servisler arasında iletişimi kolaylaştırmak için Dapr tarafından sunulan bir özelliktir. Bu, bir servisin başka bir servisi çağırmasını veya veri almasını sağlar. Service Invocation, istemci tarafında veya sunucu tarafında kullanılabilir ve aynı dilde yazılmış olmaları gerekmez. Bu, farklı dillerde ve platformlarda çalışan servisler arasında iletişimi hızlı ve güvenilir hale getirir.

## Nasıl Çalışır?

## Dapr Service Invocation, iki servis arasındaki iletişimi basit ve güvenli bir şekilde sağlar. Bu iletişimdeki temel adımları ise şunlardır :

## Servisin Kaydedilmesi: Her servis, Dapr'a kaydedilir. Kaydın ardından servis, bir kimlik (örneğin, ServisA) alır.

## İstek Gönderme: Bir servis, başka bir servisi çağırmak istediğinde, çağrı yaptığı servisin kimliği (örneğin, ServisB) ve hedef yöntemin adını belirtir.

## İstek İşleme: Dapr, isteği alır, hedef servisi ve yöntemi belirler ve isteği ileterek işlemi gerçekleştirir.

## Sonuç Döndürme: Hedef servis, işlemi tamamladığında sonucu geri döner.

## Sonuç Alma: Çağıran servis sonucu alır ve işlemi tamamlar.

## A diagram of a data processing process Description automatically generated

## Avantajları

## Dapr Service Invocation, mikro servis mimarilerinde birçok avantaj sağlamaktadır:

## Dil ve Platform Bağımsızlığı: Dapr, farklı dillerde ve platformlarda yazılmış servisler arasında iletişimi kolaylaştırır.

## İzleme ve Loglama: Dapr, iletişim trafiğini izlemek ve hataları günlükleme yeteneği sunar.

## Zamanlama ve Yeniden Deneme: İletişim sırasında zamanlamayı ve otomatik yeniden denemeyi yönetir, bu da uygulamanın daha güvenilir olmasını sağlar.

## Güvenlik: Dapr, iletişimi güvence altına almak için çeşitli güvenlik önlemleri sunar.

## Kod Tekrarını Önleme: Servisler arasındaki iletişimi kolaylaştırırken, kod tekrarını azaltır.

## Örnek:

## A diagram of a company Description automatically generated

## NodeJs uygulamasının nodeapp adında bir uygulama kimliği vardır. Pyhton uygulaması bu uygulamanın <http://localhost:3500/v1.0/invoke/nodeapp/method/neworder> yöntemine bir POST istediğinde bulunur.

## Dapr ‘ın kendi bileşeni olan name resolver component sayesinde nodejs uygulamasını konumunu keşfeder.

## Dapr isteği bu konuma iletir.

## Node’js uygulamasının dapr sidecar’ı isteği alır ve işler. Bu esnada bu isteği loglayarak kaydeder.

## NodeJs uygulaması kendi sidecar’ına cevabı iletir.

## Python uygulamasının sidecar’ına gönderir.

## Python uygulaması yanıtı alır.

## Sonuç

## Dapr Service Invocation, mikro servislerin iletişimini basit ve güvenilir hale getirir. Bu, modern uygulama geliştirme projelerinde büyük bir avantaj sağlar ve bizlere farklı dillerde ve platformlarda çalışan servisler arasında etkili iletişim kurmamızı kolaylaştırır.

**Kod Örneği:**

Aşağıdaki gibi 2 adet .net mikro servis oluşturalım.

A screenshot of a computer

Description automatically generated

Product mikroservisimiz için bir endpoint tanımlıyoruz ve döneceği response için statik bir ekran yazdırma işlemi gerçekleştiyoruz.

A screenshot of a computer

Description automatically generated

Product mikroservisimiz içinde tanımladoğımız products endpoint ini çağıracak Simulation mikroservisimizi aşağıdaki gibi şekillendiriyoruz.

A screenshot of a computer

Description automatically generated

Yukarıda gördüğünüz gibi product mikroservisine ait products end point bir çağrı da bulunuyoruz ve bunu ekrana yazdırıyoruz.

Şimdi iki mikroservisimizi Dapr ile nasıl çalıştıracağımızı inceleyelim.

Product mikroservisimizi için bir powershell pencerisi açarak aşağıdaki komutu çalıştıralım

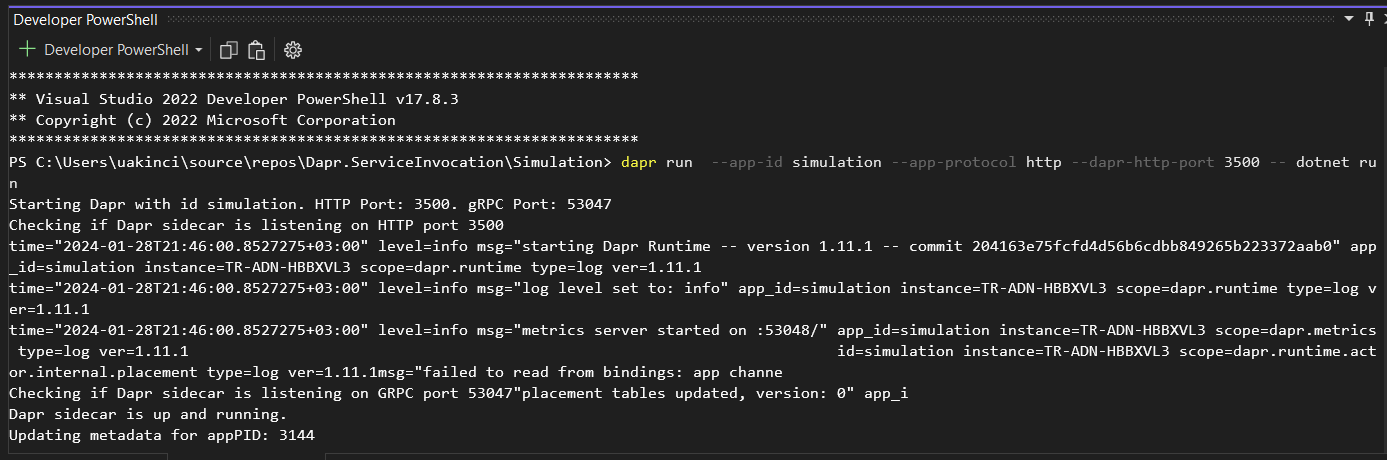
**dapr run --app-port 7001 --app-id product-service --app-protocol http --dapr-http-port 3501 -- dotnet run**

A screenshot of a computer program

Description automatically generated

Ardından product mikroservisimize istekte bulunacak Simulation mikroservisimizi çalıştıralım.

**dapr run --app-id simulator --app-protocol http --dapr-http-port 3500 -- dotnet run**



Simulator mikroservisimizin çalışmaya başlamasıyla product mikroservisimize istek atmaya başlıyor ve ekrana çıktıları yazdığını görüyoruz.

A computer screen shot of a black screen

Description automatically generated

