



Application **P**rogramming **I**nterface

API

(Uygulama Programlama Arayüzü)

28/10/2022

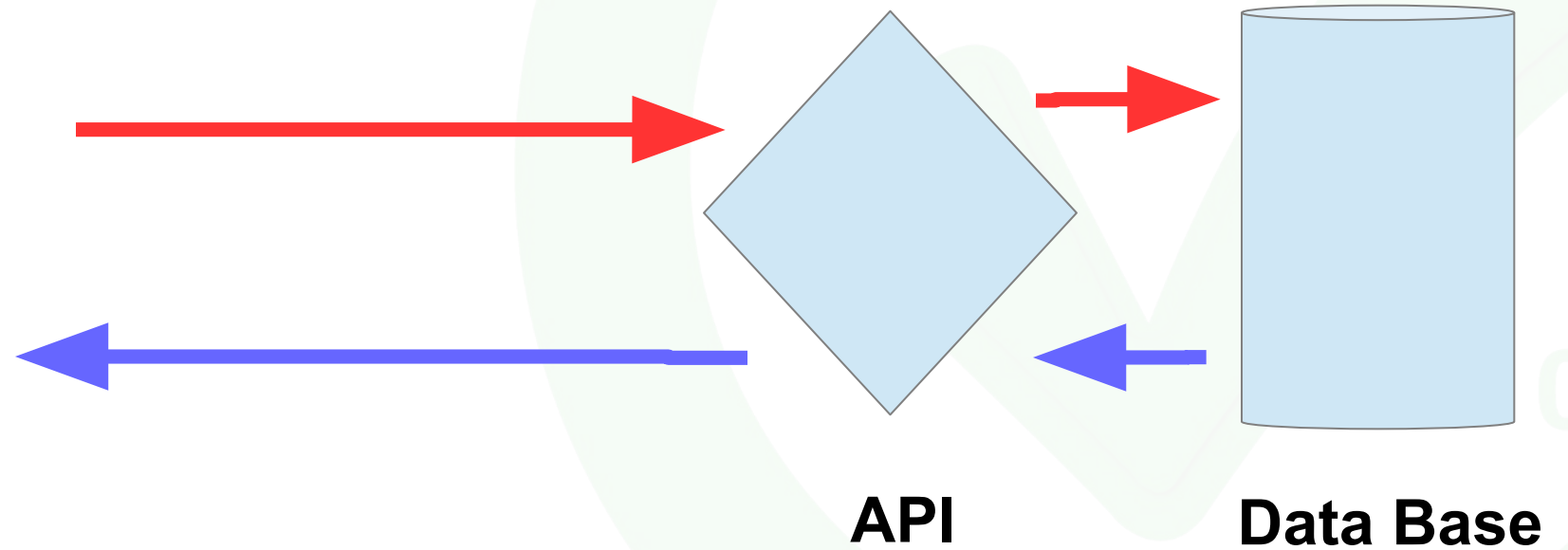


API TEST

Response response = *given().when().get(url);*

given().when().get(url)

Response response



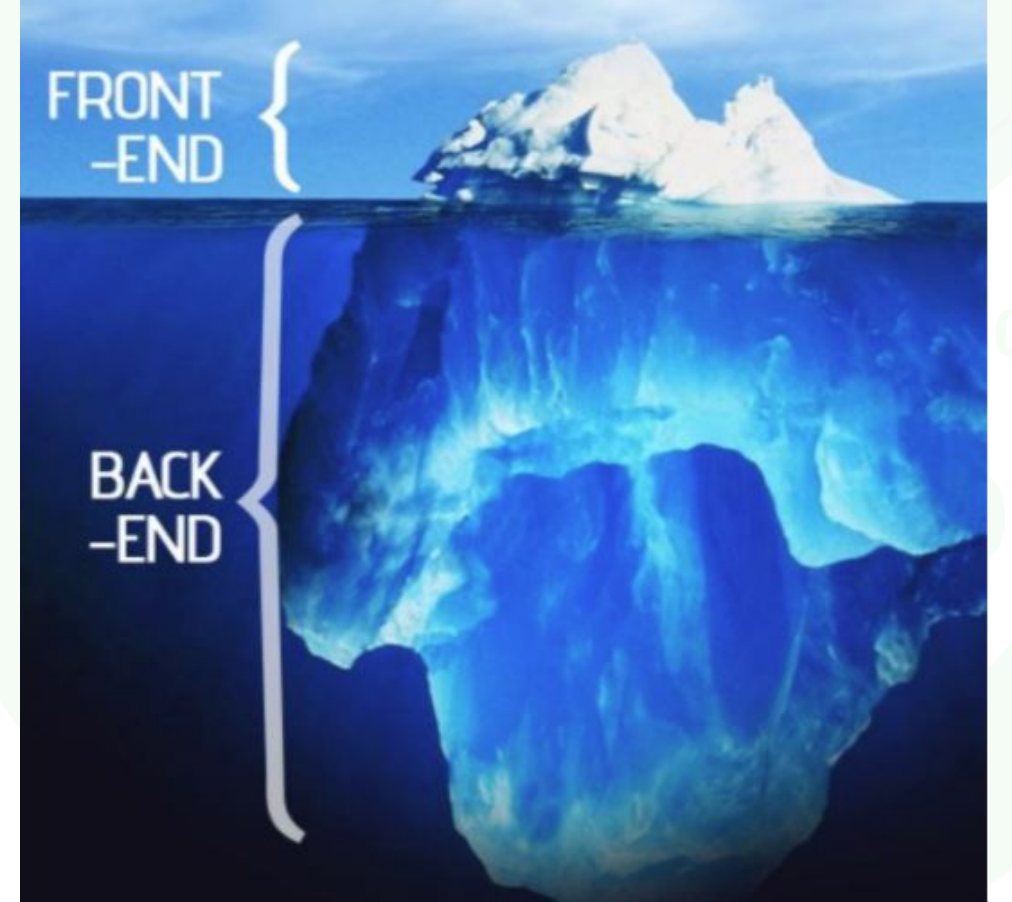


FRONTEND, BACKEND

Frontend, bir web sitesine veya uygulamaya girdiğinizde; etkileşime girdiğiniz arayüzün, tasarım ve geliştirmesidir.

Backend, bu web sitesinin veya uygulamanın perde arkasında yer alan işleyişin; server kısmı ve taban yazılımını geliştirme işine verilen adlardır.

<https://www.gmibank.com/>





FRONTEND

Bir web sitesine girdiğinizde karşılaştığınız; renkli temalar, arka fonlar, yazı tipleri, tasarımsal görseller ve bunların kullanıcıya hitap edebilecek biçimde oluşturulması, sayfaya yerleştirilmesi gibi işlemlerin hepsi frontend yani önyüz olarak adlandırılır.

Frontend'de yer alan bu öğelere eklenen bilgilerin depolanabilmesi, yani kısaca frontend'in hayata geçebilmesi için gereken alt yapı ve teknolojiyi sağlayan ise backend'dir.





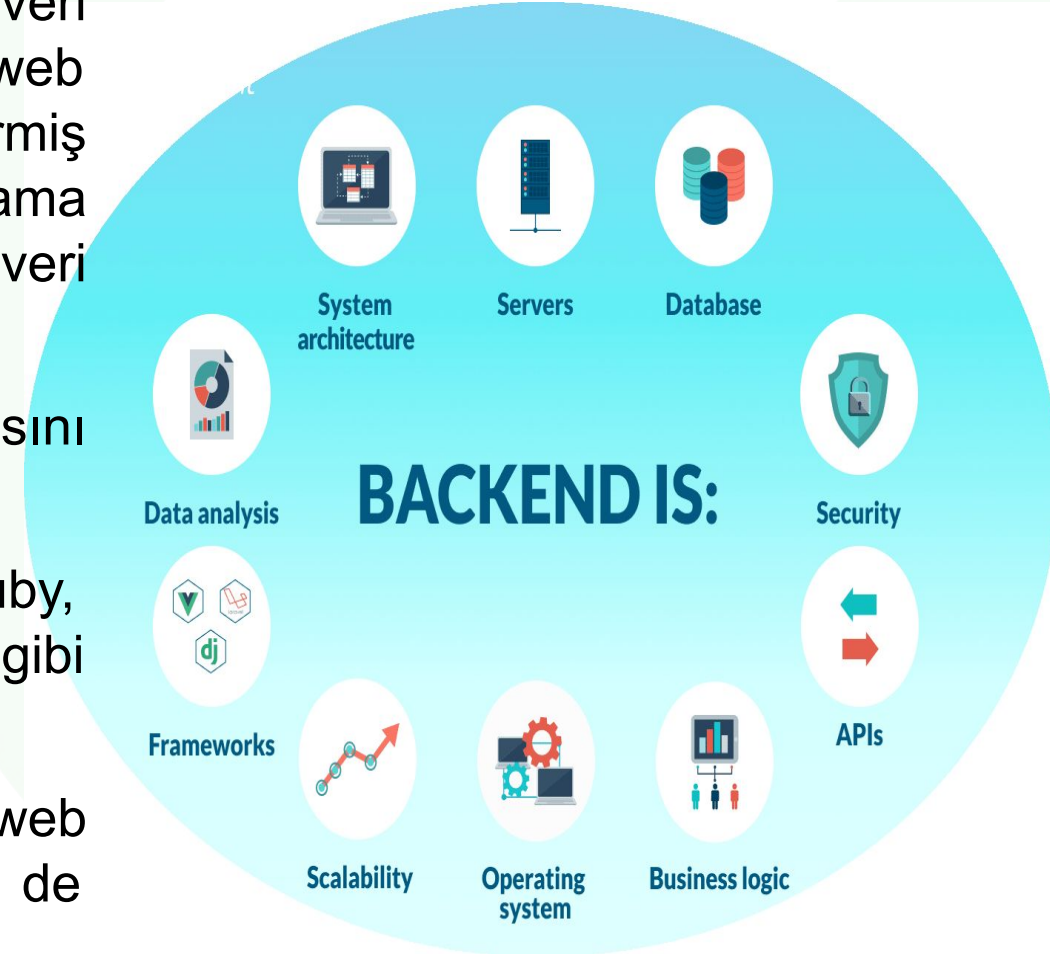
BACKEND

Backend genellikle bir sunucu, bir uygulama ve bir veri tabanından oluşur. Bir havayolu veya otobüs firmasının, web sitesine girerek bilet aldığınızda; frontend ile etkileşime girmiş olursunuz. Siz bilgilerinizi web sitesine girdiğinizde, uygulama bu bilgiyi alır ve bir sunucu üzerinde kurulmuş olan veri tabanına gönderir, size istediğiniz verileri getirir. Pegasus

Sunucu, veri tabanı ve uygulamanın birlikte çalışmasını sağlayan kısma backend denir.

Web'in backend kısmını oluşturmak için Java, PHP, Ruby, Python vb. yazılım dilleri ve MySql, PostgreSQL ve Oracle gibi veri tabanları kullanılır.

Frontend ve Backend geliştiriciler bir araya geldiklerinde bir web sitesi veya uygulama oluşturabilirler. Ancak her ikisi de birbirinden farklı işler yapar.





API NEDİR?

API (Application Programming Interface), bir yazılımın başka bir yazılımda; yalnızca belirli bölümlerine erişmesine ve kullanmasına izin veren arayüzdür.

Bir uygulamanın başka bir uygulama ile iletişim kurmasına izin veren protokol'dür.

API; web uygulaması, işletim sistemi, veritabanı, donanımlar veya yazılım kütüphanesi için kullanılabilir.

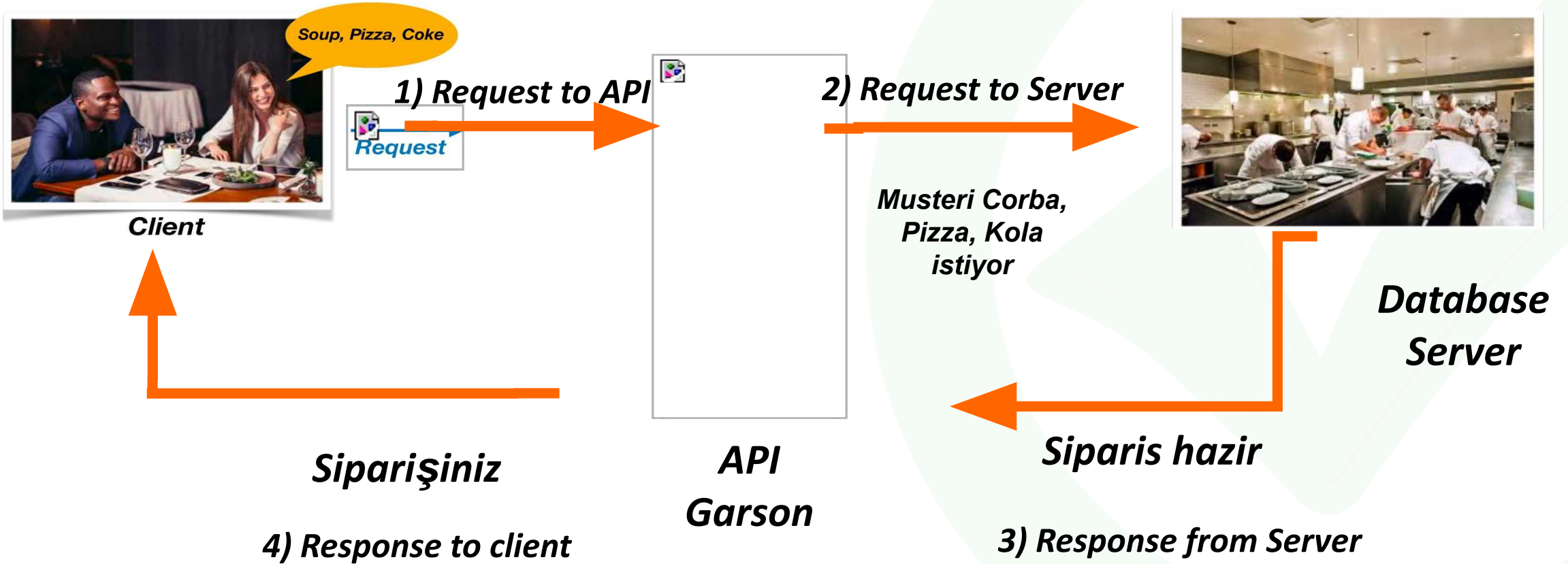
Günümüzde yoğunlukla web tabanlı uygulamalarda; istemci ve sunucu arasındaki iletişimi sağlayan bir yapı olarak kullanılmaktadır.

İstemci spesifik bir formatta veri talep eder ve sunucudan yine belirli bir formatta cevap alır. Bu yapı Web API olarak da adlandırılır.



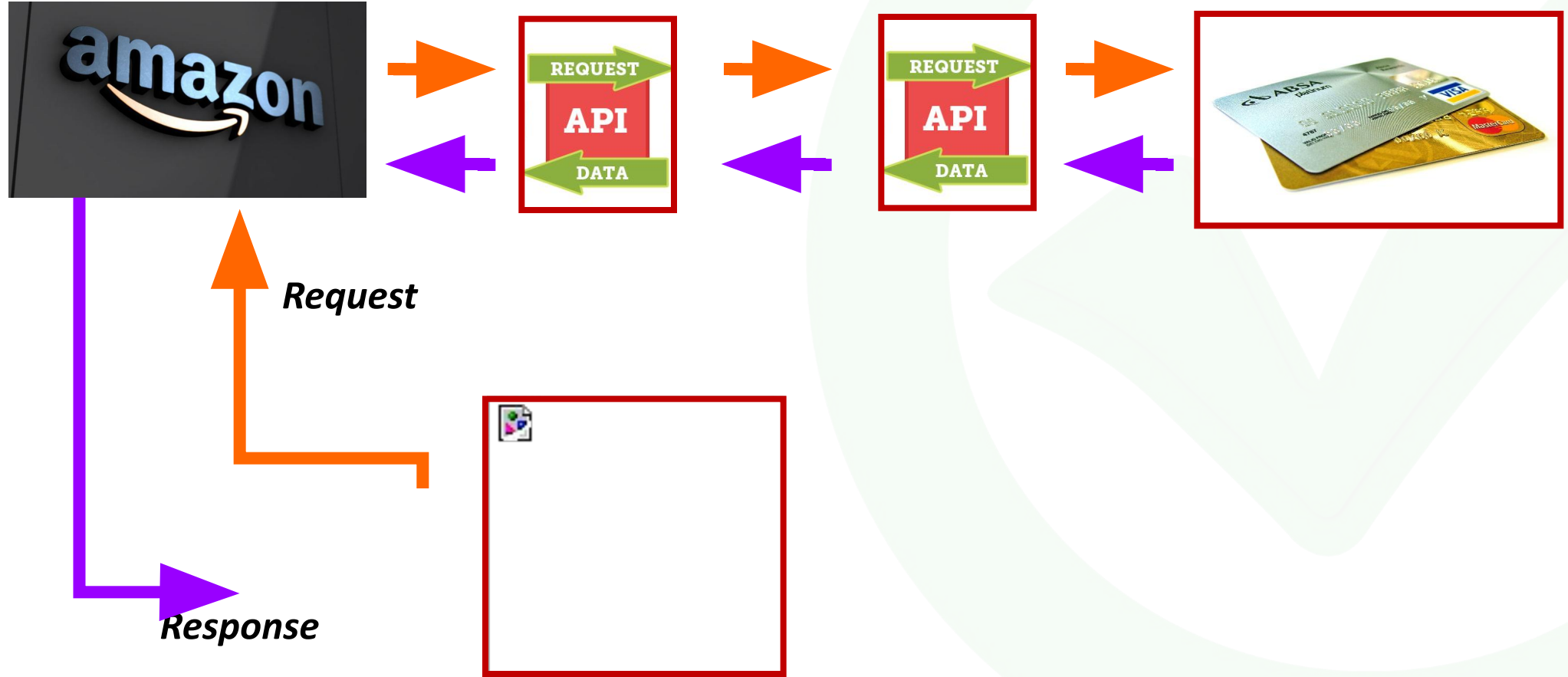


API NASIL ÇALIŞIR?



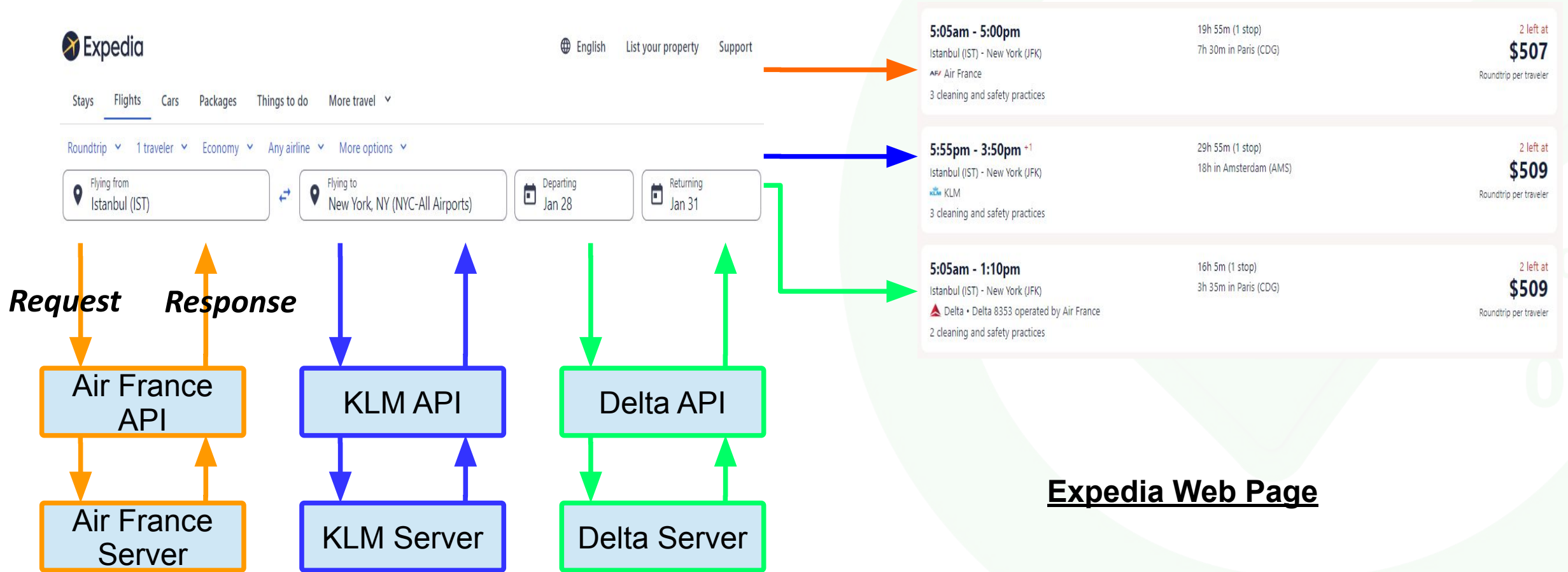


API NASIL ÇALIŞIR?



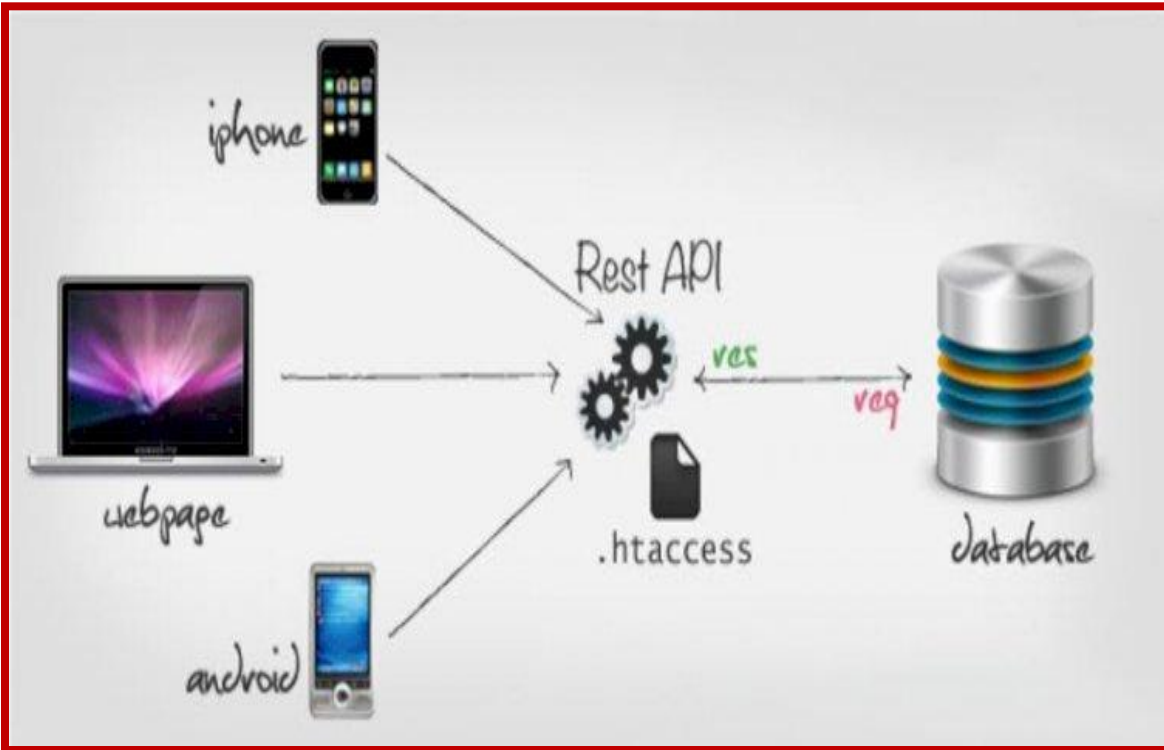


API NASIL ÇALIŞIR?





API NASIL ÇALIŞIR?



REQUEST, RESPONSE



Client

HTTP request

HTTP response



Server

Request: İstek, Talep

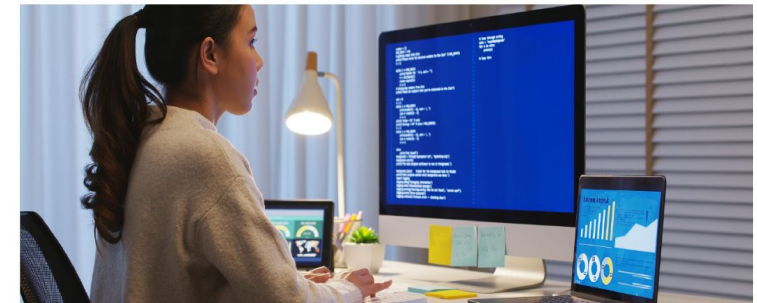
Response: Yanıt, Cevap

<https://www.techproeducation.com/>

UI



[Home](#) [Programs](#) [About Us](#) [Career Coaching](#) [Blog](#)



**Full Stack Automation
Engineer (QA)**





API UYGULAMA ÖRNEKLERİ

<https://developers.google.com/maps>

<https://developers.facebook.com/>

<https://developer.twitter.com/en>

<https://developers.garantibbva.com.tr/>

<https://sehirrehberi.ibb.gov.tr/developer/>

<https://apidocs.parasut.com/>

<https://developer.turkishairlines.com/>

<https://partner.turkcell.com.tr/>

<https://www.visualcapitalist.com/every-minute-internet-2020/>



API, WEB SERVICE

- **Web Service** : İki makinenin bir ağ üzerinden birbirleri ile iletişim kurmak için kullandığı bir yöntemdir. Bilgisayarda çalışan bir web sunucusu diğer bilgisayarlardan gelen istekleri dinler. Başka bir bilgisayardan bir istek alındığında, bir ağ üzerinden web hizmeti istenen kaynakları döndürür. Bu kaynak JSON, XML, HTML dosyası, resimler, ses dosyaları vb. olabilir.
- API ve Web Service, uygulamalar arasında iletişimi sağlar.
- **Web Service internet üzerinden iletişim sağlar. API ise internet olmadan da iletişim sağlayabilir.**
- Örneğin; Expedia, KLM Airlines DataBase'ine ulaşmak için internet kullanır (Web Service), bilgisayarımızdaki Microsoft Word gibi uygulamalar ise farklı uygulamalarla iletişim kurmak için kendi API'lerini kullanırlar. Intranet.
- **Tüm Web Servisler API'dır, ancak tüm API'lar Web Servis değildir.**
- Web servisleri, bir API'nin gerçekleştireceği tüm işlemleri gerçekleştiremeyebilir.
- Bir Web servisinin her zaman için bir ağ'a ihtiyacı olurken, bir API'nin çalışması için ağ'a ihtiyacı yoktur gerekmez.



HTTP, HTTPS

- Bilginin; sunucudan kullanıcıya nasıl ve ne şekilde aktarılacağını gösteren protokoldür. Açılımı “Hyper Text Transfer Protocol” (Üstün Metin Transfer Protokolü). Kullanıcılar, bunu aktif olarak kullanmasa da otomatik olarak browser'lar ;bu protokolü, girilen adrese ekler.
- HTTP 1990 yılından beri dünya çapında ağ üzerinde kullanılan bir iletişim protokolüdür.
- HTTP protokolü ağ üzerinden web sayfalarının görüntülenmesini sağlayan protokoldür.
- HTTP protokolü istemci (PC) ile sunucu (Server) arasındaki veri alışverişi kurallarını belirler.
- Client ve Server arasındaki tüm iletişim request ve response'lar ile olur.





SSL NEDİR?

SSL (Secure Socket Layer): Güvenli Giriş Katmanı anlamına gelen SSL, web siteniz ve ziyaretçileriniz arasındaki iletişimin; şifrelenmiş bir bağlantı üzerinden gerçekleştirilmesini amaçlar.

2018 yılında itibaren, Google Chrome gibi popüler tarayıcıların, SSL sertifikası olmayan web sitelerini “güvenli değil” şeklinde etiketlemeye başlamasıyla; SSL kullanımı yaygınlaşmıştır.

qa-environment.koalaresorthotels.com



Kullanıcı 1

HTTP

http://www.example.com

şifre : abc123



Şifreleme olmadan

korsanların gördüğü şifre “abc123”



Kullanıcı 2

HTTPS

https://www.example.com

şifre : abc123



Şifreleme varken

korsanların gördüğü şifre “xgeaDarz”



HTTP STATUS CODE

- Kullanıcılar bir web sitesini ziyaret etmek istediklerinde iki taraflı bir iletişim ortaya çıkar.
- Bu iletişimin bir tarafında tarayıcı bulunurken diğer tarafta sunucu yer alır.
- Bir web sayfasına giriş yapan kullanıcı aslında tarayıcı aracılığıyla ilgili web sayfasının yer aldığı sunucuya, sayfayı görüntülemek için bir istek gönderir.
- Sunucu ise bu isteğe üç haneli bir durum kodu ile yanıt verir.
- Sunucunun tarayıcıya verdiği üç haneli cevaplar HTTP durum kodları, HTTP status codes olarak adlandırılır.



HTTP Status Codes




API ARCHITECTURAL STYLES





API ARCHITECTURAL STYLES

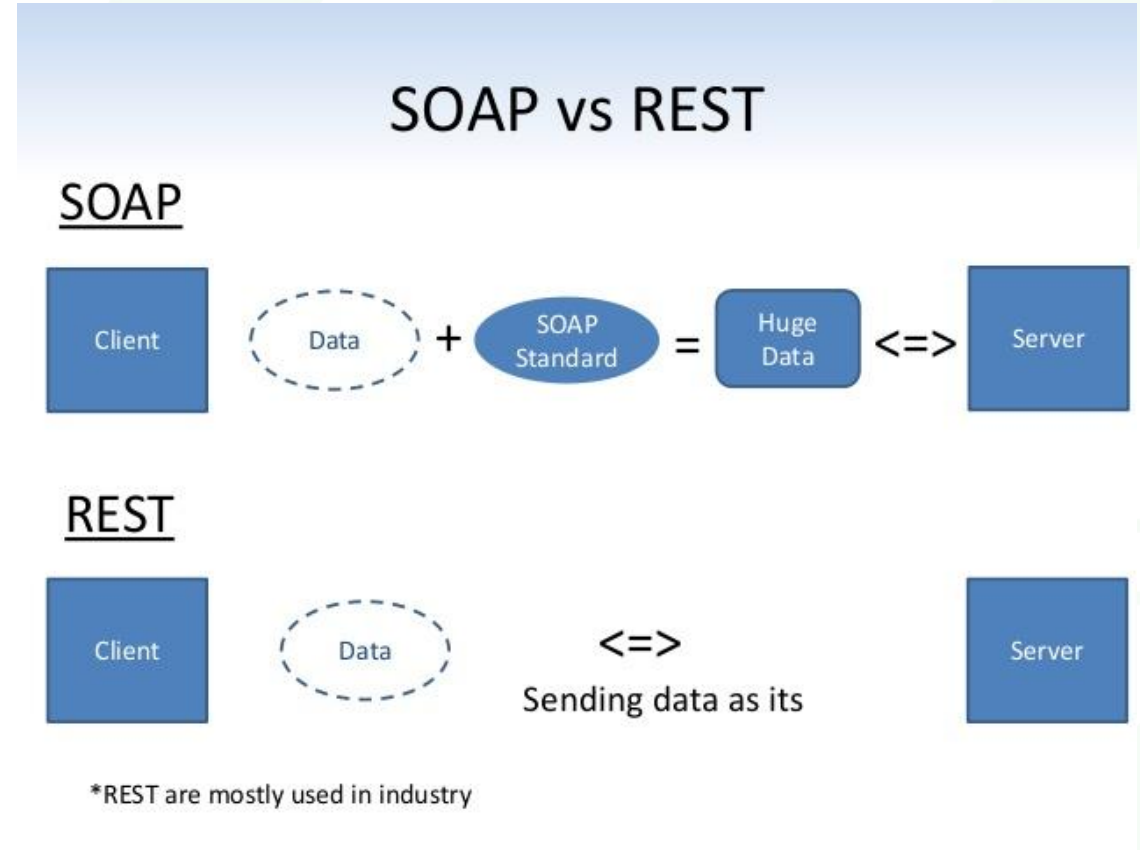
	RPC	SOAP	REST	GraphQL
Organized in terms of	local procedure calling	enveloped message structure	compliance with six architectural constraints	schema & type system
Format	JSON, XML, Protobuf, Thrift, FlatBuffers	XML only	XML, JSON, HTML, plain text,	JSON
Learning curve	Easy	Difficult	Easy	Medium
Community	Large	Small	Large	Growing
Use cases	Command and action-oriented APIs; internal high performance communication in massive micro-services systems	Payment gateways, identity management CRM solutions financial and telecommunication services, legacy system support	Public APIs simple resource-driven apps	Mobile APIs, complex systems, micro-services

**altexsoft**
software r&d engineering



SOAP, REST

- Web servis mimarisinin temeli HTTP üzerine kurulmuştur. Genel olarak web servise bir istek gelir ve web servis bu isteği yapıp, bir sonuç döndürür.
- Web servisin bu işlemi yapabilmesi için tanımlanmış farklı yöntemler bulunmaktadır. Bu yapılardan biri SOAP protokolü diğeri ise REST'dir.





SOAP (Simple Object Access Protocol)

- SOAP (Simple Object Access Protocol- Basit Nesne Erişim Protokolü) uygulamalar ile web servislerin bilgi aktarımını sağlayan, XML tabanlı bir protokoldür.
- Web servise giden bilgi XML olarak gönderilir, web servis bu bilgiyi yorumlar ve sonucunu XML olarak geri döndürür. SOAP tabanlı bir web servisin, gönderilen XML verisini nasıl yorumlayacağını tanımlanması gerekir. Bu web servis tanımlaması WSDL (Web Service Description Language- Web Servisleri Tanımlama Dili)

```
<customer>
  <customer_id> 1001 </customer_id>
  <customer_name> Mark Star </customer_name>
</customer>
```

```
<?xml version="1.0" encoding=
<definitions name="AktienKurs
  targetNamespace="http://loc
  xmlns:xsd="http://schemas.xmlsoap.or
  xmlns="http://schemas.xmlsoap.org/wsd
  <service name="AktienKurs">
    <port name="AktienSoapPort" binding
      <soap:address location="http://loc
    </port>
    <message name="Aktie.HoleWert">
      <part name="body" element="xsd:Tra
    </message>
    ...
  </service>
</definitions>
```

WSDL



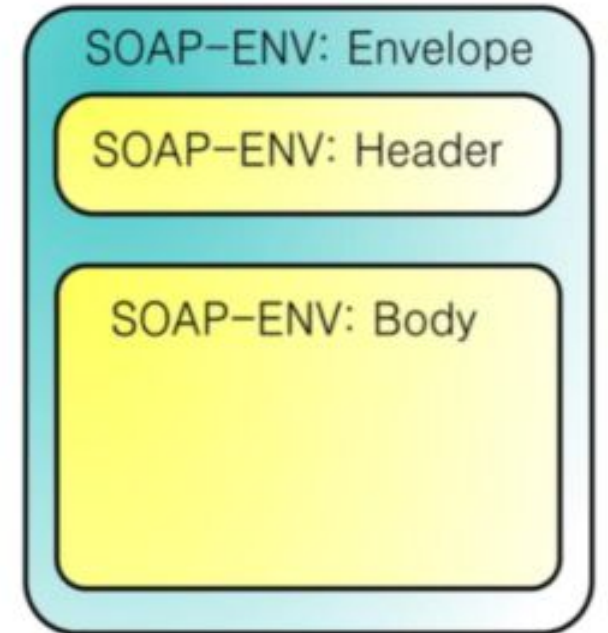
SOAP (Simple Object Access Protocol)

Bir SOAP mesajı 3 şeyi içerir.

Envelope (Zarf): Tüm mesaj içeriğinin içinde olduğu kısımdır. SOAP mesajları XML formatında olduğundan bir root elemanı olmalıdır. Envelope, SOAP mesajının root etiketidir de denebilir. Envelope'un içinde Header ve Body gibi kısımlar da bulunur.

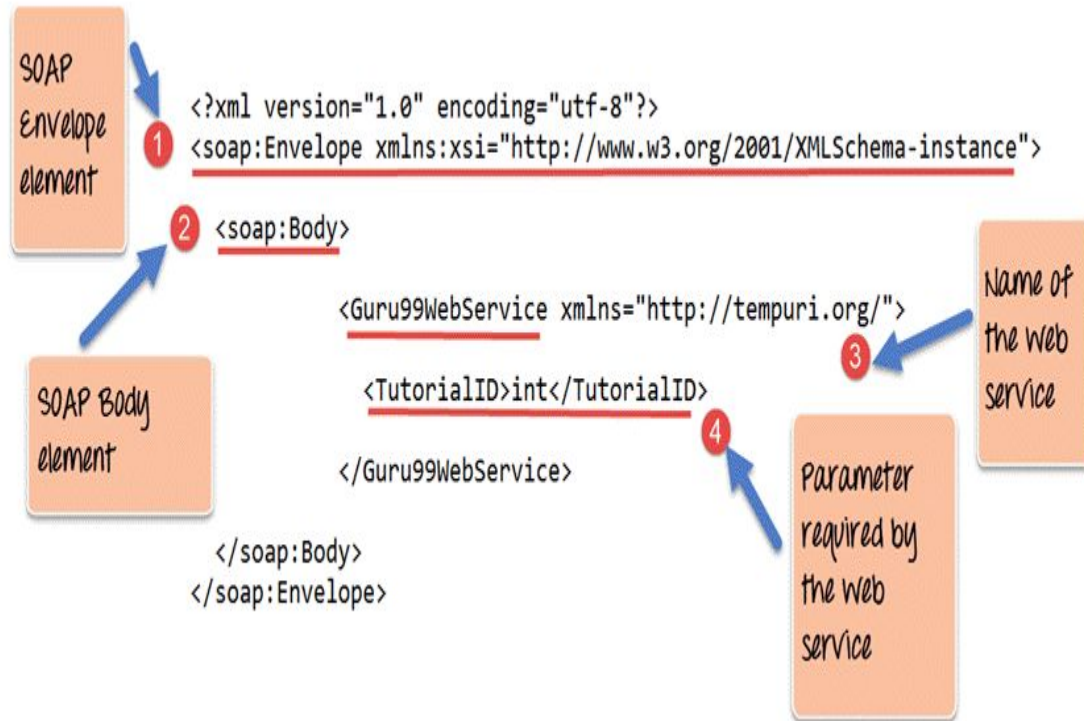
Header (Başlık): Klasik html'de bulunan <head></head> etiketine benzetilebilir. Bu bölümde mesajın meta-data bilgileri gönderilir.

Body: Soap mesajının ana içeriğini barındıran kısımdır. Bu bölümde metotlarla ilgili bilgiler veya metodun sonucu yer alır.





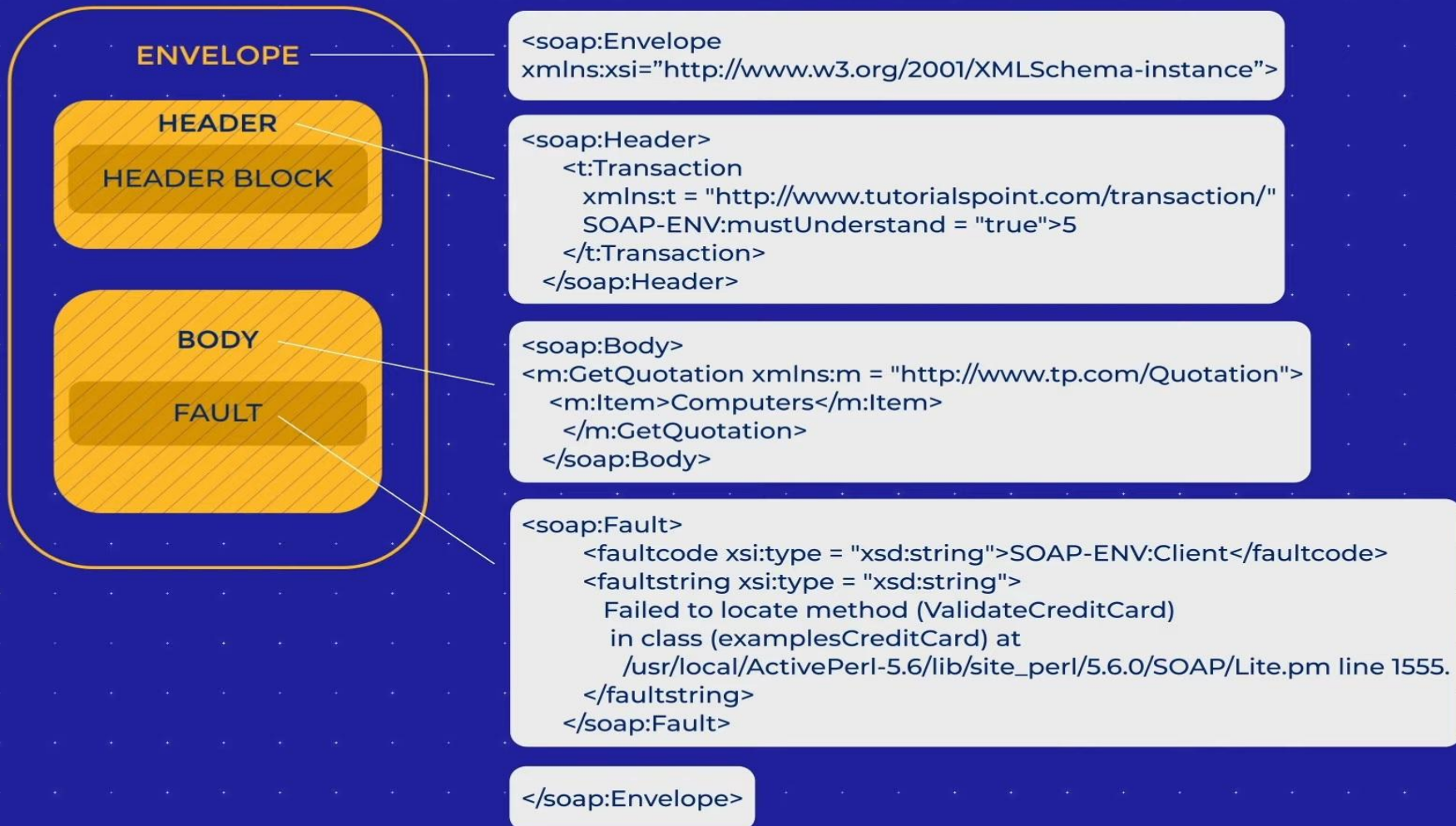
SOAP (Simple Object Access Protocol)



```
<?xml version="1.0"?>  
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://www.w3.org/2001/12/soap-envelope" SOAP-  
ENV:encodingStyle=" http://www.w3.org/2001/12/soap-encoding">  
  <soap:Body>  
    <Guru99WebService xmlns="http://tempuri.org/">  
      <TutorialID>int</TutorialID>  
    </Guru99WebService>  
  </soap:Body>  
</SOAP-ENV:Envelope>
```



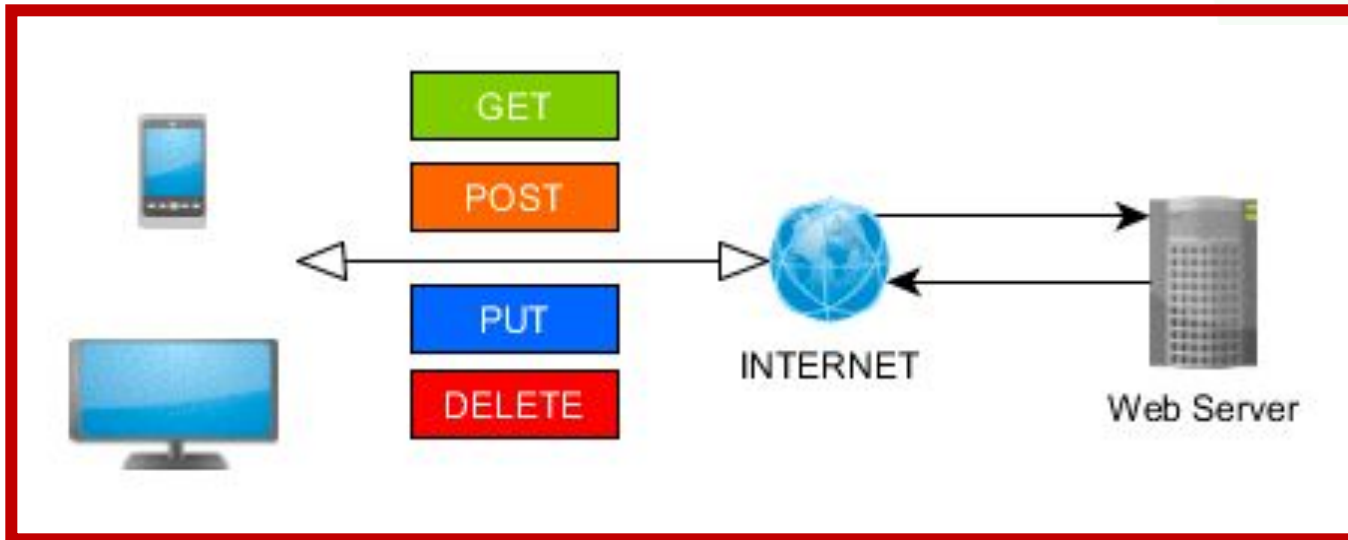

SOAP (Simple Object Access Protocol)





REST (REpresentational State Transfer)

REST API, SOAP ve WSDL tabanlı web servislerinin daha basite indirgenmiş hali olarak nitelendirilebilir. Bu amaçla; internet tarayıcısı sayfa işlemlerinin bir parçası olan HTTP protokolünü (GET, POST, PUT DELETE gibi talep tipleri) kullanır. Böylelikle REST API geliştiricilere yaygın, pratik ve oldukça esnek bir kullanım olanağı sunduğu gibi, minimum içerikle veri alıp gönderdiği için de daha hızlıdır.



```
{
  "id": 114351,
  "firstName": "Della",
  "lastName": "Heaney",
  "middleInitial": "Russell Homenick V",
  "email": "ricardo.larkin@yahoo.com",
  "mobilePhoneNumber": "123-456-7893",
  "phoneNumber": "213-456-7893",
  "zipCode": "53081",
  "address": "75164 McClure Stream",
  "city": "Gislasonburgh",
  "ssn": "823-25-7239",
  "createDate": "2021-12-05T23:00:00Z",
  "zelleEnrolled": true,
  "country": {
    "id": 3,
    "name": "USA",
    "states": null
  },
  "state": "New York43",
  "user": null,
  "accounts": null
},
```



REST (REpresentational State Transfer)

HTTP metodların REST ile kullanımı: REST tabanlı web servislerde, HTTP metodlarına özel anlamlar yüklenir ve böylece web servise bir HTTP isteği geldiği anda metod çalıştırılmış olur. Bu durumda HTTP metodlarının REST ile nasıl kullanılacağı önemlidir.

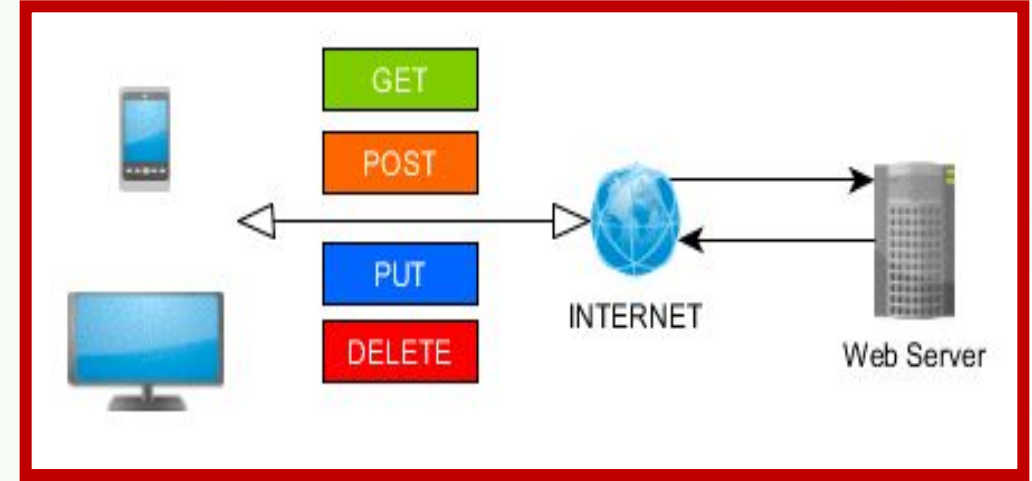
Örnek: <https://www.gmibank.com/api/tp-customers/114351>

GET: Adresi verilen nesneyi döndürmek için kullanılır. Bu metot kullanıldığında kayıtlarda bir değişiklik yapılmaz.

PUT: Var olan bir nesneyi değiştirmek için veya eğer yoksa yeni bir tane oluşturmak için kullanılır.

POST: Yeni bir nesne oluşturmak için kullanılır. Her seferinde yeni bir nesne oluşturur. PUT ile yapılan bir işlem POST ile de yapılabilir fakat aralarındaki fark tarayıcıların bu iki metodu farklı yorumlayıp iki metot için farklı tepkiler verebilmesidir.

DELETE: Adresi verilen nesneyi silmek için kullanılır.





REST (REpresentational State Transfer)

GET <https://www.gmibank.com/api/tp-customers/114351>

```
1  {
2    "id": 114351,
3    "firstName": "Della",
4    "lastName": "Heaney",
5    "middleInitial": "Russell Homenick V",
6    "email": "ricardo.larkin@yahoo.com",
7    "mobilePhoneNumber": "123-456-7893",
8    "phoneNumber": "213-456-7893",
9    "zipCode": "53081",
10   "address": "75164 McClure Stream",
11   "city": "Gislasonburgh",
12   "ssn": "823-25-7239",
13   "createDate": "2021-12-05T23:00:00Z",
14   "zelleEnrolled": true,
15   "country": {
16     "id": 3,
17     "name": "USA",
18     "states": null
19   },
20   "state": "New York43",
21   "user": null,
22   "accounts": [
```

```
23   {
24     "id": 2333,
25     "description": "musteri omer hesap3",
26     "balance": 69700,
27     "accountType": "CREDIT_CARD",
28     "accountStatusType": "ACTIVE",
29     "createDate": "2020-11-06T23:00:00Z",
30     "closedDate": "2024-11-06T23:00:00Z",
31     "employee": null,
32     "accountlogs": null
33   },
34   {
35     "id": 107250,
36     "description": "New Account_6thGenQA_01",
37     "balance": 11190,
38     "accountType": "CHECKING",
39     "accountStatusType": "ACTIVE",
40     "createDate": "2021-11-24T23:00:00Z",
41     "closedDate": "2022-11-24T23:00:00Z",
42     "employee": null,
43     "accountlogs": null
44   }
45 ]
46
```

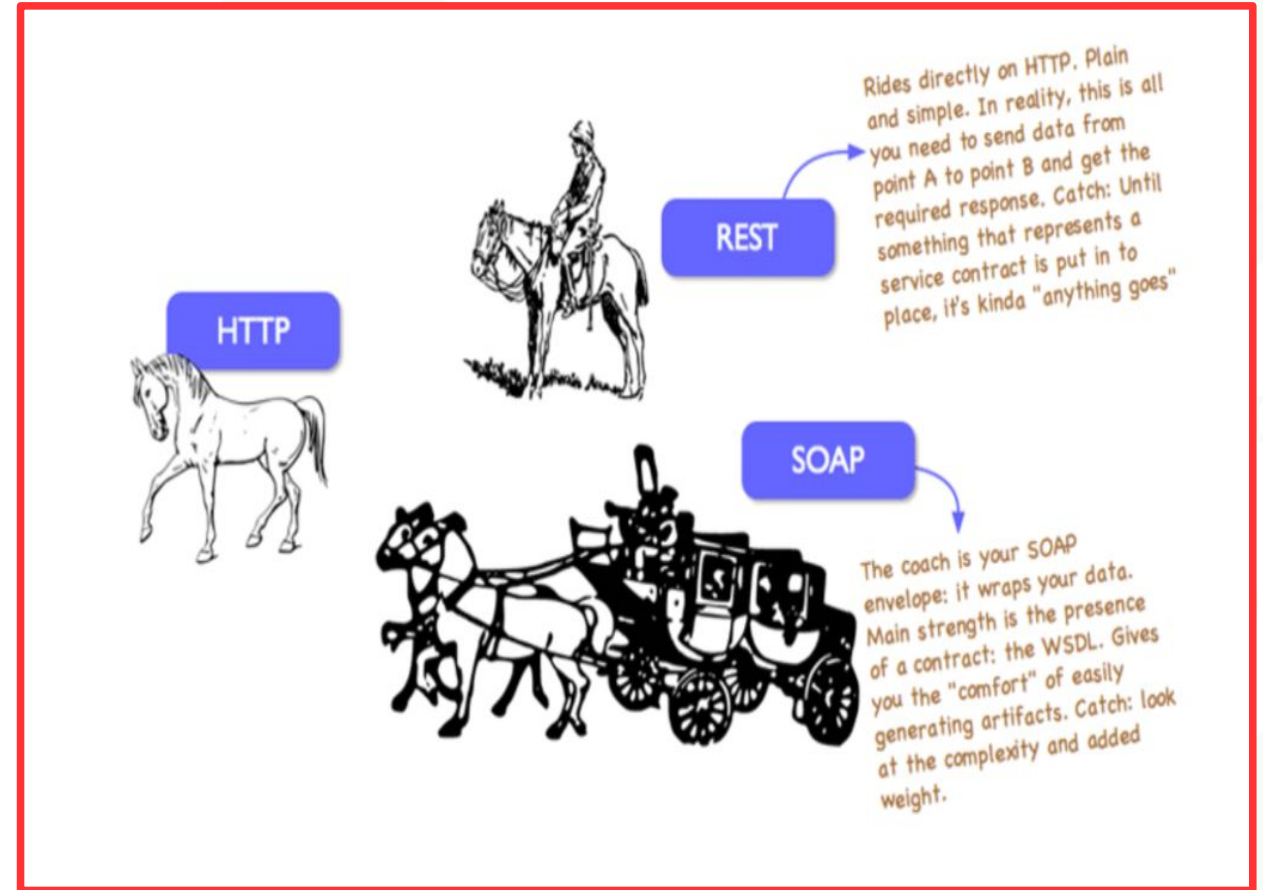
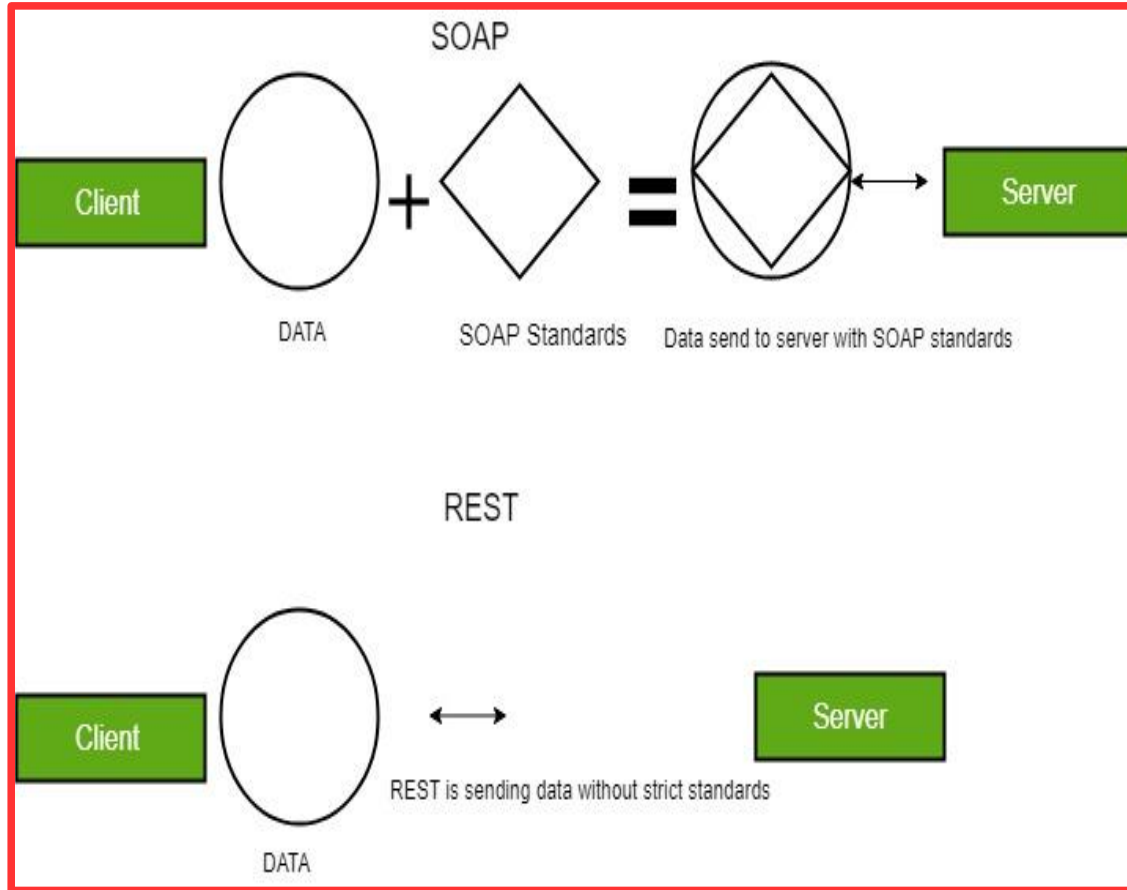


SOAP, REST

SOAP	REST
SOAP bir protokoldür	HTTP protokolünü kullanan bir mimaridir
WDSL ile tasarlama yapmak gerektiğinden kullanması daha zor	HTTP methodları ile tasarlandığı için kullanması daha kolay
Yalnızca XML format kullanır.	XML , JSON , HTML , TXT format kullanır
Önbelliği okuyamaz (can not be cached)	Önbelliği okur (can be cached)
Rest e göre daha yavaş	Soap' a göre daha hızlı
Finansal,iletişim ve ödeme noktalarında kullanılır	Sosyal medya, Web Chat , Mobil uygulamalar da kullanılır
Daha güvenlidir.	Güvenlidir.



SOAP, REST





SOAP, REST

Use in Technology driven sectors



REST

- Social Media
- Web Chat
- Mobile



SOAP

- Financial
- Telecommunication
- Payment Gateways

A postcard is faster and cheaper!

POST

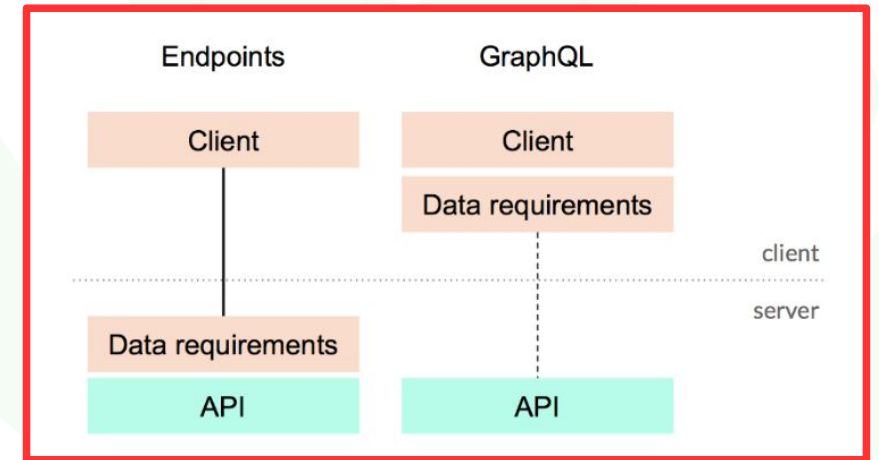
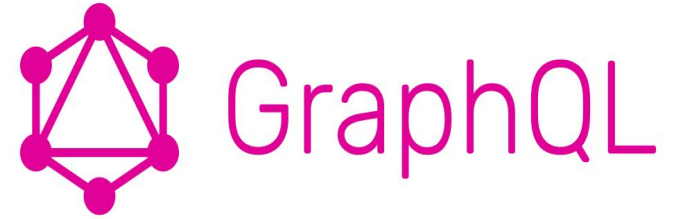
It takes a few extra steps, to access a postcard





GraphQL

- GraphQL, Facebook tarafından geliştirilmiş; verimli, etkili ve esnek bir alternatif sunan yeni bir API standartıdır. Açık kaynaklı olarak geliştirilen GraphQL'i, birçok şirket aktif olarak kullanmaktadır.
- GraphQL, API tasarlamak ve kullanmak için bir yol sunmaktadır. İstemcinin ihtiyaç duyduğu verilerin tam olarak belirtilmesini sağlayarak birden fazla kaynaktan veriye ulaşmanızı kolaylaştırmayı hedefler.
- GraphQL, genel kullanıma sunulduğundan bu yana büyüyen ve popüler hale gelen ve REST'e göre bazı büyük avantajları olan bir sorgu dilidir. Ana odak noktası, performansı optimize etmek ve esnekliği artırmaktır.





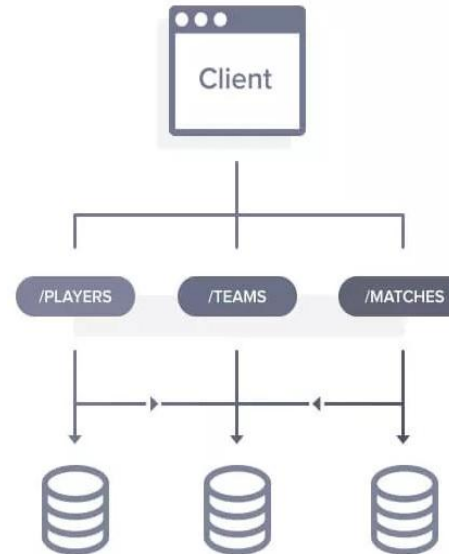
GraphQL

GraphiQL

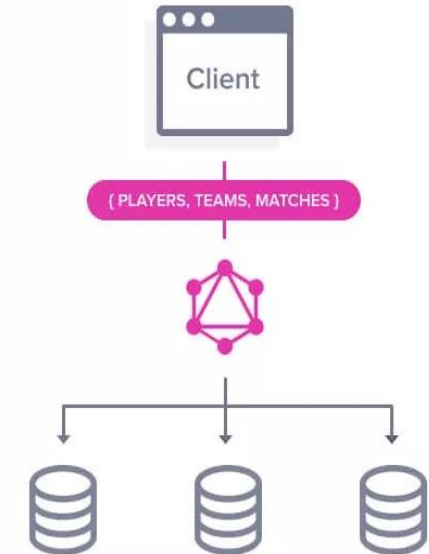
```
1 {  
2   artist{  
3     lastName  
4     firstName  
5     age  
6   }  
7 }
```

```
{  
  "data": {  
    "artist": {  
      "lastName": "Sutton",  
      "firstName": "Tierney",  
      "age": 54  
    }  
  }  
}
```

Rest API



GraphQL API



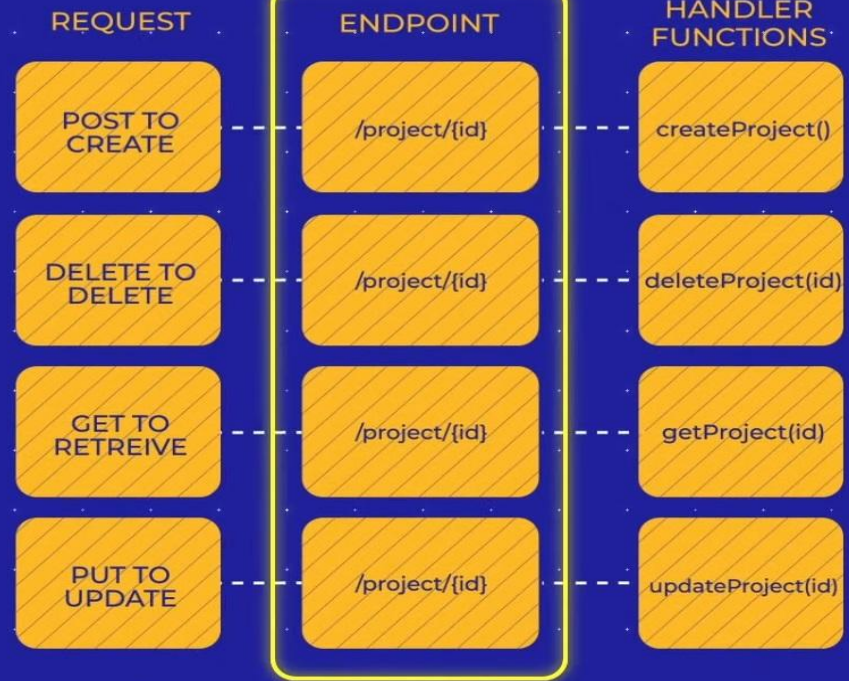
GraphQL Web Sites



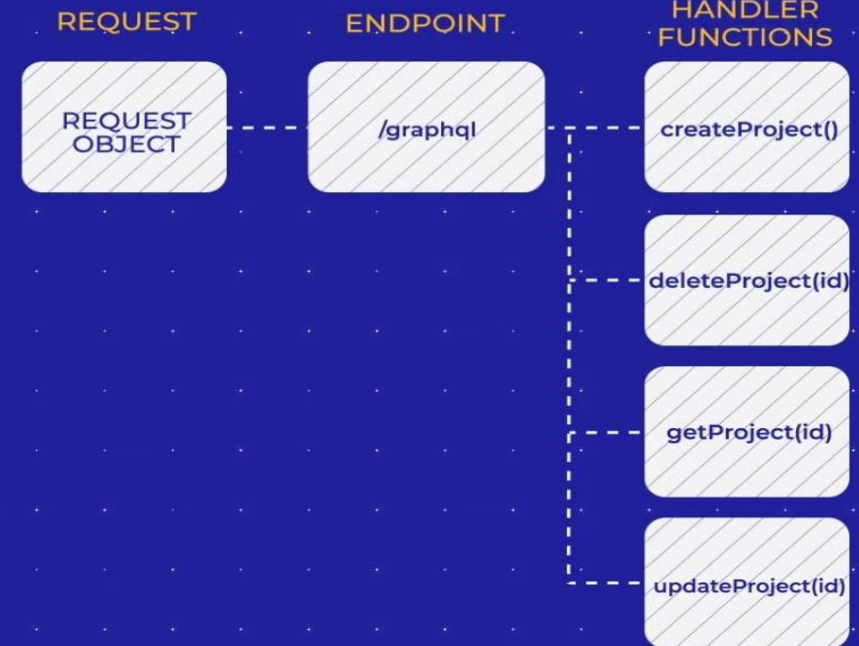
GraphQL

GraphQL

REST



GRAPHQL





POSTMAN

- Postman, backend API'lerini test etmek için oldukça kullanışlı bir tool'dur.
- Elinizde var olan bir API'yi hızlıca test etmek ve sonuçlarını incelemek isterseniz; Postman kullanıcı dostu arayüzü ile işlerinizi oldukça kolaylaştırır.
- Hazırladığınız request'leri export edip arkadaşlarınızla da paylaşabilirsiniz.
- Çok az Javascript bilgisi ile hızlıca testler yazabiliyor, yazdığınız testleri kaydedip sonra tek bir tık ile çalıştırabiliyorsunuz.

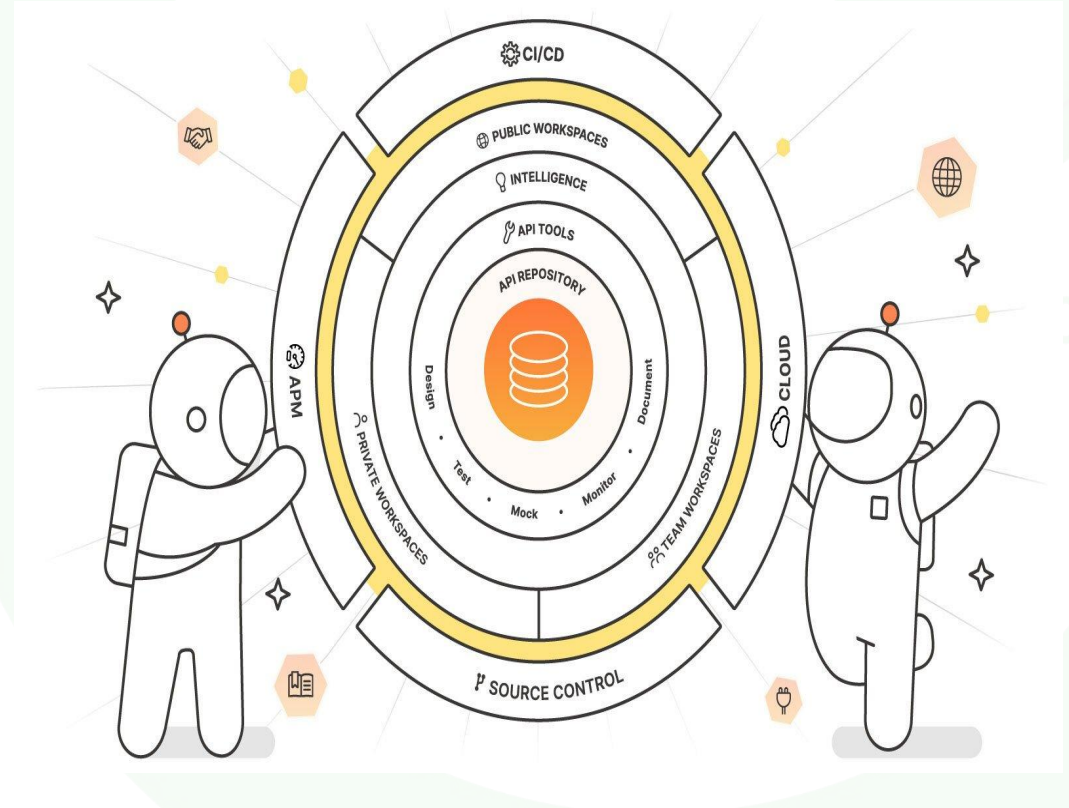


POSTMAN



POSTMAN

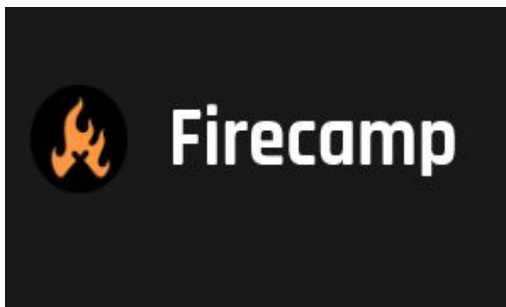
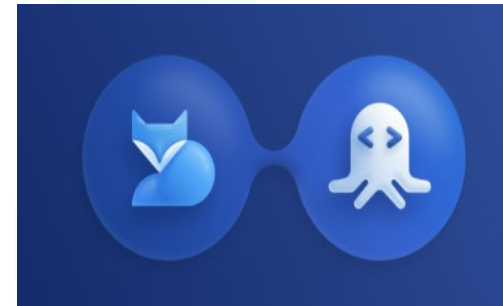
- Postman, API geliřtirmek için bir iřbirlięi platformudur.
- Postman'ı kendi bařınıza veya takım arkadaşlarınız ile birlikte API'ları tasarlamak, geliřtirmek, oluřturmak ve test etmek için kullanabilirsiniz.
- Rest, SOAP ve GraphQL sorgulamalarınız gönderebilirsiniz.
- API'inızı kullanımını ve okunmasını kolaylařtırmak için dökümantasyon oluřturabilirsiniz.
- API'lerinizin iřleyiř durumunu ve performansını kontrol edebilirsiniz.
- Takım arkadaşlarınız ile birlikte workspace dedięimiz çalışma ortamında birlikte API hazırlayıp, kullanabilirsiniz.
- Bireysel olarak ücretsizdir. Sınırsız takım çalışması ve kaynak paylaşımı gibi özellikleri istiyorsanız bunlar ücretlidir.
- Ücretsiz hesaplar için bazı özelliklere ulařımda limit vardır.





POSTMAN

POSTMAN alternatifleri





SWAGGER DOKÜMANI

- **Swagger, REST API'lerini tasarlamana, oluşturmana, belgelemenize ve kullanmamıza yardımcı olabilecek; OpenAPI Spesifikasyonu etrafında oluşturulmuş açık kaynaklı bir araçtır.**
- **Swagger'ın önemli bir amacı RestApi ler için bir arayüz sağlamaktır. Bu, insanların kaynak koda erişmeden RestApi lerin özelliklerini görmesine, incelemesine ve anlamasına olanak sağlar.**

https://app.swaggerhub.com/apis/MuratTANC/gmi-bank_backend_api/0.0.1



API İSİMLENDİRİLMESİ

1) İsimlendirme anlamlı ve sade olmalı. API'de; sonradan isim değiştirmek, ona bağlı çalışan bir çok uygulamayı etkileyecektir.

<https://www.techproeducation.com/createCostomer> --
<https://www.techproeducation.com/costomers> +

2) Hiyerarşi için / (slash) kullanılmalıdır.

<https://www.gmibank.com/api/tp-customers/114351>
<https://www.gmibank.com/api/tp-customers/114351/address>

3) İsimlendirme iki kelime olduğu zaman araya – işareti konulur.

/api/tp-employees/{id}

/api/tp-customers/accounts/{id}

GET	/api/tp-customers	getAllTPCustomers	✓↵
POST	/api/tp-customers	createTPCustomer	✓↵
PUT	/api/tp-customers	updateTPCustomer	✓↵
GET	/api/tp-customers/accounts/{id}	getTPAccounts	✓↵
POST	/api/tp-customers/transfer	getMakeTransfer	✓↵
GET	/api/tp-customers/{id}	getTPCustomer	✓↵
DELETE	/api/tp-customers/{id}	deleteTPCustomer	✓↵



API İSİMLENDİRİLMESİ

4) Versiyonlar “v1” şeklinde gösterilebilir. Yeni bir versiyon oluşturulduğunda “v1.1” veya “v2” şeklinde gösterilebilir. (Versiyon: API versiyon)

/api/v1/tp-customers/accounts/{id}

/api/v1.1/tp-customers/accounts/{id}

/api/v2/tp-customers/accounts/{id}

5) API dökümantasyonu mutlaka olmalı





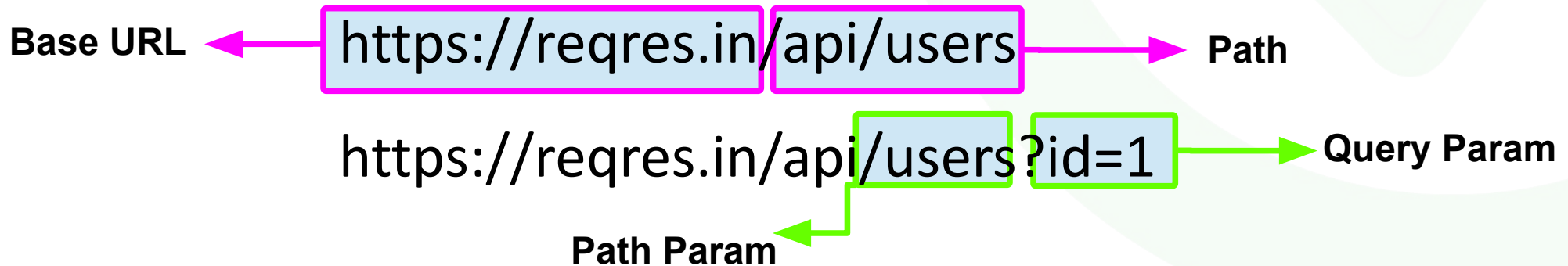
Path Param, Query Param

Bir kaynağı tanımlamak veya daha ayrıntılı nesneleri üzerinde hareket etmek istiyorsanız Path Param kullanmalısınız. Ancak öğeleri sıralamak veya filtrelemek istiyorsanız, Query Parametresi kullanmalısınız. Query parametreleri, kaynakları daha iyi bir şekilde tanımlamaya yardımcı olur.

Query parametreleri URL'de "?" İşaretinin sağ tarafında görünürken, Path parametreleri soru işareti işaretinden önce gelir.

URL'nin bir parçası oldukları için Path parametrelerindeki değerleri atlayamazsınız.

Query parametreleri key-value şeklinde kullanılır.





Path Param, Query Param

https://techproeducation.com/users

Base URL

Path

https://techproeducation.com/users?id=1

Path Param

Query Param

https://techproeducation.com/users?gender=female

https://techproeducation.com/users?gender=female&sort=ASC

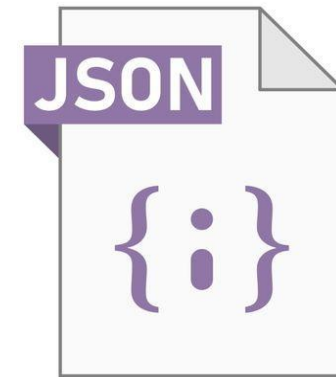
https://techproeducation.com/users?gender=female&sort=ASC&page=1



JSON (JavaScript Object Notation)

- JSON formatı 2000'lerin başında Douglas Crockford tarafından belirlenmiştir. Aralık 2005'te Yahoo! web hizmetlerinden bazılarını JSON biçiminde sunmaya başlamıştır.
- JSON, 2013 yılında bir ECMA (European Computer Manufacturers Association, Avrupa Bilgisayar Üreticileri Birliği) standardı haline gelmiştir
- JSON; bütün programlama dilleri arasında, yapılandırılmış veri değişimini kolaylaştıran bir metin biçimidir.
- Sistemler arası veri aktarımı ile çalışan teknolojilerin çoğu JSON formatını destekler.

{JSON}
JavaScript Object Notation





JSON SYNTAX

- Datarlar name-value yapısındadır.
- Name-Value yapısı arasında : olur.
- JSON isimleri " " çift tırnak içerisinde olur.
- Datarlar , virgül ile bir birilerinden ayrılırlar
- Objeler { } (curly braces) içerisinde dirler.
- Arrayler [] (square brackets) içindedir.

```
{ "Student" : [  
  { "firstName" : "Ali" , "lastName" : "Can" } ,  
  { "firstName" : "Sena" , "lastName" : "Bal" } ,  
  { "firstName" : "Burak" , "lastName" : "Cengiz" } ]  
}
```

```
1  {  
2    "bookingid": 34,  
3    "booking": {  
4      "firstname": "Ali",  
5      "lastname": "CAN",  
6      "totalprice": 111,  
7      "depositpaid": true,  
8      "bookingdates": {  
9        "checkin": "2022-01-01",  
10       "checkout": "2022-01-01"  
11     },  
12     "additionalneeds": "Breakfast, Dinner"  
13   }  
14 }
```




<https://reqres.in/>

<https://reqres.in/api/users>

Kayıtlı dataları gösterir.

<https://reqres.in/api/users/3>

3. kayıtlı datayı gösterir.

<https://reqres.in/api/users?page=1>

1. sayfadaki kullanıcıları gösterir.

<https://reqres.in/api/users?page=2>

2. sayfadaki kullanıcıları gösterir.

```
1  [
2    "page": 1,
3    "per_page": 6,
4    "total": 12,
5    "total_pages": 2,
6    "data": [
7      {
8        "id": 1,
9        "email": "george.bluth@reqres.in",
10       "first_name": "George",
11       "last_name": "Bluth",
12       "avatar": "https://reqres.in/img/faces/1-image.jpg"
13     },
14     {
15       "id": 2,
16       "email": "janet.weaver@reqres.in",
17       "first_name": "Janet",
18       "last_name": "Weaver",
19       "avatar": "https://reqres.in/img/faces/2-image.jpg"
20     },
21     {
22       "id": 3,
23       "email": "emma.wong@reqres.in",
24       "first_name": "Emma",
25       "last_name": "Wong",
26       "avatar": "https://reqres.in/img/faces/3-image.jpg"
27     },
28   ]
29 ]
```



<https://restful-booker.herokuapp.com/>

<https://restful-booker.herokuapp.com/booking>

Kayıtlı tüm datayı gösterir.

<https://restful-booker.herokuapp.com/booking/10>

Kayıtlı 10. datayı gösterir.

```
1  [
2    {
3      "bookingid": 45
4    },
5    {
6      "bookingid": 10
7    },
8    {
9      "bookingid": 6
10   },
11   {
12     "bookingid": 22
13   },
14   {
15     "bookingid": 44
16   },
17   {
18     "bookingid": 40
19   },
20  ]
```



<http://dummy.restapiexample.com/>

<http://dummy.restapiexample.com/api/v1/employees>

Kayıtlı işçilerin bilgilerini getirir.

<http://dummy.restapiexample.com/api/v1/employee/1>

1 numaralı çalışanın bilgilerini getirir.

```
1  [
2    "status": "success",
3    "data": [
4      {
5        "id": 1,
6        "employee_name": "Tiger Nixon",
7        "employee_salary": 320800,
8        "employee_age": 61,
9        "profile_image": ""
10     },
11     {
12       "id": 2,
13       "employee_name": "Garrett Winters",
14       "employee_salary": 170750,
15       "employee_age": 63,
16       "profile_image": ""
17     },
18     {
19       "id": 3,
20       "employee_name": "Ashton Cox",
21       "employee_salary": 86000,
22       "employee_age": 66,
23       "profile_image": ""
24     }
25   ]
26 }
```



JSON PATH

- JsonPath, Json Format ile verilen bir dataya ulaşmak veya değiştirmek için kullanılır.
- JSONPath, JSON verileri arasında bize sorgu yapma fırsatının verir.
- \$ isareti Json dokumanındaki tüm node'ları verir.
- Child bölümlere ulaşmak için (.) kullanılabilir. store.book bize tüm kitapları verir
- Belirli bir kitaba ulaşmak için; yapı array olduğundan index kullanabiliriz. store.book[1] , Birden fazla kitaba ulaşmak istersek virgule indexleri yazabiliriz. store.book[1,3]
- 2.kitabın price bilgisine ulaşmak için store.book[1].price kullanabiliriz
- Son kitaba ulaşmak için index olarak -1 kullanabiliriz.
- Tüm kitapların yazarlarını listelemek için store.book[*].author kullanabiliriz

<https://jsonpath.herokuapp.com/>

```
{
  "store": {
    "book": [
      {
        "category": "reference",
        "author": "Nigel Rees",
        "title": "Sayings of the Century",
        "price": 8.95
      },
      {
        "category": "fiction",
        "author": "Evelyn Waugh",
        "title": "Sword of Honour",
        "price": 12.99
      },
      {
        "category": "fiction",
        "author": "Herman Melville",
        "title": "Moby Dick",
        "isbn": "0-553-21311-3",
        "price": 8.99
      },
      {
        "category": "fiction",
        "author": "J. R. R. Tolkien",
        "title": "The Lord of the Rings",
        "isbn": "0-395-19395-8",
        "price": 22.99
      }
    ],
    "bicycle": {
      "color": "red",
      "price": 19.95
    }
  },
  "expensive": 10
}
```



NELER ÖĞRENDİK

- 1)
- 2)
- 3)





REST ASSURED

- Rest Assured, REST API'lerini test etmek ve doğrulamak için kullanılan bir Open Source (Açık Kaynak) bir Java kütüphanesidir.
- REST Assured, HTTP Builder üzerine kurulu REST tabanlı hizmetlerin test edilmesini basitleştirmek için bir Java DSL (Domain Specific Language: Özel bir uygulama alanı için kullanılan dil)'dir.
- 2010 yılında Johan Haleby tarafından literatüre kazandırılmış.
- Maven ile çok iyi entegre olur.
- XML ve JSON tabanlı web servislerini test etmek için kullanılabilir.



REST-assured



REST ASSURED

- GET, POST, PUT, PATCH, DELETE, OPTIONS, HEAD ... isteklerini destekler ve bu isteklerin yanıtını doğrulamak için kullanılabilir.
- Junit ve TestNG gibi test frameworkleri ile entegre edilebilir.
- Apache Http Client karşılaştırıldığında daha az kodlama gerektirir.
- API'ler için farklı kimlik doğrulama mekanizmaları için çok iyi destek. Kodu okunabilir kılan ve temiz kodlamayı destekleyen verilen(), while(), then() gibi BDD anahtar sözcüklerini takip eder.
- JSON ve XML yanıtını ayrıştırmaya yardımcı olan JsonPath ve XmlPath'i destekler. Rest Assured varsayılan olarak her ikisini de birleştirir.





JSON PATH

```
// https://restful-booker.herokuapp.com/booking url'ine
// accept type'i "application/json" olan GET request'i yolladigimda
// donecek cevap (response) icin
// 1) HTTP status kodunun 200
// 2) Content Type'in Json
// 3) Ve Status Line'in HTTP/1.1 200 OK
// oldugunu test edin.
```



API TEST

IntelliJ ile API testleri yapabilmek için POM Xml'e rest-assured, junit ve json dependency'lerinin yüklenmesi gerekir,

given: Genellikle, request'e basmadan önce contentType, body vb. tüm ön koşulları bu bölüme yazıyoruz.

when: get(), post(), put(), patch() ve delete() gibi göndermek istediğimiz http request yöntemini içerir.

then: istek gönderildikten sonra yapmak istediğimiz tüm doğrulamaları yazıyoruz.

and: Çoklu işlem yapılacaksa kullanılır.

```
public class GetRequest {  
  
    @Test  
    public void test01(){  
        given().contentType(ContentType.JSON).  
        when().get(s: "https://restful-booker.herokuapp.com/booking/3").  
        then().statusCode(200);  
    }  
}
```



API TEST

Response response = given().when().get(url);

given().when().get(url)

Response response

