

Uygulama Programı Arayüzü



Öğr. Gör.
Furkan DURMUŞ

Bu Haftanın Ders Kazanımları

1

FastAPI

2

İlk API Uygulaması

3

Uvicorn

4

Uygulama

5

Path Parameters

FastAPI



Nedir ?

Python ile web API geliřtirmek için kullanılan modern bir web framework'üdür.



Framework Nedir ?

Bir uygulama geliştirirken bize hazır yapı, araç ve kurallar sunan yazılım çatısıdır.

Örneğin kendimiz sıfırdan:

- HTTP request yakalama,
- URL eşleştirme,
- response oluşturma,
- JSON dönüşümü,
- veri doğrulama

gibi bütün yapıları yazmak yerine framework bunların büyük bölümünü bizim için hazırlar.



Kurulum

FastAPI kurulumu için öncelikle izole bir ortam kurmak için virtual enviroment oluşturuyoruz. Virtual enviroment sayesinde install ettiğimiz kütüphaneler sadece projemiz içerisinde çalışarak dış dünyadaki yapıyı etkilemeyecektir.

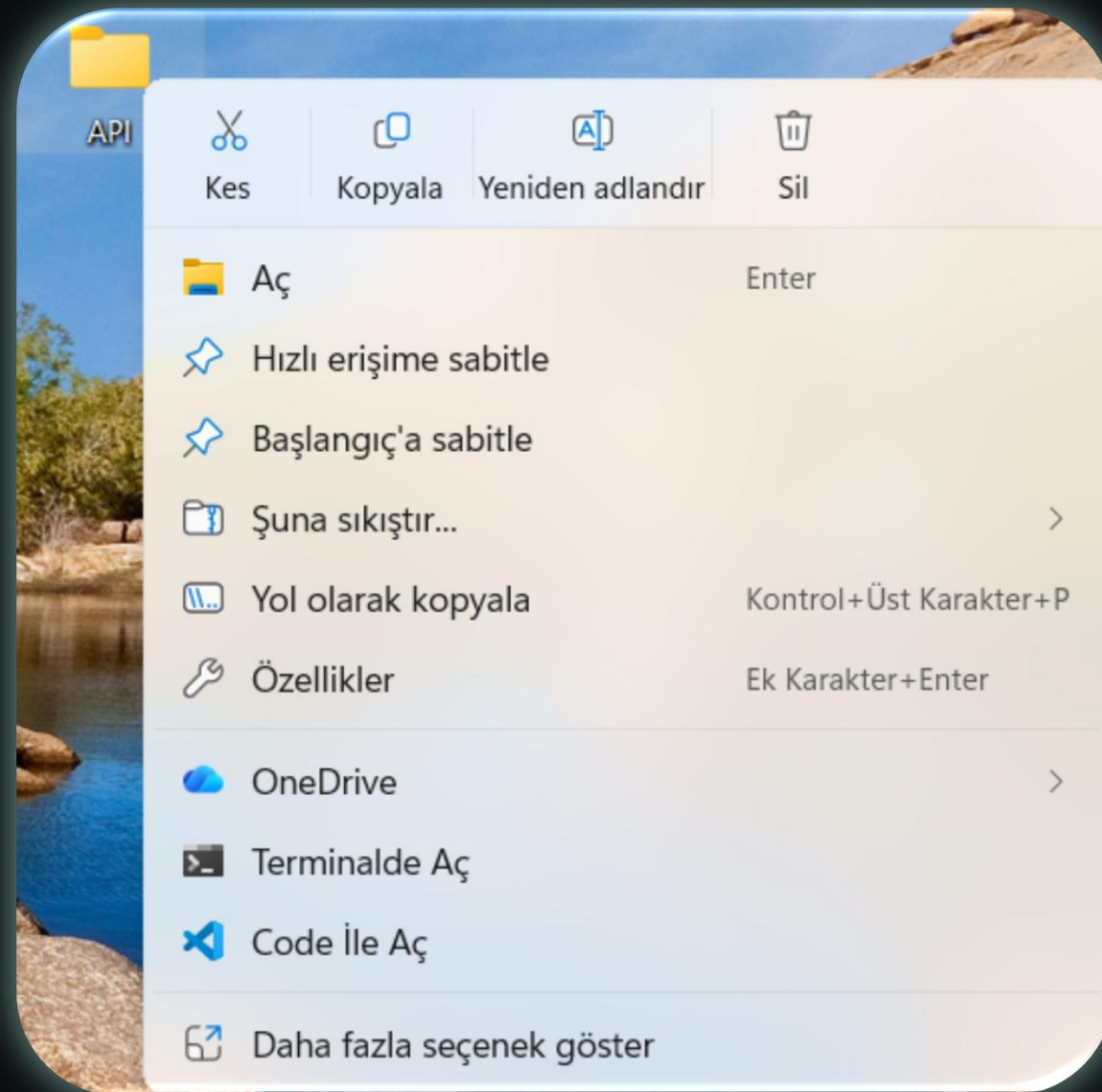
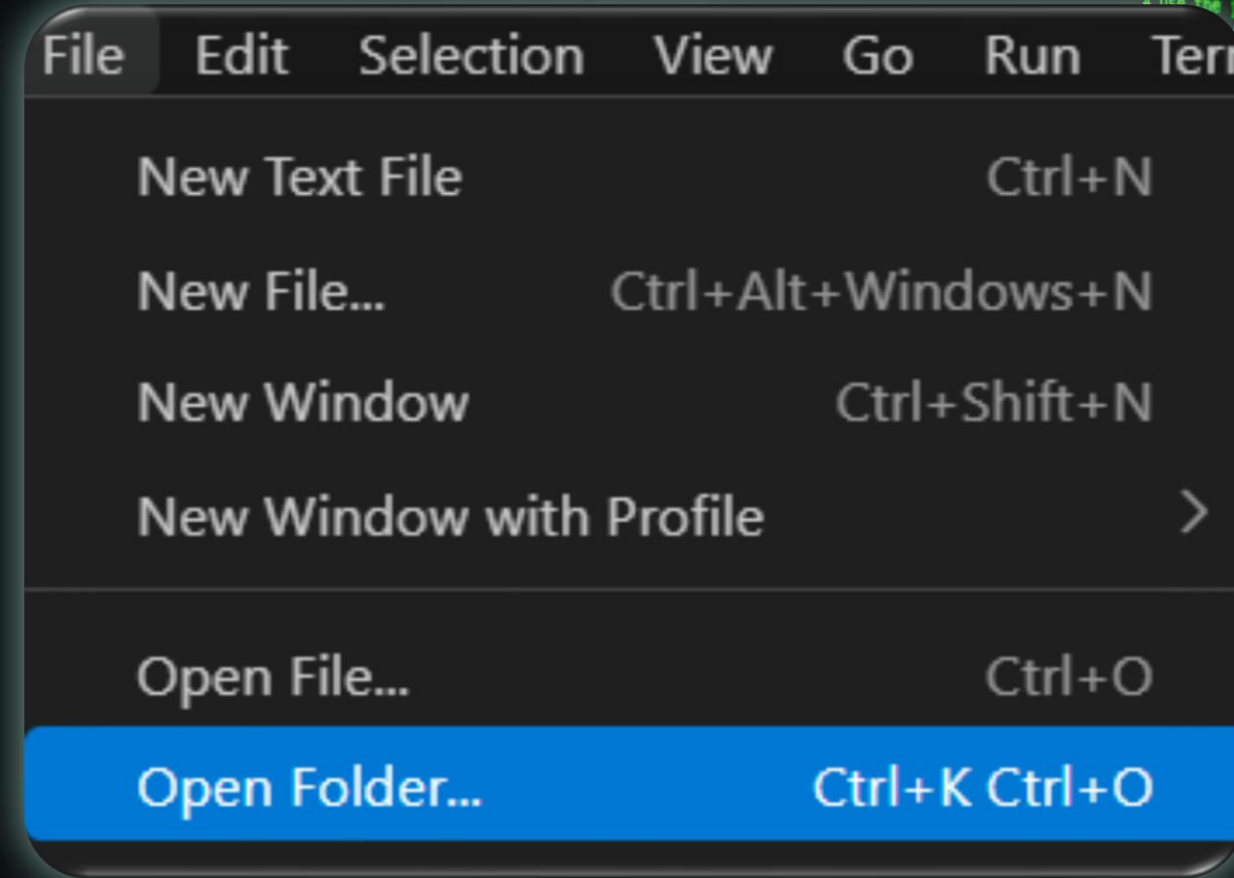
Venv kurulumu için boş bir klasörü oluşturunuz.



Kurulum

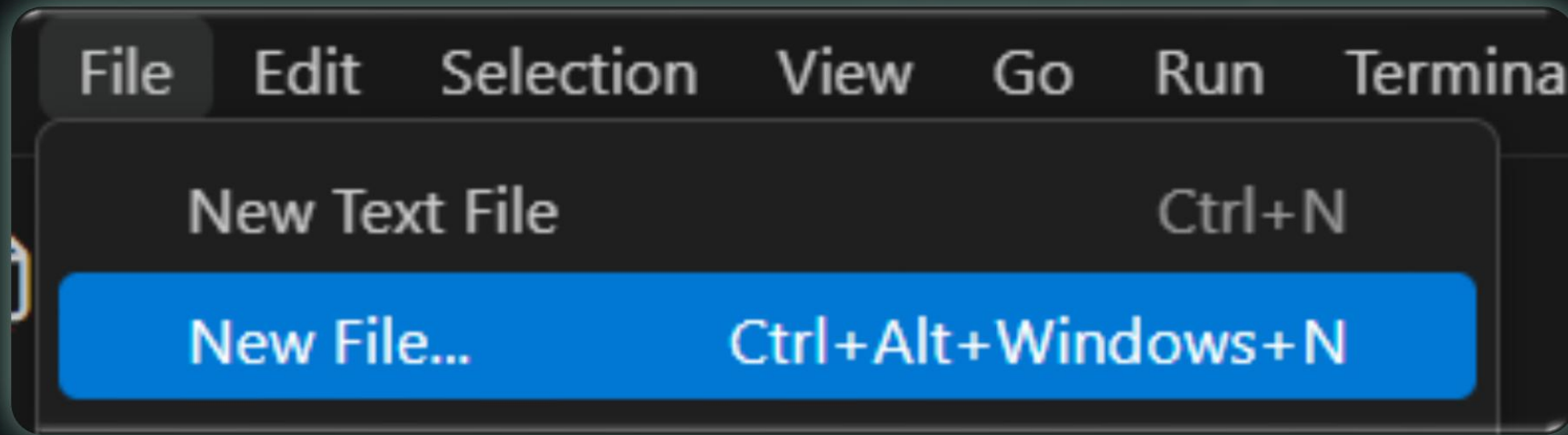
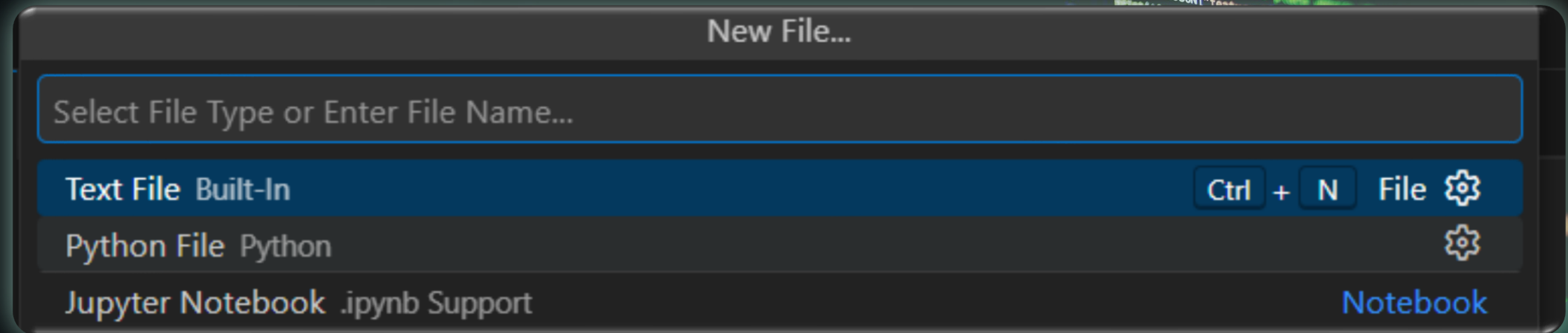
İlgili klasörü vs code ile açınız.

Bunun için klasöre sağ tık -> vs code seçebilir **yada** vs code'u açıp File -> Open Folder diyebilirsiniz.



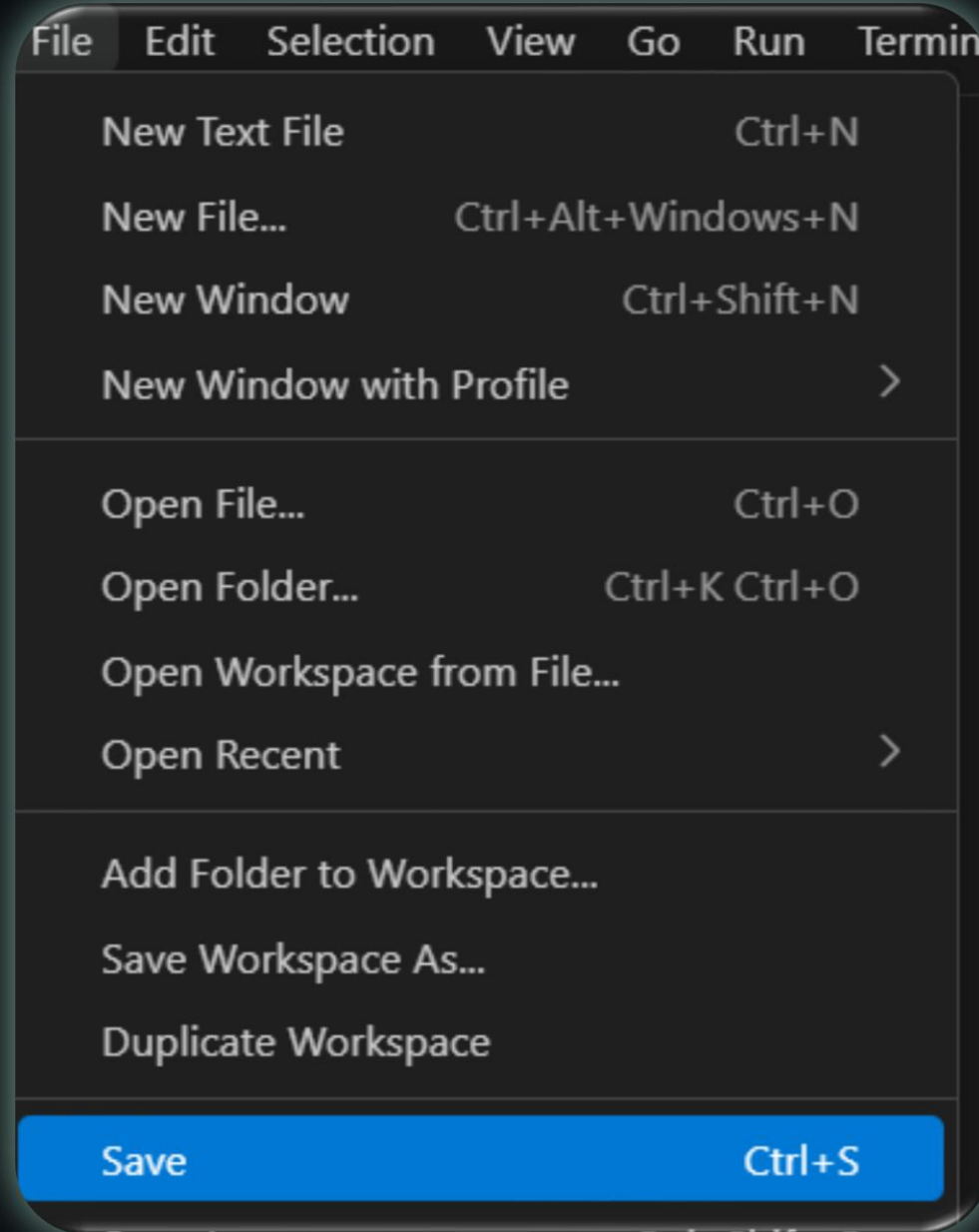
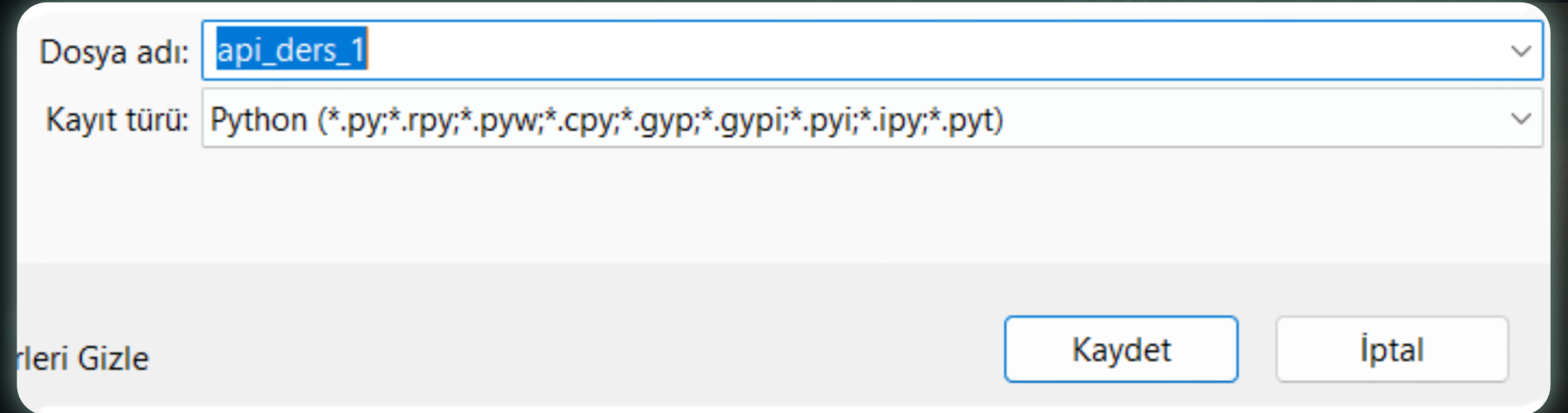
Kurulum

Açılan vs code'da File -> New File ile bir python dosyası ekleyiniz.



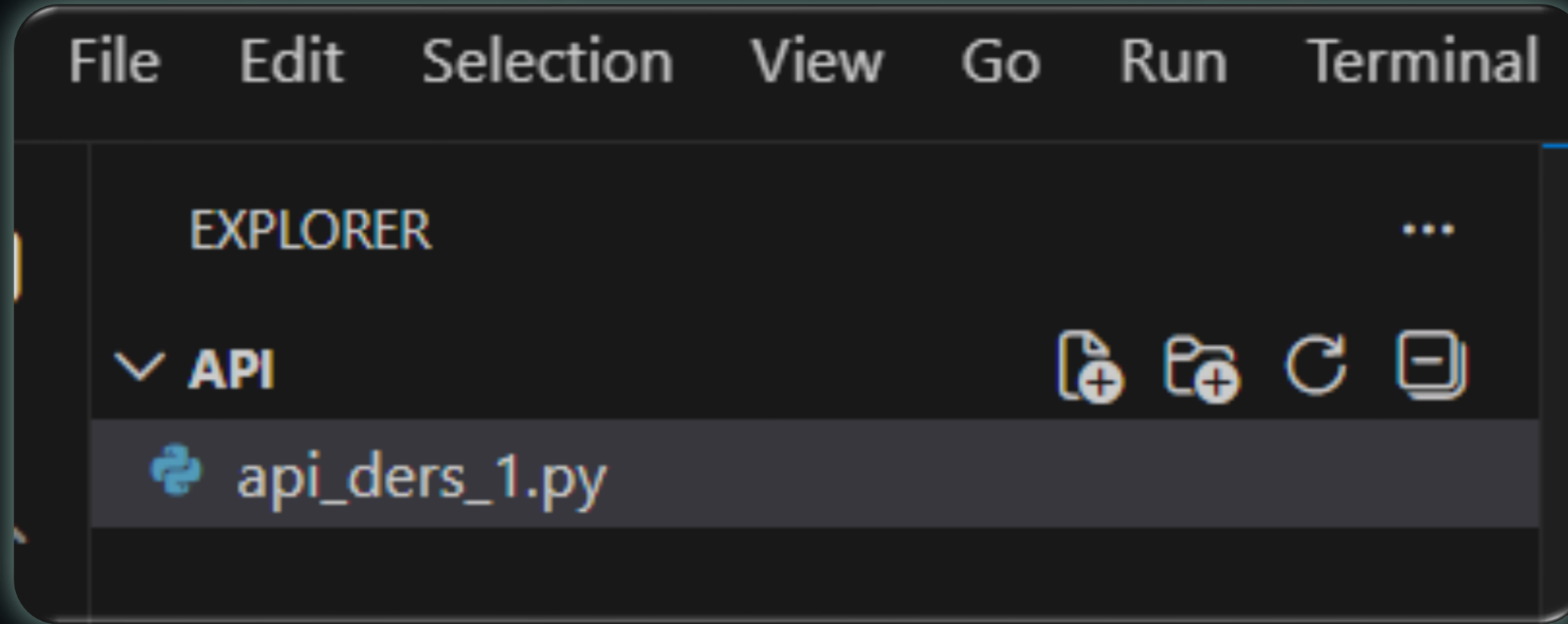
Kurulum

Oluşturduğunuz dosyayı kayıt ediniz.



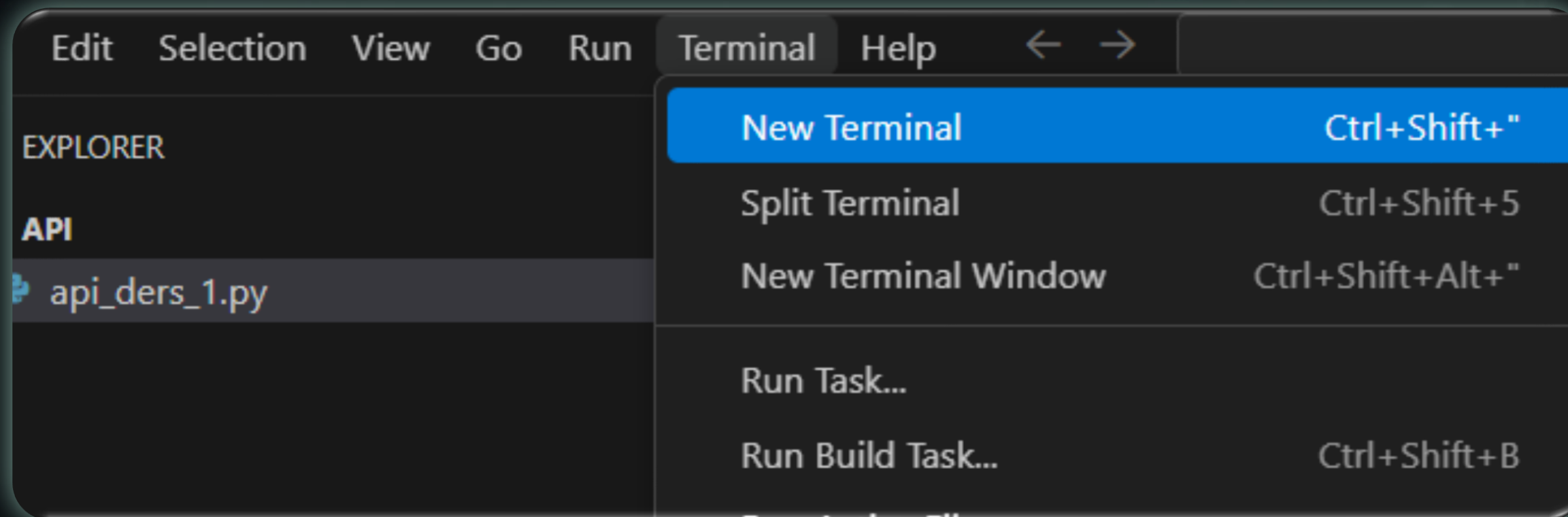
Kurulum

Oluşturduğunuz Python dosyasının vs code'da gözükmesi gerekmektedir.



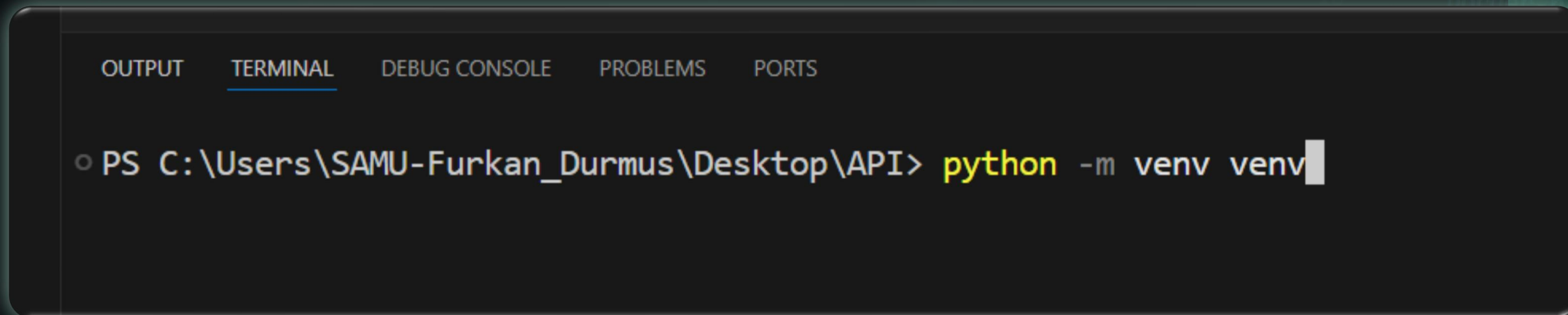
Kurulum

Terminal -> New Terminal'e tıklayarak sanal ortam kurulumu için bir terminal oluşturunuz.



Kurulum

Açılan terminalde 'python -m venv venv' ile sanal ortam oluşturunuz.

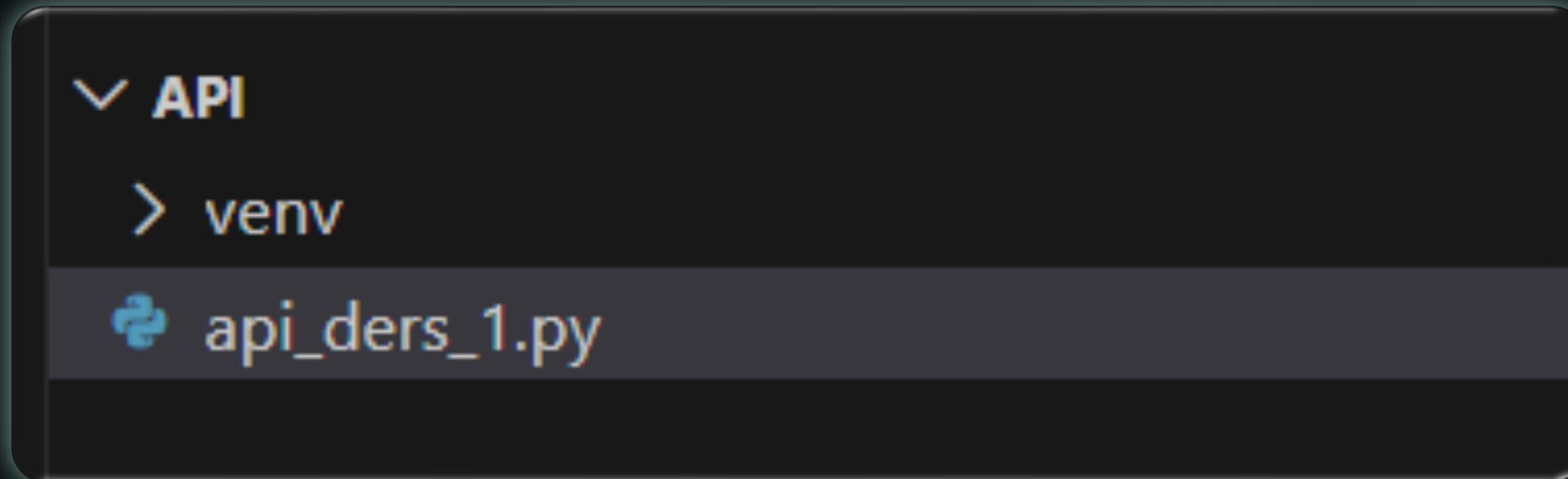


```
OUTPUT  TERMINAL  DEBUG CONSOLE  PROBLEMS  PORTS

PS C:\Users\SAMU-Furkan_Durmus\Desktop\API> python -m venv venv
```

Kurulum

Bir süre bekledikten sonra sol frame'de sanal ortamın 'venv' adı ile oluştuğunu göreceksiniz.



Kurulum

Sanal ortamı başlatmak için terminale 'venv\Scripts\activate' yazınız.

```
❖ PS C:\Users\SAMU-Furkan_Durmus\Desktop\API> venv\Scripts\activate
```

```
❖ (venv) PS C:\Users\SAMU-Furkan_Durmus\Desktop\API>
```

FastAPI Kurulum

Sanal ortam aktif olduğunda 'pip install fastapi' ile fastapi framework kurulumu yapımız.

```
(venv) PS C:\Users\SAMU-Furkan_Durmus\Desktop\API> pip install fastapi
```

FastAPI İlk Nesne

Aşağıdaki şekilde ilk FastAPI nesnenizi oluşturunuz.

```
api_ders_1.py > ...  
1  from fastapi import FastAPI  
2  
3  app = FastAPI()  
4  
5
```

İlk API Uygulaması ➔

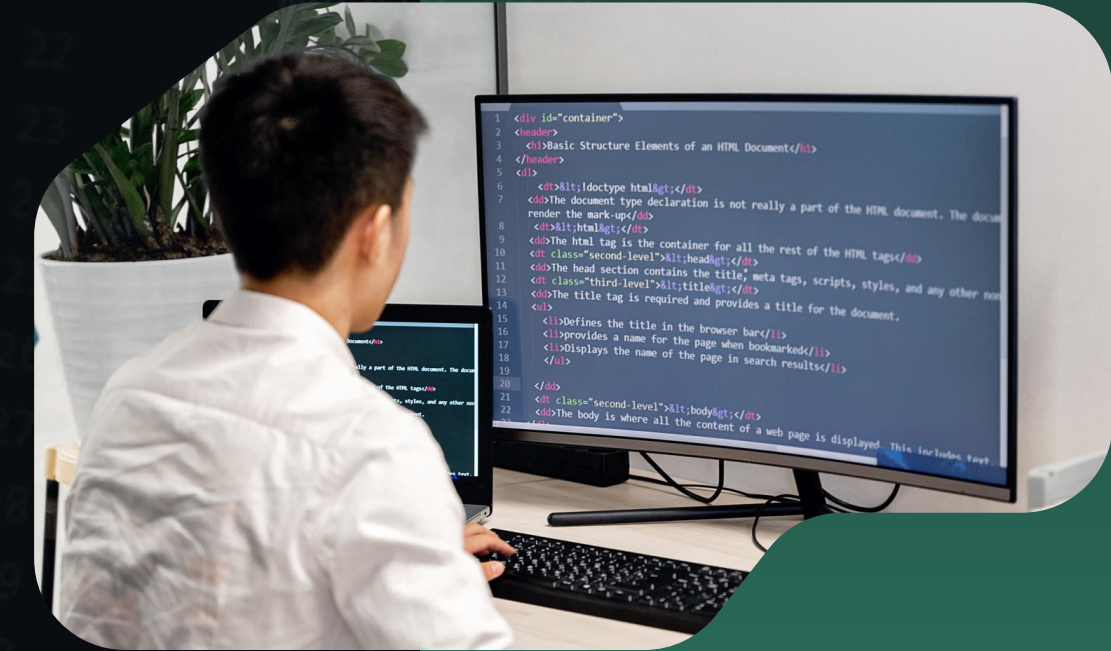


Giriş

Oluşturduğumuz fastapi nesnesi ile endpointlerimizi tasarlayabileceğiz.

@ semboli ile app nesnemize endpoint tanımlaması yapabiliriz.

```
api_ders_1.py > ...
1  from fastapi import FastAPI
2
3  app = FastAPI()
4
5
```



Kod Örneği

Yandaki kodu bilgisayarınızda yazınız.

```
api_ders_1.py X
api_ders_1.py > home
1  from fastapi import FastAPI
2
3  app = FastAPI()
4
5  @app.get("/")
6  def home():
7      return "Hello, World!"
```

Genel Yapı

api_ders_1.py X

api_ders_1.py > home

```
1 from fastapi import FastAPI
```

```
2  
3 app = FastAPI()
```

```
4  
5 @app.get("/")
```

```
6 def home():  
7     return "Hello, World!"
```

Kütüphane

Nesne

Endpoint tanımlayıcı (decorator)

HTTP Metodu

Endpoint yolu (path)

Endpoint'in çalıştıracağı fonksiyon, apinin beyni (fonksiyon ismi bağımsızdır)

Kod Örneği

Aşağıdaki kodu bilgisayarınızda yazınız ve yorumlayınız. Kodu şimdilik çalıştırmayınız.

```
api_ders_1.py > ...
1  from fastapi import FastAPI
2
3  app = FastAPI()
4
5  @app.get("/")
6  def home():
7      return {"message": "Hello, World!"}
8
9  @app.get("/hello")
10 def hello():
11     return {"message": "Hello, FastAPI!"}
12
13
```

Önemli

```
api_ders_1.py > ...
1  from fastapi import FastAPI
2
3  app = FastAPI()
4
5  @app.get("/")
6  def home():
7      return {"message": "Hello, World!"}
8
9  @app.get("/hello")
10 def hello():
11     return {"message": "Hello, FastAPI!"}
12
13
```

Birebir Eşleşme: Bir @ etiketi (decorator), yalnızca kendisinden hemen sonra gelen ilk def fonksiyonunu kapsar. Alt alta birden fazla fonksiyon yazılırsa, etiket sadece birinciyi API uç noktasına dönüştürür.

Kesintisiz Bağlantı: @ satırı ile def satırı arasına hiçbir çalışan kod (değişken, işlem, print vb.) yazılamaz. Araya sadece Python'ın çalışma sırasında yok saydığı boş satırlar veya yorum satırları (#) girebilir.

Dönüş Tipleri

Aşağıdaki kodu bilgisayarınızda yazınız ve yorumlayınız. Kodu şimdilik çalıştırmayınız.

```
api_ders_1.py •
api_ders_1.py > student
1  from fastapi import FastAPI
2
3  app = FastAPI()
4
5  @app.get("/")
6  def home():
7      return {"message": "Hello, World!"}
8
9  @app.get("/hello")
10 def hello():
11     return {"message": "Hello, FastAPI!"}
12
13 @app.get("/student")
14 def student():
15     return {
16         "id": 1,
17         "name": "Ali",
18         "age": 20
19     }
```

FastAPI dict, list, str, int gibi Python değerlerini response olarak kullanabilir ve uygun biçimde JSON'a dönüştürebilir.

```
14 @app.get("/students")
15 def students():
16     return [
17         {
18             "id": 1,
19             "name": "Ali"
20         },
21         {
22             "id": 2,
23             "name": "Ayşe"
24         }
25     ]
```



Dönüş Tipleri

```
api_ders_1.py > students
1  from fastapi import FastAPI
2
3  app = FastAPI()
4
5  @app.get("/")
6  def home():
7      return {"message": "Hello, World!"}
8
9  @app.get("/hello")
10 def hello():
11     return {"message": "Hello, FastAPI!"}
12
13 @app.get("/student")
14 def student():
15     return {
16         "id": 1,
17         "name": "Ali",
18         "age": 20
19     }
20
21 @app.get("/students")
22 def students():
23     return [
24         {
25             "id": 1,
26             "name": "Ali"
27         },
28         {
29             "id": 2,
30             "name": "Ayşe"
31         }
32     ]
```

Run tuşuna bastığımızda kod neden tepki vermiyor ?

Çalışması için ne gerekiyor ?

Uvicorn



Giriş

Bir API'ı ayağa kaldırmak için, FastAPI uygulamasını yazmak tek başına yeterli değildir.

Dışarıdan HTTP request alabilmek için bu uygulamanın bir web **server** tarafından çalıştırılması gerekir. Bu noktada **Uvicorn** kullanılır.

Uvicorn FastAPI gibi Python web uygulamalarını çalıştırabilen bir application server'dır.



Giriş



FastAPI

→ API uygulamasını tanımlar

Uvicorn

→ Bu uygulamayı çalıştırır ve HTTP requestlerini alır.

```
.box{  
    position: absolute;  
    top: 50%;  
    left: 50%;  
    transform: translate(-50%,  
    width: 400px;  
    padding: 40px;  
    background: rgba(0, 0, 0,  
    box-sizing: border-box;  
    box-shadow: 0 15px 25px  
}  
  
.box h2{  
    margin: 0 0 30px;  
    padding: 0;  
    color: #fff;  
    text-align: center;  
}  
  
.box h3{  
    margin: 0 0 10px;  
    padding: 0;  
    color: #fff;  
    text-align: center;  
}  
  
.box .input{
```

Kurulum

Uvicorn yüklemek için mevcut venv ortamında;
‘pip install uvicorn’ komutunu çalıştırınız.

```
(venv) PS C:\Users\SAMU-Furkan_Durmus\Desktop\API> pip install uvicorn
```

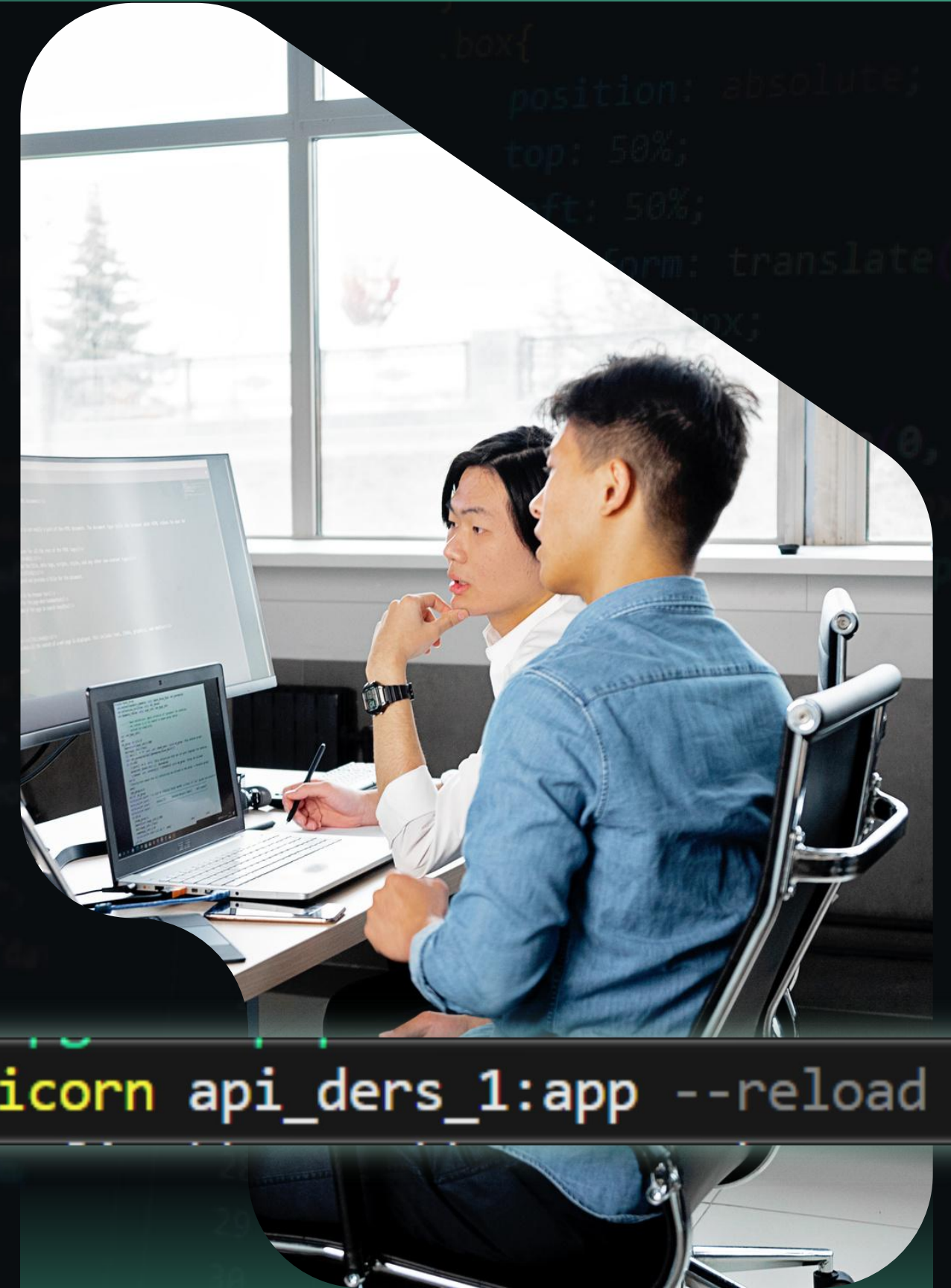


Server Başlatma

Yükleme tamamlandıktan sonra server'ı başlatmak için ;

'**uvicorn api_ders_1:app --reload**' komutunu terminale girip çalıştırınız. Buradaki – reload komutu sayesinde kodda değişiklik yapıldığında server'a değişiklik otomatik yansımaktadır.

NOT : FastAPI kodlarını yazdığınız py dosyasının adı ne ise :app'den önce o dosya adını yazınız.



```
(venv) PS C:\Users\SAMU-Furkan_Durmus\Desktop\API> uvicorn api_ders_1:app --reload
```

Local Server

Tebrikler,

İlk API sunucunuz başarıyla ayağa kalktı.

an DIRMUS\DESKTOP\API>
n Follow link (ctrl + click) rectori
<http://127.0.0.1:8000>

```
(venv) PS C:\Users\SAMU-Furkan_Durmus\Desktop\API> uvicorn api_ders_1:app --reload
INFO: Will watch for changes in these directories: ['C:\\Users\\SAMU-Furkan_Durmus\\Desktop\\API']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [24964] using StatReload
INFO: Started server process [19984]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: 127.0.0.1:55458 - "GET / HTTP/1.1" 200 OK
INFO: 127.0.0.1:55458 - "GET /favicon.ico HTTP/1.1" 404 Not Found
```

Adresine giderek az önce oluşturduğunuz API'ları kontrol edebilirsiniz.

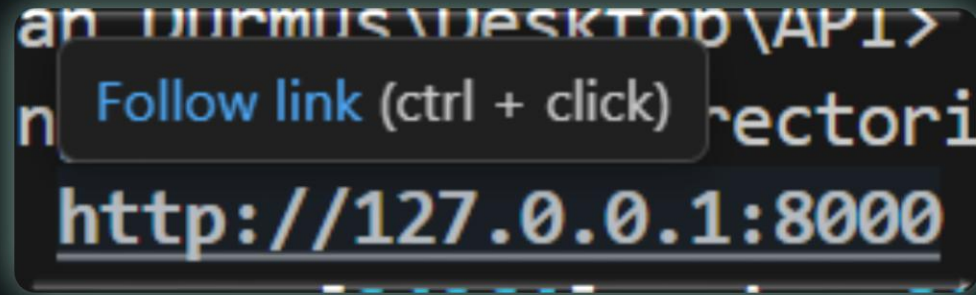
(terminalden ctrl + sol tık ile adrese direkt ulaşabilirsiniz)



Local Server

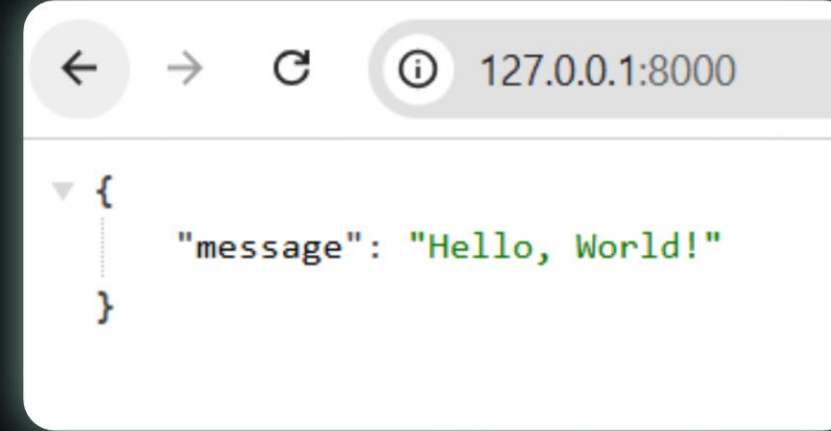
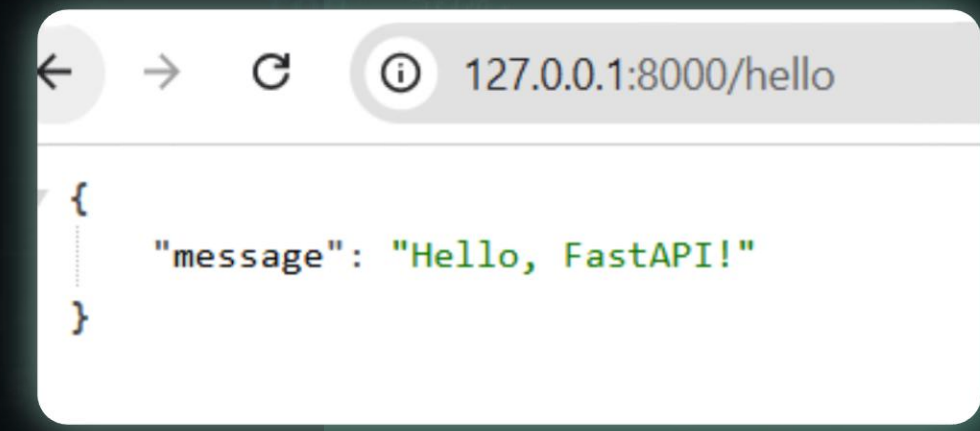
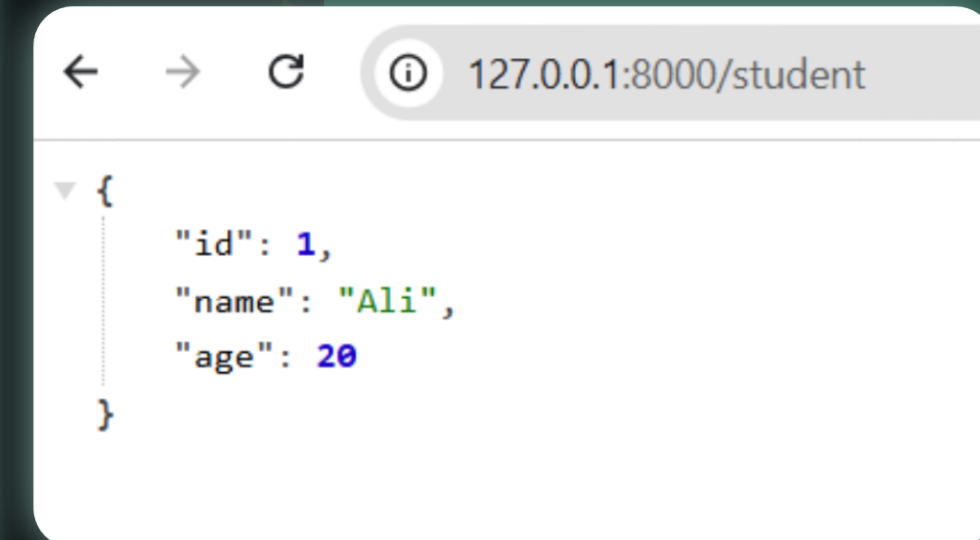
Tebrikler,

İlk API sunucunuz başarıyla ayağa kalktı.



Adresine giderek az önce oluşturduğunuz API'ları kontrol edebilirsiniz.

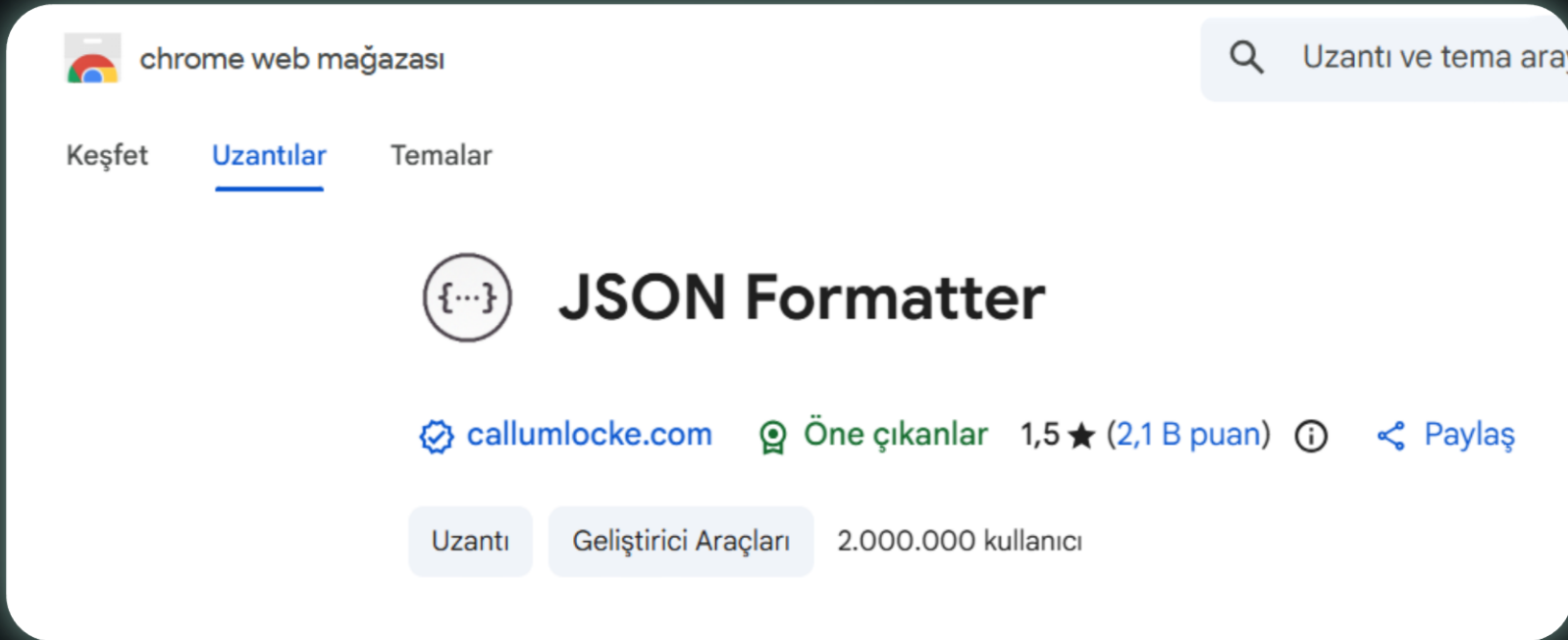
(terminalden ctrl + sol tık ile adrese direkt ulaşabilirsiniz)



Local Server

Chrome üzerinde json verilerinin daha düzenli gözükmesi için

Google'da 'json formatter extension' araması ile alttaki eklentiye kurabilirsiniz.



-- Reload Özelliği

Koddaki bir return değerini değiştirerek ctrl + s ile dosyayı kayıt ediniz. Ardından ilgili endpointi Chrome dan tekrar açınız.

Uvicorn'un reload özelliği sayesinde server'a restart atmamıza gerek kalmadan değişiklik api'mize otomatik yansdı.

```
@app.get("/students")
def students():
    return [
        {
            "id": 1,
            "name": "furkan durmuş"
        },
        {
            "id": 2,
            "name": "Ayşe"
        }
    ]
```



```
[
  {
    "id": 1,
    "name": "furkan durmuş"
  },
  {
    "id": 2,
    "name": "Ayşe"
  }
]
```

Uygulama



Örnek 1

Books adında bir endpoint tasarlayınız. Endpoint'e Get isteği atılacak ve servis liste halinde 5 adet kitap dönecektir. Her kitabın id, name, author, page bilgileri dönecektir.

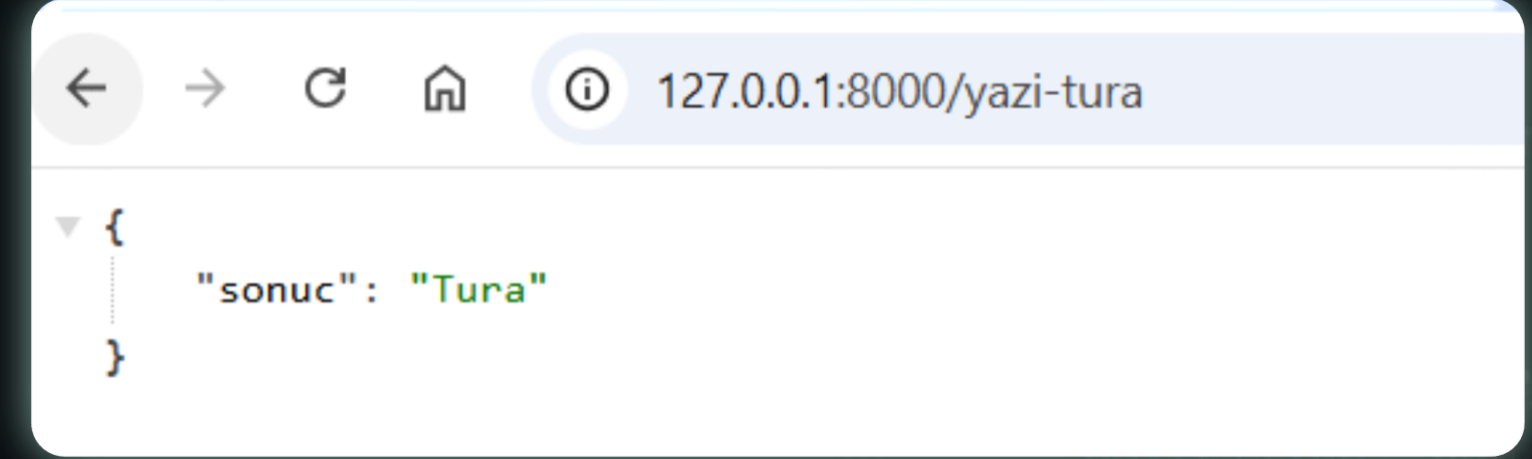
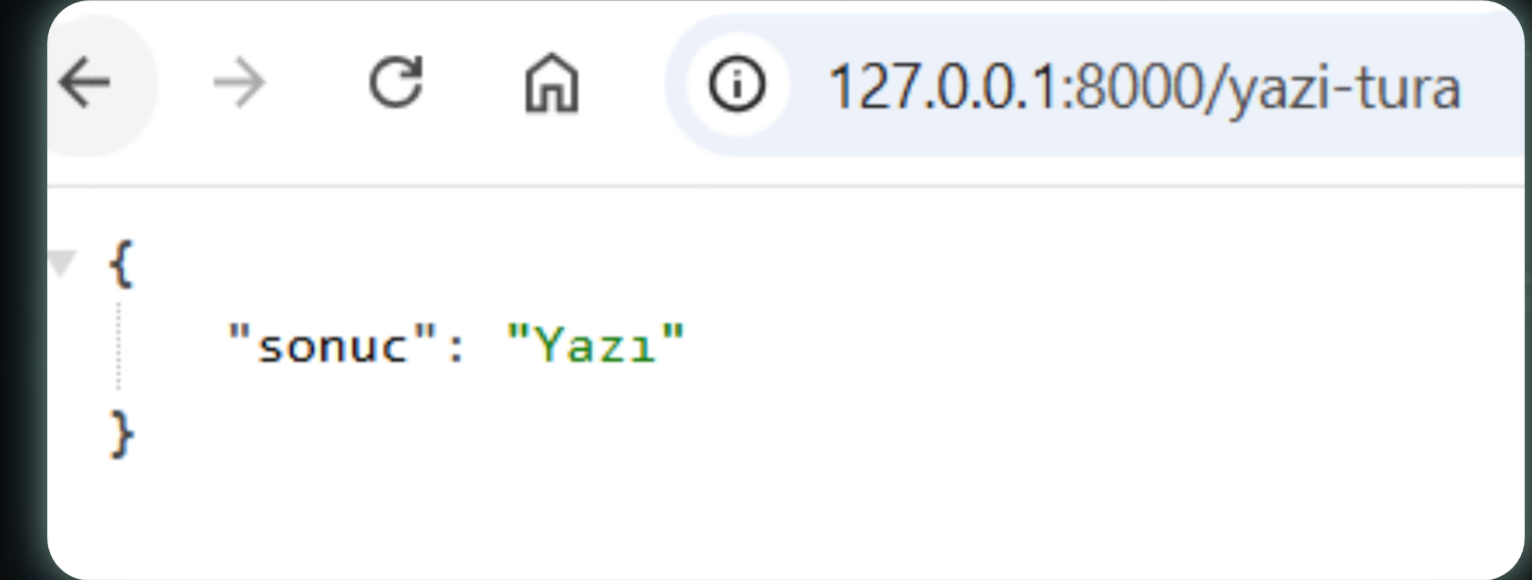


```
1 from fastapi import FastAPI
2
3 app = FastAPI()
4
5 @app.get("/books")
6 def kitaplari_getir():
7     return [
8         {"id": 1, "name": "Suç ve Ceza", "author": "Dostoyevski", "page": 687},
9         {"id": 2, "name": "Sefiller", "author": "Victor Hugo", "page": 1232},
10        {"id": 3, "name": "Simyacı", "author": "Paulo Coelho", "page": 188},
11        {"id": 4, "name": "1984", "author": "George Orwell", "page": 328},
12        {"id": 5, "name": "Satranç", "author": "Stefan Zweig", "page": 96}
13    ]
```

Örnek 2

`random.choice(liste_adı)` ile bir liste içinde rasgele seçim yapılabilir. (random kütüphanesi import edilmelidir)

`/yazi-tura` endpointi yazınız ve bu endpointe her istek attığınızda yazı tura sonucu ekrana yazılsın.



Örnek 2

`random.choice(liste_adı)` ile bir liste içinde rasgele seçim yapılabilir. (random kütüphanesi import edilmelidir)

/yazi-tura endpointi yazınız ve bu endpointe her istek attığınızda yazı tura sonucu ekrana yazılsın.

```
@app.get("/yazi-tura")
def yazi_tura_at():
    para_yuzleri = ["Yazı", "Tura"]
    gelen_yuz = random.choice(para_yuzleri)
    return {"sonuc": gelen_yuz}
```



Örnek 3

`random.randint(başlangıç, bitiş)` ile belirli bir aralıkta random sayı üretilebilmektedir.

`/pin` endpointi oluşturunuz ve bu endpointe her istek atıldığında 6 haneli random bir sayı üretmesini sağlayınız.



```
← → ↻ 🏠 ⓘ 127.0.0.1:8000/pin
{
  "islem": "Yeni şifre talebi",
  "tek_kullanimlik_sifre": 713253
}
```

```
← → ↻ 🏠 ⓘ 127.0.0.1:8000/pin
{
  "islem": "Yeni şifre talebi",
  "tek_kullanimlik_sifre": 390595
}
```

Örnek 3

`random.int(başlangıç, bitiş)` ile belirli bir aralıkta random sayı üretilebilmektedir.

`/pin` endpointi oluşturunuz ve bu endpointe her istek atıldığında 6 haneli random bir sayı üretmesini sağlayınız.



```
@app.get("/pin")
def pin_kodu_olustur():
    yeni_pin = random.randint(100000, 999999)
    return {"islem": "Yeni şifre talebi", "tek_kullanimlik_sifre": yeni_pin}
```

Örnek 4

Pythonda global etiketi ile fonksiyonun içinden fonksiyon dışındaki bir değişkene müdahale edilebilmektedir.

/bilet-al endpointi yazınız. 5 bilet ten sonra satış yapılmayacak tasarımı yapınız.



127.0.0.1:8000/bilet-al

```
{  
  "islem": "Başarılı",  
  "mesaj": "Biletiniz onaylandı. İyi eğlenceler!",  
  "bilet_no": 1  
}
```

127.0.0.1:8000/bilet-al

```
{  
  "islem": "Başarılı",  
  "mesaj": "Biletiniz onaylandı. İyi eğlenceler!",  
  "bilet_no": 5  
}
```

127.0.0.1:8000/bilet-al

```
{  
  "islem": "Başarısız",  
  "mesaj": "Üzgünüz, kapasite doldu! Bilet alamazsınız.",  
  "bilet_no": null  
}
```

Örnek 4

Pythonda global etiketi ile fonksiyonun içinden fonksiyon dışındaki bir değişkene müdahale edilebilmektedir.

/bilet-al endpointi yazınız. 5 bilet ten sonra satış yapılmayacak tasarımı yapınız.

```
bilet_no = 1
MAKSIMUM_KAPASITE = 5

@app.get("/bilet-al")
def bilet_satin_al():
    global bilet_no

    if bilet_no <= MAKSIMUM_KAPASITE:
        response = {
            "islem": "Başarılı",
            "mesaj": "Biletiniz onaylandı. İyi eğlenceler!",
            "bilet_no": bilet_no
        }
        bilet_no += 1
        return response
    else:
        return {
            "islem": "Başarısız",
            "mesaj": "Üzgünüz, kapasite doldu! Bilet alamazsınız.",
            "bilet_no": None
        }
```



Örnek 5

Pythonda global etiketi ile fonksiyonun içinden fonksiyon dışındaki bir değişkene müdahale edilebilmektedir.

/like ile beğeni arttıran /dislike ile düşüren endpointleri yazınız. 0'dan aşağı dislike düşmemelidir.



```
127.0.0.1:8000/like
{
  "islem": "Beğenildi",
  "begeni_sayisi": 1
}
```

```
127.0.0.1:8000/like
{
  "islem": "Beğenildi",
  "begeni_sayisi": 10
}
```

```
127.0.0.1:8000/dislike
{
  "islem": "Beğenilmedi",
  "begeni_sayisi": 9
}
```

```
127.0.0.1:8000/dislike
{
  "islem": "Reddedildi",
  "mesaj": "Skor zaten 0, daha fazla aşağı düşemez!",
  "begeni_sayisi": 0
}
```

Örnek 5

Python'da global etiketi ile fonksiyonun içinden fonksiyon dışındaki bir değişkene müdahale edilebilmektedir.

/like ile beğeni arttıran /dislike ile düşüren endpointleri yazınız. 0'dan aşağı dislike düşmemelidir.

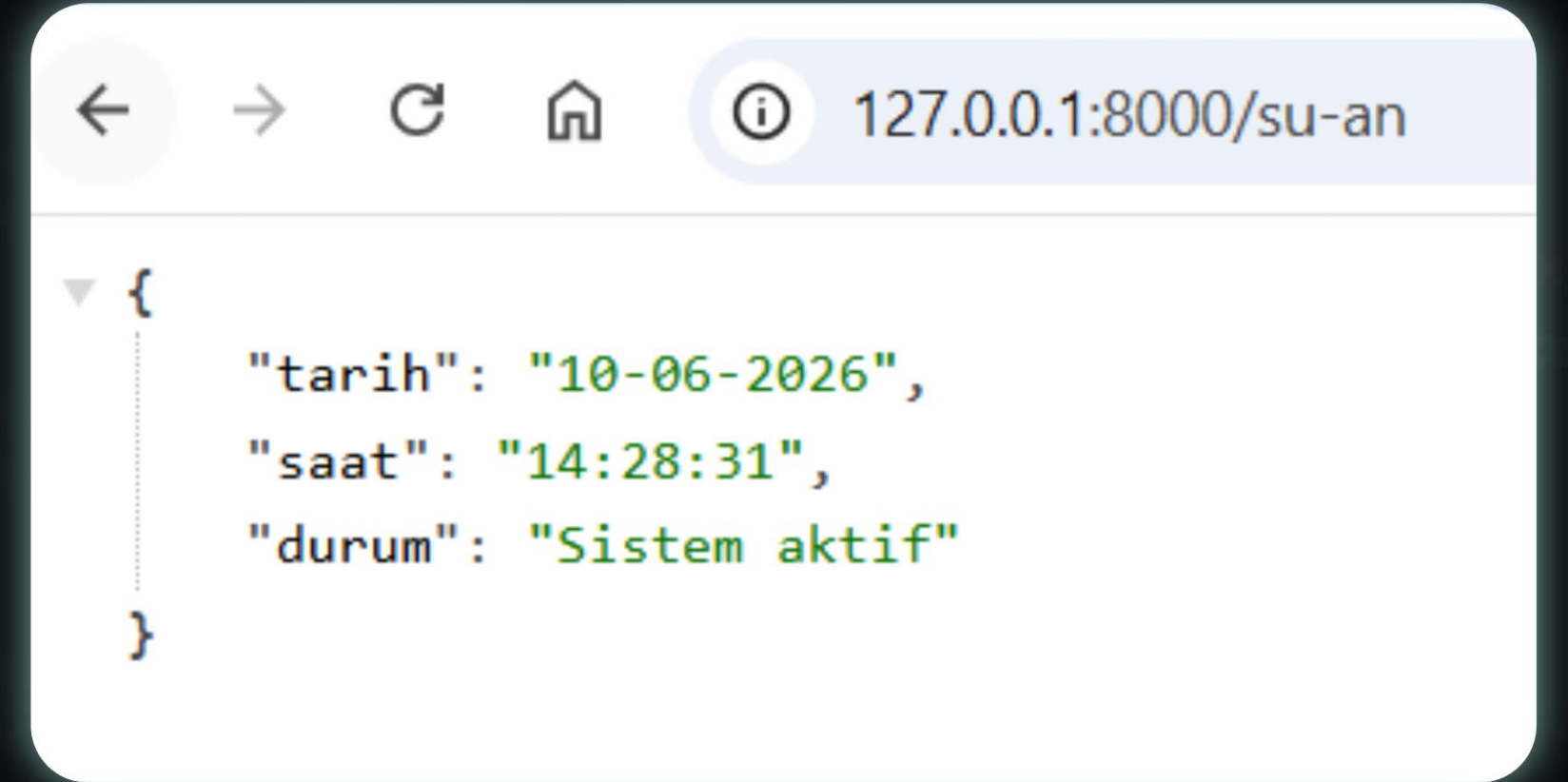


```
54     begeni_sayisi = 0
55
56     @app.get("/like")
57     def begen():
58         global begeni_sayisi
59
60         begeni_sayisi += 1
61         return {
62             "islem": "Beğenildi",
63             "begeni_sayisi": begeni_sayisi
64         }
65
66     @app.get("/dislike")
67     def begenmekten_vazgec():
68         global begeni_sayisi
69
70         if begeni_sayisi > 0:
71             begeni_sayisi -= 1
72         return {
73             "islem": "Beğenilmedi",
74             "begeni_sayisi": begeni_sayisi
75         }
76     else:
77         return {
78             "islem": "Reddedildi",
79             "mesaj": "Skor zaten 0, daha fazla aşağı düşemez!",
80             "begeni_sayisi": begeni_sayisi
81         }
```

Örnek 6

Datetime kütüphanesi ile saat ve tarih bilgisine erişebilmekteyiz.

/su-an servisi ile tarih ve saat bilgisini veriniz.



Örnek 6

Datetime kütüphanesi ile saat ve tarih bilgisine erişebilmekteyiz.

/su-an servisi ile tarih ve saat bilgisini veriniz.



```
@app.get("/su-an")
def anlik_zaman_bilgisi():
    su_an = datetime.now()
    tarih = su_an.strftime("%d-%m-%Y")
    saat = su_an.strftime("%H:%M:%S")
    return [{"tarih": tarih, "saat": saat, "durum": "Sistem aktif"}]
```

Örnek 7

`random.choice(liste_adı)` ile bir liste içinde rasgele seçim yapılabilir. (`random` kütüphanesi import edilmelidir)

`v2/yazi-tura` endpointi yazınız fakat servise her istek attığında daha önce kaç adet yazı kaç adet tura geldiği bilgisi de dönsün.



```
← → ↻ 🏠 ⓘ 127.0.0.1:8000/v2/yazi-tura  
  
▼ {  
  "sonuc": "Yazı",  
  "yazi_sayisi": 1,  
  "tura_sayisi": 0  
}
```

```
← → ↻ 🏠 ⓘ 127.0.0.1:8000/v2/yazi-tura  
  
▼ {  
  "sonuc": "Tura",  
  "yazi_sayisi": 11,  
  "tura_sayisi": 9  
}
```

Örnek 7

`random.choice(liste_adı)` ile bir liste içinde rasgele seçim yapılabilir. (random kütüphanesi import edilmelidir)

`v2/yazi-tura` endpointi yazınız fakat servise her istek attığında daha önce kaç adet yazı kaç adet tura geldiği bilgisi de dönsün.



```
yazi_sayisi = 0
tura_sayisi = 0
@app.get("/v2/yazi-tura")
def yazi_tura_at_v2():
    global yazi_sayisi, tura_sayisi

    para_yuzleri = ["Yazı", "Tura"]
    gelen_yuz = random.choice(para_yuzleri)

    if gelen_yuz == "Yazı":
        yazi_sayisi += 1
    else:
        tura_sayisi += 1

    return {
        "sonuc": gelen_yuz,
        "yazi_sayisi": yazi_sayisi,
        "tura_sayisi": tura_sayisi,
    }
```

Path Parameters ➔



Nedir?

Path Parameter, bir URL'in gövdesine doğrudan yerleştirilen ve sistemdeki belirli, spesifik bir kaynağı (örneğin tek bir kullanıcı, tek bir ürün veya tek bir sipariş) işaret etmek için kullanılan değişkenlerdir.

```
@app.get("/students/1")  
@app.get("/students/2")  
@app.get("/students/3")  
@app.get("/students/4")
```



Giriş

Her öğrenci için ayrı ayrı endpoint yazman mantıklı değildir. Path parameters sayesinde endpointlerimiz dinamik bir yapıya ulaşır.

```
@app.get("/students/1")  
@app.get("/students/2")  
@app.get("/students/3")  
@app.get("/students/4")
```



Syntax

Yandaki kodu yazınız, fakat henüz çalıştırmayınız.



```
students = [  
    {"id": 1, "name": "Ahmet", "age": 20, "city": "samsun"},  
    {"id": 2, "name": "Ayşe", "age": 22, "city": "bursa"},  
    {"id": 3, "name": "Mehmet", "age": 21, "city": "istanbul"},  
    {"id": 4, "name": "Elif", "age": 23, "city": "ankara"},  
    {"id": 5, "name": "Ali", "age": 20, "city": "izmir"}  
]  
  
@app.get("/students")  
def get_students():  
    return students  
  
@app.get("/students/{student_id}")  
def get_student(student_id: int):  
    for student in students:  
        if student["id"] == student_id:  
            return student  
    return {"error": "Öğrenci Bulunamadı", "student_id": student_id}
```

Syntax

```
students = [  
    {"id": 1, "name": "Ahmet", "age": 20, "city": "samsun"},  
    {"id": 2, "name": "Ayşe", "age": 22, "city": "bursa"},  
    {"id": 3, "name": "Mehmet", "age": 21, "city": "istanbul"},  
    {"id": 4, "name": "Elif", "age": 23, "city": "ankara"},  
    {"id": 5, "name": "Ali", "age": 20, "city": "izmir"}  
]
```

```
@app.get("/students")  
def get_students():  
    return students
```

```
@app.get("/students/{student_id}")  
def get_student(student_id: int):  
    for student in students:  
        if student["id"] == student_id:  
            return student  
    return {"error": "Öğrenci Bulunamadı", "student_id": student_id}
```

Liste tipinde veri

Tüm listeyi dönen
endpoint

Path Parameter

Path
tanımlayıcı

Parameter

Kural

```
6 @app.get("/students/{student_id}")
7 def get_student(student_id: int):
8     for student in students:
9         if student["id"] == student_id:
10             return student
11     return {"error": "Öğrenci Bulunamadı", "student_id": student_id}
```

- Path parameterlar {parametre_ismi} şeklinde endpointte tanımlanır.
- Decorator'da tanımlanan path parameter aynı isim ile def içinde çağırılmalıdır.
- Def parametresinde path parameter'ın tipi : (type hint) koyularak belirtilmelidir.

Not : Type Hint zorunlu değildir fakat api tasarımı açısından önemli bir yaklaşımdır.

Test



```
uvicorn api_ders_1:app --reload
```

Server'ı başlatınız.

Alttaki servislere istek atınız.

<http://127.0.0.1:8000/students/1>

<http://127.0.0.1:8000/students/3>

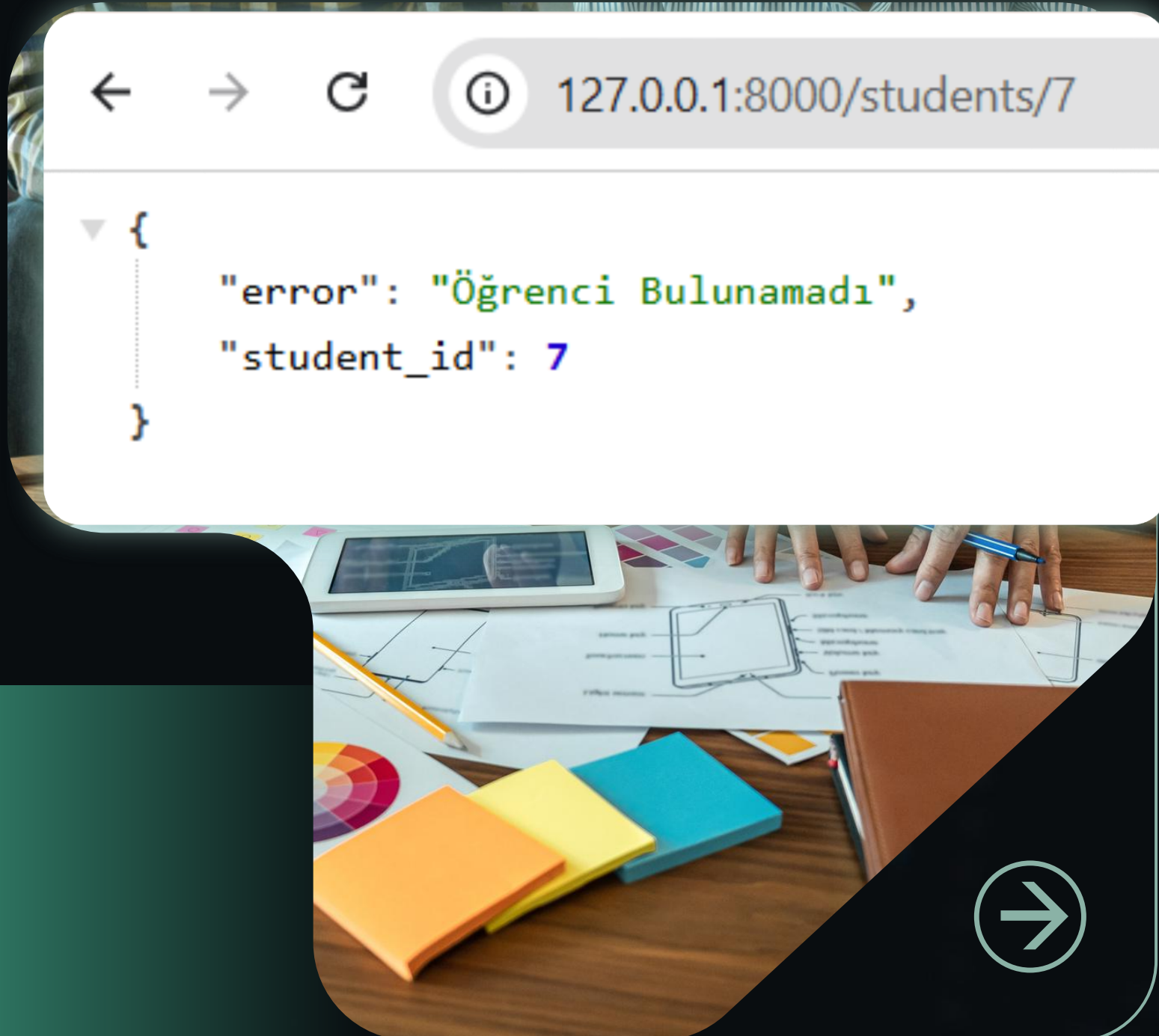
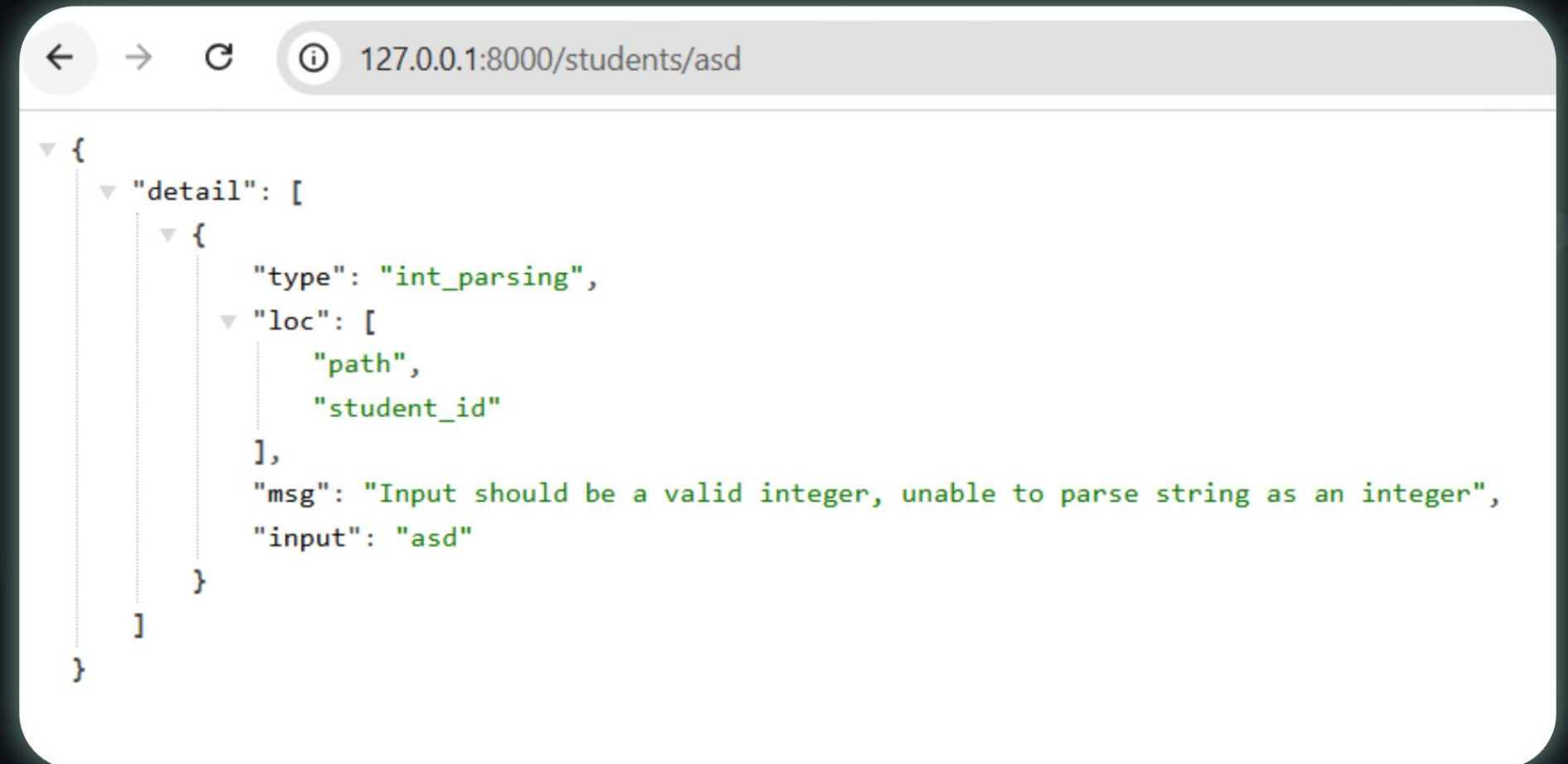
<http://127.0.0.1:8000/students/5>

<http://127.0.0.1:8000/students/7>

<http://127.0.0.1:8000/students/asd>

```
students = [  
    {"id": 1, "name": "Ahmet", "age": 20, "city": "samsun"},  
    {"id": 2, "name": "Ayşe", "age": 22, "city": "bursa"},  
    {"id": 3, "name": "Mehmet", "age": 21, "city": "istanbul"},  
    {"id": 4, "name": "Elif", "age": 23, "city": "ankara"},  
    {"id": 5, "name": "Ali", "age": 20, "city": "izmir"}  
]  
  
@app.get("/students")  
def get_students():  
    return students  
  
@app.get("/students/{student_id}")  
def get_student(student_id: int):  
    for student in students:  
        if student["id"] == student_id:  
            return student  
    return {"error": "Öğrenci Bulunamadı", "student_id": student_id}
```

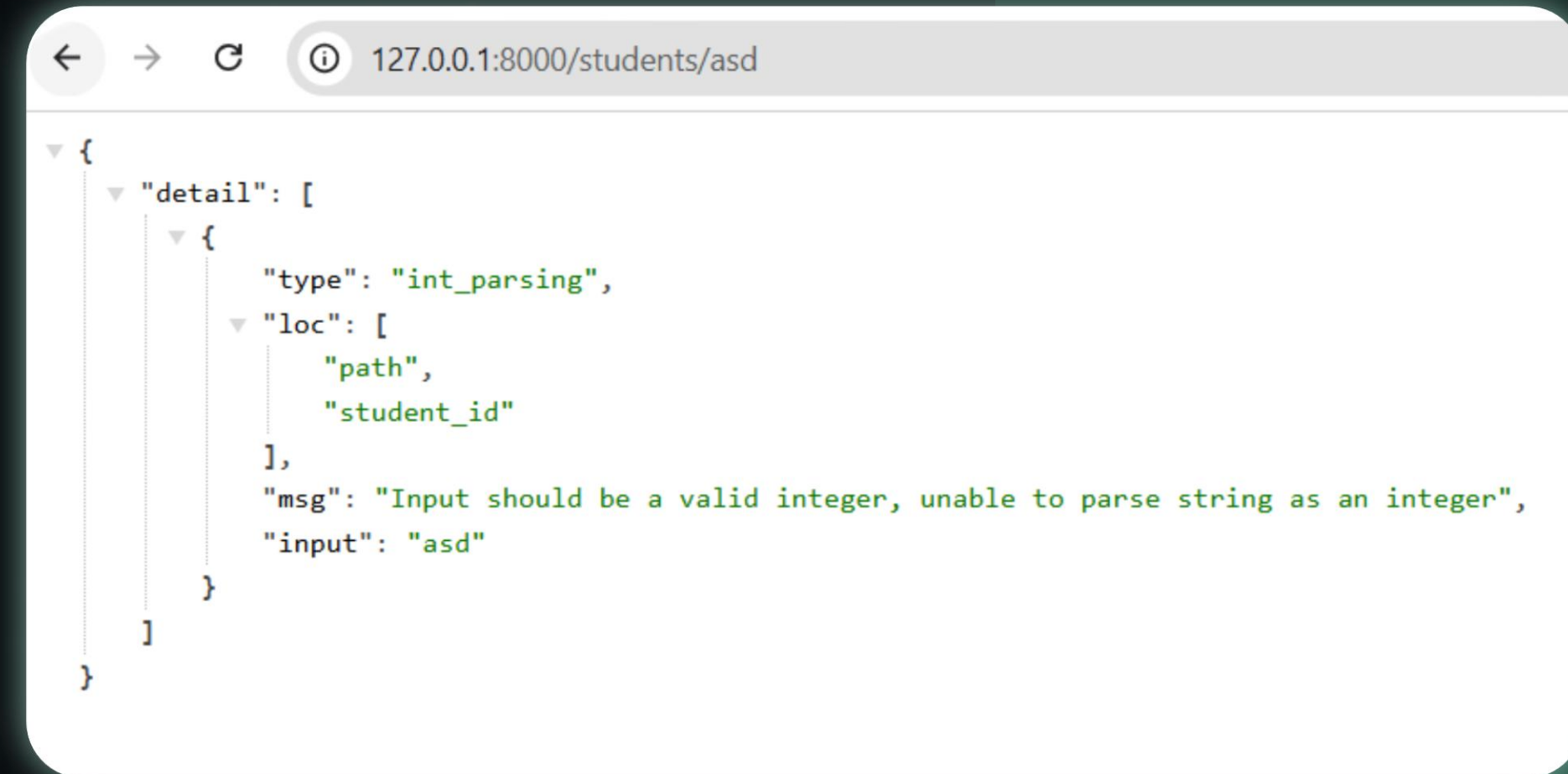
Type Hint



Farklı response vermesinin sebebi nedir ?

Type Hint

Type hint sayesinde beklenmeyen veri tiplerine karşı FastAPI framework'ünün hazır responseleri devreye girmektedir.

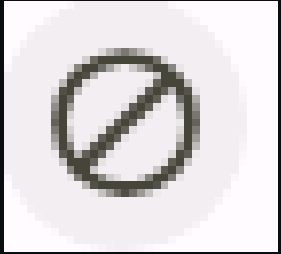


Chrome Network

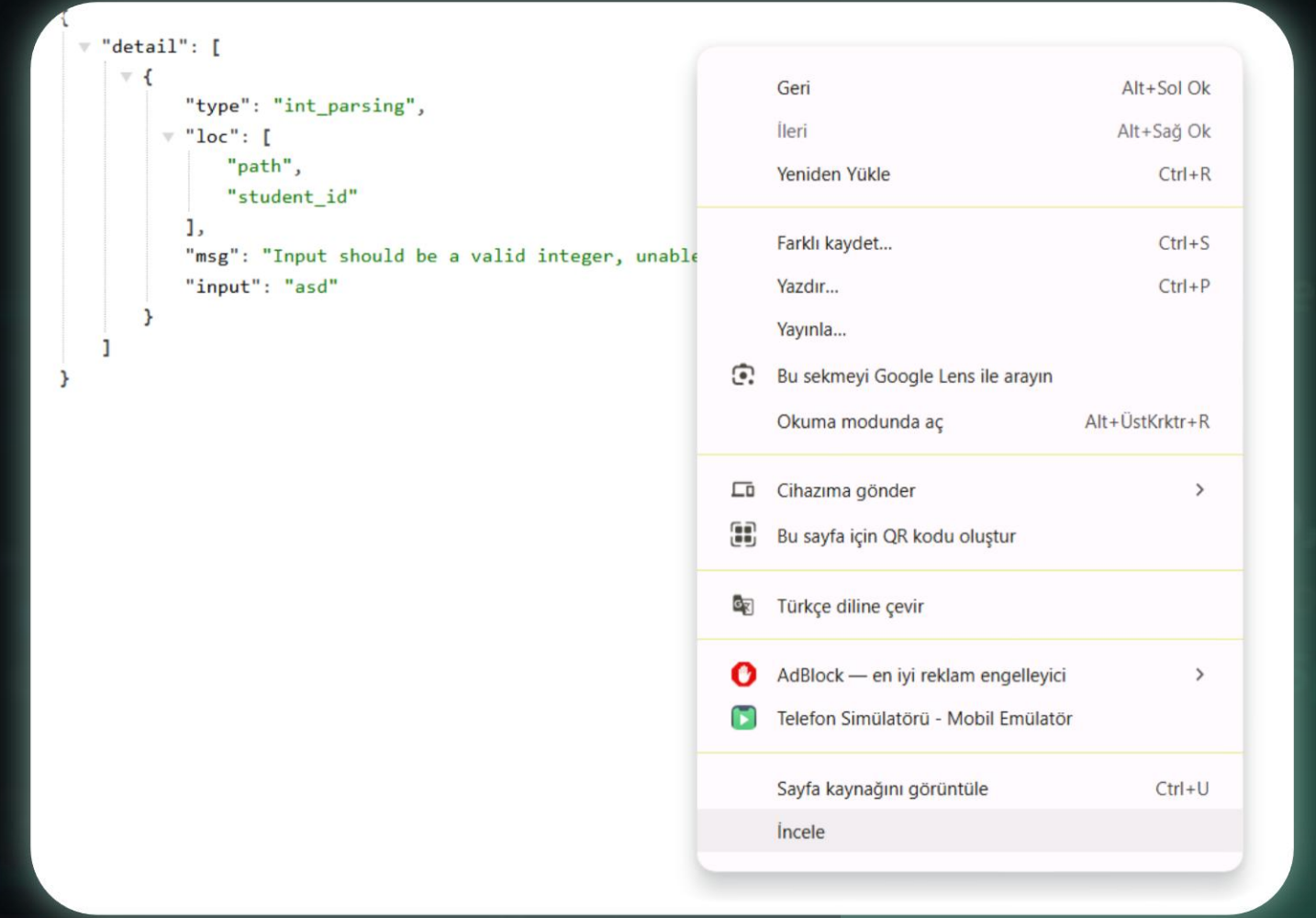
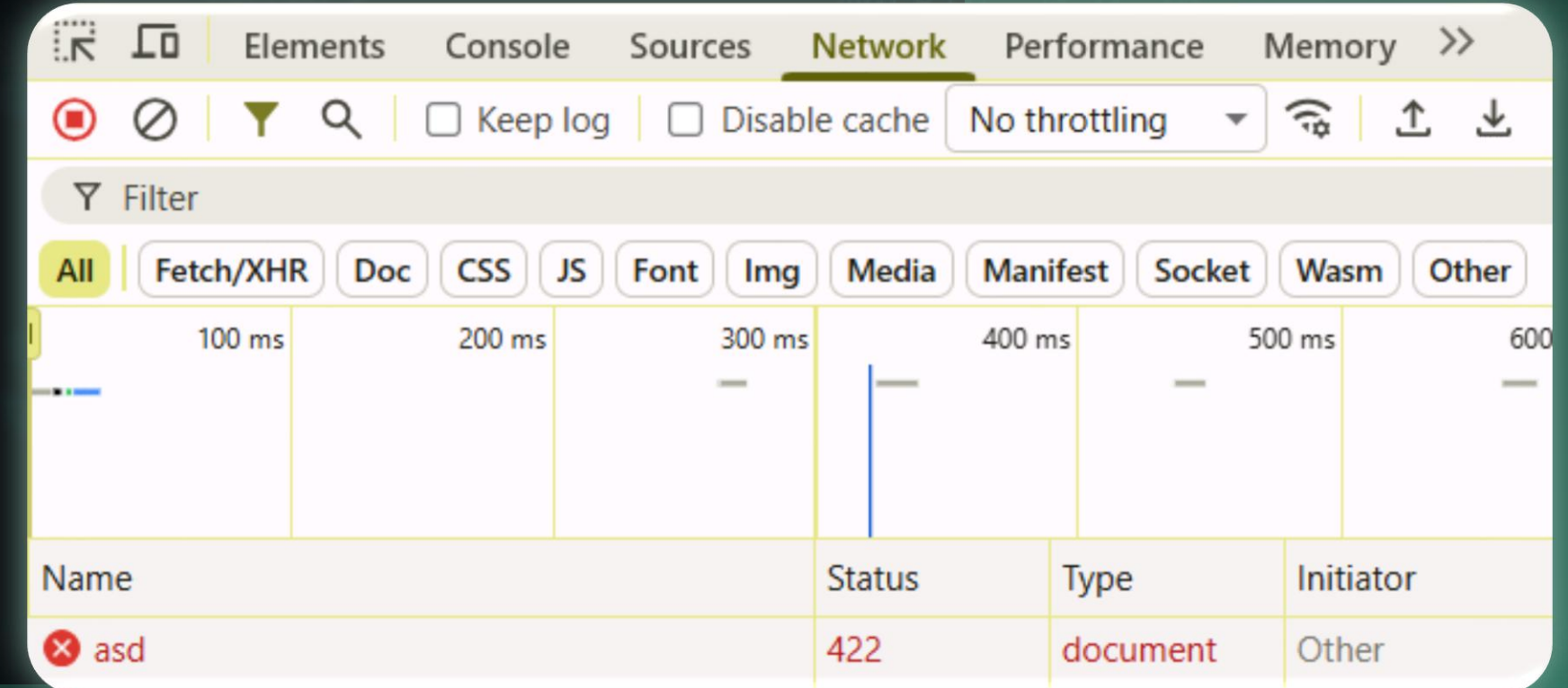
Aşağıdaki adımları ile endpoint response code ulaşılabilir.

- 1- Sağ Tuş – İncele
- 2- Network
- 3- Sayfaya tekrar istek atılır ve path (bu örnek için asd), Name listesinde bulunur.

422 status code anlamı nedir ?



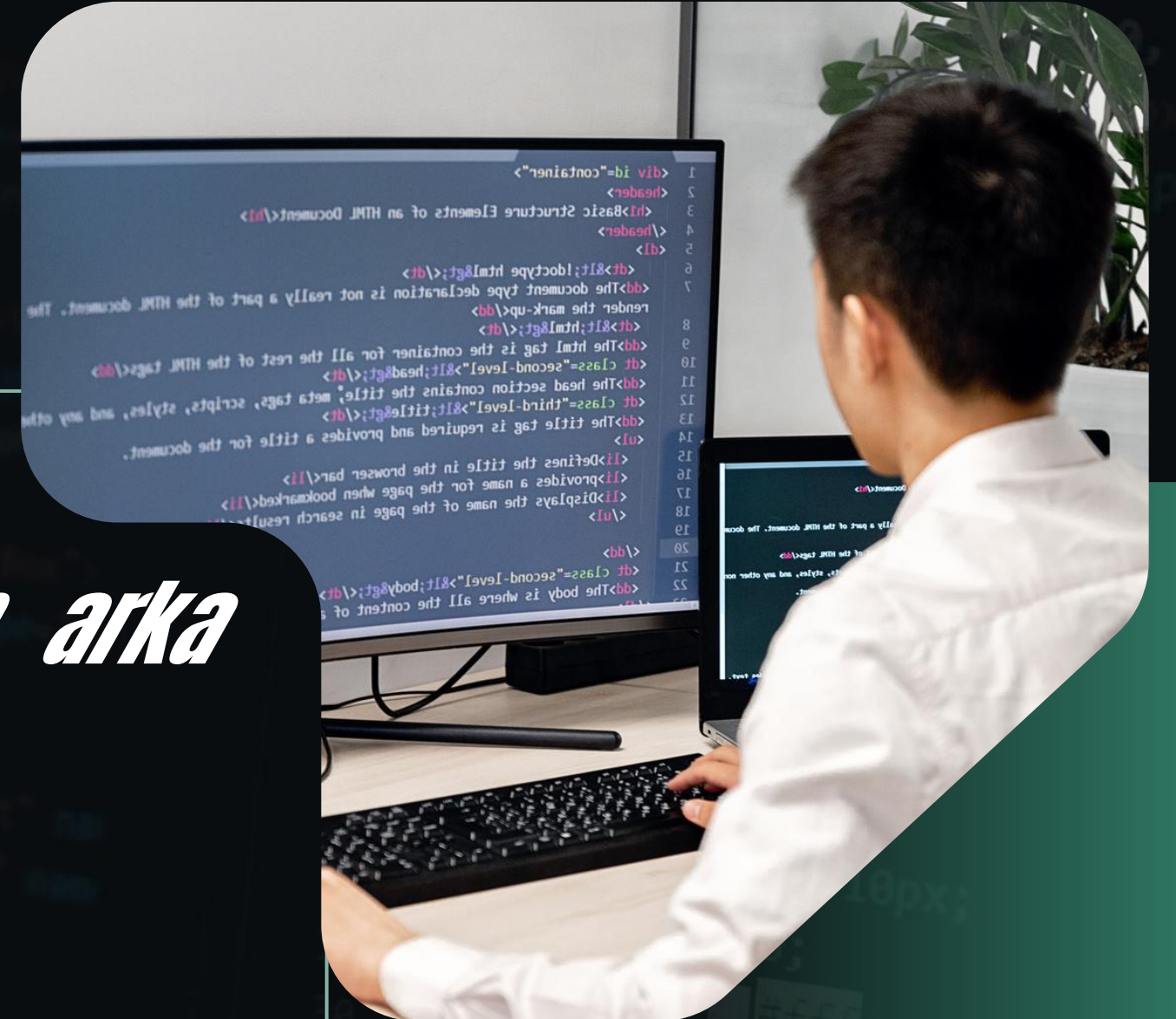
Not : Yadaki ikon'a basarak network ekranı temizlenebilir, ilgili network elemanına tıklayarak diğer detaylara ulaşabilirsiniz. Tersine mühendislik açısından Network sayfası başlangıç adımı olarak görülür.





Network Örneği

Chrome network sayesinde bir web sitesinin arka planda attığı istekleri inceleyebiliriz.



Birden Fazla Parametre

Endpoint tasarımı doğrultusunda birden fazla path parameter kullanılabilir.

```
@app.get("/users/{user_id}/orders/{order_id}")  
✓ def get_order(user_id: int, order_id: int):  
  
✓     return {  
        "user_id": user_id,  
        "order_id": order_id  
    }
```

Uygulama 1

sehirler/{plaka} adında bir endpoint oluşturunuz. Bu endpoint'e istek atıldığında ilgili şehrin **adı**, **plaka kodu**, bulunduğu **bölge** ve **nüfus** sayısı response olarak dönsün. Not : 5 adet şehir tanımlamanız yeterli, bunlar dışında istek atıldığında "hata": "Bu plaka koduna ait şehir bulunamadı." şeklinde response yapılsın.

```
.box{
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%,
width: 400px;
padding: 40px;
background: rgba(0, 0, 0,
box-sizing: border-box;
box-shadow: 0 15px 35px
```

```
# Use the json module to parse the JSON response
theJSON = json.loads(response)

# now we can access the contents of the JSON response
if "title" in theJSON["metadata"]:
    print(theJSON["metadata"]["title"])

# output the number of events, plus the magnitude of each
count = theJSON["metadata"]["count"]
print(str(count) + " events recorded")

for each event, print the place where it occurred
for i in theJSON["features"]:
    print(i["properties"]["place"])
    print("\n")

# only have a magnitude greater than 4.0
es":
  "mag": 4.0,
  "properties": {"mag": 4.0, "place": "Hawaii"}

# where at least 1 person reported feeling it
es":
  "mag": 4.0,
  "properties": {"mag": 4.0, "place": "Hawaii", "felt": 1}
```

Uygulama 1

sehirler/{plaka} adında bir endpoint oluşturunuz. Bu endpoint'e istek atıldığında ilgili şehrin adı, plaka kodu, bulunduğu bölge ve nüfus sayısı response olarak dönsün. Not : 5 adet şehir tanımlamanız yeterli, bunlar dışında istek atıldığında "hata": "Bu plaka koduna ait şehir bulunamadı." şeklinde response yapılsın.

```
sehirler_listesi = [
    {"ad": "Adana", "plaka_kodu": 1, "bolge": "Akdeniz", "nufus": 2274106},
    {"ad": "Ankara", "plaka_kodu": 6, "bolge": "İç Anadolu", "nufus": 5782285},
    {"ad": "İstanbul", "plaka_kodu": 34, "bolge": "Marmara", "nufus": 15907951},
    {"ad": "İzmir", "plaka_kodu": 35, "bolge": "Ege", "nufus": 4462056},
    {"ad": "Samsun", "plaka_kodu": 55, "bolge": "Karadeniz", "nufus": 1368488}
]

@app.get("/sehirler/{plaka}")
def sehir_getir(plaka: int):
    for sehir in sehirler_listesi:
        if sehir["plaka_kodu"] == plaka:
            return sehir

    return {"hata": "Bu plaka koduna ait şehir bulunamadı."}
```

Uygulama 2

Tek bir endpoint üzerinden toplama, çıkarma, çarpma, bölme işlemlerini yapabilecek 'hesapla' endpoint'ini yazınız.

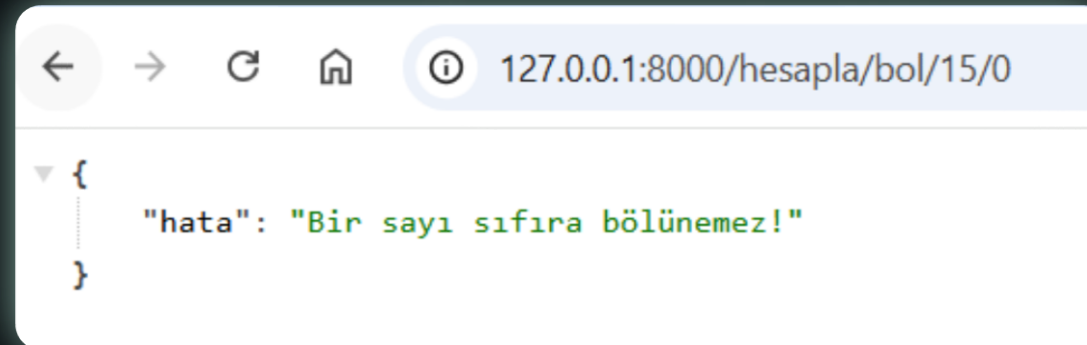
Not: Endpoint'i client ready seviyesinde (nihai hali, canlı ortama almaya hazır, ticarileştirmeye hazır) yazınız. Örn. Hesapla/selam/yazdığında hata vermeli, 0'a bölme durumunda hata vermeli, virgüllü sayı kabul etmeli.

hesapla/topla/15/2.5

hesapla/cikar/20/45

hesapla/carp/5/30

hesapla/bol/13/5



Uygulama 2

Tek bir endpoint üzerinden toplama, çıkarma, çarpma, bölme işlemlerini yapabilecek 'hesapla' endpoint'ini yazınız.

hesapla/topla/15/2.5

hesapla/cikar/20/45

hesapla/carp/5/30

hesapla/bol/13/5

```
@app.get("/hesapla/{islem}/{sayi1}/{sayi2}")
def matematik_islemi(islem: str, sayi1: float, sayi2: float):
    if islem == "topla": sonuc = sayi1 + sayi2
    elif islem == "cikar": sonuc = sayi1 - sayi2
    elif islem == "carp": sonuc = sayi1 * sayi2
    elif islem == "bol":
        if sayi2 == 0: return {"hata": "Bir sayı sıfıra bölünemez!"}
        sonuc = sayi1 / sayi2
    else:
        return {"hata": "İşlem sadece topla, cikar, carp, bol olabilir."}

    return {"islem_turu": islem, "sonuc": sonuc}
```



Tesekkürler



www.furkandurmus.com

