API Nedir? Nasıl Çalışır?

API (Application Programming Interface), programcılık ve web tabanlı uygulamalar konusunda adı çok geçen bir kavramdır. Türkçede ise Uygulama Programlama Arayüzü olarak ifade ediliyor. API arayüzleri, çok çeşitli alanlarda ve çok çeşitli amaçlarla sıklıkla kullanılıyor. API, bir programın verilere, sunucu yazılımına veya diğer programlara ulaşabilmek için kullandığı bir bağlantı arayüzüdür. İki makinenin nasıl konuştuğunu belirleyen bir dizi kuralları içerir. Örneğin, bir E-ticaret sitesi, API sayesinde bir bankanın verdiği kredi kartından para çekilmesine izin vererek alışveriş yapılmasını sağlayabiliyor.

Ya da bir Linkedin App (Android veya iOS), bir API sayesinde Linkedin sunucularından size ait verileri çekip size gösterebiliyor. API ler yardımıyla daha önceden geliştirilmiş bir altyapıya ulaşıp, tekrar altyapı oluşturmaya gerek kalmadan onu kullanma şansına erişiyoruz. Özetle birçok özel veriyi, web üzerinde veya akıllı telefonlarımızla API'ler sayesinde kolayca alabiliyor ve bazı ilave özellikleri edinebiliyoruz.

API Çeşitleri Nelerdir?

API'ler çeşitli şekillerde sınıflanabiliyor. API'ler genel olarak Web tabanlı servislerde kullanılıyor. Şimdi isterseniz biraz da kullanım şekilleri ve mimarilerine göre nasıl sınıflandıklarına odaklanalım.

Kullanım Amaçlarına Göre API Çeşitleri

Dahili API'ler

Sadece belirli kişilerce veya belirli alanda kullanılan örneğin bir şirketin departmanları için tanımlanmış API'lere, dahili API denir.

Herkese Açık API'ler

Herkesin kullanımına açık olan API'lerdir. Open API olarak da isimlendirilirler. Sadece bir şirket veya kuruma özel tanımlanmış Dahili API'lerden burada farklılık gösterir.

Partner API'ler

Partner API'ler belirli iş ortaklığı ile iş yürüten şirketlerin ortak operasyonlarını yürütebilmek ve birbirlerinin sistemlerinin koordinasyon sağlayabilmesine olanak sağlarlar.

Mimarilerine Göre API Çeşitleri

REST API

REST API, HTTP protokolünü kullanarak çalışan bir API servisidir. Bu modelde oluşturulan API'ler URI (Uniform Resource Identifier) adreslerini kullanarak veri tabanlarına ulaşırlar. HTTP isteklerinden (Request) GET, POST, PUT, DELETE gibi istek komutları yardımıyla veri alışverişine imkân tanınır.

REST (Representational State Transfer) API'lerde veriler, JSON (JavaScript Object Notation) formatında taşınırlar. REST API, oldukça kullanışlı ve hafif bir yapıda oldukları için çok tercih edilen bir mimaridir.

SOAP

SOAP (Simple Object Access Protocol), yani Basit Nesne Erişim Protokolü daha sıkı bir güvenlik yapısıyla veri transferini sağlayan API mimarisidir. Bu mimaride veri akışı XML formatı kullanılarak sağlanır. Yapılandırması Rest API'ye göre daha zor olsa da bu mimari daha güvenli bir bağlantı sağladığı söylenir.

API Nasıl Çalışır?

API'ler yazılımlar veya veri tabanları arasında güvenli ve kontrollü bir kapı açmaya yarayan uygulama arayüzüdür.

API'nin çalışma şekli aşağıdaki gibidir.

- Alıcı program bir istek (API Çağrısı) yapar. Bu istek URI aracılığıyla Web sunucusuna istek fiili olarak işlenir.
- İstek alındıktan sonra API, harici program veya web sunucusuna çağrıda bulunur.
- Sunucu istenen bilgileri API'ye bir yanıt olarak gönderir.
- API, isteği yapan alıcı programa aldığı verileri aktarır.
- API çalışma sistemini bir de şu şekilde tanımlayabiliriz;

Büyük, çok çeşitli kitap ve dokümanların bulunduğu bir kütüphane düşünün. Örneğin işinizi yapabilmeniz için bu kütüphaneden bazı verileri düzenli olarak almanız gerek. Kütüphanenin bazı kat veya odaları özel veriler içerdiği için herkese açık değil ve bilgi edinmek isteyenlere yetki sınırlarını dahilinde sadece ilgili kapıları açan bir anahtar veriliyor diyelim.

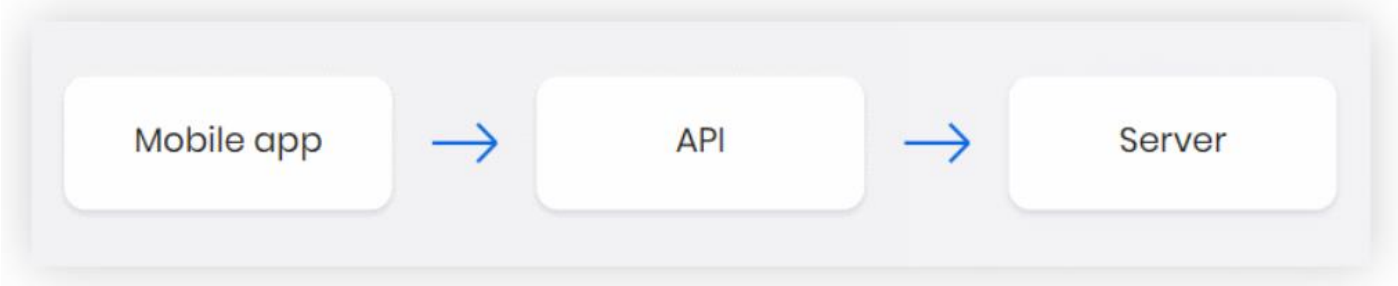
Kütüphane güvenlik birimi sizin yetkinize göre bir anahtar çıkarıyor ve size veriyor. Bu anahtar yardımıyla işinizi yapabileceğiniz kat veya odalara girerek bilgileri alabiliyor ve çalışmanızı yürütebiliyorsunuz. İşte burada size verilen anahtar API'yi temsil eder.

Bu kart sayesinde bir daha güvenlik birimine uğramadan size açılan bölümlerdeki verilere dilediğiniz şekilde ulaşabilirsiniz. Bu örneğe göre her anahtar sahibi ziyaretçi de istemci programlardır.

Son dönemlerde çok sık kullanılan İnternet bankacılığı uygulamaları da benzer şekilde API'ler yardımıyla çalışır. Akıllı telefonunuza kurduğunuz aplikasyon, bankanın tanımladığı API yardımıyla banka hesap bilgileriniz arasında bir köprü oluşturur.

API entegrasyonu sonrası sizin banka hesabınızda işlem yapmanıza izin verir. Normal şartlarda bankanın işletim sistemi ve kullandığı yazılım sizin telefonunuzdaki işletim sistemi ve yazılımdan farklıdır.

API bu farklı işletim sistemi ve yazılımlar arasında iletişim kurmaya ve sorunsuz bir şekilde işlem yapılmasına olanak tanır. API'ler sayesinde programlama dili veya cihazlar farklılık gösterse bile işlemler çalışır ve bir köprü oluşturulur.



API'leri birbirini tanımayan iki kişiye (İki farklı uygulama veya sunucu), ikisini de tanıyan bir kişinin (API) bilgi taşıması olarak da tarif edebiliriz.

API Nasıl Kullanılır? API Entegrasyonu Nasıl Yapılır?

API'leri verdiğiniz hizmette kolaylık sağlamak için kullanılabilir. Sisteminize entegre ettiğiniz, örneğin anlık döviz kuru takip eden bir API ile verdiğiniz hizmeti anlık kur fiyatlarına göre otomatik güncelleyebilirsiniz ve ödemelerin otomatik bir şekilde güncel kur verilerine göre yapılmasını sağlayabilirsiniz.

Aşağıda GET metodu ile oluşturulmuş API (Çağrısı) İsteği aşağıdaki gibi görünür.

Request	URL
origins: Vancouver+BC|Seattle destinations: San+Francisco|Victoria+BC mode: driving key: API_KEY	https://maps.googleapis.com/maps/api/distancematrix/json?origins=Vancouver+BC|Seattle&destinations=San+Francisco|Victoria+BC&key=YOUR_API_KEY

Bir örnek API mimarisi örneğide aşağıdaki gibi olmaktadır. API'nin hangi amaçlarla kullanıldığı ve ne gibi kabiliyetler edinildiği görebileceğiniz bu örnek ile bir API Gateway sayesinde neler yapılabileceği görülüyor.

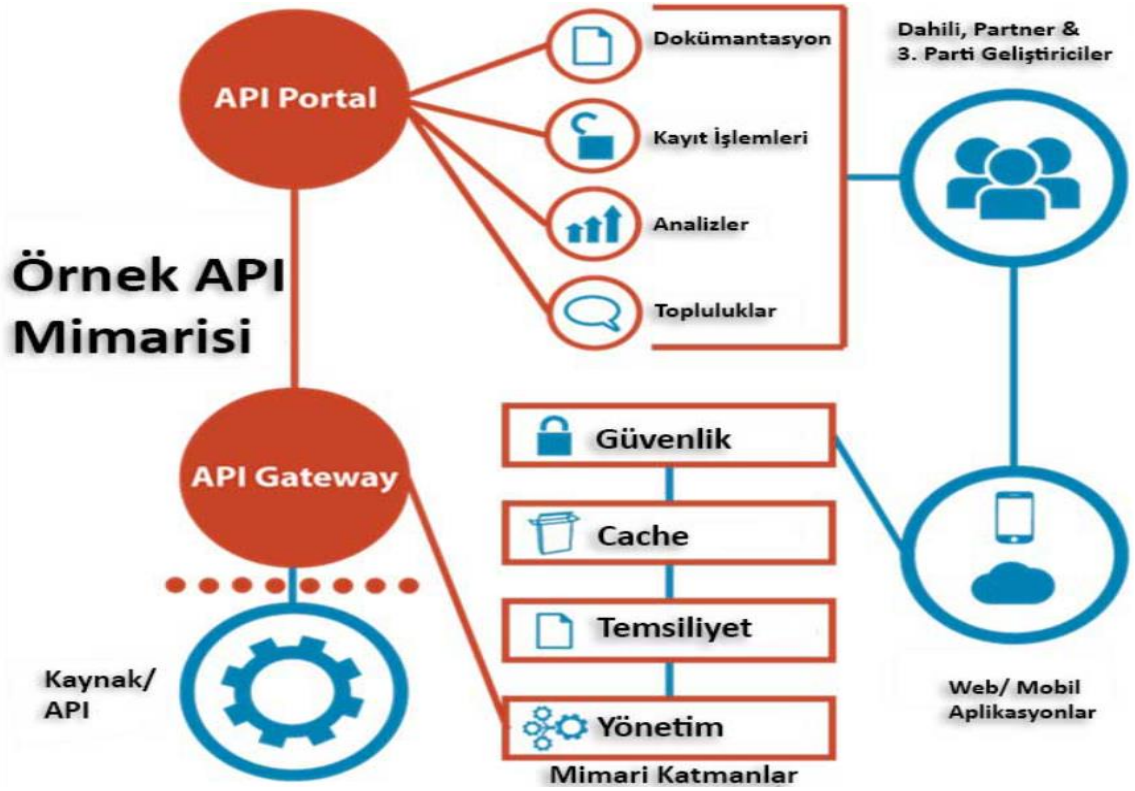
API entegrasyonu yapılabilmesi için istemcinin kendisini tanıtacak bir API key yani anahtara ihtiyacı vardır. API hizmetini veren uygulama veya sunucu, istemciden API key ve şifre ister.

Şifre ile Authentication (Yetkilendirme) alan uygulama kendisini tanımlayan anahtar vasıtasıyla sunucuyla sürekli bir bağlantı kanalı oluşturur. Böylece düzenli veri alışverişi gerçekleşmeye başlar.

API Entegrasyonunu oluşturabilmek için bu API Key ve şifreleri ihtiyacınız olduğunu söylemiştik. Entegrasyon işlemi Backend tarafında gerçekleştirilir. İstemciye verilecek API Key veritabanında oluşturulan bir tabloda saklanır.

Belirlenen API Key ve şifre ile Authentication oluşturulduktan sonra istemci örneğin HTTP isteği yaparken Header ve gövdeye API Key'i ekleyerek göndermelidir.

Bu key sayesinde istemcinin kimliği tanımlanır ve daha önceden Authentication sağlandığı için güvenli bağlantı sağlanarak talep ettiği veriler istemciye açılır. İsteğe bağlı olarak ilave güvenlik önlemi sağlayabilmek adına, istemcinin IP adresi de istenebilir bu sayede API Keylerini başkalarının kullanabilmesinin önüne geçilmesi sağlanır.



Postman Nedir?

Postman, API geliştirme için bir iş birliği platformudur. Postman'ın özellikleri, bir API oluşturmanın her adımını basitleştirir ve iş birliğini kolaylaştırarak daha iyi API'leri daha hızlı oluşturabilmenizi sağlar.

Temel özellikleri şu şekildedir:

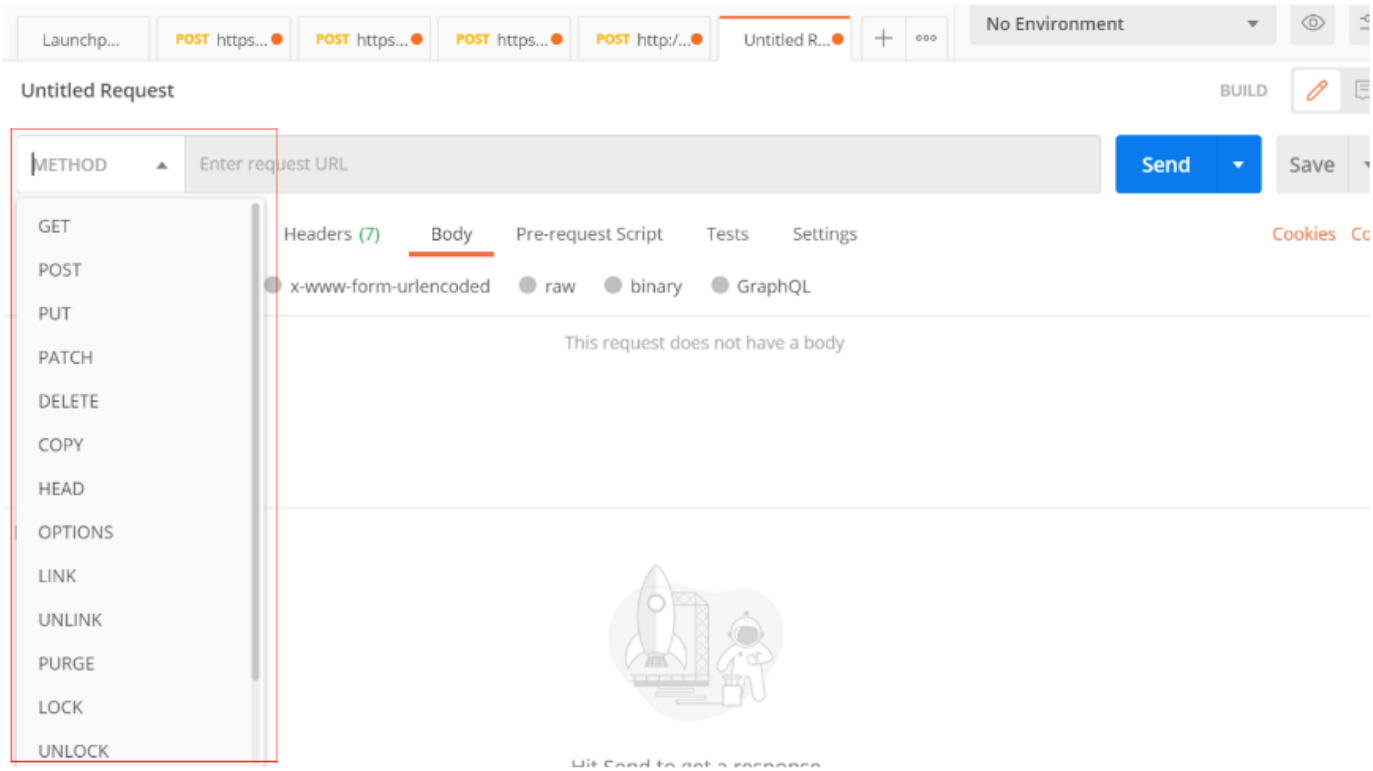
- **Api Client**: Postman ile hızlı ve kolay bir şekilde Rest ve Soap istekleri oluşturabilirsiniz. Client yerine kullanabilirsiniz.
- **Automated Tests:** Testler, tekrar tekrar çalışabilen test grupları oluşturularak otomatik hale getirilir. Postman; birim testleri, fonksiyonel testler, entegrasyon testleri, uçtan-uca testler, regresyon testleri vb.. dahil olmak üzere birçok test türünü otomatikleştirmek için kullanılabilir. Otomatik test, insan hatasını önler ve testi kolaylaştırır.
- **Documentation:** Postman, dökümanlarınınızı hızlı ve kolay bir şekilde yayınlamanıza olanak tanır. Postman, dokümantasyon sayfanızı dinamik örneklerle ve makine tarafından okunabilir talimatlarla doldurmak için örnek requestlerinizi otomatik olarak çeker, böylece API'nizi dünyanın geri kalanıyla kolayca paylaşabilirsiniz.

Postman, Chrome uzantısı olarak kullanabileceğimiz veya direct indirip bilgisayarımıza yükleyebileceğimiz bir uygulamadır. Rest Client olarak da tanımlanabilir.

API (Application Programming Interface) farklı uygulama yazılımlarının birbirleri ile etkileşim sağlamasına olanak sağlar. Client (Android) ve Backend(java) yazılımlarının Restfull Api ile iletişim kurması buna örnek verilebilir.

Postman sayesinde uzun uzun kodlar yazmak yerine API'lerimizi kolayca test edebiliriz. Birçok özelliği sayesinde kolay bir şekilde istek hazırlayıp gelen cevap değerlerini kullanabiliriz.

POST-GET-PUT-DELETE



Yukarıdaki resimin sağ tarafında kırmızı ile işaretlenen yerde istek metod türlerimiz bulunmakta

En çok kullanılan GET, POST, PUT, DELETE metodlarını ne zaman kullanmalıyız aşağıda açıklıyor olacağım.

Note: CRUD operasyonları Create, Read, Update, Delete olarak adlandırılıyor.

1) GET: Sunucudan sadece veri çekmek(okuma) istiyorsak yani veri üzerinde herhangi bir değişiklik (ekleme, slime, modifiye) yapılmayacaksa GET metodunu kullanmamız tavsiye ediliyor.

CRUD operasyonlarından Read'e karşılık geldiğini söyleyebiliriz.

Ör: GET /students kullandığımızda bize öğrenciler listesini dönmesi.

2) POST: Server Api'e body kısmını doldurarak ve veri üzerinde değişiklik yapmak istediğimizde kullanabiliriz

Değişilik yapmak ile kastedilen CRUD operasyonlarından Create ve Update kısımlarını kapsar.

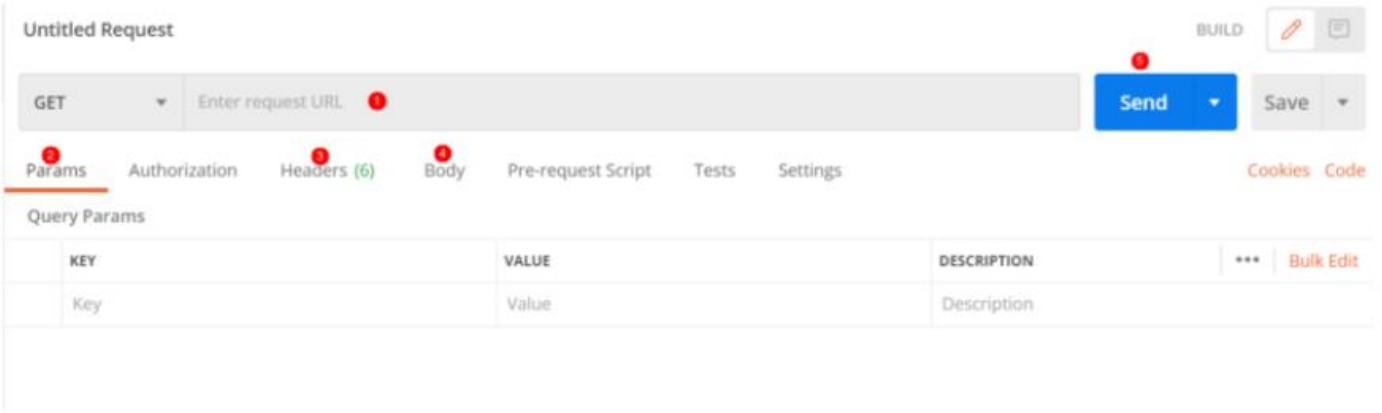
Ör: Post /createUser ile body kısmına kullanıcı bilgileri girip veritabanında bir kullanıcı oluşturulması istenmesi

3) PUT: Post isteğinin özelliklerine sahiptir. Yani CRUD operasyonlarından Create ve Update operasyonlarını yapmak istediğimizde kullanıyoruz.

Post'dan ayrılan tarafı Put isteğinin idempotent ve not cacheable olarak tanımlanması

4) DELETE: CRUD operasyonlarından Delete'e karşılık gelir. Bir veriyi silmek istediğimizde kullanılması tavsiye ediliyor.

PARAMS-HEADER-BODY-PATH-SEND



Yukarıdaki resimde Postman body kısmında bulunan numaralandırılmış olan yerlerden bahsedeceğim.

Bir istek hazırladığımız zaman genellikle aktif olarak bu 5 alan kullanılıyor

1 numara ile gösterilen yer Api Url girdiğimiz yer.

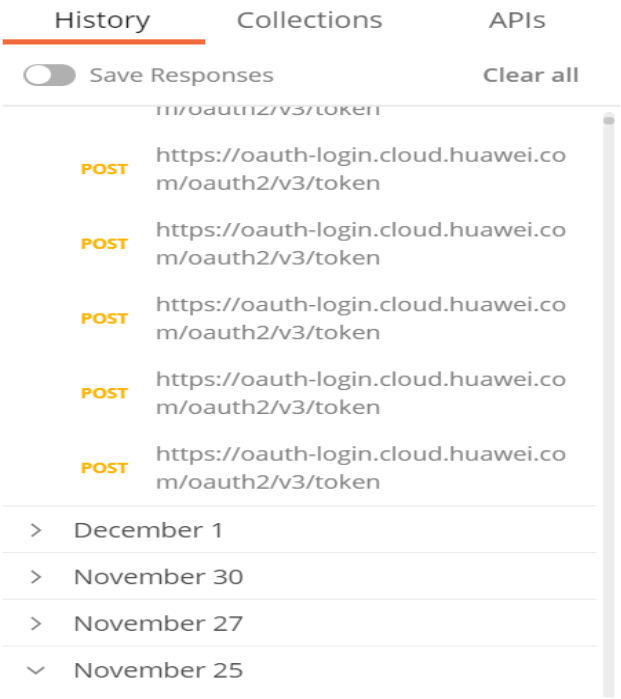
2 numara ile gösterilen yer Params değerlerini tanımladığımız alan

3 numara ile gösterilen yer Headers değerlerini tanımladığımız alan

4 numara ile gösterilen yer Body kısmını dolduracağımız alan raw, binary gibi seçenekler mevcut

5 numara ile gösterilen yer isteğimizi hazırladığımızda Send'e basarak isteğimizi atabiliyoruz

HISTORY



Postman'nin sol tarafında bulunan History sekmesi önceden kullandığımız istekleri gün gün tutuyor

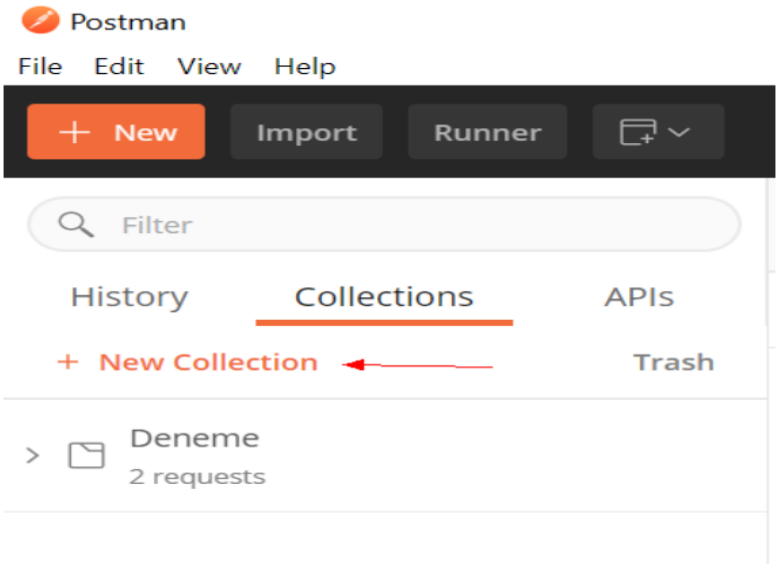
Herhangi birine tıklayınca yeni sekmede tıklanılan istek açılıyor

COLLECTİON

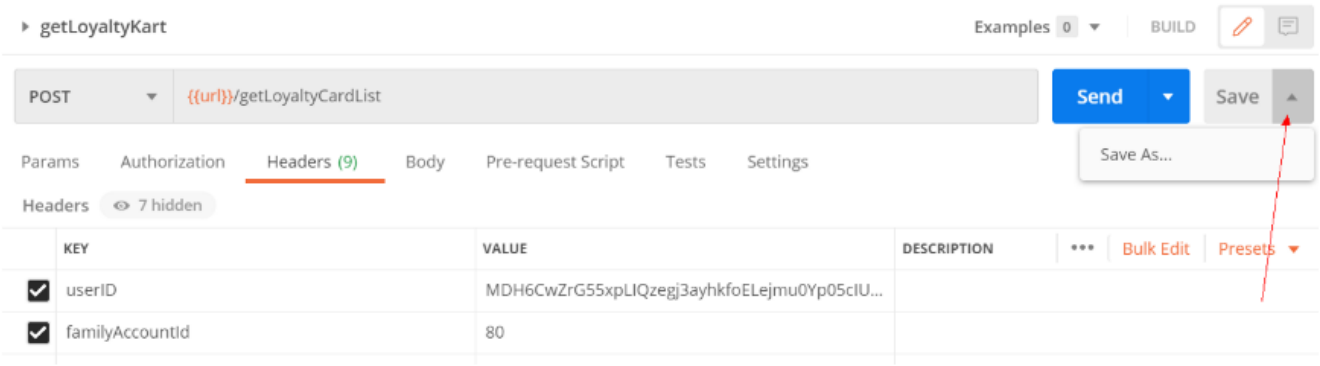
Collection bize birden fazla isteği grup halinde kaydetmemizi, gerektiğinde başkalarıyla paylaşmamızı sağlıyor. Bu sayede bir projede çalışırken hali hazırda kullandığımız istekleri düzenli bir şekilde elimizin altında tutmamızı sağlıyor.

Öncelikle aşağıdaki gibi New Collection sekmesinden yeni bir Collection oluşturuyoruz.

Collection ismini verip Create demeniz yeterli



Yeni oluşturduğumuz Deneme Collection'a eklemek üzere bir istek hazırlıyoruz aşağıdaki resim gibi ve Save buton'nun yanındaki ok'a tıklayıp Save As… diyoruz.



Save as ekranından istek ismini ve gerekiyorsa açıklamasını girip hangi Collection'a kaydedilmesi gerektiğini seçiyoruz. Sonrasında Deneme Collection'nın altında eklediğimiz isteği görebiliriz.

Ek olarak Deneme Collection'nın altına add folder diyerek GET, POST, DELETE gibi dosyalar açıp oluşturduğumuz istekleri bu dosya düzenine göre yerleştirmek faydalı olacaktır.

Import Özelliği

Genelde hazır, kayıtlı bir postman collection'ını yüklemek için kullanılır. Ama test etmek istediğiniz API'ınız varsa ve swagger vb. bir araç ile doküman haline getirilmişse, **import** alanındaki link özelliği ile tüm endpointleri Postman'e hızlıca kaydedebilirsiniz.

Örnek olarak swagger'in test API'yı olarak sağladığı

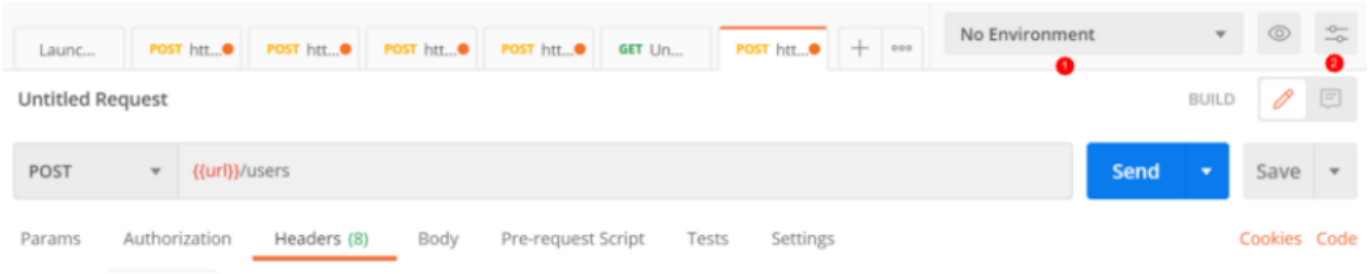
https://petstore.swagger.io/ url'indeki

https://petstore.swagger.io/v2/swagger.json linkini kullanıyorum.

Daha sonra Confirm sekmesinden istediğim özelliklere göre **import** işlemimi tamamlıyorum.

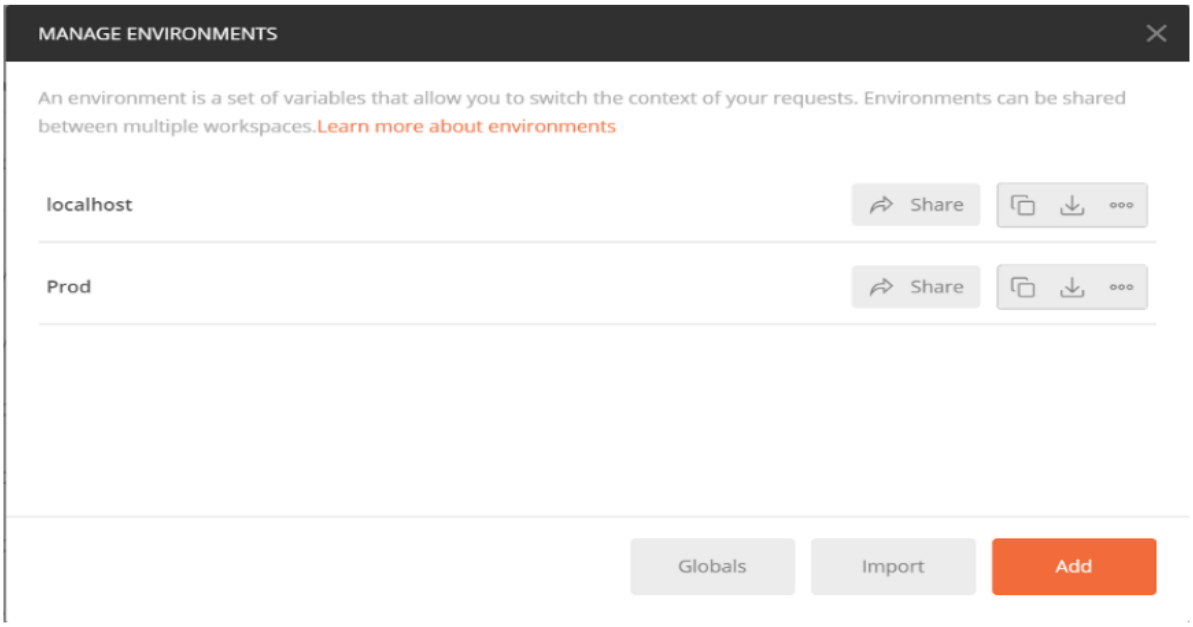
Tabi bu seçeneklerden bağımsız olarak **New** butonuyla yeni sorgularınızı oluşturabilirsiniz.

Environment



Environment tanımlamadaki amacımız sabit değişkenleri belirlemek.

Environment tanımlamak için 2 ile gösterilen sekmeyi tıklayıp Manage Environments'a geçiyoruz.



Add tıklıyoruz ve aşağıdaki örnekteki gibi environment'larımızı tanımlıyoruz.

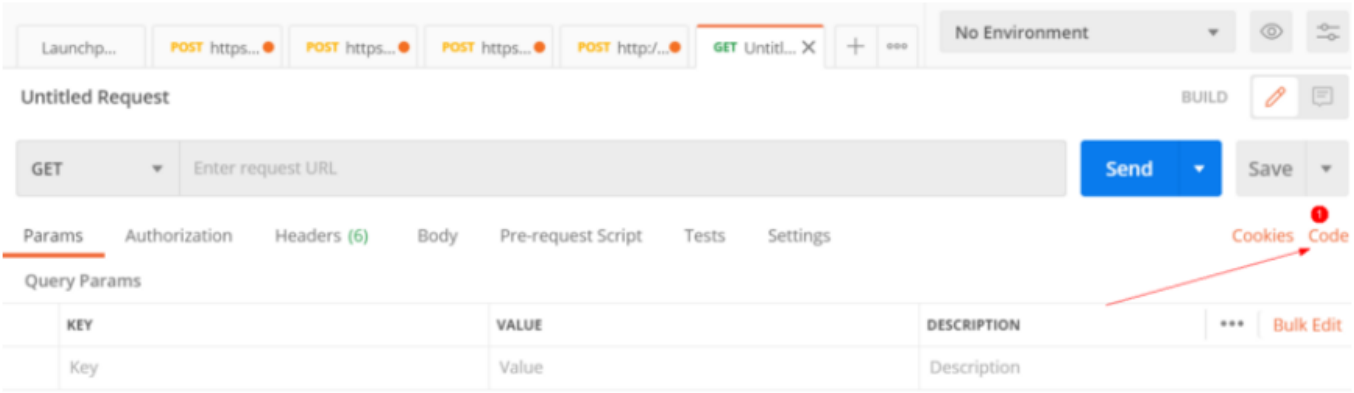
Ör: Bir environment tanımladım ismi: Localhost, variable: url, value: http://localhost:8080

Bir environment daha tanımladım ismi: Prod, variable: url, value: http:// 159.159.59.159

Artık Api url kısmında {{url}}/users dediğimde ve 1 numara ile gösterilen yerden Localhost seçtiğimde {{url}}/users, http://localhost:8080/users olarak davranıyor.

Benzer şekilde Prod seçersem http:// 159.159.59.159/users olarak algılanıyor

Code



Yukarıdaki resimde görüldüğü üzere Postman sağ tarafında Code diye bir seçenek var.

Bu seçenek bize hazırladığımız isteğin koda çevrilmesini sağlıyor.

Code buton'a bastığımızda bizi yukarıdaki resim karşılıyor.

Resmin sol üstte kırmızıya alınan yerden kodlama yapacağımız dili seçebiliyoruz.

Örnekte Java dili OkHttp kütüphanesi kullanılarak, bizim hazırladığımız isteğe karşılık gelen kodu sağ tarafta görebiliyoruz.

Bu özellik bize Postman'de istek hazırladım çalıştırdım ama kod tarafında nasıl yapmalıyım sorusunun cevabını veriyor.

KAYNAK:

- https://www.hosting.com.tr/bilgi-bankasi/api/
- https://app.patika.dev/courses/net-core/2-postman-nedir-nas%C4%B1l-kullan%C4%B1l%C4%B1r
- https://medium.com/huawei-developers-tr/postman-nedir-83eeaa5ed6ac
- https://omeratli.medium.com/api-testi-i%C3%A7in-postman-kullan%C4%B1m%C4%B1