

API’ ı bir restoranda garson olarak düşünelim.

Bir garson olarak, API bir istemciden bir istek alır (Request), bu isteği veri tabanına alır,

istenen verileri veri tabanından alır ve istemciye bir yanıt (Response) döndürür.

**API Nedir?**

- API ==> Application Programming Interface.

- API’ nin UI (User Interface - Kullanıcı Arayüzü) yoktur.

**API Testi Nedir?**

API testi, API‘ ın bir uygulamanın işlevselliği, güvenilirliği, performansı ve güvenliği açısından beklentileri karşılayıp karşılamadığını kontrol etmek için yapılır. API testi uygulamanın görünümü ile ilgilenmez.

**End to End (E2E) Senaryosu:**

1) Create/Update/Delete kullanıcı arayüzü kullanarak verileri oluşturabilir, silebilir veya güncelleyebiliriz.

3) Create/Update/Delete SQL komutlarını kullanarak verileri oluşturabilir, silebilir veya güncelleyebiliriz.

2) Create/Update/Delete API kullanarak verileri oluşturabilir, silebilir veya güncelleyebiliriz.

Not: Her şirket kendi API'larını oluşturur ve bu hizmetleri satarak para kazanabilirler.

**API ve Web Service Arasındaki Fark?**

API ve Web Service uygulamalar arasında iletişim oluştururlar.

Web Service Aralarındaki tek fark Web Service iki uygulama arasındaki iletişimi internet aracılığı ile yapar.

API Microsoft Word gibi masaüstü uygulamaları, internete ihtiyaç duymayan kendi API‘ larını kullanır.

Bu yüzden Microsoft Word kullandığımızda Microsoft Word API kullanıyoruz.

**Not**: Bütün Web Service’ler API’dır. Ancak bütün API’lar Web Service değildir.

**HTTP Nedir?**

http: Hyper Text Transfer Protocol İstemci (Client) bilgisayarlar ve web sunucuları arasındaki iletişim,

HTTP Request ve HTTP Response göndererek yapılır. İstemciler (Client) genellikle tarayıcılardır (Chrome, Edge, Safari),

ancak herhangi bir program veya cihaz türü olabilirler. Sunucular (Server) genellikle buluttaki bilgisayarlardır.

Client ve Server arasındaki iletişim, request ve response lar ile yapılır:: 1.Bir Client (bir tarayıcı) web'e bir HTTP Request gönderir.

2.Bir Web Server Request’i alır.

3.Server, Request’i işlemek için bir uygulama çalıştırır.

4.Server tarayıcıya bir HTTP Response (çıktı) gönderir.

5.Client (Browser) Response’u alır.

**API Testinde Tam Olarak Neyin Doğrulanması Gerekir?**

API testinde, bizler API’a bilinen verilerle bir istek göndeririz ve cevabı test ederiz.

1)HTTP Status Kodları

2)Data Doğruluğu

**Ortak HTTP status Kodları**

1) 200 (OK) ==> Talep karşılandı. Request gerçekleşti.

2) 201 (CREATED) ==> Database’ e gönderilen datanın oluşturulduğunu söyler. (200 olarak da gelebilir)

3) 204 (NO CONTENT) ==> Talep yerine getirildi ancak size cevap (response) dönemiyoruz.

4) 400 (BAD REQUEST) ==> Syntax yanlış, yüksek boyutlu data ya da diğer kullanıcı hatası...

5) 403 (FORBIDDEN) ==> Bu dataya erişim izniniz yok. (400 olarak da gelebilir)

6) 404 (NOT FOUND) ==> Aranılan data bulunamadı.

7) 500 (INTERNAL SERVER ERROR) ==> Server kaynaklı bir hatadır, server’ın düzelmesini bekleriz.

**API Testlerinde Hani Programları Kullanırız?**

1) Postman, geliştiricilerin API’ları oluşturmasını,

paylaşmasını ve test etmesini ve belgelemesini sağlayan popüler bir uygulamadır.

2) SOAP UI SOAP: Simple Object Access Protocol.

(Basit Nesne Erişim Protokolü) Web uygulamalarının internet üzerinden haberleşebilmesi önemlidir.

HTTP, tüm internet tarayıcıları ve sunucuları tarafından desteklendiğinden, uygulamalar arasında iletişim kurmanın en iyi yolu HTTP üzerindendir.

SOAP bunu başarmak için oluşturulmuştur.

SOAP, farklı teknolojiler ve programlama dilleri ile farklı işletim sistemlerinde çalışan uygulamalar arasında iletişim kurmak için bir yol veya biçim sağlar.

SOAP platformlardan bağımsızdır.

3) REST / RestFul

REST: REpresentational State Transfer

* REST sistemlerin birbirleri ile iletişim kurmalarını kolaylaştırır.
* REST mimarisinde, client’ lar dataları almak (retrive) veya değiştirmek (modify) için request gönderir ve server’lar bu request’lere response gönderir.
* REST , XML ve JSON (Java Script Object Notation) formatlarının ikisini de kullanabilir. Bu sebeple SOAP dan daha yaygın bir kullanımı vardır.

Bir REST Request’i Nelerden Oluşur?

1) Bir HTTP yöntemi, ne tür bir işlemin gerçekleştirileceğini tanımlar.

A) GET, dataları okumak içindir. ==> Bu sitede hotel arayalım www.hotels.com

B) POST, yeni data oluşturmak içindir. ==> Sitede hotel rezervasyonu yapalım www.hotels.com

C) PUT, dataları tamamen güncellemek içindir. ==>Tüm rezervasyon datasını değiştirelim www.hotels.com

D) PATCH, dataları kısmen güncellemek içindir. ==> Sadece rezervasyon tarihini değiştirelim www.hotels.com

E) DELETE, dataları silmek içindir. ==> Rezervasyonu iptal edelim. www.hotels.com

2) HEADER(Başlık)‘da, istemci(Client) sunucudan alınacak içeriğin türünü gönderir. Header(Başlık),

sunucunun istemci tarafından anlaşılamayan veya işlenemeyen verileri göndermemesini sağlar.

Bir kaynağa PATH(Yol) (Endpoint/URL/URI) oluşturma:

İstekler(Request), işlemin gerçekleştirileceği bir kaynağa giden yolu içermelidir.

**API Endpoint Nedir?**

API’ın istek gönderdiği ve kaynağın yaşadığı yer endpoint’tir.

API developer’lar bir API oluşturduğunda genel endpoint’ler(URL) oluştururlar

ve hangi endpoint’in hangi HTTP Request yöntemleriyle çalışacağına karar verirler.

**Swagger Nedir?**

Swagger, API’ nızın, zayıf kaynak kodu bilgisine sahip olsalar bile geliştiriciler

ve testerlar tarafından okunması ve anlaşılması kolay ortak bir dil kullanarak tanımlamak için bir framework’tür.

Bunu bir ev planı gibi düşünebilirsiniz.İstediğiniz yapı malzemesini kullanabilirsiniz ancak evin planının dışına çıkamazsınız.

API belgeleri için SWAGGER kullandım. SWAGGER bana API endpoint’leri veriyor ve bunların nasıl kullanılacağı hakkında beni bilgilendiriyor.

**GET Request Nasıl Çalışır?**

Postman, end-point URL’ini girdiğimiz backend testi için kullanılır. Request’i sunucuya gönderir ve sunucudan yanıtı alır.

A) Parameters(Yol Parametresi) (Optional)

B) REST API URL/Endpoint(Mandatory), Query Parameters (Sorgu Parametresi) ya da Path

C) Headers‘da kullanılacak cevap formatını seçmek. XML ya da JSON (Optional) olabilir.

**POST Request Nasıl Çalışır?**

Note: Yanıt aldıktan sonra öncelikle status code kontrol ederiz.

Note: POST Request yapabilmeniz için yetkilendirilmiş olmanız gerekmektedir.

**PUT Request Nasıl Çalışır?**

Note: Yanıt aldıktan sonra öncelikle status code kontrol ederiz.

Note: PUT Request yapabilmeniz için yetkilendirilmiş olmanız gerekmektedir.

**PATCH Request Nasıl Çalışır?**

Note: Yanıt aldıktan sonra öncelikle status code kontrol ederiz.

Note: PATCH Request yapabilmeniz için yetkilendirilmiş olmanız gerekmektedir.

**DELETE Request Nasıl Çalışır?**

Note: Yanıt aldıktan sonra öncelikle status code kontrol ederiz.

Note: DELETE Request yapabilmeniz için yetkilendirilmiş olmanız gerekmektedir.

**Postman‘de Testler Nasıl Yapılır ?**

1) Tests’ e tıklanır

2) Seçim

3) Güncellemeye ihtiyacınız varsa

4) Gönder

5)İsterseniz kayıt edebilirsiniz

**API Test Case‘leri Nasıl Yapılır ?**

Aşağıdaki testleri Postman Requestlerini kullanarak yapalım (Manually)

1. When REST API ye GET sorgusu gönderdiğimizde

URL: https://restful-booker.herokuapp.com/booking

And Accept type: “application/JSON” olmalıdır

Then HTTP Status Code: 200 olmalıdır

And Response formatı: "application/JSON"

1. When REST API ye GET sorgusu gönderdiğimizde

URL https://restful-booker.herokuapp.com/booking/5

And Accept formatı “application/JSON” olmalıdır

Then HTTP Status Code 200 olmalıdır

And first name “Mary” olmalıdır

And total price 814 olmalıdır

1. When REST API ye GET sorgusu gönderdiğimizde

URL http://dummy.restapiexample.com/api/v1/employees And Accept formatı “application/JSON” olmalıdır

Then HTTP Status Code 200 olmalıdır And “Gerret Winters” data da görünmelidir.

1. When REST API ye GET sorgusu gönderdiğimizde URL http://dummy.restapiexample.com/api/v1/employee/1

And Accept formatı “application/JSON” olmalıdır Then HTTP Status Code 200 olmalıdır

And employee name “Tiger Nixon” olmalıdır And employee salary “320800” olmalıdır And employee age “61” olmalıdır