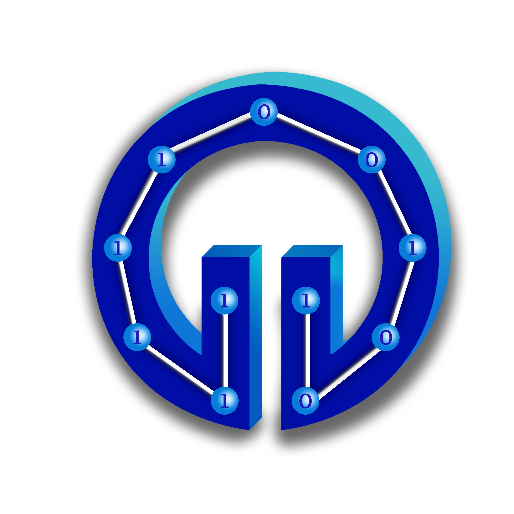
**KARADENİZ TEKNİK ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

****

**PROJENİN KONUSU**

**BİTİRME PROJESİ**

**Adı SOYADI**

**2020-2021 GÜZ DÖNEMİ**

**KARADENİZ TEKNİK ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**DOCKER İÇİN YARDIMCI ARAYÜZ UYGULAMASI**

**BİTİRME PROJESİ**

**Sevilay ERKAN**

**2022-2023 GÜZ DÖNEMİ**

|  |  |  |
| --- | --- | --- |
| ieee.jpg | **IEEE Etik Kuralları**  **IEEE Code of Ethics** | **ieee.jpg** |

Mesleğime karşı şahsi sorumluluğumu kabul ederek, hizmet ettiğim toplumlara ve üyelerine en yüksek etik ve mesleki davranışta bulunmaya söz verdiğimi ve aşağıdaki etik kurallarını kabul ettiğimi ifade ederim:

1. Kamu güvenliği, sağlığı ve refahı ile uyumlu kararlar vermenin sorumluluğunu kabul etmek ve kamu veya çevreyi tehdit edebilecek faktörleri derhal açıklamak;
2. Mümkün olabilecek çıkar çatışması, ister gerçekten var olması isterse sadece algı olması, durumlarından kaçınmak. Çıkar çatışması olması durumunda, etkilenen taraflara durumu bildirmek;
3. Mevcut verilere dayalı tahminlerde ve fikir beyan etmelerde gerçekçi ve dürüst olmak;
4. Her türlü rüşveti reddetmek;
5. Mütenasip uygulamalarını ve muhtemel sonuçlarını gözeterek teknoloji anlayışını geliştirmek;
6. Teknik yeterliliklerimizi sürdürmek ve geliştirmek, yeterli eğitim veya tecrübe olması veya işin zorluk sınırları ifade edilmesi durumunda ancak başkaları için teknolojik sorumlulukları üstlenmek;
7. Teknik bir çalışma hakkında yansız bir eleştiri için uğraşmak, eleştiriyi kabul etmek ve eleştiriyi yapmak; hatları kabul etmek ve düzeltmek; diğer katkı sunanların emeklerini ifade etmek;
8. Bütün kişilere adilane davranmak; ırk, din, cinsiyet, yaş, milliyet, cinsi tercih, cinsiyet kimliği, veya cinsiyet ifadesi üzerinden ayırımcılık yapma durumuna girişmemek;
9. Yanlış veya kötü amaçlı eylemler sonucu kimsenin yaralanması, mülklerinin zarar görmesi, itibarlarının veya istihdamlarının zedelenmesi durumlarının oluşmasından kaçınmak;
10. Meslektaşlara ve yardımcı personele mesleki gelişimlerinde yardımcı olmak ve onları desteklemek.

IEEE Yönetim Kurulu tarafından Ağustos 1990’da onaylanmıştır.

**ÖNSÖZ**

Yazılım geliştiricilerin Docker kullanımına yardımcı olmayı hedefleyen bu proje, Karadeniz Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü’nde bahar dönemi bitirme projesi olarak hazırlanmıştır.

Öncelikle tez danışmanım Doç. Dr. Hüseyin Pehlivan’a saygılarımı ve teşekkürlerimi sunarım. Konu seçiminde yardımcı olup Docker ve DevOps konularında tecrübe kazanmamı sağlayan Netaş’ta yaptığım stajdaki ekip şefim Gürcan Pehlevan’a teşekkürü borç bilirim.  
  
Projenin yapımı sürecinde manevi destekleri için kuzenlerim Ceyda ve Yahya Erdem Bora’ya, Kamp.us komünitesine ve bu süre zarfında sürekli yanımda olup destek oldukları için Twitch’deki takipçilerime çok teşekkür ederim.

Sevilay Erkan

Trabzon 2023

**İÇİNDEKİLER**

|  |  |
| --- | --- |
|  | Sayfa No |
| IEEE ETİK KURALLARI...................................................................................... | II |
| ÖNSÖZ.................................................................................................................... | III |
| İÇİNDEKİLER........................................................................................................ | IV |
| ÖZET....................................................................................................................... | V |
| 1. GENEL BİLGİLER............................................................................................. | 1 |
| * 1. Giriş.................................................................................................................. | 2 |
| * 1. .......................................................................................................................... | 3 |
| * 1. .......................................................................................................................... | 4 |
| 2. YAPILAN ÇALIŞMALAR................................................................................ | 5 |
| 2.1. .......................................................................................................................... | 6 |
| 2.2. .......................................................................................................................... | 7 |
| 2.3. .......................................................................................................................... | 8 |
| 3. SONUÇLAR....................................................................................................... | 9 |
| 4. ÖNERİLER......................................................................................................... | 10 |
| 5. KAYNAKLAR.................................................................................................... | 11 |
| 6. EKLER (varsa).................................................................................................... | 12 |
| STANDARTLAR ve KISITLAR FORMU............................................................ | 13 |

**ÖZET**

Bu projenin amacı bir konteyner teknolojisi olan Docker’ı yeni başlayan veya konteyner teknolojileri hakkında bilgi sahibi olmayan yazılım geliştiricilerin bu teknolojiyi projelerinde hızlı ve kolay bir kullanabilmesine olanak verebilmektir.

Proje genel olarak iki kısımdan oluşmaktadır:

1. Temel Docker komutlarının kullanımına yardımcı olacak terminal ve web uygulaması
2. Kullanıcının istediği herhangi bir proje için Docker kullanılabilmesini sağlayacak scripti oluşturmasına yardımcı olacak “Docker Compose Oluşturucu” terminal ve web uygulaması

Yapılan proje tüm Docker komutlarını içermese de bir yazılım geliştiricinin temel Docker konteyner ihtiyacını komut kullanmadan karşılamasına olanak verecek yapıda Python ve Flask web frameworkü kullanılarak tasarlanmıştır. Bu tezde de kullanılan teknolojiler anlatılarak, programın akışı ve yapım süreci açıklanmaya çalışılmıştır.

**1. GENEL BİLGİLER**

* 1. **Giriş**

Projenin Amacı: Günümüzde yazılım geliştirmede Docker gibi konteyner çözümlerinin aktif olarak kullanıldığını ve işleri kolaylaştırdığını gözlemlemekteyiz. Bu konteyner çözümleri bu teknolojiyi ilk defa kullanmak isteyen yazılımcılar/bilişimciler için ciddi bir öğrenme eğrisi gerektirmesiyle sıkıntı yaşatabilmektedir. Bu nedenle bir arayüz üzerinden kullanıcıların Docker komutlarını terminal veya web arayüzü üzerinde komutları ezberlemek zorunda olmadan çalıştırabilecekleri bir uygulama oluşturulması amaçlanmaktadır.

Projenin Kapsamı: Projede bir ara yüz üzerinden kullanıcıların Docker komutlarını terminal veya web arayüzü üzerinden çalıştırabilecekleri bir uygulama oluşturulması projenin kapsamı dâhilindedir.

Bu amaç ve kapsamı gerçekleştirmek için bir geliştiricinin çalışma koşullarında ihtiyacı olan Docker komutları belirlenmiş ve bu komutlar Python dili ve bu dildeki Flask web altyapısı kullanılarak son kullanıcının kolayca kullanabileceği bir yapı oluşturulmaya çalışılmıştır. Tasarımı zenginleştirmek amacıyla Docker Compose dosyası oluşturucu da sonradan projeye dahil edilmiştir.

Proje genel olarak iki kısımdan oluşmaktadır:

1.Temel Docker komutlarının kullanımına yardımcı olacak terminal ve web uygulaması

2.Kullanıcının istediği herhangi bir proje için Docker kullanılabilmesini sağlayacak scripti oluşturmasına yardımcı olacak “Docker Compose Oluşturucu” terminal ve web uygulaması

**1.2 Konteyner Teknolojileri**

**1.2.1 Docker**

**1.3 Geliştirme Ortamı**

Geliştirme ortamı için kurulumlara başlamadan şu an üstlendiğim proje ve kullanabileceğimiz teknolojilere göre bir planlama yaptım. Bu planlamada DevOps süreçlerini düşündüm ve buna göre bir liste çıkardım. Bu listeye yazılım geliştirme ve deployment için çeşitli programları da eklediğimde listemin son hali şu şekilde oldu.

IDE/Kod Editörü: VS Code, Notepad ++

Versiyon kontrolü: Git

Yazılım Geliştirme: Python

İşletim Sistemi Ortamları: WSL, Ubuntu, Debian

Diğer: Docker, Docker Desktop

**1.3.1 VS Code**

Visual Studio Code, Microsoft tarafından açık kaynak olarak geliştirilen hafif ve güçlü bir kod editörüdür. Kullanımı kolay ve anlaşılır bir arayüze sahiptir. Farklı programlama dillerine destek verir ve geliştirme sürecini hızlandırmak için birçok özelliği vardır.

VS Code'un sağladığı avantajlar arasında hızlı ve hafif olması yer almaktadır. Kısa başlatma süresi sayesinde projelerinizi hızlı bir şekilde açabilir ve düzenleyebilirsiniz. Ayrıca, çapraz platform desteği sayesinde farklı işletim sistemlerinde sorunsuz bir şekilde çalışır.

VS Code, geniş bir eklenti ekosistemine sahiptir. Bu eklentileri kullanarak kod düzenleme, hata ayıklama, otomatik tamamlama gibi işlevleri özelleştirebilir ve kendi çalışma ortamınızı oluşturabilirsiniz.

Kullanıcılar, VS Code'un zengin özellik setinden de faydalanabilirler. Bu özellikler sayesinde kod tamamlama, otomatik biçimlendirme, kod analizi gibi geliştirme süreçlerini daha verimli hale getirebilirsiniz.

VS Code'un önemli özellikleri şunlardır:

**Hızlı ve Hafif:** VS Code, düşük kaynak tüketimi sayesinde hızlı ve hafif bir performans sunar. Projelerinizi hızlı bir şekilde açabilir ve düzenleyebilirsiniz.

**Çapraz Platform Desteği:** Windows, macOS ve Linux'ta sorunsuz bir şekilde çalışır. Hangi işletim sistemini kullanırsanız kullanın, aynı kullanıcı deneyimini yaşayabilirsiniz.

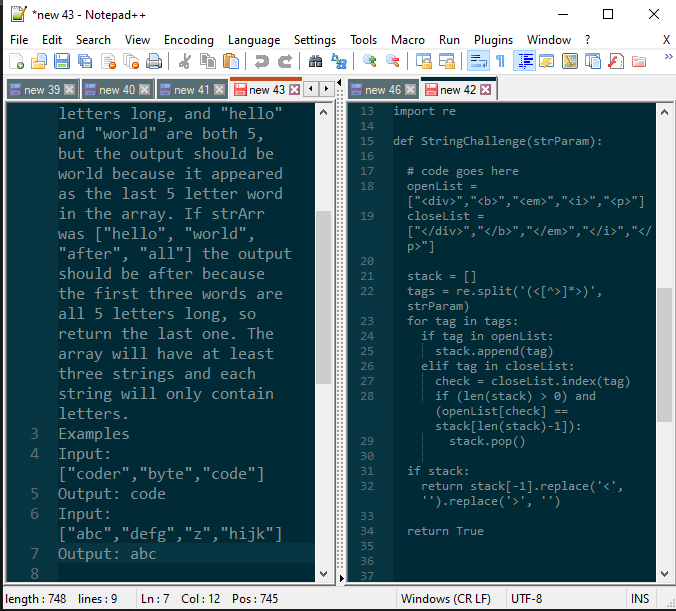
**Eklenti Desteği:** VS Code, zengin bir eklenti koleksiyonuna sahiptir. Bu eklentileri kullanarak, kod düzenlemeyi ve otomatik tamamlamayı özelleştirebilir, hata ayıklama ve daha fazlasını yapabilirsiniz.

**Gelişmiş Özellikler:** Kod tamamlama, otomatik düzeltme, hata ayıklama gibi gelişmiş özellikleri bulunur. Bu özellikler, kod yazma sürecinizi daha verimli hale getirir.

VS Code, hem yeni başlayanlar hem de deneyimli geliştiriciler için ideal bir seçenektir. Basit ve kullanıcı dostu arayüzü sayesinde kolayca öğrenilebilir ve kullanılabilir. Kod düzenleme ve proje yönetimi işlerinizi daha etkili ve keyifli hale getiren bu açık kaynak kod editörü, birçok geliştirme ortamında tercih edilen bir araçtır.

**1.3.2 Notepad++**

Not alma, dokümantasyon takibi tutma amacıyla kişisel olarak da kullanabildiğimiz bu uygulama aynı zamanda kod editörü olarak da kullanılabiliyor. 2003 yılında Don Ho tarafından Windows işletim sistemine sahip cihazlar için C++ ile geliştirilen Notepad++ bir metin ve kaynak kod editörü olarak kullanılabiliyor. Sekmeli çalışma yapmamıza olanak verir, 78’den fazla programlama dili için syntax desteği sunar, ekran bölme, satır kaydetme, metin tamamlama, plugin/macro desteği gibi çeşitli güzel özellikleri vardır. Ayrıca GNU General Public lisansıyla lisanslanmış özgür bir yazılımdır. Dünya çapında ve benim bu uygulamayı tercih etmemdeki en önemli etmen ise geliştirildiği teknolojiler sebebiyle uygulama boyutunun küçük olması ve CPU gibi sistem kaynaklarını daha az kullanması. Bu sayede bilgisayarın güç kullanımı düşüyor ve çevreye daha az karbon ayak izi bırakmış oluyoruz.



Şekil 1.3.2.1 Notepad++ Ekranı

**1.3.3 Git Versiyon Kontrol Sistemi**

Günümüzün yazılım geliştirme süreçleri, birçok geliştiricinin aynı projede bir arada çalışmasını gerektirir. Bu nedenle, bir proje üzerinde birden fazla kişi çalıştığında, tüm değişiklikleri izlemek ve kontrol etmek önemli hale gelir. İşte tam burada sürüm kontrol sistemleri devreye girer.

Versiyon kontrol sistemleri, projelerde yapılan değişiklikleri izlemek, paylaşmak ve yönetmek için kullanılan yazılımlardır. Merkezi ve dağıtık olmak üzere iki tür versiyon kontrol sistemi vardır. Merkezi versiyon kontrol sistemlerinde, tüm veriler merkezi bir sunucuda saklanırken, dağıtık git versiyon kontrol sistemlerinde, her kullanıcının kendi kopyası vardır ve herhangi bir değişiklik yapılabilir.

Git, 2005 yılında Linus Torvalds tarafından Linux çekirdeği geliştirme sürecinde kullanılmak üzere oluşturulmuş bir sürüm kontrol sistemi olarak ortaya çıkmıştır. Git'in ortaya çıkışı, çok sayıda Linux çekirdeği geliştiricisinin proje yönetimi için bir önceki sürüm kontrol sistemi olan BitKeeper'ı tercih etmesiyle başlamıştır. Andrew Tridgell, bir takım tersine-mühendislik yöntemleriyle BitKeeper protokolüne müdahalelerde bulunmuş, ancak BitKeeper'ın telif haklarını elinde bulunduran Larry McVoy, BitKeeper'ın ücretsiz kullanımını reddederek konuyu hukuki platforma taşıyınca BitKeeper'ın kullanımından vazgeçilmiş, böylece Git'in temelleri atılmıştır.

Git, bir sürüm kontrol sistemi olarak birçok avantaja sahiptir. Bu avantajlar şunlardır:

* Kolay işbirliği ve kod paylaşımı
* Değişiklikleri izleme ve önceki sürümlere geri dönme yeteneği
* Paralel geliştirme için dallanma ve birleştirme yetenekleri
* Çevrimdışı çalışma için dağıtık mimari
* İşbirliği ve Ekip Çalışması

Git, geliştiricilerin kod tabanı üzerinde işbirliği yapmasını kolaylaştırır. Her geliştiricinin kendi kod tabanı kopyasının olması nedeniyle, farklı geliştiriciler aynı anda kodun farklı parçalarında çalışabilirler ve birbirleriyle etkileşime girmezler. Git, farklı geliştiriciler tarafından yapılan değişiklikleri gözden geçirme ve birleştirme araçları da sağlar.

Git'in branch ve merge yetenekleri, kodun farklı sürümleri üzerinde aynı anda çalışmayı kolaylaştırır. Geliştiriciler, özellikler veya düzeltmeler üzerinde çalışmak için yeni branchler oluşturabilir ve bu değişiklikleri hazır olduklarında ana kod tabanına mergeleyebilirler.

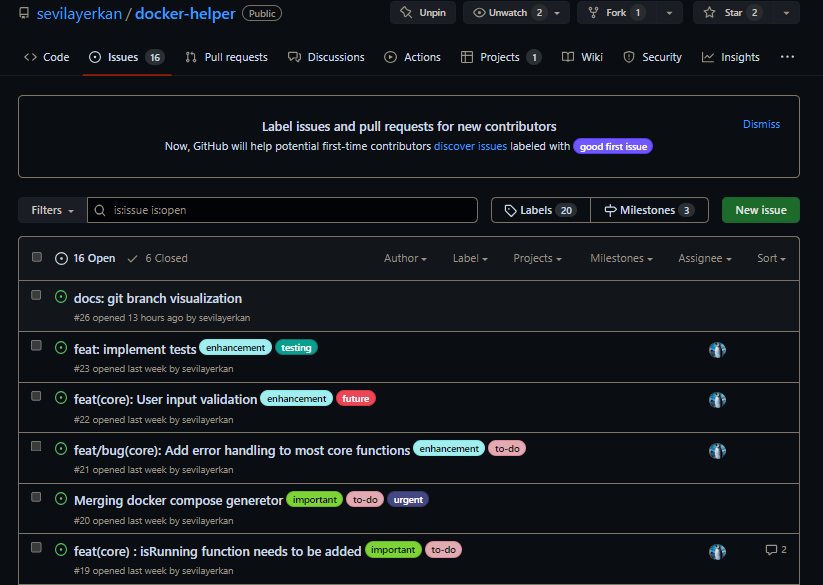
Gitte her bir değişiklik, "commit" olarak adlandırılan bir işlemle takip edilir. Bu sayede, herhangi bir zamanda, bir önceki sürüme geri dönüp hataları düzeltmek veya daha önce yapılan bir değişikliği geri getirmek mümkündür.

Git kullanmanın en temel yollarından biri, öncelikle bir Git deposu oluşturmaktır. Bunun için şu komutu kullanabilirsiniz: git init

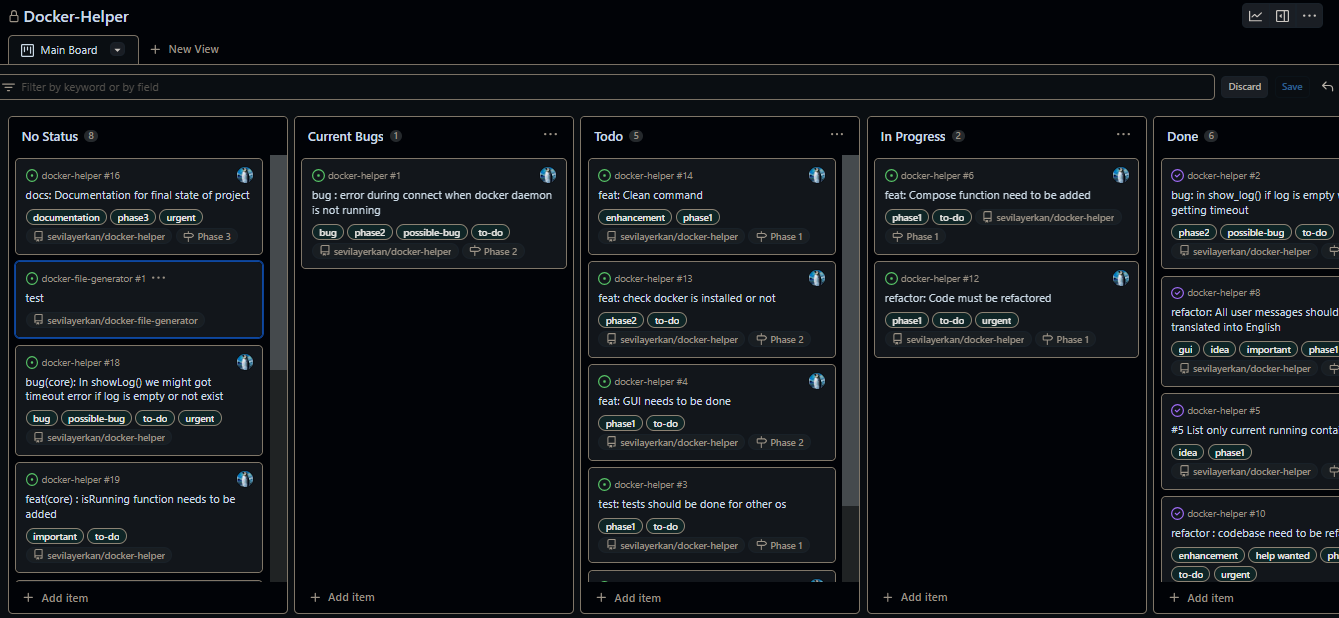
Depoyu oluşturduktan sonra, Git'te birçok komut kullanılabilir. Bazı temel komutlar şunlardır:

* Dosyaları ekleme: git add dosya adı
* Veya tüm dosyaları eklemek için git add -A
* Değişiklikleri kaydetme: git commit -m "açıklama"
* Depo durumunu kontrol etme: git status
* Değişiklik geçmişini görüntüleme: git log
* Değişikleri göndermek için uzak sunucu eklemek için: git remote add uzaksunucuismi uzaksunucuadresi
* Değişikleri uzak git sunucusuna yollamak için: git push

Bu anlatılan teknolojiyi kullanmak için ilgili uygulamaları indirdim ve git sunucusuna bağlantı yapabilmek için GitHub hesabımdan yeni SSH anahtarı oluşturup git shell i üzerinde tanımladım. Ve VS Code üzerinde gerekli clientleri kullanarak da GitHub hesabımı yazılım geliştirme ortamıma bağlamış oldum. Bu kurulumunu sağladığım GitHub reposunu hata/özellik takibi için ve kanban tablosu olarak da kullandım.



Şekil 1.3.3.1 GitHub’ın Özellik ve Hata Takibinde Kullanımı



Şekil 1.3.3.2 GitHub’ın Proje Yönetim Sürecinde Kullanımı

* + 1. **WSL**

Çalışmak için Linux bir makineye/shelle gereksinimim olabileceğinden Windows’un Windows Subsystem for Linux olarak bilinen Linux komutlarının çalıştırılmasını sağlayan uyumluluk katmanını Ubuntu işletim sistemi kullanarak denedim.

Windows Subsystem for Linux (WSL), geliştiricilere Windows işletim sisteminde Linux tabanlı uygulamaları sorunsuz bir şekilde çalıştırma olanağı sunan önemli bir özelliktir. Microsoft'un Windows 10 ve sonraki sürümlerinde sağladığı bu teknoloji, Linux ekosistemindeki araçlardan ve geliştirme deneyiminden yararlanmak isteyen kullanıcılar için pratik bir çözüm sunar.

WSL, Windows ve Linux arasında bir köprü olarak görev yapar. Kullanıcılar, Windows Mağazası aracılığıyla tercih ettikleri Linux dağıtımını (Ubuntu, Debian, Fedora, vb.) kolayca indirip kurabilirler. Ardından, Linux dağıtımını başlattıklarında, Windows işletim sistemi içerisinde çalışan bir Linux çekirdeğine erişim sağlayarak Linux komut satırını kullanabilirler.

Bu teknolojinin getirdiği önemli avantajlar bulunmaktadır:

**Kolay Kurulum:** WSL, Windows Mağazası üzerinden basit ve hızlı bir şekilde kurulabilir. Kullanıcılar, karmaşık ayarlamalara ve ek bir sanal makine oluşturmaya gerek duymadan Linux dağıtımını çalıştırabilirler.

**Linux ve Windows Uyumlu Ortam:** WSL, Linux tabanlı geliştirme araçlarını ve komut satırı programlarını Windows ortamında sorunsuzca çalıştırma olanağı sunar. Böylece, geliştiriciler Linux ekosisteminden yararlanırken Windows'un kullanıcı dostu arayüzünden de faydalanabilirler.

**Dosya Paylaşımı:** WSL, Windows dosya sistemi ile entegre bir şekilde çalışır. Bu sayede, Windows dosyaları WSL içinde doğrudan erişilebilir ve Linux komutları ile işlenebilir. Aynı şekilde, Linux dosyaları da Windows uygulamalarıyla düzenlenebilir.

**Geliştirme Araçları:** WSL, Python, Node.js, Ruby gibi popüler programlama dilleri ve Git, Docker gibi geliştirme araçlarının Windows üzerinde çalışmasını sağlar.

WSL, hem Windows hem de Linux dünyasından en iyi özellikleri birleştirerek geliştiricilere daha esnek bir geliştirme deneyimi sunar. Geliştiriciler, Windows platformunda Linux uygulamalarını geliştirebilir, test edebilir ve dağıtabilirler. Bu özellik, Linux'a aşina olan kullanıcıların Windows üzerinde daha rahat ve üretken bir ortamda çalışmasını mümkün kılar.

**1.4 Python**

Python, son yıllarda büyük bir popülerlik kazanmış ve geniş bir kullanıcı tabanına sahip olan yüksek seviyeli bir programlama dilidir. Kullanıcıların göreceli olarak kolay öğrenebileceği ve anlayabileceği bir sözdizim yapısı sunarak, programcılara etkili bir geliştirme deneyimi sunmaktadır. Python, geniş kütüphane desteği ve taşınabilirliği sayesinde çeşitli alanlarda kullanılan birçok uygulama ve projenin geliştirilmesinde tercih edilen bir araç haline gelmiştir.

Python'un bazı önemli özellikleri şöyledir:

**1. Kullanıcı Dostu ve Okunabilir Sözdizim:** Python, kullanıcı dostu ve anlaşılır bir sözdizim yapısına sahiptir. Bu dilin tasarımı, kodun okunabilirliğini ve anlaşılırlığını ön planda tutmaktadır. Doğal dil yapılarına benzerlik gösteren bu sözdizim, programcılara karmaşıklığı azaltarak daha hızlı ve verimli kod yazma olanağı sağlar.

**2. Yüksek Seviyelilik ve Taşınabilirlik:** Python, yüksek seviyeli bir programlama dili olarak kabul edilir. Bu, programcıların daha genel bir perspektifte programlama yapabilmelerini ve karmaşık detaylara fazla odaklanmamalarını sağlar. Ayrıca Python platform bağımsız bir dildir, yani farklı işletim sistemlerinde (Windows, Linux vb.) tutarlı bir şekilde çalışabilir. Bu taşınabilirlik özelliği, Python'un yaygın olarak kullanılmasını sağlayan faktörlerden biridir.

**3. Geniş Kütüphane Desteği:** Python, zengin bir kütüphane ve paket ekosistemine sahiptir. Bu kütüphaneler, çeşitli alanlarda (veri analizi, yapay zeka, web geliştirme, bilimsel hesaplama vb.) hazır fonksiyonlar ve araçlar sunar. Örneğin, NumPy, Pandas ve Matplotlib gibi kütüphaneler, veri analizi ve görselleştirme gibi yaygın görevleri kolaylaştıran araçlardır. Bu kütüphaneler, geliştiricilere daha hızlı ve verimli bir şekilde projelerini geliştirme olanağı sunar.

**4. Çok Amaçlı ve Geniş Uygulama Alanı:** Python, çok yönlü bir programlama dilidir ve birçok farklı uygulama alanında kullanılabilir. Veri bilimi, makine öğrenmesi, yapay zeka, web geliştirme, oyun geliştirme ve otomasyon gibi birçok alanda Python tercih edilen bir dil haline gelmiştir. Python'un esnekliği, geliştiricilere çeşitli projelerde ve sektörlerde kullanabilecekleri güçlü bir araç sunar.

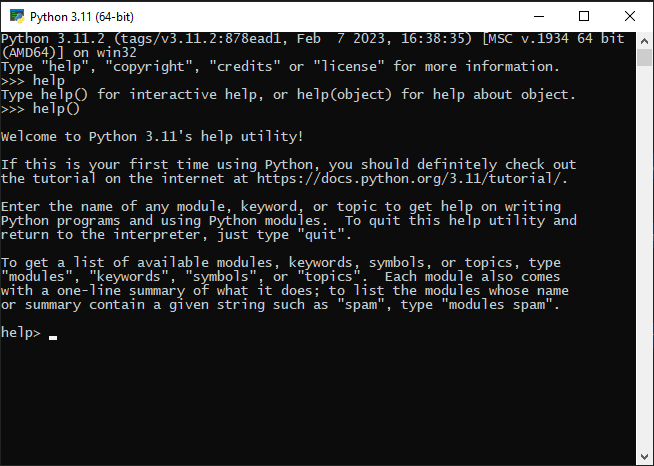
Python, kullanıcı dostu sözdizimi, geniş kütüphane desteği ve taşınabilirliği ile dikkat çeken güçlü bir programlama dilidir. Geliştiricilerin etkili bir şekilde kod yazmalarına ve projelerini hızlı bir şekilde geliştirmelerine olanak tanır. Bu nedenle, Python, programlama dünyasında geniş bir kabul ve kullanım alanına sahiptir.

Açık kaynaklı bir programlama dili olarak, Python'ın kaynak kodu herkese açıktır ve kullanıcılar tarafından değiştirilebilir ve geliştirilebilir. Bu, Python topluluğunun büyümesine ve dilin sürekli olarak iyileştirilmesine olanak tanır.

Python'ın zengin bir dokümantasyonu vardır. Python resmi belgeleri, kullanıcıların dilin her yönünü anlamalarına ve kullanmalarına yardımcı olacak kapsamlı bir kaynaktır. Bu dokümantasyon, dilin temellerinden başlayarak ileri düzey konulara ve modüllere kadar geniş bir yelpazede bilgi sunar. Python kütüphanelerinin dokümantasyonları da genellikle ayrıntılı ve anlaşılır şekilde hazırlanmıştır.

Python programlama dilini bilgisayarınıza kurmak için aşağıdaki adımları izleyebilirsiniz:

* Python'un resmi web sitesine (https://www.python.org/downloads/) gidin.
* İndirme sayfasından işletim sisteminize uygun olan Python sürümünü seçin.
* Yükleyiciyi indirin ve çalıştırın.
* Kurulum sırasında gerekli ayarları seçin veya varsayılanları kullanın.
* Kurulumun tamamlanmasını bekleyin.
* Python'un doğru şekilde kurulduğunu test etmek için bir terminal veya komut istemi açın ve "python" komutunu çalıştırın.



Şekil 1.4.1 Python Shell Ekranı

**1.4.1 Flask**

Flask, Python programlama diliyle web projeleri geliştirmek için kullanılan bir microframework'tür. 2004 yılında kurulan uluslararası bir Python meraklıları grubu olan Pocoo'dan Armin Ronacher tarafından yaratılan Flask 1 Nisan şakası olarak ortaya çıkmıştır ve Werkzeug ve Jinja 2 tabanlıdır. Belirli araçlar veya kütüphaneler gerektirmediği için bir mikro çerçeve olarak sınıflandırılır. Microframework (mikro çerçeve) temel özellikleri içeren ancak ihtiyaca göre çeşitli kütüphanelerle genişletilebilen bir web çatısı anlamına gelir.

Flask çerçevesini kullanan uygulamalar arasında Pinterest ve LinkedIn bulunmaktadır.

Ronacher ve Georg Brandl 2004 yılında Python ile yazılmış bir ilan tahtası sistemi oluşturduklarında, Pocoo projelerinden Werkzeug ve Jinja da geliştirildi. Nisan 2016'da Pocoo ekibi dağıldı ve Flask ve ilgili kütüphanelerin geliştirilmesi yeni kurulan Pallets projesine geçti.

Flask, aşağıdaki özellikleriyle öne çıkar:

**Minimalist Tasarım:** Flask, minimal ve hafif bir yapıya sahiptir. Bu sayede gereksiz karmaşıklığı ortadan kaldırır ve sadece temel işlevleri sunar. Flask, gereksinimlerinize göre şekillendirebileceğiniz bir temel oluşturmanıza olanak tanır.

**Esneklik:** Flask, geliştiricilere büyük bir esneklik sağlar. Uygulamanızın gereksinimlerine uyacak şekilde çatıyı özelleştirebilirsiniz. Flask, herhangi bir ORM (Object-Relational Mapping) veya veritabanı motoruna bağımlı değildir, bu da geliştiricilere daha fazla kontrol sağlar.

**Taslak (Blueprint) Desteği:** Flask, taslaklar aracılığıyla uygulamalarınızı modüler hale getirir. Taslaklar, farklı işlevlere veya bileşenlere ait kodu gruplamak için kullanılır. Bu sayede büyük ölçekli projeleri daha iyi organize edebilir ve bakımı kolaylaştırabilirsiniz.

**Geniş Ekosistem:** Flask, geniş bir ekosisteme sahiptir. Birçok eklenti, kütüphane ve araç tarafından desteklenir. Flask eklentileri, oturum yönetimi, form işleme, veritabanı entegrasyonu ve daha fazlası gibi işlevleri kolayca entegre etmenizi sağlar. Bu da geliştirme sürecini hızlandırır ve verimliliği artırır.

1. **Yapılan Çalışmalar**
   1. **Temel Docker Komutlarının Gerçeklenmesi**
   2. **Docker Compose Dosyası Oluşturucu**
2. **Sonuçlar**

Proje planında “Docker İçin Arayüz Uygulaması” için belirlediğim,

**Minimum Hedef** olan “Kullanıcıların Docker komutlarını bir arayüz üzerinden kullanıp yönetebilmesine imkan sağlanması.” ;

**Belirlenen Hedef** olan “Projenin amacı ve gereksinim analizindeki problem tanımında anlatılanların doğru bir şekilde çözülerek gerçeklenmesi. Kullanıcıların bir arayüz ve web arayüzü üzerinden Docker ile ilgili işlemlerini yapmalarına imkan vermek.” hedeflerini gerçekleştirmiş oldum.

Projenin planlanmasında yaptığım ekte görebileceğiniz akış şemasındaki ilk fazlara ait çoğu aşamayı gerçekledim. Bunun yanında projeye sonradan dahil ettiğim Docker Compose dosyası oluşturma özelliğiyle de planlarımın üstünde bir çalışma yapmış oldum. Bu proje sonucunda Python, Docker, DevOps konularında büyük bir tecrübe kazanmış oldum. En yakın zamanda fırsat yaratıp geliştirmeye opensource olarak devam edip, bu projenin bir kısmını bir cloud altyapısı üzerinden son kullanıcıya ulaştırma niyetindeyim.

1. **Öneriler**

Hemen hemen her projede olduğu gibi burada da sistem daha da geliştirilebilir. Ekteki akış şemalarında görülebilecek bazı fonksiyonlar eklenmemiştir bunlar eklenerek ve kullanıcılara erişilebilir göze hitap eden bir tasarım sunarak proje geliştirilebilir. Bunun yanında “Docker Compose Üretici” bir bulut platformuna entegre edilerek bütün yazılım geliştiricilerin kullanımına sunulabilir. Projeye otomatize entegrasyon testleri ve unit testler de ilave edilerek geliştirme sürecinin daha sağlıklı ilerlemesi sağlanabilir.

1. **Kaynaklar**
2. <https://www.python.org/doc/essays/blurb/>, What is Python? 20.04.2023.
3. <https://docs.python.org/3/faq/general.html>, Python Official Documentation, 20.04.2023.
4. <https://en.wikipedia.org/wiki/Python_(programming_language)>
5. <https://github.com/microsoft/vscode> 20.04.2023.
6. <https://flask.palletsprojects.com/en/2.3.x/> 20.04.2023.
7. <https://medium.com/@awesome_nyn/python-ile-flask-microframework-kullanarak-nas%C4%B1l-web-projesi-olu%C5%9Fturulur-fbca456e7c71> 20.04.2023.
8. <https://peps.python.org/pep-0008/>, PEP 8, 1.07.2023.
9. <https://innovate.ieee.org/ieee-software-engineering-standards/> 1.07.2023.
10. **Ekler**

**STANDARTLAR ve KISITLAR FORMU**

Projenin hazırlanmasında uyulan standart ve kısıtlarla ilgili olarak, aşağıdaki soruları cevaplayınız.

1. Projenizin tasarım boyutu nedir? (Yeni bir proje midir? Var olan bir projenin tekrarı mıdır? Bir projenin parçası mıdır? Sizin tasarımınız proje toplamının yüzde olarak ne kadarını oluşturmaktadır?)

|  |
| --- |
| Proje tamamıyla özgün bir projedir. |

1. Projenizde bir mühendislik problemini kendiniz formüle edip, çözdünüz mü? Açıklayınız.

|  |
| --- |
| Evet çözdüm. |

1. Önceki derslerde edindiğiniz hangi bilgi ve becerileri kullandınız?

|  |
| --- |
| “Sistem Programlama” ve “Sistem Laboratuvarı” derslerinde öğrendiğim Unix ve infra becerilerini ayrıca “Yazılım Mühendisliği” dersinde öğrendiğim proje yönetimi ve versiyon kontrolü konularını bu projenin yapımında kullandım. |

1. Kullandığınız veya dikkate aldığınız mühendislik standartları nelerdir? (Proje konunuzla ilgili olarak kullandığınız ve kullanılması gereken standartları burada kod ve isimleri ile sıralayınız).

|  |
| --- |
| IEEE Software Engineering Standards, IEEE/EIA 12207, Standard for Information  Technology – Software Life Cycle Processes, PEP 8 tyle Guide for Python Code |

1. Kullandığınız veya dikkate aldığınız gerçekçi kısıtlar nelerdir? Lütfen boşlukları uygun yanıtlarla doldurunuz.

a) Ekonomi

|  |
| --- |
| Uygulama geliştirilirken kullanılacak platform ve araçlar açık kaynak ve ücretsiz araçlardan tercih edilmiştir. |

b) Çevre sorunları:

|  |
| --- |
| Projede herhangi bir çevre sorunu oluşturacak bir araç kullanılmamıştır. Güç tüketimi konusunda tasarruflu araçlar tercih edilmiştir. |

c) Sürdürülebilirlik:

|  |
| --- |
| Yapılan proje kullanıcıların hayatını kolaylaştıracağından sürdürülebilirdir. |

d) Üretilebilirlik:

|  |
| --- |
| Projenin kullanım kolaylığı daha da artırılarak, erişilebilir bir tasarım da eklendiğinde çeşitli bulut platformlarına entegre edilerek üretimi yapılabilir. |

e) Etik:

|  |
| --- |
| Yapılan projede doğa, insan yaşamı, toplum değerleri ve yasaları etkileyecek bir sorun  bulunmamaktadır. |

f) Sağlık:

|  |
| --- |
| Yapılan proje insan sağlığı açısından herhangi bir sorun teşkil etmemektedir |

g) Güvenlik:

|  |
| --- |
| Bu projede herhangi bir güvenlik riski bulunmamaktadır. |

h) Sosyal ve politik sorunlar:

|  |
| --- |
| Bu projede herhangi bir sosyal ve politik sorun bulunmamaktadır. |