# **FLASK**

Flask, Python dilinde yazılmış web uygulamaları geliştirmek için kullanılan bir **web framework**’üdür.

## **Web Framework Nedir?**

Bir Web Framework’ü, web uygulaması geliştiricilerinin protokol, iş parçacığı yönetimi vb. gibi düşük düzeyli ayrıntılar hakkında endişelenmeden uygulamalar yazmasına olanak tanıyan bir kitaplıklar ve modüller koleksiyonunu temsil eder.

Flask, dinamik içerikli web siteleri oluşturmak için tasarlanmıştır. Basitliği ve esnekliği ile popüler bir seçenektir. Açık kaynak kodlu bir **micro** **framework'dür**.

## **Micro Framework Nedir?**

Flask'a genellikle mikro framework denir. Uygulamanın çekirdeğini basit ve ölçeklenebilir tutmak için tasarlanmıştır.

Veritabanı desteği için bir soyutlama katmanı (abstraction layer) yerine Flask, uygulamaya bu tür yetenekleri eklemek için uzantıları (extensions) destekler.

Micro kavramını için **basit ama genişletilebilir** denilebilir. Şu şekilde açıklanabilir:

Kendinize bir araba oluşturduğunuzu varsayın. Motor, kapılar, tampon, bagaj gibi birçok parçaya ihtiyacınız var. Bu motorun parçalarını almak için iki farklı yere gidiyorsunuz. İlk yer ihtiyacınız olan parçaları ve daha fazlasını paket halinde vermek istiyor. İkinci yer ise sadece motoru veriyor ve dilerseniz eklemeler yaparsınız diyor. İşte gittiğimiz ikinci yer micro framework’ün ta kendisi. İstediğiniz parçayı kullanma veya kullanmama özgürlüğü tanıyan özünde basit bir yapıdır.

Bu parçalara bilgisayar dünyasından örnek vermek gerekirse; veri tabanı, kimlik doğrulama, form doğrulama gibi sistemlerdir. Bu sistemlere, ihtiyaca göre ekleme yapabilme özgürlüğü vardır.

## **Flask Kullanan Global Şirketler:**

🔎Netflix 🔎MIT

🔎Reddit 🔎Uber

🔎Airbnb 🔎Mozilla

🔎Lyft 🔎Red Hat

🔎Patreon 🔎MailGun

Flask, **Werkzeg WSGI** araç setini ve **Jinja2** şablonu temel alır.

## **WSGI**

WSGI (Web Server Gateway Interface), Python ile web uygulamaları geliştirmek için kullanılan bir standarttır. Bu, web sunucuları ile web uygulamaları arasında iletişim kurmak için bir yol sağlar.

## **Werkzeug**

Werkzeug, istekleri, yanıtları ve yardımcı işlevleri yöneten bir araç setidir (toolkit). Bu araç seti, bir web framework oluşturmak için kullanılabilir. Flask frameworkü, temelini oluştururken Werkzeug'u kullanır.

## **jinja2**

Jinja2, Python için yaygın olarak kullanılan bir şablon aracıdır. Bu araç, web sayfalarını oluşturmak için kullanılan şablonları ve verileri birleştirerek dinamik web sayfaları oluşturmaya yardımcı olur.

Bu, Python değişkenlerini aşağıdaki gibi HTML şablonlarına aktarmanıza olanak tanır:

<html>  
 <head>  
 <title> {{ title }} </title>  
 </head>  
 <body>  
 <h1>Hello {{ username }} </h1>  
 </body>  
</html>

## **Neden Flask?**

Flask, Python programlama dilini kullanarak web uygulamaları oluşturma işlemini çok kolaylaştırır. Önemli özellikleri arasında şunlar yer alır:

⭐️Basit yapısı ile öğrenmesi ve kullanması kolaydır.

⭐️Minimum gereksinimleri vardır.

⭐️Esnek ve genişletilebilirdir.

⭐️Ölçeklenebilir ve güvenilirdir.

Bunun üzerine, çok açık bir yapıya sahiptir, bu da okunabilirliği artırır. "Hello World" uygulamasını oluşturmak için sadece birkaç satır kod gereklidir.

İşte bir örnek temel kod yapısı:

A black screen with a blue line

Description automatically generated

A screenshot of a computer

Description automatically generated

Ardından, yalnızca kendi bilgisayarınızda erişilebilir olan bir web sunucusu başlatır. Bir web tarayıcısında 5000 numaralı bağlantı noktasında localhost'u açarsanız (URL), "Hello World!"ü görürüz.

A screenshot of a phone

Description automatically generated

Kodu inceleyelim: Bu kod ne yapıyor?

1-İlk olarak, Flask kütüphanesinden Flask sınıfını içeri aktarıyoruz. Bu sınıf, uygulamamızın temelini oluşturacak.

2-Sonra, Flask sınıfından bir örnek oluşturuyoruz. Bu örneği oluştururken, uygulamamızın adını (name) veriyoruz. Bu, Flask'ın şablonlar, statik dosyalar vb. için nereye bakacağını belirtmemize yardımcı olur.

3-@app.route('/') ifadesi ile '/' yolunu belirtiyoruz. Bu, tarayıcıda ana URL olarak görünecek olan yolu belirtir.

4-Son olarak, hello\_world() adında bir fonksiyon tanımlıyoruz. Bu fonksiyon, kullanıcının tarayıcısında Hello, World!' metnini görüntüleyecek.

## **Flask Temel İşlemler**

### **1- Flask URL Yapısı ve Dinamik URL**

URL yapısı şu şekildedir:

A screenshot of a computer

Description automatically generated

Arama yerine[**http://127.0.0.1:5000/search**](http://127.0.0.1:5000/search) yazdığımız zaman karşımıza çıkan sayfa:

A screenshot of a computer

Description automatically generated

**Dinamik URL:**

Bir sayfamız olduğunu düşünelim ve bu sayfada yüzlerce, binlerce makalemiz olsun. delete/2, delete/54, delete/100… şeklinde her makale için route yazmak mantıklı mı? Tabi ki değil. Dinamik URL burda yardımımıza yetişir:

A screenshot of a computer

Description automatically generated

id nin 54 olduğu düşünelim:

A screenshot of a phone

Description automatically generated

### **2- Jinja Templateler**

A screen shot of a computer

Description automatically generated

A computer screen shot of text

Description automatically generated

Respond şu şekilde olur:

A screenshot of a phone

Description automatically generated

HTML sayfamıza veri de gönderebiliriz. Python verisi nasıl gönderilir? index.hmtl sayfamıza iki tane number göndermek istiyoruz ve number değişkenin değerlerinin 10 ve 20 olduklarını kabul edelim:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

Respondumuz şu şekilde olur:

A screenshot of a computer

Description automatically generated

### **3- Templatelerde Koşullu Durum ve Döngüler**

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a phone

Description automatically generated

**FOR Döngüsü:**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

respond:

A screenshot of a computer

Description automatically generated

### **4- GET ve POST Request, Form İşlemleri**

A screenshot of a computer program

Description automatically generated

A computer screen shot of text

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated A screenshot of a message

Description automatically generated**5- Redirect – URL Yönlendirme**

[**http://127.0.0.1:8080/toplam**](http://127.0.0.1:8080/toplam) bu adress gittiğimizi yani bir get request yaptığımızı düşünelim. A screenshot of a message

Description automatically generated

Böyle bir uyarı vermek yerine kullanıcıyı ana sayfaya yönlendirmek istiyoruz. O zaman:

A computer screen shot of a computer program

Description automatically generatedA computer screen shot of text

Description automatically generated

Öncesi:

A screenshot of a message

Description automatically generated

Ve sonrası:

A screenshot of a computer

Description automatically generated

## **Flask vs Django**

A person with red buttons

Description automatically generated

Python web framework’leri söz konusu olduğunda, **Django** ve **Flask** en popüler olan iki framework olarak karşımıza çıkmaktadır. Her ikisi de Python ile yazılan **açık kaynaklı** web framework’ü olsa da, Django daha **geniş kapsamlı** bir framework’ken, Flask **mikro** bir framework’dür.

**Django**

Django, Python kullanarak web siteleri oluşturmayı kolaylaştıran bir Python framework’dür. Ücretsiz ve açık kaynaklı bir framework’dür. Django, geliştiricilerin hızlı bir şekilde zengin özelliklere sahip, güvenli ve ölçeklenebilir bir web uygulaması oluşturmasına yardımcı olur.

DRY (Don't Repeat Yourself) olarak da adlandırılan bileşenlerin yeniden kullanılabilirliğini vurgulayan Django, oturum açma sistemi, veri tabanı bağlantısı ve CRUD işlemleri (Create Read Update Delete) gibi kullanıma hazır özelliklerle birlikte gelir.

**Django’nun Temel Özellikleri**

**Çok yönlü:** Django çeşitli alanlarda kullanılır. İçerik yönetim sistemleri (CMS), sosyal ağ siteleri ve bilgi işlem platformları gibi çeşitli uygulamalar oluşturmak için Django kullanılır.

**Taşınabilir:** Django, Python'da yazılmıştır. Herhangi bir platformda çalışmak için taşınabilirlik özelliği sağlar.

**Güvenli:** Django, kullanıcı kimlik doğrulama sistemi sunar. Tıklama hırsızlığı, siteler arası komut dosyası çalıştırma ve bunlar gibi yaygın güvenlik sorunlarından korunmaya yardımcı olur.

**Ölçeklenebilir:** DRY ilkelerinin birleşimiyle kodun yeniden kullanımı ve bakımı ölçeklenebilirlik sağlar.

**Uyarlanabilirlik:** JSON, HTML, XML ve daha fazlası dahil olmak üzere çeşitli teknolojilerle uyumludur.

**Popüler:** Spotify, Instagram, Dropbox, Pinterest, Mozilla, YouTube, NASA ve National Geographic gibi şirketler Python framework’ü olarak Django kullanır.

**Flask**

Flask, geliştiricilerin tercih ettikleri veri tabanını ve eklentileri seçme esnekliği sunan Python tabanlı bir mikro framework’dür. Flask, açık kaynak kodlu bir web framework’üdür. Küçük ve orta ölçekli uygulamaların geliştirilmesi için ideal bir araçtır.

Flask, kolay entegrasyon sağlaması ve geniş bir kullanıcı tabanına sahip olması nedeniyle popülerdir. Kullanımı kolay ve basit bir API'ye sahiptir ve hızlı bir şekilde öğrenilir.

**Flask’ın Temel Özellikleri**

**Hafif ve Genişletilebilir:** Geliştiriciler, uygulama mimarisi, kitaplıklar ve uzantılar üzerinde kontrole sahiptir.

**Basit Arayüz:** Flask’ın kullanımı kolaydır.

**İstek İşleme:** HTTP ve RESTful isteklerini destekler.

**Test Etme ve Hata Ayıklama:** Flask, test ve hata ayıklayıcı araçlarını sunar.

**Esnek ve Ölçeklenebilir:** WSGI şablonları desteği, esneklik ve ölçeklenebilirlik sağlar.

**Popüler:** Netflix, Reddit, Uber, Lyft, Zillow, Patreon, Airbnb ve MIT gibi büyük şirketler tarafından kullanılır.

**Flask ve Django Ne Zaman Kullanılır?**

Aşağıdakileri yapmanız gerektiğinde **Django** kullanabilirsiniz:

* Kesin teslim tarihleri ​​olan çok sayfalı ve büyük projelerde
* Geliştirmeyi hızlandırmak istediğinizde
* Erişim desteği almak istediğinizde
* Güvenli projeler oluşturmak istediğinizde
* Gelecekte projeleri büyütme veya daha karmaşık hale getirme planınız varsa
* Yerel ORM desteği ile web uygulamaları oluşturmak istediğinizde

Aşağıdakileri yapmanız gerektiğinde **Flask** kullanabilirsiniz:

* Daha küçük projeler üzerinde çalışırken
* Veri tabanı desteğine ihtiyacınız olduğunda
* Kitaplıkları ve uzantıları seçme esnekliğine ve özgürlüğüne sahip olmak istediğinizde
* Gelecekte projeye yeni uzantılar eklemek istiyorsanız
* Statik web siteleri, hızlı prototipler ve RESTful web hizmetleri oluşturmak istiyorsanız