



Ulak

Kullanıcı Kılavuzu

Ulak, yerel çalışan bir API istemcisidir. Postman'de yaptığınız işleri (koleksiyon, environment, istek gönderme, token alma) daha hızlı ve sade bir uygulamada yapar. Bu kılavuz kurulumdan günlük kullanıma kadar her adımı ekran görüntüleriyle anlatır. Teknik bilgi gerekmez.

İçindekiler

- 1 Ulak nedir?
- 2 Kurulum ve ilk açılış
- 3 Ekranı tanıyalım
- 4 İçerik aktarma: Postman, Swagger, cURL
- 5 İlk isteğinizi gönderin
- 6 Değişkenler ve environment
- 7 Login ile token almak
- 8 Digest ve OAuth 2.0
- 9 Eksik değer uyarıları
- 10 Cevabı okumak ve hatalar
- 11 Akış halindeki cevaplar
- 12 WebSocket
- 13 Geçmiş ve hızlı arama
- 14 Tema, dil ve ayarlar
- 15 Klavye kısayolları
- 16 Sık sorulanlar ve sınırlar

1. Ulak nedir?

Ulak, web servislerine (API) istek gönderip cevabını görmenizi sağlayan bir masaüstü uygulamasıdır. Postman'i bilen biri Ulak'ı birkaç dakikada kullanmaya başlar: aynı kavramlar, aynı dosyalar.

Ne işe yarar?

- İstekleri **koleksiyon** ve **klasör** halinde düzenler.
- Aynı istekleri farklı ortamlara (Lokal, Test, Canlı) **environment** seçerek gönderir.
- Login isteğinden gelen **token**'ı otomatik saklar ve diğer isteklerde kullanır.
- Postman ve Swagger / OpenAPI dosyalarını **kayıpsız** içe ve dışa aktarır.
- Canlı ortama gönderilecek isteklerde sizden **onay** ister.
- **OAuth 2.0** ve **Digest** ile korunan servislere bağlanır, token'ı kendisi alır ve yeniler.
- Canlı akışları (**SSE**) ve **WebSocket** bağlantılarını izler, mesaj gönderir.

Neden Ulak?

- **Hızlı:** açılışı ve gönderimi beklemezsiniz. Bir istek çalışırken başkasını gönderebilirsiniz.
- **Yerel:** hesap açmak, internete veri göndermek yok. Her şey kendi bilgisayarınızda.
- **Türkçe ve İngilizce:** arayüz iki dilde kullanılabilir.

2. Kurulum ve ilk açılış

Kurulum

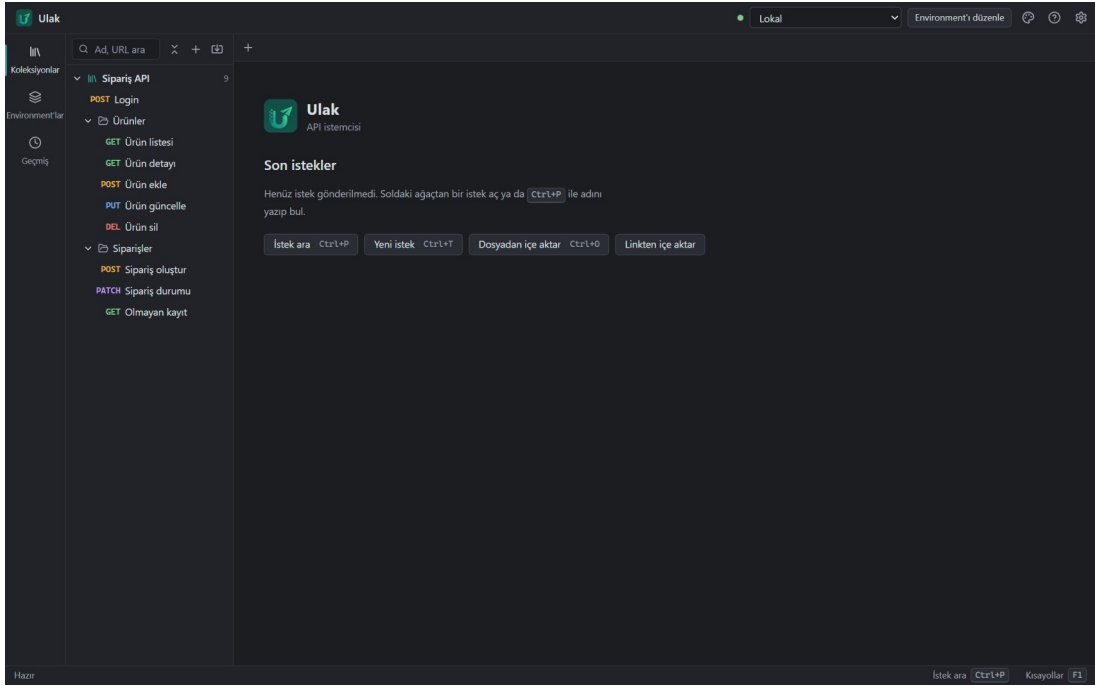
İki seçenek var. İkisi de yönetici yetkisi istemez.

- **Kurulumlu:** Ulak_1.0.0_x64-setup.exe dosyasını çalıştırın. Ulak, Başlat menüsüne eklenir.
- **Kurulumsuz:** ulak.exe dosyasını istediğiniz klasöre koyun ve çift tıklayın.

Verileriniz nerede? Koleksiyonlar, environment'lar ve geçmiş %LOCALAPPDATA%\Ulak klasöründe durur. Ayarlar penceresindeki **Veri klasörünü aç** düğmesi bu klasörü açar.

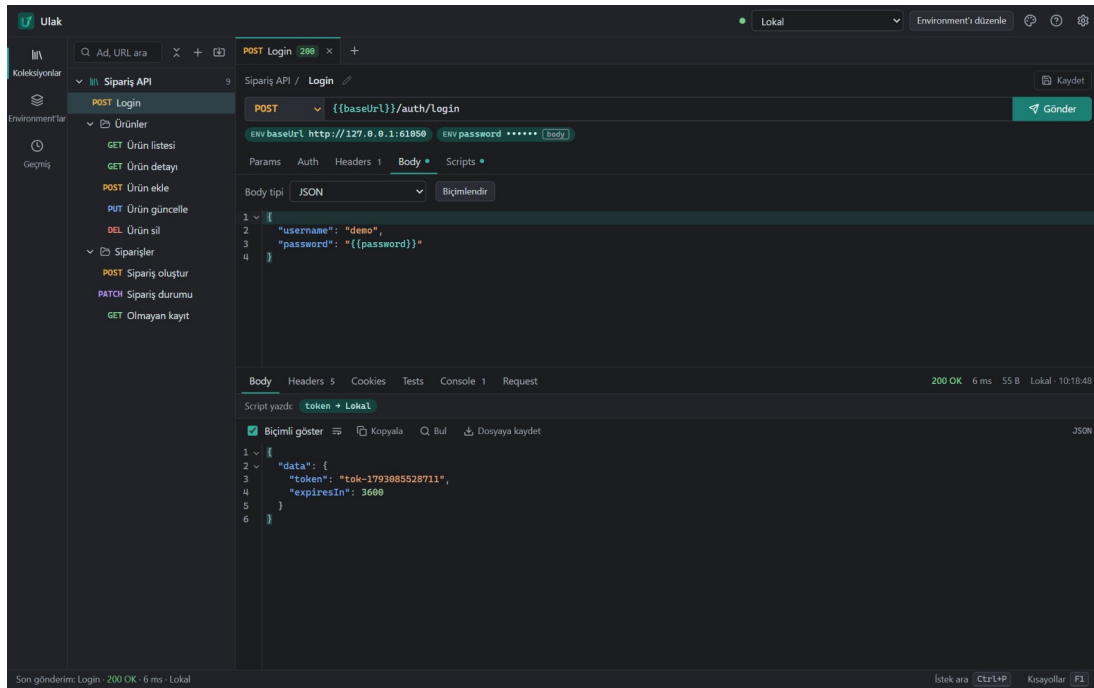
İlk açılış

Ulak ilk açıldığında koyu temayla ve sistem dilinizle gelir. Sol tarafta koleksiyonlarınız, sağda başlangıç ekranı görünür. Başlangıç ekranındaki düğmelerle istek arayabilir, yeni istek açabilir ya da dosya içe aktarabilirsiniz.



Ulak'ın başlangıç ekranı: solda koleksiyon ağacı, sağda hızlı başlangıç düğmeleri.

3. Ekranı tanıyalım



Bir isteğin açık olduğu ekran.

Ekran beş bölümden oluşur:

1. **Üst çubuk:** aktif environment seçimi, Environment'ı düzenle, tema, yardım ve ayarlar. Environment'ın yanındaki renkli nokta ortamın türünü söyler: yeşil lokal, sarı test, kırmızı canlı.
2. **Sol menü:** Koleksiyonlar, Environment'lar ve Geçmiş arasında geçiş yapar. Üstteki arama kutusu ad ya da URL ile arar.
3. **Sekmeler:** açık istekler. Turuncu nokta kaydedilmemiş değişiklik, yeşil sayı son cevabın durum kodudur.
4. **İstek alanı:** metot, URL, Gönder düğmesi ve altında Params, Auth, Headers, Body, Scripts sekmeleri.
5. **Cevap alanı:** gelen cevabın gövdesi (Body), başlıkları (Headers), çerezleri, testleri ve gönderilen isteğin kendisi.

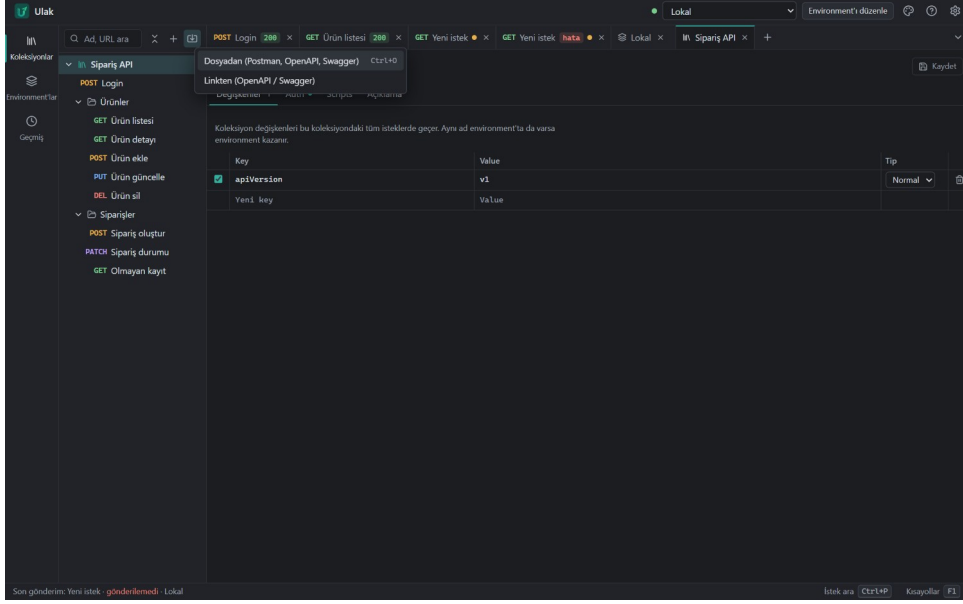
En alttaki çubuk son gönderimi özetler: istek adı, durum, süre ve environment.

İpucu Postman'deki **Key, Value, Body, Params, Headers** gibi terimler Ulak'ta da İngilizce bırakıldı. Postman'den alıştığınız adlarla arayabilirsiniz.

4. İçe aktarma: Postman, Swagger, cURL

Postman'den geçiş

1. Postman'de koleksiyona sağ tıklayın → **Export** → **Collection v2.1**. Environment'lar için Environments → ... → **Export**.
2. Ulak'ta sol üstteki **İçe aktar** düğmesine basın (ya da **Ctrl+0**) ve **Dosyadan**'ı seçin. Birden çok dosyayı aynı anda seçebilirsiniz.
3. Dosyaları pencerenin üstüne sürükleyip bırakmak da olur.



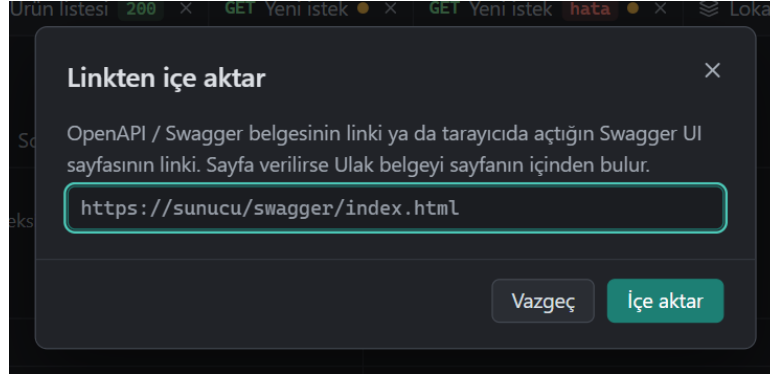
İçe aktar menüsü: dosyadan ya da linkten.

Ulak dosyanın türünü kendisi tanır: koleksiyon, environment, globals ya da Swagger / OpenAPI. Aynı koleksiyonu ikinci kez içe aktarırsanız eskisi silinmez, kopya olarak eklenir.

Swagger / OpenAPI linkinden

Servisinizin Swagger sayfası varsa linkini vermeniz yeterli. Ulak belgeyi bulur ve her uç noktayı bir istek olarak ekler.

1. **İçe aktar** → **Linkten**'i seçin.
2. Swagger sayfasının ya da belgenin adresini yapıştırın (örnek: <https://sunucu/swagger/index.html>).
3. **İçe aktar**'a basın. Ulak belgeyi indirir, koleksiyonu oluşturur.



Linkten içe aktarma penceresi.

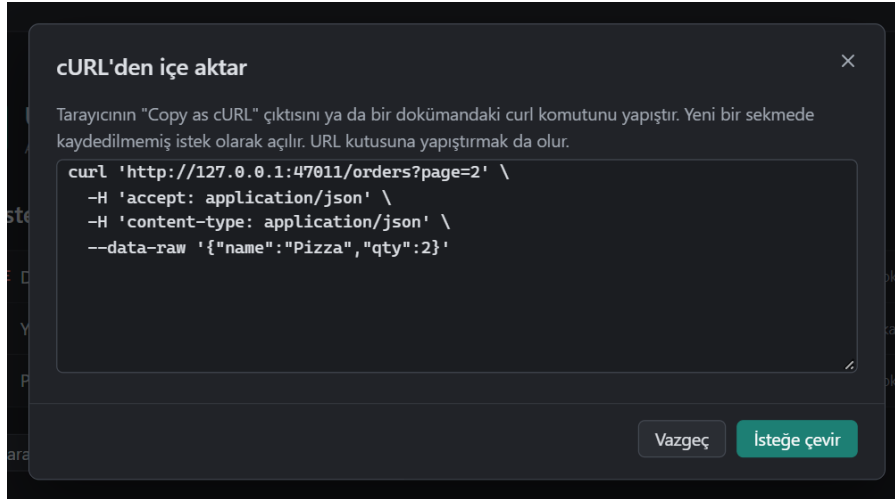
Swagger'dan gelen koleksiyonda her **tag** bir klasör olur. Sunucu adresi **baseUrl** adlı değişkene yazılır. Token gerekiyorsa koleksiyonun Auth ayarı hazır gelir; siz sadece token değerini girersiniz.

Dışa aktarma Koleksiyona sağ tıklayıp **Dışa aktar** dersiniz Postman biçiminde, **Dışa aktar (OpenAPI 3 / Swagger)** dersiniz Swagger biçiminde kaydeder.

cURL komutundan

Tarayıcının geliştirici araçlarında bir isteğe sağ tıklayıp **Copy as cURL** dersiniz ya da bir dokümanda `curl ...` komutu görürseniz, onu doğrudan isteğe çevirebilirsiniz.

- **En hızlısı:** komutu URL kutusuna yapıştırın. Metot, adres, header'lar ve gövde kendiliğinden dolar.
- **Yeni sekmede:** İçe aktar → **cURL komutundan** (ya da **Ctrl+P** → "cURL'den içe aktar"), komutu yapıştırıp **İsteğe çevir**'e basın.

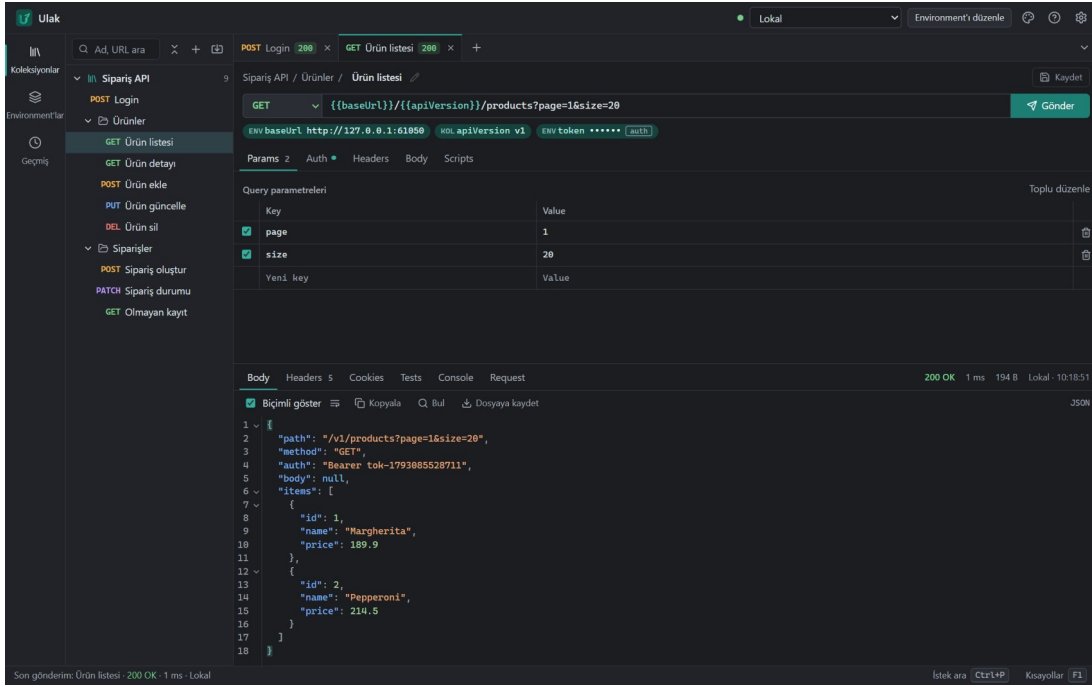


cURL'den içe aktarma penceresi.

Kullanıcı adı ve şifre (-u) Auth sekmesine Basic ya da Digest olarak geçer. Karşılığı olmayan bayraklar (örneğin -k) bildirimde "atlanmalar" olarak yazılır.

5. İlk isteğinizi gönderin

1. Soldaki ağaçtan bir isteğe tıklayın. Tek tık önizleme olarak açar; çift tık kalıcı sekmede açar.
2. Üst çubuktan doğru environment'ı seçin (örnek: **Lokal**).
3. **Gönder** düğmesine basın ya da **Ctrl+Enter**.
4. Cevap aşağıda görünür. Sağ üstte durum (200 OK), süre ve boyut yazar.



Ürün listesi isteği gönderildi; cevap aşağıda.

Yeni istek ve kaydetme

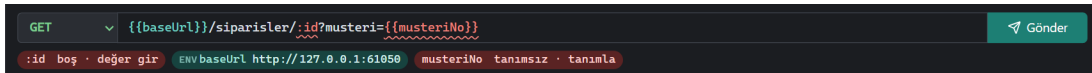
- Yeni istek için **Ctrl+T** ya da sekme çubuğundaki **+**.
- Kaydetmek için **Ctrl+S**. İstek henüz bir koleksiyonda değilse hangi koleksiyon ve klasöre kaydedileceği sorulur.
- İsteği çoğaltmak için ağaçta sağ tık → **Çoğalt** (ya da **Ctrl+D**).
- Yeniden adlandırmak için ağaçta **F2**. Ad satırın içinde düzenlenir; Enter kaydeder.
- Silinen istek, ekrandaki bildirimdeki **Geri al** ile ya da **Ctrl+Z** ile geri gelir.

6. Değişkenler ve environment

Değişkenler, adreste ya da gövdede tekrar eden değerleri tek yerde tutar. Yazılışı Postman'deki gibidir: `{{baseUrl}}`.

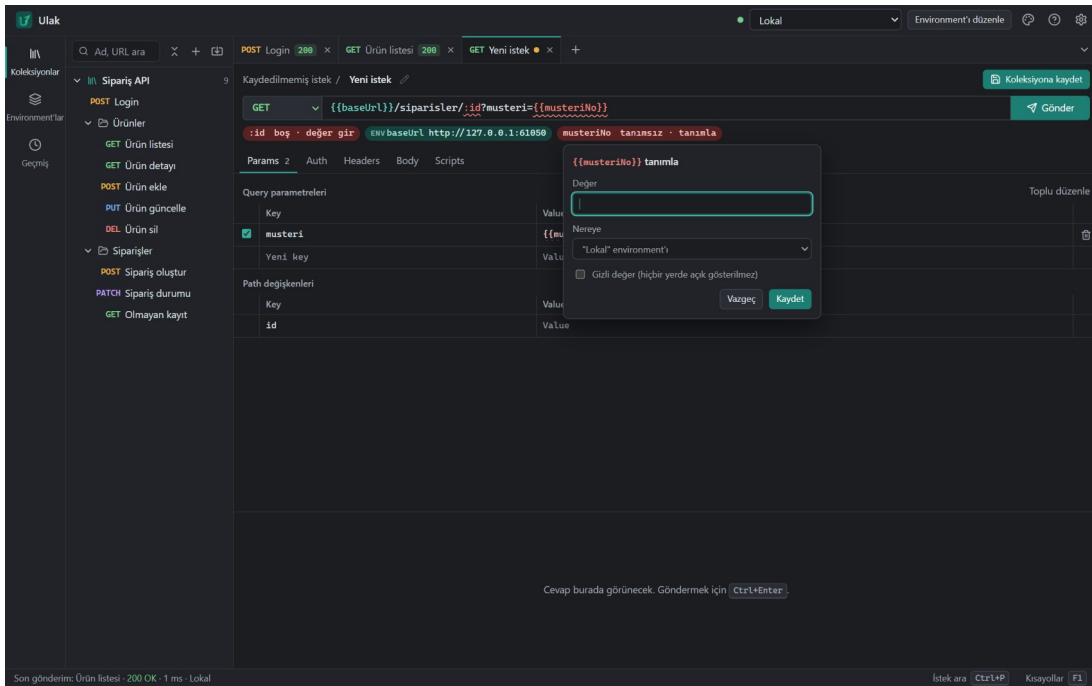
URL altındaki çipler

Bir istekte kullanılan her değişken, URL'nin hemen altında küçük bir çip olarak görünür. Çip değişkenin o anki değerini ve nereden geldiğini gösterir.



URL altındaki çipler.

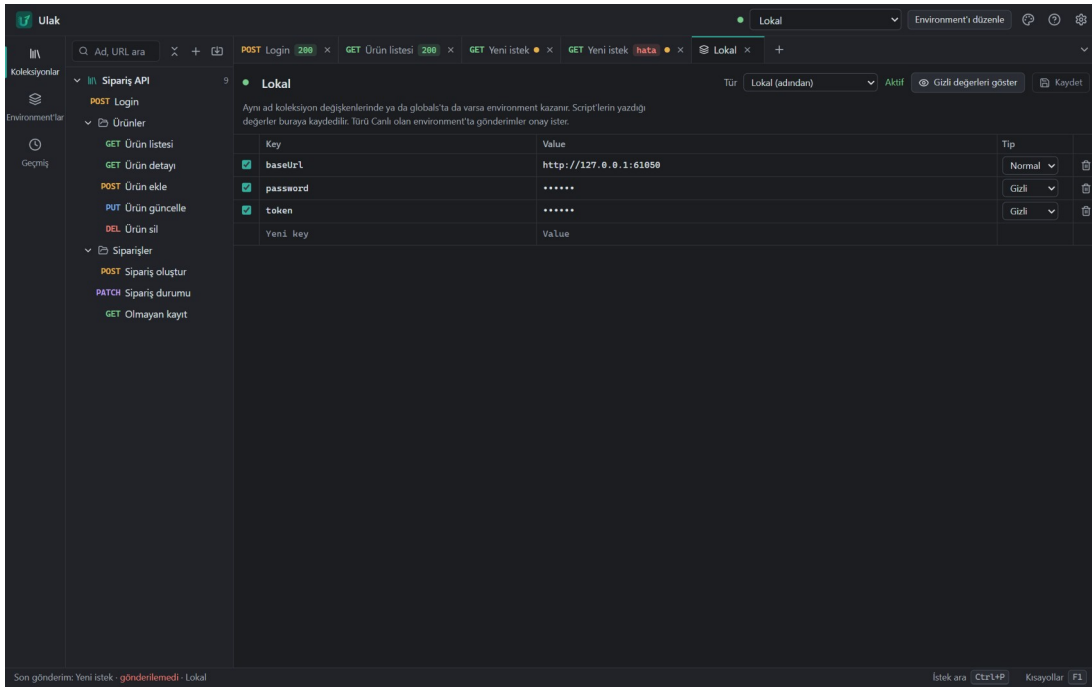
- **Yeşil çip:** değişken tanımlı. Baştaki küçük etiket nereden geldiğini söyler: env (environment), kol (koleksiyon), glob (globals).
- **Kırmızı çip:** değişken tanımsız ya da boş. Üstüne tıklayınca küçük bir panel açılır, değeri oradan girersiniz.
- **Boş `:id`:** adresteki path değişkeninin değeri yoksa kırmızı görünür. Çipe tıklayınca Params'taki alana gider.



Kırmızı çipe tıklayınca açılan panel: değer, nereye kaydedileceği, gizli mi.

Environment

Environment, aynı isteklerin hangi ortama gideceğini belirler. Örneğin **Lokal**'de `baseUrl` sizin bilgisayarınızı, **Test**'te test sunucusunu gösterir. Üst çubuktan environment değiştirtince bütün istekler yeni adrese gider.



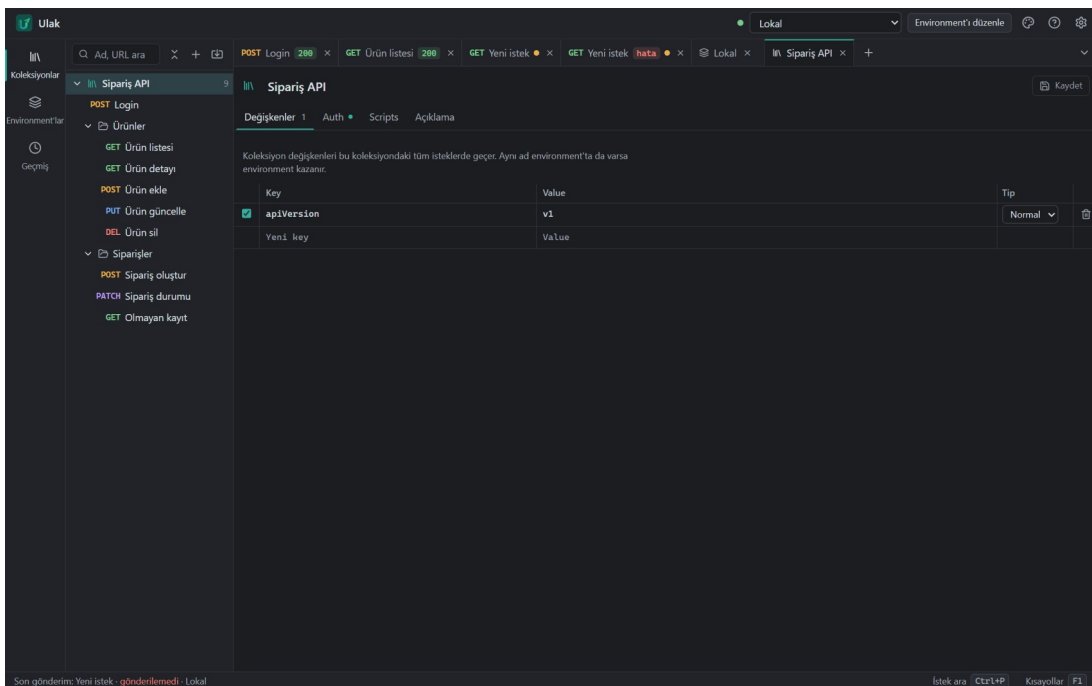
Environment düzenleme ekranı.

- **Tür:** Lokal, Test ya da Canlı. Ulak bunu addan tahmin eder; isterseniz kendiniz seçersiniz.
- **Gizli değerler:** şifre ve token gibi değerlerin Tip'ini **Gizli** yapın. Hiçbir yerde açık yazılmaz.
- Değişiklikler **Kaydet**'e basana kadar gönderimde kullanılmaz.

Hangi değer kazanır? Aynı ad birden çok yerde varsa sıra şöyledir: globals < koleksiyon < environment. Yani environment'taki değer koleksiyondakini ezer.

Koleksiyon ayarları

Koleksiyon satırındaki ayar simgesi koleksiyonun değişkenlerini, ortak Auth ayarını ve script'lerini açar. Buradaki Auth, içindeki bütün isteklere geçer (istek **Üstten devral** seçiliyse).



Koleksiyon ayarları: değişkenler, Auth, Scripts.

7. Login ile token almak

Çoğu API önce giriş yapıp bir token almanızı ister. Ulak bu token'ı otomatik saklayıp diğer isteklerde kullanabilir. Bir kez kurulur, sonra kendiliğinden çalışır.

1. Login isteğini açın, **Scripts** → **Post-response** bölümüne şu iki satırı yazın:

```
const body = pm.response.json();  
pm.environment.set("token", body.token);
```

2. Koleksiyon ayarlarında **Auth** → **Bearer Token** seçin, değere {{token}} yazın.
3. Login isteğinin kendi Auth'unu **Auth yok** yapın.
4. Login'i bir kez gönderin. Cevabın üstünde "Script yazdı: token → Lokal" yazısını görürsünüz.

Artık koleksiyondaki diğer istekler token'ı kendiliğinden kullanır. Token uygulama kapansa da kaybolmaz.

Kolaylık Token yoksa ve bir isteği gönderirseniz Ulak bunu fark eder ve **Önce "Login", sonra gönder** seçeneğini sunar.

8. Digest ve OAuth 2.0

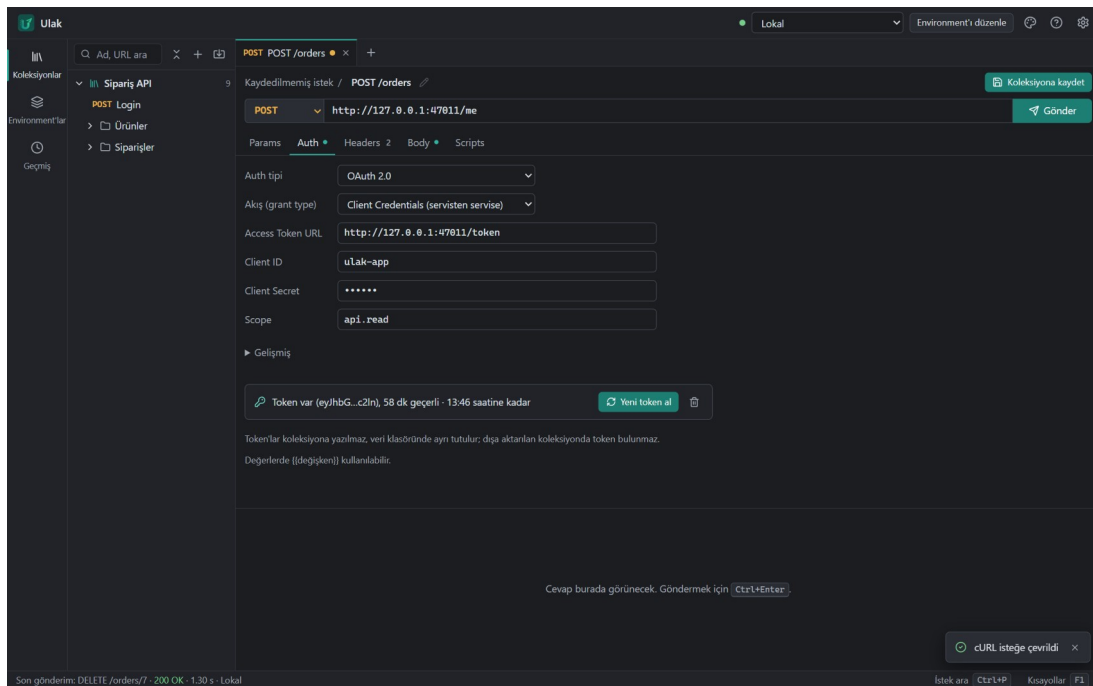
Auth sekmesindeki **Auth tipi** listesinde Bearer Token, Basic Auth ve API Key'in yanında iki tip daha var.

Digest Auth

Kullanıcı adı ve şifreyi yazmanız yeterli. Sunucu ilk isteğe 401 ile cevap verir; Ulak gelen bilgilerle isteği imzalayıp kendisi tekrar gönderir. Kameralar, ağ cihazları ve bazı kurumsal servisler bunu kullanır.

OAuth 2.0

Modern servislerin çoğu (Azure AD, Keycloak, Google...) önce bir erişim token'ı almanızı ister. Ulak token'ı alır, saklar, süresi dolunca yeniler ve her isteğe ekler.



OAuth 2.0 ayarları ve token durumu.

1. Auth tipi olarak **OAuth 2.0**'ı seçin.
2. **Akış**'ı seçin: servisten servise bağlantı için **Client Credentials**, kullanıcı adı/şifre için **Password**, tarayıcıda giriş için **Authorization Code + PKCE**.
3. Sağlayıcının verdiği **Access Token URL**, **Client ID**, **Client Secret** ve **Scope** değerlerini girin (değişken de kullanılabilir).
4. **Token al**'a basın. Altteki kutu token'ın kaç dakika geçerli olduğunu gösterir.

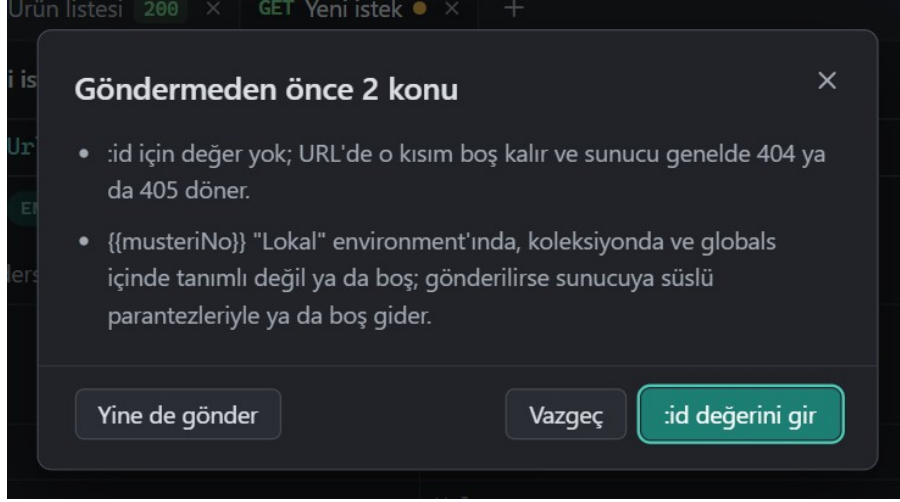
Tarayıcıda giriş Authorization Code akışında giriş ekranı tarayıcıda açılır; giriş bitince sağlayıcı `http://localhost:53682/callback` adresine döner ve Ulak dönüşü yakalar. Bu adresi sağlayıcıdaki uygulama kaydına (redirect / callback URL) bir kez eklemeniz gerekir.

- Client Credentials ve Password akışlarında **Token al**'a basmasanız da ilk gönderimde token kendiliğinden alınır.
- Süresi dolan token, sağlayıcı yenileme token'ı verdiyse gönderimde sessizce yenilenir.

- Token'lar koleksiyon dosyasına yazılmaz; koleksiyonu paylaştığınız da token paylaşılmaz.
- Ayarı koleksiyonda yaparsanız içindeki bütün istekler (**Üstten devral**) aynı token'ı kullanır.

9. Eksik değer uyarıları

Ulak, gönderim yanlış gidecekse önceden haber verir. Örneğin adreste değeri girilmemiş bir `:id` ya da tanımsız bir değişken varsa tek bir pencerede hepsini listeler.



Göndermeden önce çıkan uyarı penceresi.

- Sağdaki vurgulu düğme ilk sorunu düzeltir (örnek: **:id değerini gir**).
- **Vazgeç** isteği göndermez.
- **Yine de gönder** isteği olduğu gibi gönderir.

Canlı ortam koruması

Canlı türündeki bir environment seçiliyken Gönder düğmesi kırmızı olur ve **Canlıya gönder** yazar.

Canlıda ilk okuma isteği bir kez, veri değiştiren istekler (POST, PUT, PATCH, DELETE) her seferinde onay ister.

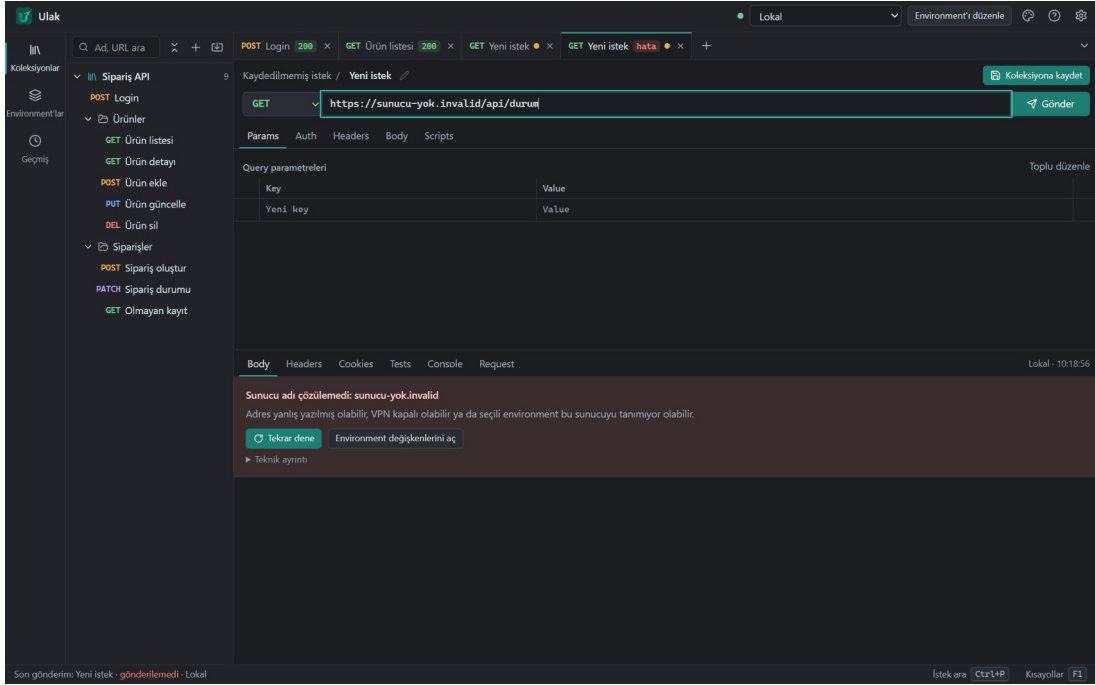
10. Cevabı okumak ve hatalar

Cevap alanı

- **Body:** cevabın gövdesi. **Biçimli göster** JSON'u okunur hale getirir.
- **Kopyala** gövdenin tamamını panoya alır. **Bul** (ya da **Ctrl+F**) gövdede arar.
- **Headers** ve **Cookies:** sunucunun gönderdiği başlıklar ve çerezler. **Tümünü kopyala** hepsini alır.
- **Request:** gerçekte gönderilen adres, başlıklar ve gövde. Gizli değerler maskelenir.
- Cevabın sağ üstünde hangi environment ile, saat kaçta alındığı yazar. İsteği değiştirirseniz üstte "Bu cevap isteğin önceki haliyle alındı" uyarısı çıkar.

İstek gönderilemezse

Sunucuya ulaşılamazsa Ulak sebebi sade bir dille yazar ve ne yapabileceğinizi önerir.



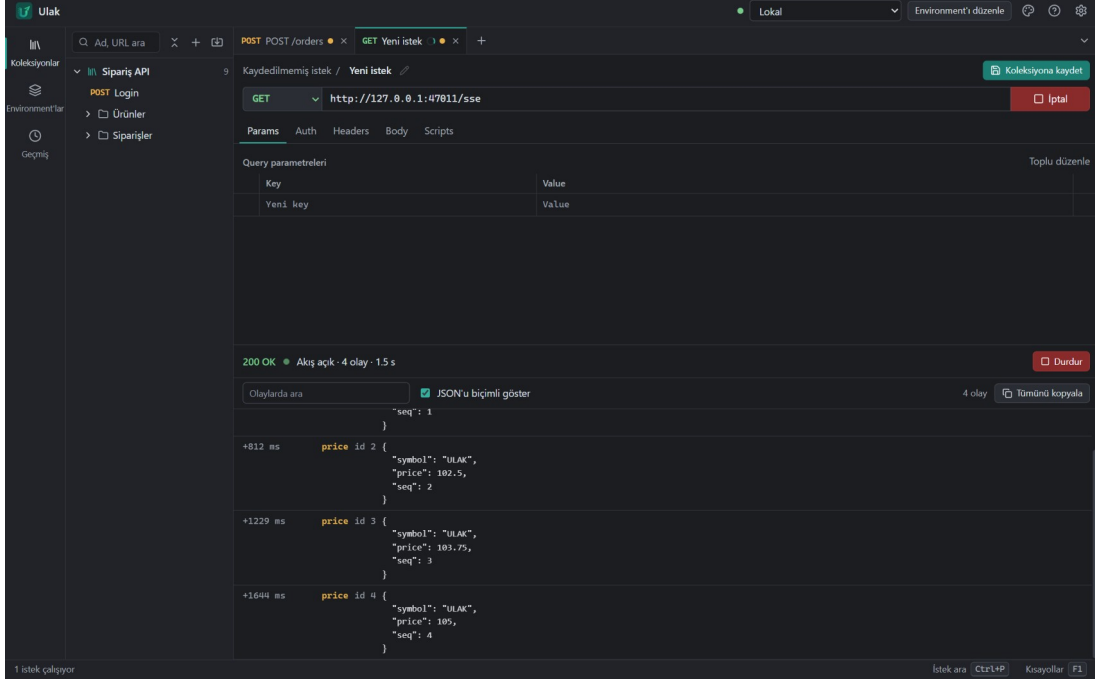
Sunucu adı çözülmediğinde çıkan hata kartı.

- **Tekrar dene:** isteği yeniden gönderir.
- **Environment değişkenlerini aç:** adres yanlışsa düzeltmeniz için environment'ı açar.
- **Teknik ayrıntı:** ham hata metni (destek isterken işe yarar).

11. Akış halindeki cevaplar

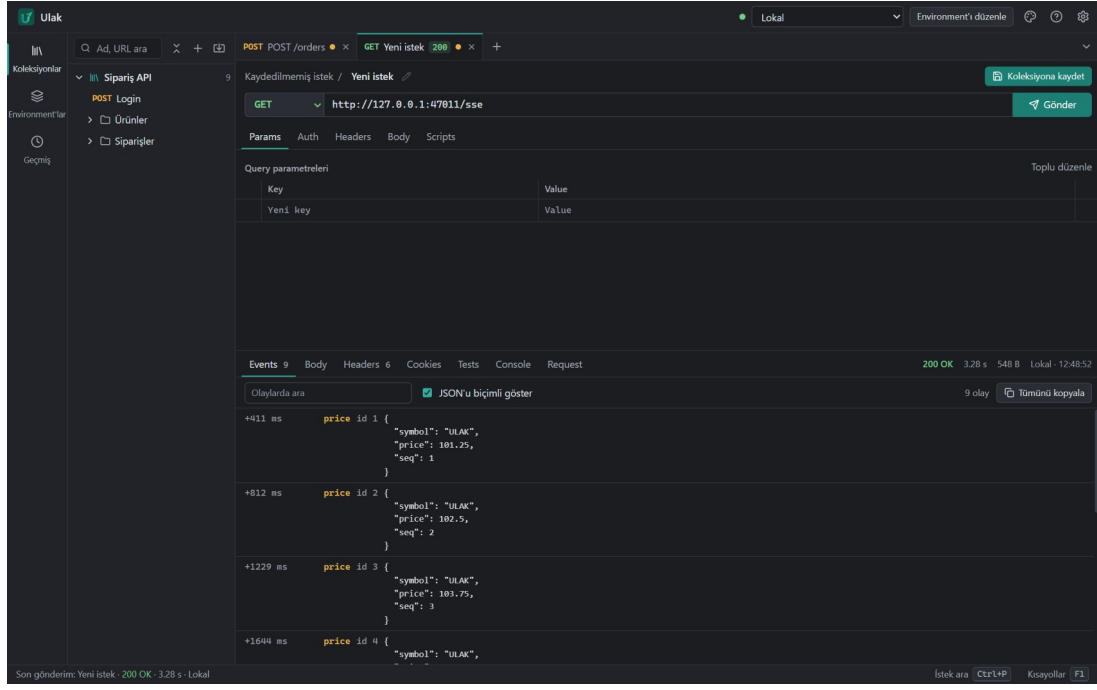
Bazı servisler cevabı tek seferde değil, parça parça gönderir: canlı fiyatlar, bildirim akışları, uzun işlerin ilerlemesi, yapay zekâ servislerinin kelime kelime gelen cevapları. Bunlara Server-Sent Events (SSE) ya da NDJSON akışı denir.

Böyle bir isteği normal şekilde gönderin. Ulak cevabın bitmesini beklemez; olaylar geldikçe listelenir ve üstte kaç olay geldiği ile geçen süre yazar.



Açık bir akış: olaylar geldikçe listeye eklenir.

- **Durdur** akışı keser; o ana kadar gelen olaylar kaybolmaz.
- Akış bitince cevap alanında **Events** sekmesi açılır: her olayın geliş zamanı, adı, id'si ve verisi.
- Olaylarda arama yapabilir, tek bir olayı ya da hepsini kopyalayabilirsiniz. Ham metin Body sekmesindedir.
- Akış açık kaldığı sürece zaman aşımı uygulanmaz.

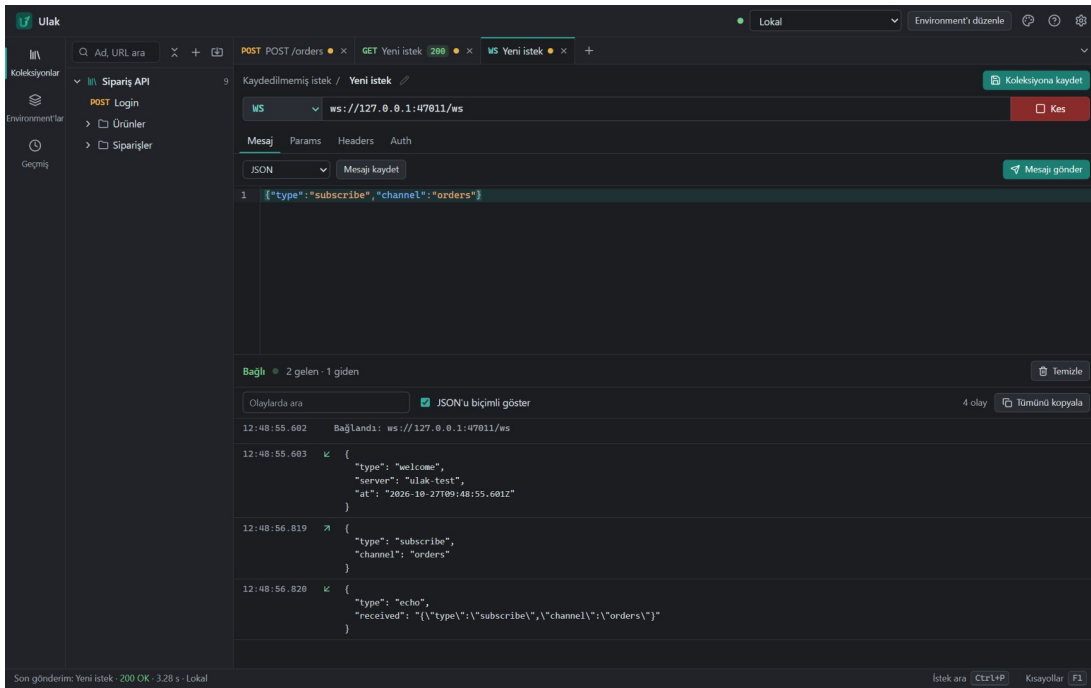


Akış bitince Events sekmesi.

12. WebSocket

WebSocket, sunucuyla sürekli açık kalan bir bağlantıdır; iki taraf da istediği an mesaj gönderir. Canlı skor, sohbet, borsa verisi ve bildirim sistemlerinde kullanılır.

1. Yeni bir WebSocket isteği açın: ağaçta klasöre sağ tık → **Yeni WebSocket**, **Ctrl+P** → "Yeni WebSocket", ya da bir isteğin metod listesinden **WS**'yi seçin.
2. Adresi yazın (**ws://** ya da **wss://** ile başlar) ve **Bağlan**'a basın (ya da URL kutusunda Enter).
3. **Mesaj** sekmesinde mesajınızı yazın (JSON, Text ya da Binary) ve **Mesajı gönder** (**Ctrl+Enter**).
4. Alt bölümde gelen (✓) ve giden (↗) mesajlar saatleriyle listelenir. İşiniz bitince **Kes**'e basın.



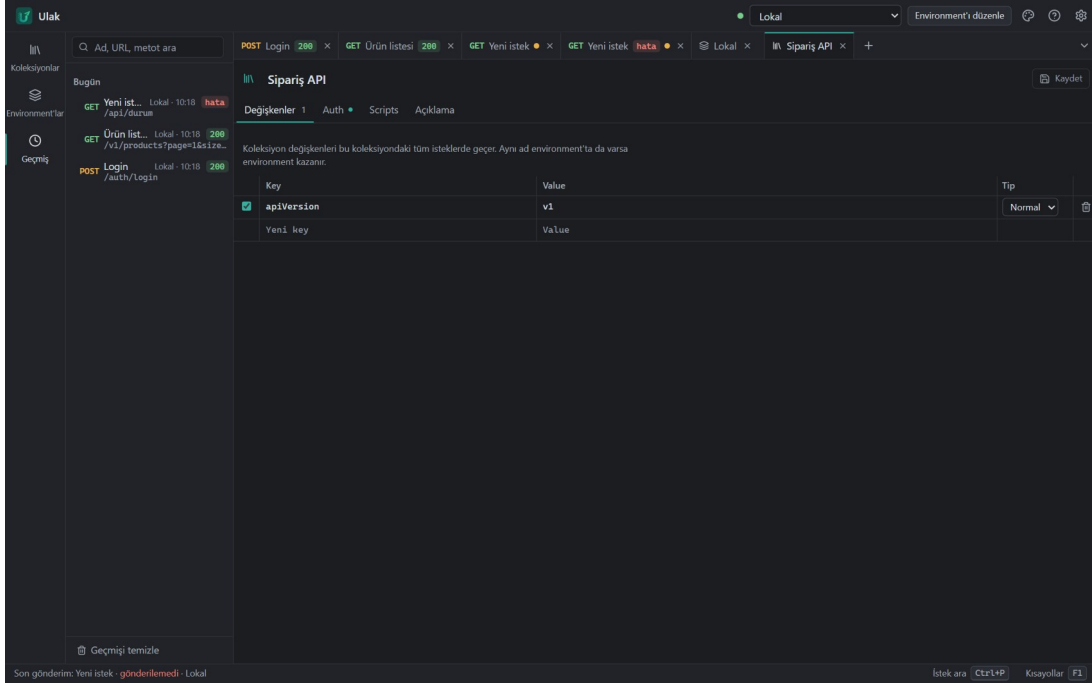
Bağlı bir WebSocket: gönderilen mesaj ve sunucunun cevabı.

- Sık kullandığınız mesajları **Mesajı kaydet** ile isteğe kaydedin; tek tıkla geri gelir.
- Headers ve Auth sekmeleri HTTP isteklerindeki gibi çalışır; mesajlarda {{değişken}} kullanılabilir.
- WebSocket istekleri koleksiyonla birlikte saklanır, ancak Postman biçiminde karşılıkları olmadığı için Postman'e dışa aktarılan dosyaya yazılmaz.

13. Geçmiş ve hızlı arama

Geçmiş

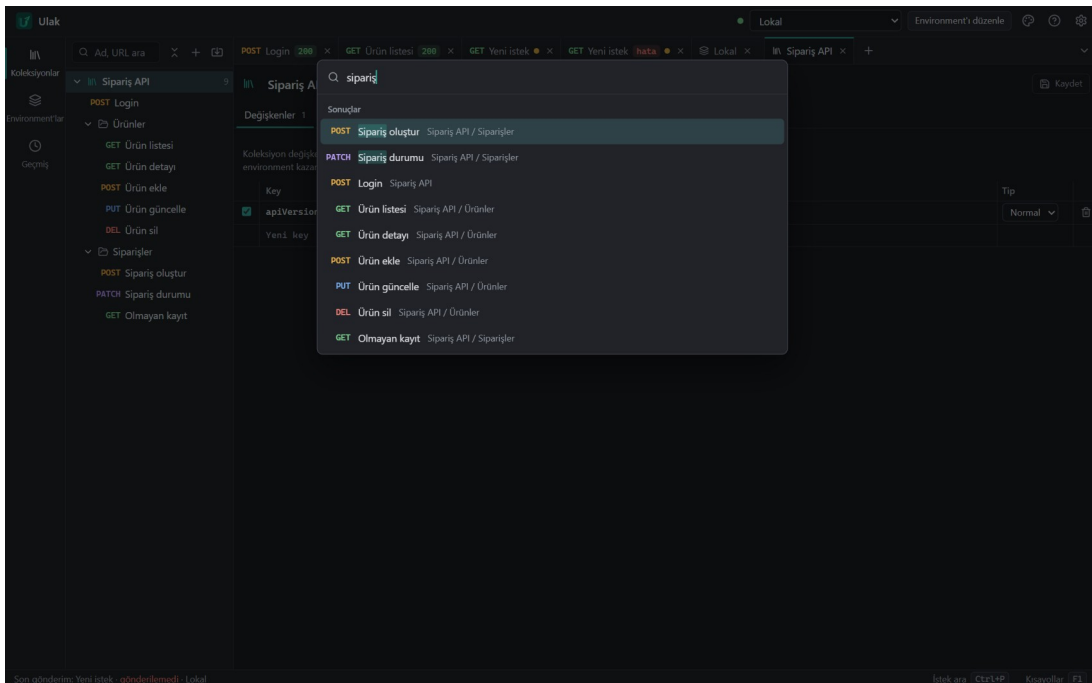
Gönderdiğiniz her istek **Geçmiş** bölümünde saklanır. Satıra tıklayınca istek, o zaman dönen cevapla birlikte açılır. **Şimdi gönder** ile tekrar gönderilir.



Geçmiş: istek adı, environment, saat ve durum kodu.

Komut paleti

Ctrl+P ile açılan palet, bütün istekleri ve komutları tek kutuda arar. İstek adı, klasör, metot ya da URL'nin bir kısmını yazın, Enter ile açın. Türkçe harf yazmanız gerekmez: "siparis" yazınca "Sipariş" bulunur.

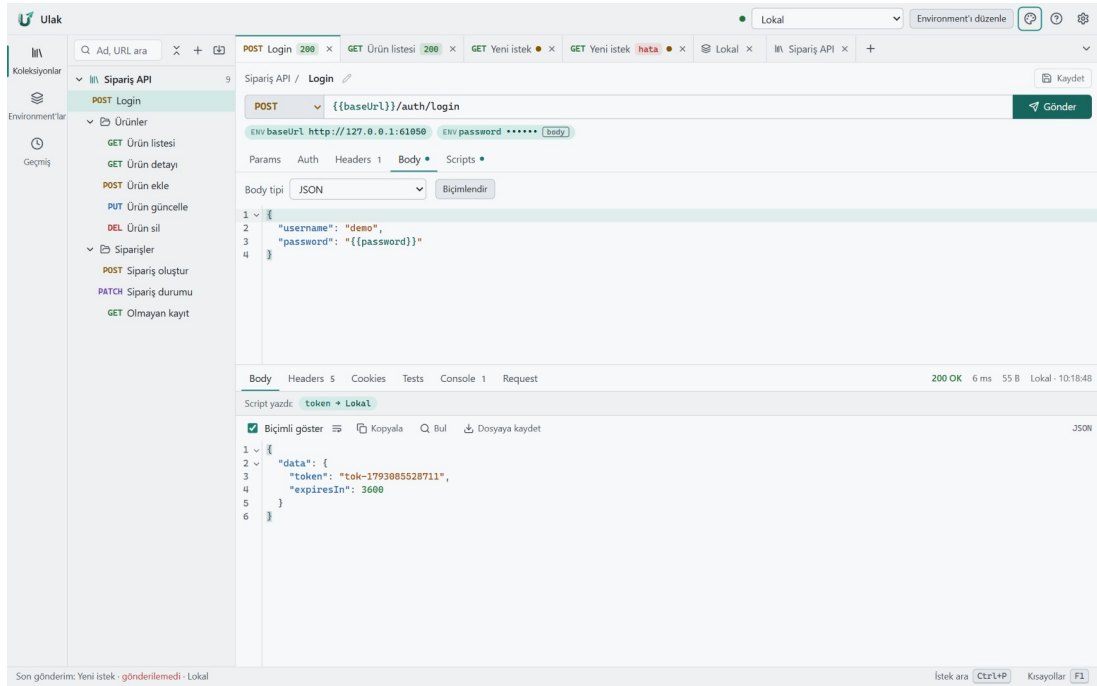


Komut paleti: "sipariş" araması.

14. Tema, dil ve ayarlar

Tema

Üst çubuktaki palet simgesi dört tema sunar: **Koyu** (varsayılan), **Gece**, **Açık** ve **Kum**. Seçim hemen uygulanır ve hatırlanır.



Açık tema.

Dil

Ulak Türkçe ve İngilizce kullanılabilir. **Ayarlar** → **Görünüm** → **Dil** alanından ya da **Ctrl+P** ile "dil" yazarak değiştirin. Koleksiyon ve istek adlarınız çevrilmez; Postman terimleri iki dilde de İngilizcedir.

Ayarlar

Ayarlar

İstek

Zaman aşımı (ms) 30 saniye · 0 = sınırsız

Yönlendirmeleri takip et ☒

SSL sertifikasını doğrula ☒ Kendinden imzalı (self-signed) sertifika kullanan test sunucuları için kapat.

En büyük cevap (MB) Üstü kesilir, cevapta uyarı görünür.

Görünüm

Tema Sağ üstteki palet düğmesinden de değişir; seçim hatırlanır.

Dil Arayüz ve hata mesajları bu dilde; Postman terimleri (Key, Value, Body...) iki dilde de İngilizce.

Kayıt

Geçmişte tutulacak istek

Log seviyesi Uygulama yeniden açılınca geçerli olur.

Ulak 0.1.0 · veri klasörü C:\Users\Muhammed\AppData\Local\Temp\ulak-review-MJw9Zi

Ayarlar penceresi.

- **Zaman aşımı:** cevap bu süreden uzun sürerse istek durur (0 = sınırsız).
- **SSL doğrulama:** test sunucusunda kendi imzalı sertifika varsa kapatın.
- **En büyük cevap:** bu boyuttan büyük cevaplar kesilir, uyarı gösterilir.
- **Geçmiş** kayıt sayısı ve **log** seviyesi.

15. Klavye kısayolları

Tüm kısayolların listesi uygulamada F1 ile açılır. En çok kullanılanlar:

Kısayol	Ne yapar
Ctrl+Enter	İsteği gönderir (WebSocket'te bağlanır; mesaj kutusunda mesajı gönderir)
Ctrl+S	İsteği kaydeder
Ctrl+T	Yeni istek açar
Ctrl+P	İstek ve komut arar
Ctrl+O	Dosyadan içe aktarır
Ctrl+D	İsteği çoğaltır
Ctrl+W	Sekmeyi kapatır
Ctrl+Shift+T	Kapatılan sekmeyi geri açar
Ctrl+F	Cevap gövdesinde arar
Ctrl+Z	Silmeyi geri alır
F2	Ağaçta yeniden adlandırır
F1	Kısayollar ve ipuçları

16. Sık sorulanlar ve sınırlar

Postman'e geri dönebilir miyim?

Evet. Ulak dosyaları Postman biçiminde saklar. Koleksiyona sağ tık → **Dışa aktar** ile aldığınız dosya Postman'e olduğu gibi içe aktarılır.

Verilerim bir yere gönderiliyor mu?

Hayır. Ulak'ın sunucusu ya da hesabı yoktur. Verileriniz yalnızca kendi bilgisayarınızda durur.

Kuruma özel sertifikalar tanınıyor mu?

Evet. Ulak Windows'un sertifika deposunu kullanır. Test sunucusu kendi imzalı sertifika kullanıyorsa Ayarlar'dan SSL doğrulamayı kapatabilirsiniz.

Henüz olmayanlar

- Auth tiplerinden Bearer, Basic, API Key, Digest ve OAuth 2.0 uygulanır (OAuth'un eski implicit akışı yok).
- Script'lerde `pm.sendRequest` ve zamanlayıcılar yok; WebSocket isteklerinde script çalışmaz.
- Socket.IO, MQTT, gRPC, toplu çalıştırma (Collection Runner), mock ve monitor yok.

Hata bildirirken Hata kartındaki **Teknik ayrıntı** metnini ve log klasöründeki (Ayarlar → **Logları aç**) o günün log dosyasını eklemeniz sorunu bulmayı hızlandırır.