T.C. KIRKLARELİ ÜNİVERSİTESİ

TEKNİK BİLİMLER MESLEK YÜKSEKOKULU

BİLGİSAYAR PROGRAMCILIĞI BÖLÜMÜ PYTHON PROGRAMLAMA DERSİ

PROJE ÖDEVİ

Projeyi Veren:

Ad-Soyad: Nadir Subaşı

Projeyi Hazırlayan:

Ad-Soyad:Betül Sağlık

Numara:1247008036

İÇİNDEKİLER

İÇİNDEKİLER.....2

KISALTMALAR.....3

1.GİRİŞ..... 4

1.1 Flask Nedir?.....4

1.2. Geliştirici Firma ve Topluluk.....4

1.3.Lisans.....5

2.Hangi Sorunları Çözer?.....5

3.Flask'ın Temel Özellikleri.....7

3.1.Minimal ve Microframework Yapısı.....8

3.2. Routing (URL Yönlendirme) Sistemi.....8

3.3. Jinja2 Template Motoru.....8

3.4. Genişletilebilir Yapı.....8

3.5. Debugger ve Development Server.....9

3.6. RESTful API Kolaylığı.....9

3.7. Blueprint Mimarisi.....9

3.8. Werkzeug ve WSGI Altyapısı.....9

3.9. Python Ekosistem Desteği.....10

4.İstek Döngüsü Aşamaları.....10

5.Nasıl Kurulur?.....10

6.Çalışmak İçin Gerekli Bağımlılıklar 11

7.Ekosistemdeki Yeri ve Alternatifleri.....11

7.1Flask'ın En Önemli Alternatifleri.....12

8.Avantaj ve Dezavantajları.....12

9.Kullanım Alanları.....13

9.1Günlük Hayat Örnekleri.....13

9.2.Flask Bu Uygulamalarda Neden Tercih Edilir.....13

10.Sonuç.....14

11.Kaynakça.....14

KISALTMALAR:

ORM : Object-Relational Mapping

API : Application Programming Interface

EST : Representational State Transfer

MVP : Minimum Viable Product

WSGI : Web Server Gateway Interface

XSS : Cross-Site Scripting

CLI : Command Line Interface

GİRİŞ

Flask Nedir?

Flask; Python programlama dilinin gücünden faydalanan, hafif (lightweight) ve son derece esnek (highly flexible) bir mikro web framework'tür. "Mikro" olarak adlandırılmasının nedeni, Flask'ın çekirdeğinde veritabanı soyutlama katmanı (ORM) veya form doğrulama gibi karmaşık bileşenleri zorunlu olarak bulundurmamasıdır. Bu tercih edilen sade yapı, geliştiricilere proje üzerinde tam kontrol sağlar ve yalnızca ihtiyaç duyulan uzantıların eklenmesine olanak tanır. Flask, kaynak tüketimi düşük ve hızlı çalışmasıyla öne çıkar; özellikle RESTful API geliştirme süreçlerinde sıkça tercih edilir.

Minimal ve modüler yapısı sayesinde geliştirici, uygulamanın mimarisini kendi ihtiyaçlarına göre özgürce tasarlayabilir. Karmaşık ve ağır framework'lerin getirdiği sınırlamalardan uzak kalmak, Flask'ı hem öğrenmesi kolay hem de özelleştirilebilir bir çözüm hâline getirir. Küçük projelerde hızlı prototip geliştirme imkânı sunması ve sadece gerekli bileşenlerin projeye eklenebilmesi, Flask'ı özellikle özgün ve özelleştirilmiş web çözümleri geliştirmek isteyenler için cazip kılar.

Geliştirici Firma ve Topluluk

Flask, başlangıçta bir firma tarafından değil, bireysel geliştiriciler ve açık kaynak topluluğu tarafından hayata geçirilmiş bir projedir. İlk olarak 2010 yılında Armin Ronacher tarafından geliştirilmiştir. İlginç bir şekilde, proje başlangıçta bir 1 Nisan şakası olarak ortaya çıkmış, ancak kısa sürede Python topluluğu tarafından ciddiye alınmış ve popülerlik kazanmıştır.

Flask'ın geliştirilmesinde önemli bir rol oynayan topluluk, Pocoo (veya Poocco) olarak bilinen Python meraklılarından oluşan bir gruptu. Pocoo, 2004 yılında kurulmuş ve birçok temel Python projesini (Werkzeug, Jinja2 gibi) desteklemiş bir topluluktur.Ayrıca Ronacher'da bu topluluğun bir üyesiydi. Flask, bu ekosistem içerisinde Werkzeug ve Jinja2 gibi temel kütüphanelerle birlikte geliştirilmiş ve dağıtılmıştır. Günümüzde Flask'ın bakımı ve geliştirilmesi, dünya genelinde geniş bir gönüllü Python geliştirici topluluğu tarafından sürdürülmektedir.

Bu topluluk destekli yapı, Flask'ın hızlı gelişmesini, çeşitli platformlarda kullanılmasını ve sürekli güncellenmesini sağlamıştır. Ayrıca, açık kaynaklı olması sayesinde hem bireysel geliştiriciler hem de şirketler, Flask'ı kendi projelerine adapte edebilmekte ve katkıda bulunabilmektedir. Flask'ın bu özgür ve topluluk odaklı yaklaşımı, diğer ticari ve kapalı kaynak web frameworklerinden ayrılmasını sağlayan temel faktörlerden biridir.

Lisans

Flask, açık kaynak dünyasında yaygın olarak kullanılan BSD 3-Clause Lisansı altında dağıtılmaktadır. Bu lisans modeli, geliştiricilere büyük bir özgürlük sunar; yazılımın kopyalanması, değiştirilmesi, yeniden dağıtılması ve ticari projelerde kullanılması herhangi bir ek kısıtlama olmaksızın mümkündür. Tek zorunluluk, orijinal telif hakkı bildiriminin ve lisans metninin korunmasıdır.

Flask ekosisteminde yer alan temel bileşenler Werkzeug, Jinja2, MarkupSafe ve ItsDangerous'de aynı lisans ile dağıtılmaktadır. Bu sayede geliştiriciler, tüm ekosistemi hem eğitim hem de ticari projelerde özgürce kullanabilir. Lisansın sağladığı esneklik, Flask'ı yalnızca bireysel geliştiriciler için değil, büyük ölçekli kurumsal projeler için de uygun hâle getirmektedir.

BSD 3-Clause lisansının en önemli avantajları şunlardır:

-Ticari kullanım özgürlüğü: Şirketler, Flask'ı kendi ürün ve servislerinde lisans ücreti ödemeden kullanabilir.

-Kod değişikliği ve özelleştirme: İhtiyaçlara göre framework üzerinde değişiklik yapmak ve özelleştirmeler eklemek mümkündür.

-Geniş topluluk desteği: Açık kaynak olması, dünya çapında birçok geliştiricinin katkı sunmasını sağlar ve bu da sürekli güncellenen bir ekosistem yaratır.

Bu lisans yapısı sayesinde Flask, hem bireysel hem de kurumsal geliştiriciler için güvenilir ve esnek bir web framework hâline gelmiştir. Geliştiriciler, proje gereksinimlerine göre gerekli modülleri seçip entegre ederek hem minimal hem de genişletilebilir uygulamalar geliştirebilirler.

Hangi Sorunları Çözer?

Flask, yazılım geliştiricilerin hızlı, esnek ve yönetilebilir web uygulamaları geliştirme ihtiyacını karşılamak için tasarlanmış bir web framework'tür. Geleneksel web geliştirme süreçlerinde, geliştiriciler hem sunucu tarafı mantığını kurmak hem de URL yönetimi, form işlemleri, istek–yanıt döngüsü, oturum yönetimi ve veri tabanı işlemleri gibi temel web işlevlerini sıfırdan yazmak zorunda kalıyordu. Bu durum, geliştirme sürecini yavaşlatıyor, kod karmaşıklığını artırıyor ve projelerin bakımını zorlaştırıyordu.

Flask, bu noktada devreye girerek gereksiz karmaşıklığı ortadan kaldıran, modüler ve minimal bir web framework sunar. Geliştiriciler yalnızca ihtiyaç duydukları bileşenleri ekleyerek projelerini oluşturabilir ve böylece hem öğrenmesi kolay hem de genişletilebilir bir yapıya sahip olur. Flask, özellikle küçük ve orta ölçekli projelerde, prototipleme süreçlerinde ve API geliştirme çalışmalarında yüksek verimlilik sağlar.

Flask'ın çözdüğü temel problemler şunlardır:

1.Web Uygulaması Geliştirmede Basitlik ve Hız Sağlama
Flask, karmaşık yapılandırmalar ve ağır bağımlılıklar olmadan çalışır. "Microframework" olarak adlandırılmasının nedeni, temel web geliştirme ihtiyaçlarını karşılayıp kullanıcıya fazladan yük bindirmemesidir. Bu sayede sıfırdan bir web projesi kurmak çok daha hızlı ve pratiktir. Özellikle startup projelerinde veya hızlı prototip geliştirme süreçlerinde bu hız, zaman ve maliyet tasarrufu sağlar.

2.Esneklik İhtiyacı
Bazı büyük frameworkler (ör. Django), proje mimarisini kullanıcıya dayatır ve belirli dosya yapıları ile kurallara uyulmasını zorunlu kılar. Flask ise hiçbir zorunlu dosya yapısı veya kurallara bağlı kalmadan çalışır. Geliştirici, projesini istediği gibi tasarlayabilir ve farklı modülleri kendi mantığına göre organize edebilir. Bu esneklik, özellikle küçük projelerde, araştırma prototiplerinde ve MVP (Minimum Viable Product) süreçlerinde büyük avantaj sağlar.

3.Modülerlik ve İhtiyaca Göre Genişleyebilme
Flask yalnızca temel çekirdeği sağlar:

-URL yönlendirme

-HTTP istek--yanıt yönetimi

-Template render etme (Jinja2 ile)

İhtiyaç duyulduğunda, geliştirici proje gereksinimlerine uygun olarak aşağıdaki özellikleri kolayca entegre edebilir:

-Veritabanı yönetimi ve ORM (SQLAlchemy gibi)

-Kimlik doğrulama ve kullanıcı yönetimi

-Form işleme ve doğrulama

-REST API geliştirme

Bu yapı, Flask'ı "yalın ama genişletilebilir" bir framework hâline getirir ve gereksiz şişkinlik sorununu ortadan kaldırır.

4.Hızlı Prototipleme ve Öğrenme Kolaylığı
Flask, anlaşılması ve kullanılması son derece kolay bir framework'tür. Bu nedenle hem öğrenciler hem de profesyonel geliştiriciler, hızlı bir şekilde çalışan prototipler ve küçük uygulamalar oluşturabilir. Araştırma projelerinde, MVP süreçlerinde veya kısa süreli web servislerinde Flask, öğrenme kolaylığı ve hızlı geliştirme avantajı sunar.

5.API Geliştirme Süreçlerini Basitleştirme
Modern uygulamaların çoğu REST API'lere ihtiyaç duyar. Flask, API geliştirmeyi kolaylaştıran dekoratör yapısı ve minimal tasarımı ile karmaşık sunucu altyapılarını basite indirger. Geliştirici, sadece gerekli endpoint'leri tanımlayarak işlevsel ve bakımı kolay API'ler oluşturabilir. Flask ayrıca JSON tabanlı veri alışverişini ve HTTP metodlarını kolayca yönetebilir, böylece modern web ve mobil uygulamalar için ideal bir temel sağlar.

6.Esnek Entegrasyon ve Özelleştirme
Flask, üçüncü taraf kütüphanelerle ve modern Python paketleriyle kolayca entegre edilebilir. Örneğin

-Veri tabanı işlemleri için SQLAlchemy

-Form yönetimi için WTForms

-API doğrulama ve serileştirme için Marshmallow

Bu esneklik, projeyi geliştiricinin ihtiyacına göre özelleştirmeyi kolaylaştırır ve yazılım geliştirme sürecini hızlandırır.

7.Küçük ve Büyük Ölçekli Projelerde Kullanılabilirlik
Her ne kadar Flask küçük ve orta ölçekli projeler için idealse de, mikroservis tabanlı mimarilerde büyük projelerde de etkin bir şekilde kullanılabilir. Örneğin Netflix, Lyft ve Airbnb, dahili araçlar ve mikroservislerde Flask kullanarak bu esnek ve minimal yapının avantajlarından yararlanmaktadır.

8.Kaynak Yönetimi ve Hafiflik
Flask, minimal tasarımı sayesinde sunucu kaynaklarını verimli kullanır. Büyük frameworklerde görülen gereksiz kütüphane yüklemeleri veya otomatik eklenen modüller, sistem kaynaklarını tüketebilir. Flask, yalnızca gerekli bileşenleri kullanarak hafif bir yapı sunar ve uygulamanın hızlı çalışmasını sağlar.

Kısacası:

Flask, modern web geliştirme süreçlerinde karşılaşılan temel sorunları çözmek amacıyla geliştirilmiş hızlı, esnek ve modüler bir mikro web framework'tür. Karmaşık ve ağır frameworklerin getirdiği zorunlu yapılandırmaları ortadan kaldırarak geliştiriciye kontrolü elinde tutma imkânı sunar. Bu sayede hem küçük ve orta ölçekli projelerde hem de mikroservis tabanlı büyük sistemlerde etkili bir şekilde kullanılabilir.

Flask'ın en önemli avantajları arasında minimal ve öğrenmesi kolay yapısı, modülerliği ve üçüncü taraf kütüphanelerle genişletilebilme imkânı yer alır. Geliştiriciler, sadece ihtiyaç duydukları özellikleri ekleyerek projelerini optimize edebilir, gereksiz şişkinliklerden uzak durabilirler. Ayrıca REST API geliştirme süreçlerini basitleştirmesi ve hızlı prototip geliştirmeye olanak sağlaması, Flask'ı araştırma projeleri, MVP süreçleri ve startup projeleri için ideal bir çözüm hâline getirir.

Sonuç olarak Flask, basitlik, esneklik ve hız gibi temel ihtiyaçları karşılamak için tasarlanmış bir framework olarak, Python ekosisteminde hem eğitim hem de endüstriyel projelerde geniş bir kullanım alanı bulmaktadır. Kullanıcıya sağladığı özgürlük ve modüler yapı sayesinde, özgün ve özelleştirilmiş web çözümleri oluşturmak isteyenler için vazgeçilmez bir araçtır.

Flask'ın Temel Özellikleri

Flask, minimal tasarımına rağmen web uygulaması geliştirme sürecini kolaylaştıran bir dizi önemli özellikle donatılmıştır. Bu özellikler, hem yeni başlayan geliştiricilerin öğrenmesini kolaylaştırır hem de profesyonel projelerde ihtiyaç duyulan esnekliği sağlar. Flask'ın temel

özellikleri, modüler yapısı ve genişletilebilir tasarımıyla birleşerek framework'ü oldukça güçlü bir hâle getirir. Aşağıda Flask'ın en dikkat çeken özellikleri ayrıntılı bir şekilde açıklanmaktadır:

1. Minimal ve Microframework Yapısı

Flask'ın en bilinen özelliklerinden biri, bir "microframework" olmasıdır. Bu terim, Flask'ın az özellik içerdiği anlamına gelmez; aksine, gereksiz bileşenler barındırmayan, yalnızca temel web işlevlerini sunan bir yapıya sahip olduğunu ifade eder.
Bu sayede kullanıcı:

-Gerek duymadığı modüllerle uğraşmadan hızlıca geliştirmeye başlayabilir,

-Projede tam kontrol sağlayabilir,

-Sadece gerekli gördüğü paketleri kendisi ekleyebilir.

Bu yapı özellikle karmaşadan uzak, sade, özelleştirilebilir projeler geliştirmek isteyenler için büyük avantaj sağlar.

2. Routing (URL Yönlendirme) Sistemi

Flask, URL'leri fonksiyonlara bağlayan gelişmiş bir routing sistemi sunar. @app.route() dekoratörü sayesinde hangi URL'nin hangi işlevi çalıştıracağı kolayca belirlenir.
Bu sistem:

-Temiz URL yapısı sağlar,

-API endpoint oluşturmayı kolaylaştırır

-Farklı HTTP metotlarını yönetebilmeye (GET, POST, PUT, DELETE) olanak tanır.

Routing mekanizması, modern web servislerinin temelini oluşturduğu için Flask'ın en güçlü yönlerinden biridir.

3. Jinja2 Template Motoru Desteği

Flask, HTML şablonlarını dinamik hâle getiren güçlü Jinja2 template motorunu kullanır. Bu sayede:

-HTML içinde Python benzeri ifadeler kullanılabilir,

-Değişkenler, döngüler, koşullar gibi yapılar şablonlara entegre edilebilir,

-Büyük web projelerinde tekrar eden tasarım parçaları (navbar, footer vb.) "template inheritance" ile kolayca yönetilir.

Bu özellik hem frontend hem backend arasında köprü görevi görür ve kullanıcıya büyük esneklik sağlar.

4. Genişletilebilir Yapı ve Zengin Eklenti Desteği

Flask'ın temel prensibi şudur:
"Ne istiyorsan ekle, istemediğini yükleme."
Bu yaklaşım sayesinde Flask gerçek bir "tamamlayıcı framework" hâline gelir.

Kullanabileceğiniz popüler eklentiler:

-Flask-SQLAlchemy -> ORM ve veritabanı yönetimi

-Flask-Login -> Kullanıcı girişi, oturum, kimlik doğrulama

-Flask-Mail -> Mail gönderimi

-Flask-WTF -> Form yönetimi ve doğrulama

-Flask-Migrate -> Veritabanı migration işlemleri

Bu eklentiler sayesinde Flask, istenirse Django kadar güçlü bir yapıya dönüştürülebilir.

5. Geliştirici Dostu Yapı: Debugger ve Development Server

Flask, geliştirme sürecini hızlandırmak için yerleşik bir geliştirme sunucusuna sahiptir.
Sunucu otomatik olarak:

-Kod değişikliklerini algılar,

-Uygulamayı yeniden başlatır,

-Hata olduğunda tarayıcı üzerinde interaktif bir debug ekranı gösterir.

Bu sistem özellikle hata ayıklamayı kolaylaştırır ve geliştiricinin verimliliğini artırır.

6. RESTful API Geliştirme Kolaylığı

Flask'ın hafif yapısı, onu modern API geliştirme için ideal hâle getirir.
Geliştirici:

-JSON veri döndürebilir,

-HTTP metodlarını yönetebilir,

-API endpoint'lerini kolayca oluşturabilir.

Bu yüzden Flask, mikroservis mimarisinde sık tercih edilir.

7. Modüler Uygulama Mimarisi (Blueprints)

Flask, büyük projelerde düzeni korumak için Blueprint yapısını sunar. Bu özellik:

-Modüler dosya yapısı oluşturmanı,

-Projeyi parçalara ayırmanı,

-Farklı fonksiyonları bağımsız modüller hâline getirmeni sağlar.

Orta ve büyük ölçekli projelerde bu yapı kod tekrarını azaltır ve sürdürülebilirlik sağlar.

8. WSGI ve Werkzeug Üzerine Kurulu Yapı

Flask'ın alt yapısı Werkzeug WSGI kütüphanesine dayanır.

Bu sayede:

Güvenli,hızlı,standartlara uygun bir sunucu tarafı iletişimi sağlanır.

Flask'ın başarısının temel sebeplerinden biri de bu sağlam temel üzerine kurulmuş olmasıdır.

9. Python Topluluğu ve Ekosisteminin Gücü

Flask, Python dünyasının geniş topluluğu tarafından desteklenir.
Bu sayede:

-Çok sayıda öğretici kaynak,

-Blog yazısı,

-StackOverflow cevabı,

-Ücretsiz eğitim materyalleri bulunur.

İstek Döngüsü Aşamaları

İstek Alma (Werkzeug): Kullanıcının tarayıcıdan gönderdiği tüm HTTP verileri (URL, başlıklar, form bilgileri) Werkzeug tarafından yakalanır ve Flask'a ait bir Request nesnesi hâline getirilir.

Yönlendirme (Routing): Flask, Request nesnesindeki URL bilgisini alır ve uygulama içindeki @app.route() dekoratörleri ile eşleştirir.

İşlev Çağrısı (View Function): Eşleşen URL'ye karşılık gelen Python fonksiyonu (View Function) çalıştırılır. Bu fonksiyonda veritabanı sorguları yapılır, hesaplamalar gerçekleştirilir ve sayfa için gerekli veriler toplanır.

Yanıt Oluşturma (Jinja2): Toplanan veriler, Jinja2 şablon motoruna gönderilir. Jinja2 bu verileri HTML iskeletine yerleştirir ve son HTML çıktısını oluşturur. (Eğer bir API isteği ise, yanıt JSON formatında serileştirilir.)

Yanıt Gönderme: Flask, oluşturulan HTML veya JSON yanıtını Werkzeug aracılığıyla bir HTTP Response nesnesine dönüştürür ve bu yanıtı son kullanıcıya geri gönderir.

Bu döngü, Flask'ın mimarisindeki modülerliği ve hızı sağlayan temel mekanizmadır.

Nasıl Kurulur?

Öncelikle, Python'ın güncel bir sürümü resmi web sitesinden indirilip kurulur ve bu sırada PATH ortam değişkenine eklenmesi sağlanır. Kurulumun doğruluğu Komut İstemcisi'nde 'python –version' komutu ile kontrol edildikten sonra, projeye başlamak için' cd desktop', 'mkdir demo' ve 'cd demo' komutları ile yeni proje klasörüne geçiş yapılır. Ardından, projeye özel bağımlılık izolasyonu için gerekli olan sanal ortam (virtual environment), önce 'pip install virtualenvwrapper-win' komutu ile yöneticisi kurularak ve devamında 'mkvirtualenv flask_env' komutu ile oluşturulup aktif hale getirilir. Sanal ortam (flask_env) aktif duruma geçtiğinde, 'pip install flask' komutu çalıştırılarak ana çerçeve kurulur; bu komut Flask'ı ve onun zorunlu bağımlılıkları olan Werkzeug, Jinja2, ItsDangerous, Click gibi tüm paketleri otomatik olarak indirir. Kurulumun başarılı olduğunu ve hangi paketlerin eklendiğini görmek için pip freeze komutu kullanılır ve son olarak Python kabuğunda import flask ile hata kontrolü yapılır.

Çalışmak İçin Gerekli Bağımlılıklar

Flask, minimal bir çerçeve olmasına rağmen, arka planda çalışmak için aşağıdaki zorunlu Python kütüphanelerine ihtiyaç duyar. Programın kurulumunda 'pip install Flask ' yazıldığında bu kütüphanler otomatik olarak kurulur.

1. Werkzeug: Flask'ın temelini oluşturur. Gelen HTTP isteklerini yönetir, URL yönlendirmesi (routing) ve yanıtları (response) oluşturma işlerini halleder.

2. Jinja2: Dinamik HTML sayfaları oluşturmak için kullanılır. Python verilerini alıp HTML şablonlarıyla birleştirerek son kullanıcıya gönderilecek çıktıyı hazırlar.

3. ItsDangerous: Özellikle Flask oturumlarını (session) çerezlerde güvenli bir şekilde saklamak için kullanılır. Oturum verilerinin değiştirilmediğinden emin olmak için veriyi imzalar.

4.Click: Flask uygulamasının komut satırı üzerinden yönetilmesini sağlayan güçlü bir araçtır. Flask'ın flask run, flask shell, flask routes gibi komutlarının çalışmasını sağlar. Kullanıcıya özel komutlar yazma imkânı tanır ve uygulamanın yönetimini kolaylaştırır. Flask'ın CLI cephesi Click sayesinde çalışır ve bu kütüphane olmadan framework'ün komut satırı özellikleri kullanılamaz.

5. MarkupSafe: Jinja2 şablon motoruyla birlikte çalışarak, HTML çıktılarında XSS (Cross-Site Scripting) saldırılarına karşı koruma sağlamak için veriyi otomatik olarak kaçırır (escaping).

Ek Yazılımlar

-Üretim WSGI Sunucusu: Geliştirme sunucusu yerine, daha sağlam ve performanslı bir sunucuya ihtiyaç duyulur (Örn: Gunicorn veya uWSGI).

-Ters Vekil Sunucu (Reverse Proxy): Güvenlik, yük dengeleme ve statik dosya sunumu için kullanılır (Örn: Nginx veya Apache).

-Veritabanı Sistemi: Projeniz veri saklayacaksa (neredeyse her zaman böyledir), bir veritabanı sunucusuna ihtiyacınız vardır (Örn: PostgreSQL, MySQL, MongoDB).

Ekosistemdeki Yeri ve Alternatifleri

Flask, Python ekosisteminde minimal ve son derece esnek bir web framework olarak öne çıkar. Özellikle küçük ve orta ölçekli web uygulamaları, prototipler veya RESTful API projeleri geliştirmek için idealdir. Flask, temel olarak HTTP request-response yönetimi, routing (yönlendirme), template engine (Jinja2) ve basit middleware desteği gibi kritik web geliştirme araçlarını sunar. Minimal yapısı sayesinde geliştirici, uygulamanın mimarisini kendi ihtiyaçlarına göre özgürce tasarlayabilir ve gereksiz karmaşıklıktan uzak kalabilir. Bununla birlikte, kullanıcı yönetimi, veri tabanı işlemleri veya admin paneli gibi gelişmiş özellikler Flask'ın kendi içinde bulunmaz; bu nedenle bu tür ihtiyaçlar için ek kütüphaneler veya modüller kullanmak gerekir. Bu yapı, Flask'ı hem öğrenmesi kolay hem de esnek bir framework hâline getirirken, aynı zamanda geliştiricinin projenin her aşamasında kontrolü elinde tutmasına olanak sağlar. Küçük projelerde hızlı prototip geliştirme imkânı sunması ve modüler yapısıyla yalnızca gerekli bileşenlerin eklenebilmesi, Flask'ı özellikle özgün ve özelleştirilmiş web çözümleri geliştirmek isteyenler için cazip kılar.

Flask'ın En Önemli Alternatifleri

1.FastAPI

2.Django

Avantaj ve Dezavantajları

Flask:

Avantajları: Minimal ve esnek yapısı sayesinde kullanıcıya uygulamanın mimarisini özgürce kurma imkanı verir. Öğrenmesi kolaydır ve küçük projelerde hızlı geliştirme olanağı sağlar. Geniş Python topluluğu ve bol dokümantasyon desteği vardır. İhtiyaca göre modüler bir şekilde ek kütüphanelerle genişletilebilir.

Dezavantajları: Büyük ve karmaşık projelerde pek çok ek kütüphane ve yapılandırma gerekir. ORM, admin paneli veya kullanıcı yönetimi gibi gelişmiş özellikler kendi başına yoktur. Asenkron programlamayı doğrudan desteklemez, performans odaklı API projelerinde ek yapı gerekebilir.

Django:

Avantajları: “Batteries included” yaklaşımıyla ORM, admin paneli, kullanıcı yönetimi ve güvenlik gibi pek çok özellik hazır olarak gelir. Büyük ve karmaşık projeler için güçlü ve kapsamlı bir temel sağlar. Çok büyük bir topluluk ve kapsamlı dokümantasyon desteği vardır.

Dezavantajları: Flask’a kıyasla daha az esnektir; framework kuralları uygulamanın mimarisini sınırlayabilir. Küçük projelerde fazla ağır ve karmaşık olabilir, gereksiz yük oluşturabilir.

FastAPI:

Avantajları: Modern ve yüksek performanslıdır, asenkron programlamayı doğrudan destekler. Pydantic ile veri doğrulama ve tip kontrolü sağlar. Otomatik API dokümantasyonu (Swagger/OpenAPI) ile geliştirici deneyimini artırır. REST API geliştirmek için idealdir ve performans odaklı projelerde avantaj sağlar.

Dezavantajları: Flask kadar köklü bir topluluğa sahip değildir ve ekosistemde eklenti sayısı sınırlıdır. Öğrenme eğrisi biraz dik olabilir; tip anotasyonları ve async mantığını anlamak gereklidir.

Kullanım Alanları

Flask, esnek ve minimal yapısı sayesinde farklı ölçeklerdeki web uygulamaları ve servisler için tercih edilen bir framework’tür. Küçük ve orta ölçekli projelerde hızlı prototip geliştirme, API servisleri oluşturma ve modüler yapıda uygulamalar geliştirme için idealdir. Aynı zamanda mikroservis tabanlı mimarilerde veya veri görselleştirme araçlarında da etkin bir şekilde kullanılabilir. Flask, geliştiricilere yalnızca ihtiyaç duydukları bileşenleri ekleyerek projeyi özelleştirme imkânı verir ve böylece hem geliştirme süreci hızlanır hem de bakım ve genişletme işlemleri kolaylaşır. Bu özellikleri sayesinde Flask, araştırma projeleri, MVP (Minimum Viable Product) geliştirme süreçleri ve profesyonel uygulamalar için güvenle kullanılabilir.

Günlük Hayat Örnekleri:

1. Netflix
Netflix, mikroservis tabanlı altyapısında bazı dahili araçları ve prototip servisleri geliştirmek için Flask kullanıyor. Flask’ın minimal ve esnek yapısı, Netflix’in hızlı prototip geliştirme ve küçük hizmetler oluşturma ihtiyaçları için uygun.

2. Lyft
Lyft, araç paylaşım uygulamasının bazı API servislerinde Flask kullanıyor. Özellikle yüksek performans gerektirmeyen mikroservisler ve dahili araçlar için Flask tercih edilmiş.

3. Airbnb
Airbnb, dahili yönetim araçları ve veri görselleştirme servislerinde Flask kullanıyor. Flask, esnekliği sayesinde veri ekiplerinin hızlı bir şekilde web tabanlı dashboard’lar ve yönetim panelleri oluşturmasına olanak sağlıyor.

Flask Bu Uygulamalarda Neden Tercih Edilir?

Flask, minimal tasarımı sayesinde hızlı geliştirme yapılmasına olanak tanır ve özellikle mikroservis mimarisine oldukça uyumludur. Python'un zengin veri işleme kütüphaneleriyle kolayca entegre olabilmesi, veri odaklı uygulamalar geliştirmeyi pratik hâle getirir. Ayrıca Jinja2 template motoru sayesinde dinamik ve hızlı bir şekilde arayüz oluşturmak mümkündür. Genel olarak öğrenmesi kolay bir yapıya sahip olduğundan, hem başlangıç seviyesindeki geliştiriciler hem de profesyoneller için düşük bakım maliyetiyle verimli bir geliştirme süreci sunar.

Sonuç

Flask; sade, esnek ve geliştirici dostu yapısı sayesinde modern web uygulamalarının geliştirilmesinde güçlü bir çözüm sunar. Minimal tasarımı sayesinde yalnızca ihtiyaç duyulan bileşenleri kullanmayı mümkün kılarak hem performans hem de kontrol açısından avantaj sağlar. Mikroservis mimarisine uygunluğu, geniş Python ekosistemiyle uyumu ve öğrenmesi kolay yapısı, Flask'ı hem öğrenciler hem de profesyonel geliştiriciler için tercih edilir hâle getirir. Gerçek dünya projelerinde de yaygın olarak kullanılmasının nedeni, sağladığı hız, özgürlük ve genişletilebilirliktir. Bu yönleriyle Flask, küçük projelerden büyük ölçekli sistemlere kadar farklı boyutlardaki yazılım ihtiyaçlarını karşılayabilen etkili bir web framework'üdür.

KAYNAKÇA

https://flask.palletsprojects.com/

https://learn.microsoft.com/tr-tr/visualstudio/python/learn-flask-visual-studio-step-01-project-solution?view=visualstudio

https://onlyjs.com/yazilim-sozlugu/flask

https://medium.com/kodlayan-nesil/flask-nedir-9364c1bb5f41

https://coderspace.io/sozluk/flask-nedir/

https://jinja.palletsprojects.com/

https://realpython.com/

https://www.w3schools.com/python/python_flask.asp

https://www.geeksforgeeks.org/

https://opensource.org/licenses/BSD-3-Clause