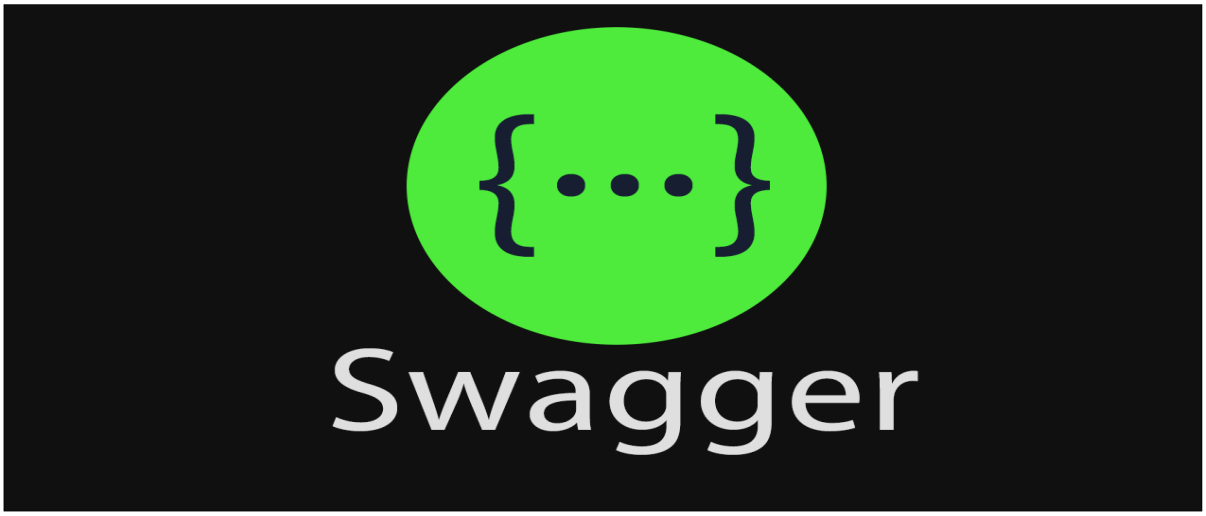
Swagger Nedir

Swagger, özellikle RESTful API'ler için kullanılan açık kaynaklı bir API belirtme ve belgelerleme aracıdır. Swagger, API'lerin nasıl kullanılacağını anlamak ve bu API'lerle nasıl etkileşimde bulunulacağını belgelemek için kullanılır. Swagger, API geliştiricilerinin API'lerini daha anlaşılır hale getirmelerine, kullanıcıların API'leri keşfetmelerine ve kullanmalarına yardımcı olur. Swagger, API'lerin metotlarını, girdi ve çıktı parametrelerini, hata kodlarını ve diğer bilgileri tanımlamak için bir dizi açık standart kullanır. Bu tanımlar, Swagger tarafından otomatik olarak işlenir ve interaktif bir API belgelemesi oluşturmak için kullanılabilir. Bu belgeler, API kullanıcılarının API'nin nasıl kullanılacağına dair kolayca anlaşılabilir bir rehber sunar.



Tarihçe

Swagger, başlangıçta 2011 yılında Tony Tam tarafından oluşturulan bir açık kaynak API belgeleme aracı olarak hayata geçti. Tony Tam, Wordnik adlı bir online sözlük ve dil kaynağı API'si üzerinde çalışırken Swagger'ı geliştirdi. Wordnik'in geliştiricileri, API'lerini daha iyi belgelemek ve kullanıcıların Wordnik API'sini daha kolay bir şekilde kullanmasını sağlamak için bir çözüm arıyordu.

Swagger, API'leri açıklamak için bir standart tanımlama biçimi sunan bir spesifikasyondu. Bu spesifikasyon, API'lerin rotalarını, parametrelerini, yanıt kodlarını ve diğer detaylarını tanımlamak için kullanıldı. Swagger spesifikasyonu, API belgelemesi için bir standart haline geldi ve API'lerin nasıl kullanılacağına dair anlaşılır belgelerin oluşturulmasını kolaylaştırdı.

Swagger, 2015 yılında SmartBear Software tarafından satın alındıktan sonra, OpenAPI Specification adı altında daha geniş bir topluluk tarafından yönetilmeye başlandı. OpenAPI Specification, Swagger spesifikasyonuna dayanan ve daha geniş bir katılımı teşvik eden bir

standart haline geldi. Bu standart, API belgelerinin oluşturulması ve paylaşılmasında yaygın olarak kullanılmaktadır.

Swagger/OpenAPI'nin gelişimi, RESTful API'lerin popülaritesinin artmasıyla paralel olarak devam etti. Geliştiriciler, API'lerini daha iyi belgelemek ve kullanıcıların API'lerini daha kolay bir şekilde kullanmasını sağlamak için Swagger/OpenAPI'nin sunduğu araçları kullanmaya devam etmektedirler.

Swagger Bileşenleri

Swagger Editor

Swagger tanımlama dosyalarını oluşturmak, düzenlemek ve doğrulamak için kullanılan bir çevrimiçi veya yerel düzenleme aracıdır. Swagger Editor, kullanıcıların API tanımlarını yazarken hızlı geri bildirim almasını sağlar. Ayrıca, JSON veya YAML formatında tanımlamaları kolayca oluşturabilir veya düzenleyebilirsiniz. Editör aynı zamanda açıkça hataları tespit eder ve kullanıcıya doğru çözümleri önerir.

Swagger UI

Swagger/OpenAPI tanımlama dosyalarından otomatik olarak oluşturulan interaktif bir API belgesidir. Swagger UI, API'nin rotalarını, parametrelerini ve yanıtlarını görselleştirir. Kullanıcılar, bu belgeleme aracılığıyla API'nin nasıl kullanılacağını anlamak ve hızlıca denemek için gerçek zamanlı olarak etkileşimde bulunabilirler. Bu, API'nin kullanımını anlamak ve test etmek için kullanıcılara büyük bir kolaylık sağlar.

Swagger Codegen

Swagger Codegen, Swagger/OpenAPI tanımlama dosyalarından başlangıç seviyesi istemci ve sunucu kodlarının otomatik olarak oluşturulmasını sağlar. Bu, geliştiricilere API'yi kullanmak için gerekli olan kodları hızlı bir şekilde üretme yeteneği sunar. Codegen, birçok programlama dilini ve çerçeveyi destekler, böylece geliştiriciler API'ye erişmek için tercih ettikleri teknolojiyi kullanabilirler.

Swagger Hub

Swagger Hub, Swagger/OpenAPI belgelerini paylaşmak, işbirliği yapmak ve yönetmek için bulut tabanlı bir platformdur. Geliştiriciler, Swagger Hub üzerinde API belgelerini oluşturabilir, paylaşabilir, revize edebilir ve yayınlayabilirler. Ayrıca, Swagger Hub, API'lerin takım içinde işbirliği yaparak geliştirilmesine olanak tanır ve API belgelerinin farklı sürümlerini yönetmeyi kolaylaştırır.

Swagger Inspector

Swagger Inspector, API isteklerini test etmek ve belgelemek için kullanılan bir tarayıcı tabanlı

araçtır. Geliştiriciler, bu araç aracılığıyla API isteklerini kolayca oluşturabilir, gönderip yanıtları gözlemleyebilir ve bu işlemleri otomatik olarak belgeleyebilirler. Bu, API'leri hızlı bir şekilde test etmek ve hata ayıklamak için kullanışlı bir araçtır.

API Dökümantasyon Önemi ve Swagger

API'ler kullanılmak amacıyla tasarlandığı için tüketicilerin API'nizi hızlı bir biçimde uygulayabilmesini ve anlaşılmasını sağlamak son derece önemli bir nokta. Bu nedenle ki API dokümantasyonu olmazsa olmaz bir konumda bulunuyor. Tüketici konumunda yer alan bir insan dokümana baktığı zaman "Tanımı nedir, bu servis ne iş yapar, içerisinde ne tür operasyonları barındırıyor, bu operasyonların kullanım biçimi hangi tarzda olmalı, giriş çıkış parametreleri hangi türde?" benzeri soruların kolay bir şekilde cevaplanması akıllarda şüphe bırakmaması gerekiyor.

Eğer geleneksel usul takip edilmek istenmiyorsa biraz daha otomatik tarzda, yani biraz daha basit ve çok yönlü bir dokümantasyon yapılmak istenirse ortaya swagger çıkıyor. Swagger sadece dokümantasyon amacıyla kullanılmıyor.API client kodununda üretimi yapılıyor. Oradan hareketle direkt istek gönderilip sonuçlara ulaşılabiliyor. Postman tarzı bir uygulama kullanılmadan da yapılan iş çözüme kavuşabiliyor.

Swagger Neden Kullanılır?

Swagger, API geliştirme ve belgeleme süreçlerinde kullanılan temel bir araçtır. API'lerin doğru bir şekilde belgelenmesi ve kullanıcılarının API'leri nasıl kullanacaklarını anlamaları, API ekosisteminin sağlıklı bir şekilde işlemesi için hayati öneme sahiptir. Swagger, bu ihtiyacı karşılamak için çeşitli bileşenler sunar.

Öncelikle, Swagger Editor, geliştiricilere API tanımlama dosyalarını (JSON veya YAML formatında) oluşturma, düzenleme ve doğrulama imkanı sağlar. Bu, API'nin yapısını ve işlevselliğini tanımlamak için kullanılır ve daha sonra API belgeleri için bir temel oluşturur. Ardından, Swagger UI, bu tanımlama dosyalarından otomatik olarak interaktif bir API belgesi oluşturur. API'nin rotalarını, parametrelerini ve yanıtlarını görsel olarak sunar, böylece API kullanıcıları API'yi daha iyi anlayabilir ve test edebilirler.

Bunun yanı sıra, Swagger Codegen, API tanımlama dosyalarından istemci ve sunucu kodlarını otomatik olarak oluşturur. Bu, geliştiricilere API'yi kullanmak için gereken kodları hızlı bir şekilde üretme ve kullanma imkanı sağlar. Ayrıca, Swagger Hub gibi platformlar, ekiplerin bir arada çalışarak API belgelerini oluşturmasına ve yönetmesine olanak tanır.

Tüm bu bileşenler, API geliştirme sürecini daha verimli ve etkili hale getirirken, API belgelerinin daha iyi oluşturulmasını, daha kolay keşfedilmesini ve daha hızlı bir şekilde geliştirilmesini sağlar. Böylece, Swagger, API ekosisteminin daha sağlıklı ve verimli bir şekilde işlemesine katkıda bulunur, geliştiricilere daha iyi bir API deneyimi sunar ve genel olarak yazılım projelerinin başarısını artırır.

Swagger Avantajları ve Dezavantajları

Avantajları

- **Dokümantasyon:** Swagger, API belgelerini standartlaştırır ve kullanıcı dostu bir arayüzde görüntüleyerek API'nin anlaşılır ve tutarlı bir şekilde belgelenmesini sağlar. API belgeleri, API'nin hangi operasyonları desteklediğini, hangi parametrelerin kullanılması gerektiğini, hangi cevapların döndüğünü ve diğer teknik detayları içerir. Swagger, belgelerin otomatik olarak güncellenmesini sağlar, böylece API'nin güncel durumunu yansıtır ve kullanıcıların doğru bilgilere erişmesini sağlar.

- **Test etme:** Swagger UI, API kullanıcılarına API'leri test etme imkanı sunar. Kullanıcılar, API belgelerini kullanarak API isteklerini oluşturabilir ve API'nin nasıl çalıştığını görebilir. Bu, API'nin doğru çalıştığından emin olmak için kullanıcı dostu bir arayüzde API'leri test etme ve hataları tespit etme imkanı sunar.

- **Hızlı geliştirme:** Swagger Codegen, API istemcileri ve sunucularını otomatik olarak oluşturarak geliştirme sürecini hızlandırır ve koddaki hataları en aza indirir. Swagger belgeleri, Swagger Codegen kullanılarak farklı programlama dilleri ve çerçeveler için API istemcileri veya sunucuları otomatik olarak oluşturulabilir. Bu, geliştiricilere API'leri hızlı bir şekilde kullanma imkanı sunar ve hataların en aza indirilmesine yardımcı olur.

- **Standartlaşma:** Swagger, API belgelerini standart bir format olan JSON veya YAML formatında tanımlamanıza olanak tanır. Bu, API belgelerini kolayca paylaşmanızı ve diğer geliştiricilerin API'nizi daha iyi anlamalarını sağlar. Aynı zamanda Swagger, API'lerin belgelenmesi için yaygın bir standart olan OpenAPI Specification'ı (eski Swagger Specification olarak da bilinir) kullanır, böylece API belgeleriniz standartlaştırılmış bir format kullanarak oluşturulur ve güncellenir.

- **İletişim ve işbirliği:** Swagger, geliştiriciler, testçiler, ürün yöneticileri ve diğer paydaşlar arasında daha iyi iletişim ve işbirliği sağlar. API belgeleri, API'nin nasıl kullanılacağına dair net bir rehber sunar ve farklı ekipler arasında API'nin kullanımı konusunda ortak bir anlayış geliştirmeyi sağlar. Swagger UI, kullanıcı dostu bir arayüzde API'leri görselleştirdiği için, farklı paydaşların API'nin nasıl çalıştığını daha iyi anlamasını sağlar.

- **Müşteri deneyimi:** API kullanıcıları, Swagger UI aracılığıyla API'leri kolayca test edebilir, API isteklerini doğru bir şekilde oluşturabilir ve cevapları görüntüleyebilir. Bu, kullanıcıların API'nizi kullanırken daha az hata yapmalarını ve daha iyi bir müşteri deneyimi yaşamalarını sağlar. API belgelerinin kullanıcı dostu bir arayüzde sunulması,

API kullanıcılarının daha kolay ve hızlı bir şekilde API'nizi anlamalarını ve kullanmalarını sağlar.

Dezavantajları

- **Karmaşıklık:** Swagger, API belgeleme ve geliştirme için bir dizi araç ve bileşen sunar, ancak bu araçlar bazen kullanıcılar için karmaşık olabilir. Özellikle yeni başlayanlar için Swagger'ı başlangıçta anlamak ve kullanmak zaman alabilir.

- **Bağımlılık:** Swagger, API geliştirme sürecinde kullanıcıların Swagger ekosistemine bağımlı olmasını gerektirir. Bu, bazı geliştiriciler için API tanımlama dosyalarını başka bir formatta oluşturma veya yönetme ihtiyacını artırabilir.

- **Öğrenme Eğrisi:** Swagger'ı etkili bir şekilde kullanmak için bir öğrenme eğrisi vardır. API tanımlama dosyalarının nasıl yazılacağı, Swagger Editor ve diğer araçların nasıl kullanılacağı gibi konularda bilgi sahibi olmak zaman alabilir.

- **Kod Üretim Kalitesi:** Swagger Codegen gibi araçlar, API tanımlama dosyalarından otomatik olarak kod üretir. Ancak, bu otomatik olarak oluşturulan kodun kalitesi ve uyumluluğu bazen beklenen düzeyde olmayabilir. Bu nedenle, el ile yazılmış kodlara göre daha az esneklik ve kalite sağlayabilir.

- **Proje Boyutu ve Karmaşıklık:** Büyük ölçekli projelerde, Swagger belgeleme dosyaları ve Swagger ekosistemi, proje boyutu ve karmaşıklığı artırabilir. Bu da yönetim ve bakım açısından zorluklar yaratabilir.

- **Sürüm Uyumsuzluğu:** Swagger'ın farklı sürümleri arasında uyumsuzluklar olabilir. Bu, API tanımlama dosyalarının geçişlerinin zorlaşmasına ve bazı özelliklerin eski sürümlerle uyumsuz olmasına neden olabilir.

Swagger ile JWT ve Basic Auth yapılarını test etmek

1. **Swagger Belgesi Oluşturma**: İlk adım, API'nin Swagger belgesini oluşturmak veya mevcut bir Swagger belgesini kullanmaktır. Swagger belgesi, API'nin endpoint'lerini, parametrelerini, istek ve yanıt türlerini vb. açıklar.

2. **Gerekli Kimlik Doğrulama Şemalarını Belirtme**: Swagger belgesinde, API'nin JWT veya Basic Auth gibi kimlik doğrulama yöntemlerini nasıl kullanacağını belirtmelisiniz. Bu, her endpoint için ayrı ayrı yapılabilir veya genel olarak tüm API için belirtilebilir.

3. **Swagger UI Kullanma**: Swagger belgesini görsel bir arayüzde sunan Swagger UI'yi kullanın. Bu arayüzde, API'yi test etmek için kullanabileceğiniz bir deneme ortamı bulunur. Swagger UI, belirlediğiniz kimlik doğrulama yöntemlerini kullanarak istek yapmanıza olanak tanır.

4. **JWT ile Kimlik Doğrulama**: Swagger UI üzerinden JWT ile kimlik doğrulama yapmak için, JWT token'ını Authorization başlığı altında göndermeniz gerekir. Swagger UI'de, isteği yaparken Authorization başlığına "Bearer {JWT Token}" formatında token'ı ekleyebilirsiniz. Bu şekilde, API'ye istek yaparken doğru kimlik doğrulamasını sağlayabilirsiniz.

5.Basic Auth ile Kimlik Doğrulama: Basic Auth ile kimlik doğrulama yapmak için, Swagger UI'de isteği yaparken kullanıcı adı ve parolayı Authorization başlığı altında Base64 kodlanmış olarak göndermeniz gerekir. Authorization başlığına "Basic {Base64(KullanıcıAdı:Parola)}" formatında bilgileri ekleyerek isteği gönderebilirsiniz.

6. **İstek Yapma ve Yanıtı Kontrol Etme**: Belirlediğiniz kimlik doğrulama yöntemini kullanarak API'ye istek yapın ve gelen yanıtı kontrol edin. Doğru kimlik doğrulaması yapıldığında beklenen yanıtı almalısınız. Eğer yanıt beklenenden farklıysa veya hata varsa, API'nin kimlik doğrulama mekanizmasını tekrar gözden geçirin ve hatayı gidermeye çalışın.