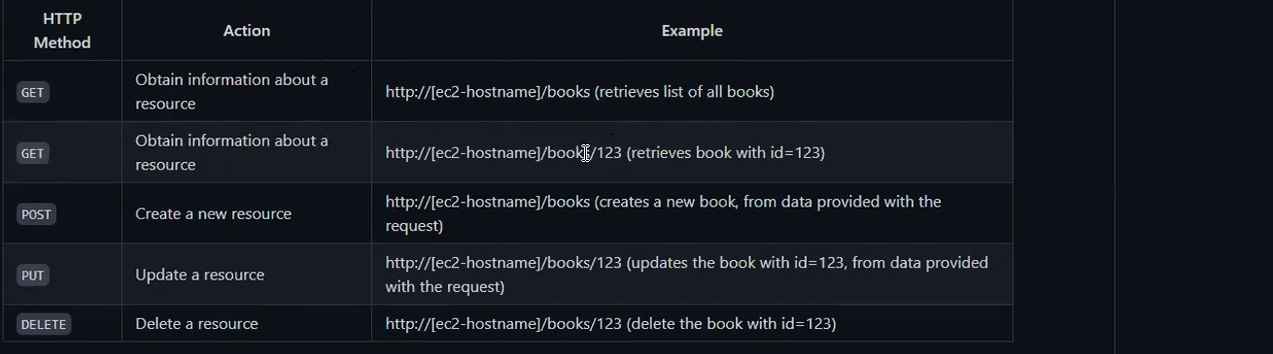
**27.03.2023**

**dockerization-bookstore-api-on-python-flask-mysql-project**

* Api, kullanıcıya veri girme ve alma imkanı tanıyor. Kitap uygulamasındaki kitap listesini kırtasiye uygulaması kullanmak istiyor listeyi yani veriyi api ile çekebiliyor.
* Json formatında bir sayfa göreceğiz.
* İki cont. ayağa kaldıracağız biri uyg. kısmı (web kısmı) diğeri de datayı tutan database kısmı.
* Docker file yazacağız ve cont. ayağa kadıracak docker compose yazacağız. Bu işlemi yapabilmek için instance docker yüklü bir terraform yazmamız gerekiyor. Makine ayağa kalktığında docker file ile imaj oluşacak ve docker compose ile iki kont dan uyg lamayı almaya çalışacağız. Terraform ile her şeyi otomatize edeceğiz. Uygulamayıda gihubdan alacağız bunuda terraformda yazacağız.



http protokolünü kullanarak Uygulamada veriye erişim sağlayabiliyoruz. GET metodu ile kitapları listeleyebilir ya da ıd numarasını yazarak istediğimiz kitabı getirebiliriz. POST metodu ile uyg. ya kitap ekleyeblirsin. PUT metodu ile update edebilir veriyi güncelleyebilirsin delete ile silebilirsin bunları CURL komutu ile yapabilirsin.

Site oluştuğunda Browser da ip yanına /books eklemeyi unutmayalım!

* Bize terraform yüklü bir yer gerekli lokalde çalışabiliriz ya da ec2 ya terraform yükleyebiliriz.

**TERRAFORM İNSTALL KOMUTLARI**

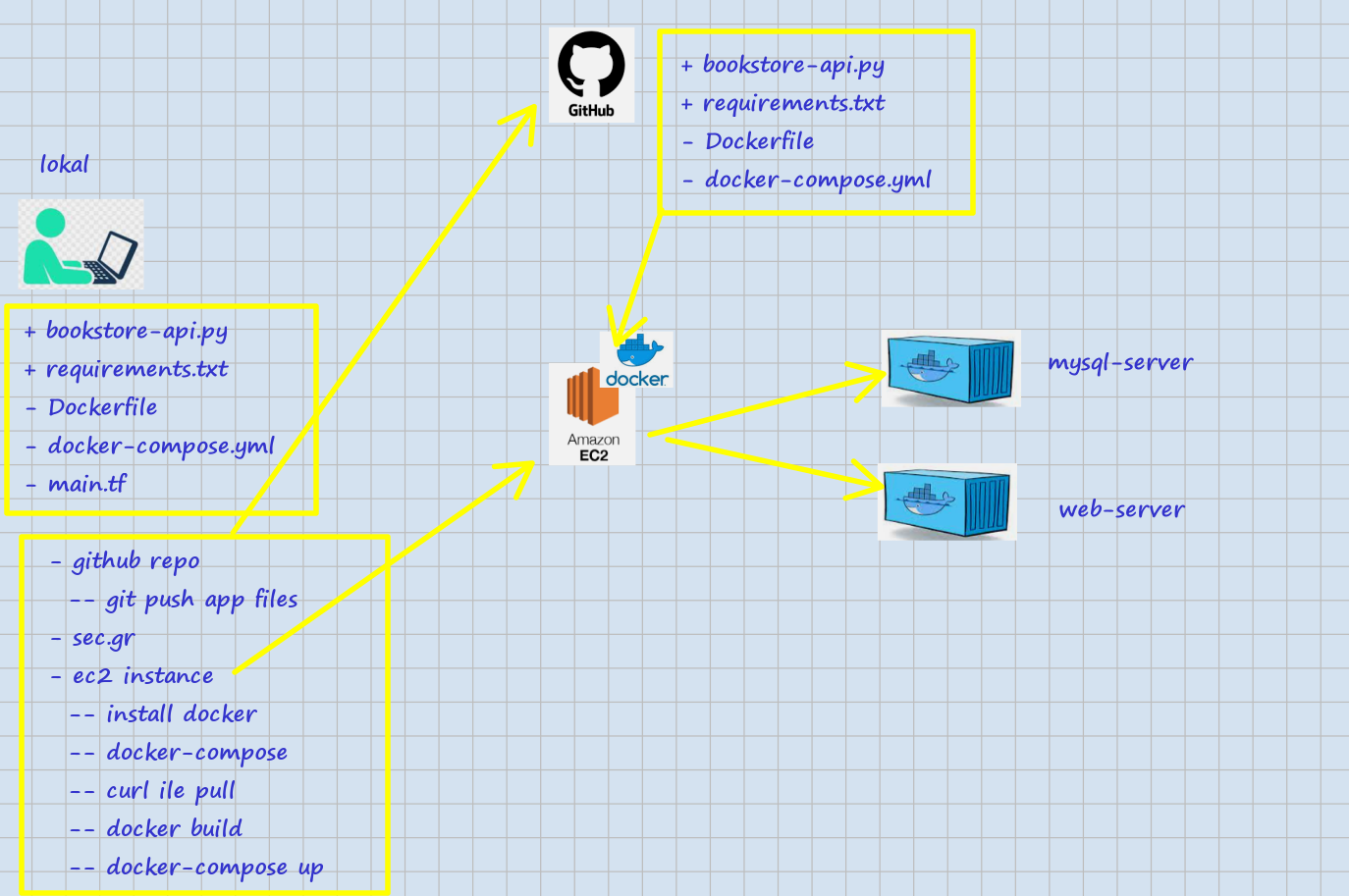
**sudo yum update -y**

**sudo yum install -y yum-utils**

**sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo**

**sudo yum -y install terraform**

**terraform --version**



* Çalışmada bize gerekli olan bazı dosyalar var uygulamanın kendi dosyası bookstore-api.py ve requirements.txt bunları bize devaloper lar veriyor.
* İlk yapmamız gereken dockerfile yazmak ardından docker-compose.yml ve son olarak main.tf yazmak.
* Main.tf içeriğinde:

- github repo oluşturacağız sonra repoya resourceları gönderme işlemleri yapacağız (git push).

- Sec grup oluşturacağız.

- ec2 instance oluşturacağız, içeriğinde install docker, docker-compose olacak ve instance a githuba pushladığımız dosyaları curl ile çekeceğiz (pull). DOCKER BUİLd ve docker-compose up diyeceğiz. Ve uygulamanın gelmesini bekleyeceğiz. Sonuç olarak instance içerisinde mysql ve web server (cont.) olacak. Ve githuba gönderdiğimiz dosyalar olacak.

**- Terraform yüklü ec2 da herhangi bir klasör oluşturalım, içine bookstore-api.py ve requirements.txt dosyalarını kopyalayıp yapıştıralım (taşıyalım). Ardından aynı dizinde Dockerfile , docker-compose.yml ve main.tf dosyalarını oluşturalım (terminalde touch ile ya da vsc de direkt oluşturabilirsin).**

Bookstore-api.py inceleyecek olursak burada dikkat etmemiz gereken bazı hususlar var çünkü biz bu uyg çalışırdığımızda cont. oluşturacağız ve database ile bağlanacağız. DATABASE ENV variable lara dikkat etmeli ! database host user password isimlerine dikkat etmek gerek neden ? çünkü database\_host ile bulacak hangi database bağlanacağını, database\_db ile de içinde hangi database olacağını (bookstore\_db) belirliyor, hangi user bağlanacağı ve şifresinin ne olacağı burada belirliyoruz. Docker compose yazarken bu variable lere dikkat etmeli! Bu bilgileri composa girmeliyim ki web server cont. database cont. na bağlanabilsin. Bunların birbirne bağlanabilmesi için iletişim için aynı ağda olmaları gerek aynı NETWORK de olmalı biz kendimiz bir network ismi vereceğiz fakat vermesek de aynı defoult network de oluşacaklar.

**Dockerfile yazmaya başlayalım... touch Dockerfile demiştik..**

1. **İlk yapmamız gereken nedir? Uygulamaya göre imaj tercihi yapacağız. İçinde python ve pip yüklü bir imaj tercih edeceğiz flaske gerek yok. Docker Hub a gidip search kısmına python yazıyorum ve içinde python geçen imajlar geliyor. Python un official imajı var ona tıklıyorum. Kullanacağımız python alpine ona tıklayalım ve içeriğine bakalım. İmajın docker file nı inceleyelim içine RUN ile kurulu olan programları inceleyebilirsin versionunu incelyebilirsin sonunda pip –version demiş yani pip de yüklenmiş bana ne lazım flask mysql in yüklenmesi lazım bunu pip ile indireceğim..İlk zorunlu satır FROM satırı peki buna nereden bakıyorduk docker docks dan bakıyorduk browsera dockerfile refference yazıyorum örneklerini parametrelerini görebilirim komutların..**

**FROM python:alpine**

1. **Daha sonra COPY ile içine yüklemem gereken dosyaları bookstore-api.py ve requirements.txt ı, app adında bir klasör oluşturup içine kopyalayacağım (nokta bulunduğun yerdeki tüm dosyaları at demek ne varsa app e atar). (copy i aşağıda alabiliriz çok değişen kısımları en sona alabiriz.)(WORKDIR ya da COPY fark etmez hangisi önce ise klasör oluşturma işlemini gerçekleştirir).**

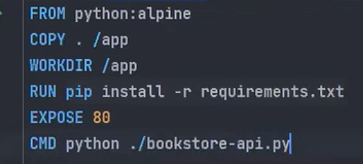
**COPY ./app**

1. **Ardından WORKDIR ile app klasörüne geçeceğim.**

**WORKDIR /app**

1. **Daha sonra RUN ile requirements.txt içerisinde bulunan flask ve flask-mysql i pip ile indirmemiz gerekiyor. İmajın içinde pip yüklü olduğu için direkt çalıştırıyor ve imajın daha küçük olmasını sağlıyor. (Bu komut stırında ki –r requirements demek bunuda pip in kendi dokümantasyonuna giderek görebilirsin python nun paket yöneticisidir pip (pip.pypa.io))**

**RUN pip install –r requirements.txt**



1. **Açıklama satırı olan EXPOSE satırını giriyorum 80 portundan çalışacağını izah ediyorum.**

**EXPOSE 80**

1. **En son ki satırım olan CMD satırını giriyorum bu uygulamamı çalıştıracak olan komut.**

**CMD python ./bookstore-api.py**

**Docker-compose.yml dosyamızı yazalım...**

1. **Docker docs. a gidip nasıl yazacağımıza bakabiliriz (docker compose reference). Önce hangi docker compose versionunu kullanmak istiyorsak onu belirtmeliyiz. En son versionu kullanma gibi bir zorunluluk yok hangisi çalışıyorsa onu kullanabilirsiniz.**

**Version: “3.7 ”**

1. **Services kısmını yazacağız (bu yapıda services aslında containerlar ve bir tane olmak zorunda. Birde docker swarm vardı burada cont sayısını belirleyebiliyoruz 3-5 fakat compose de belirleyemiyoruz.). Database ve Myapp services hazırlayalım..**

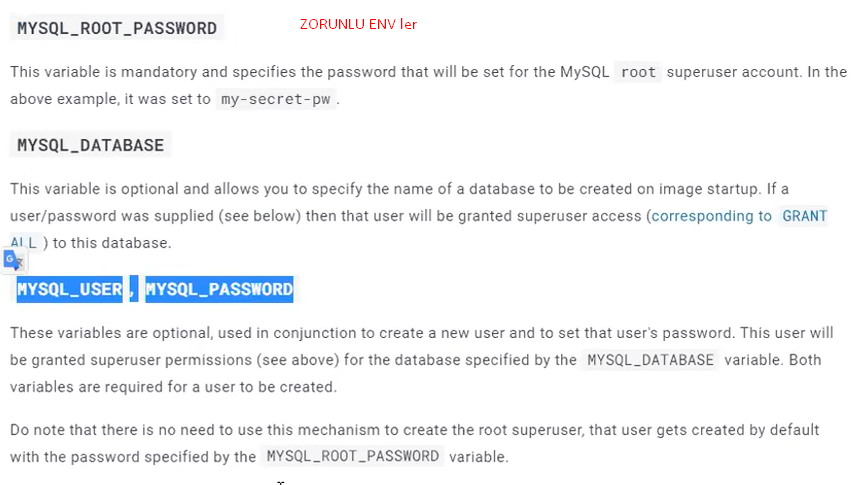
**-Burada database in aynı olması gerek (şifre,user..)değiştiremiyorduk. isimle ulaşmak için aynı olmalı.**

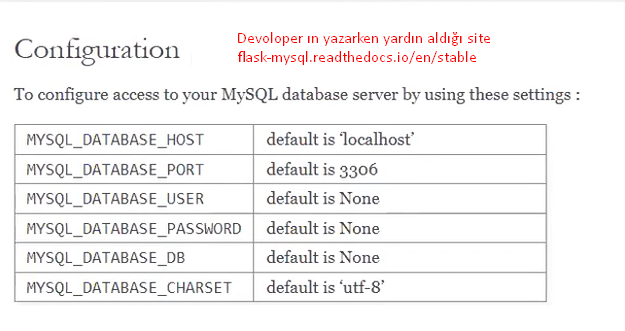
**-Database in imajı mysql:5.7 imajı nereden çekecek docker hubdan , docker hub a gidip imajı inceleyebilirsin. Burada environment variables var ne demiş burada mysql\_root\_password var zorunlu bir alan (imajın istediği root şifre belirlemen gerek). Zorunlu girmem gereken bazı env ler var.**

**services:**

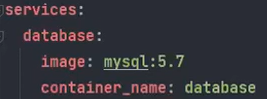
**database:**

**image: mysql:5.7**

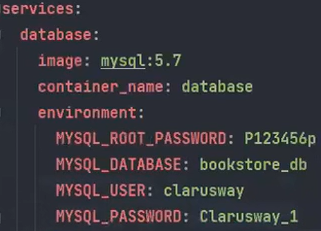


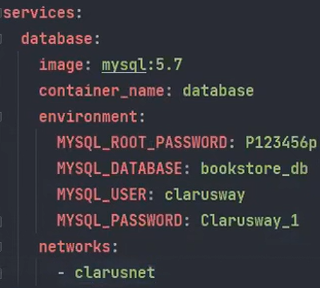


* **Database için imaj oluşturduktan sonra istersek buraya container name atayabiliriz ve database aynı ismi verebiliriz fakat isim vermesek de olur ama ping atarken ismi ile atabiliriz bu kolaylık var ip ye gerek kalmayacak.**



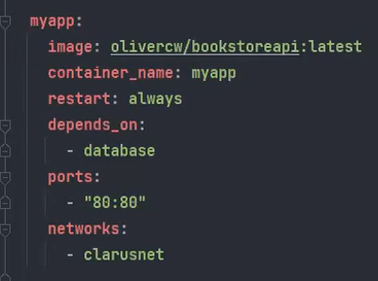
* **Environment leri girelim.. docker hub da zorunlu olan yerleri docker hub a gidip kopyalayıp yml yapıştırıyorum. Root şifresini kendim atıyorum ve diğer değişkenleri bookstore-api.py e göre düzenliyorum.**



* **Networku tanımlayalım Env ile aynı hizada olmalı bir isim verelim.** 

**Databae cont. ile girilecek bilgileri tamamladık... ve şimdi web için oluşturacağımız cont. hazırlayalım.**

* **Myapp için bir imaj oluşturacağız fakat burada bir durum var ortada bir imaj yok docker build yapmadan önce ve docker composu çalıştırmadan önce bir imaj ismi oluşturmalıyım ve kendi docker hub hesabımada gönderebilme ihtimalime karşı havvanur94, registry url kısmınıda girmeliyim! Terraformda docker build diyeceğim..**
* **Cont ismini giriyorum myapp de verebiliriz farklı bir isim de verebilirz... Restart kısmını always giriyorum herhangi bir sebeple durursa devamlı çalıştırsın...**
* **depends\_on kısmına önce hangi cont. kurulmasını istiyorsam onu yazmalıyım (DATABASE SERVİSİNİN ADI ne ise onu yazmalı) burada database ilk bağlanayım ki web cont a verileri (kitap yazar bilgileri) atabileyim...**
* **Uygulamamı publish etmek istediğim için burada b:ir port numaraası gerekli readme de 80 istenmiş imajın defoult portuda 80. 80:80...**
* **Network bilgisini giriyorum clarusnet, aynı network olmalı database ile...**



1. **Son olarak services gibi Networks ve Volmes parametrelerini girmem gerek.**

**Oluşturacağım neworks adını giriyorum clarusnet, normalde dökümantasyonda driver ı da eklememizi ister fakat burada defoult driver bridge olduğu için yazmasak da sorun olmaz. Network belirlemeseydik cont içeriside defoult networkü atayacaktı. Volume bu projede hata oluşturuyor.**



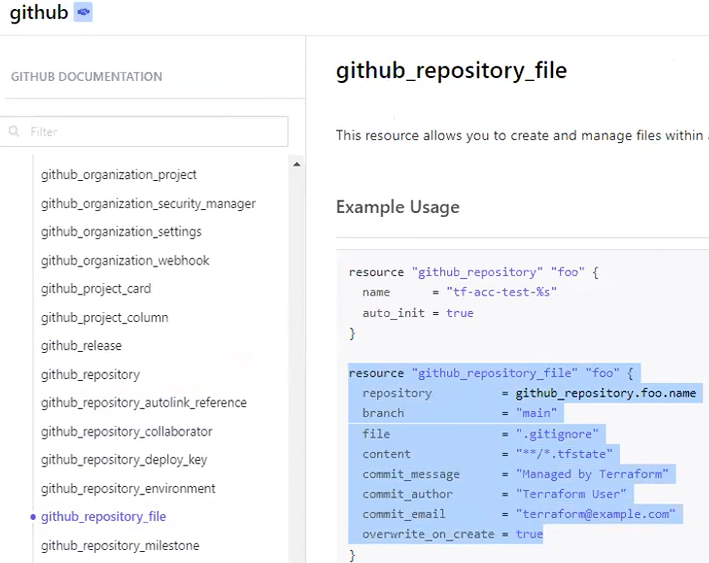
**DOCKER COMPOSE TAMAMLADIK**

**MAİN.TF OLUŞTURACAĞIZ**

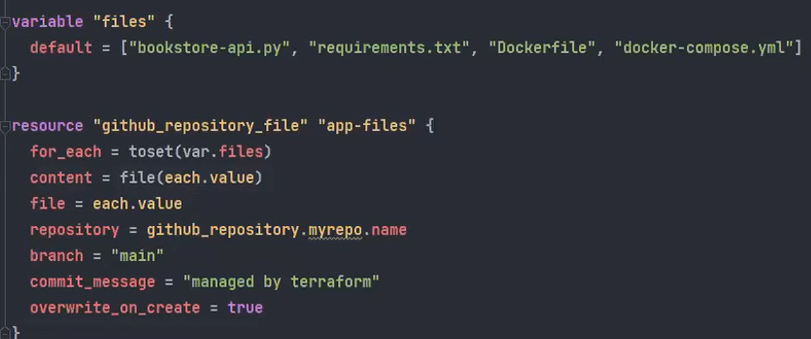
**Main.tf i yazalım**



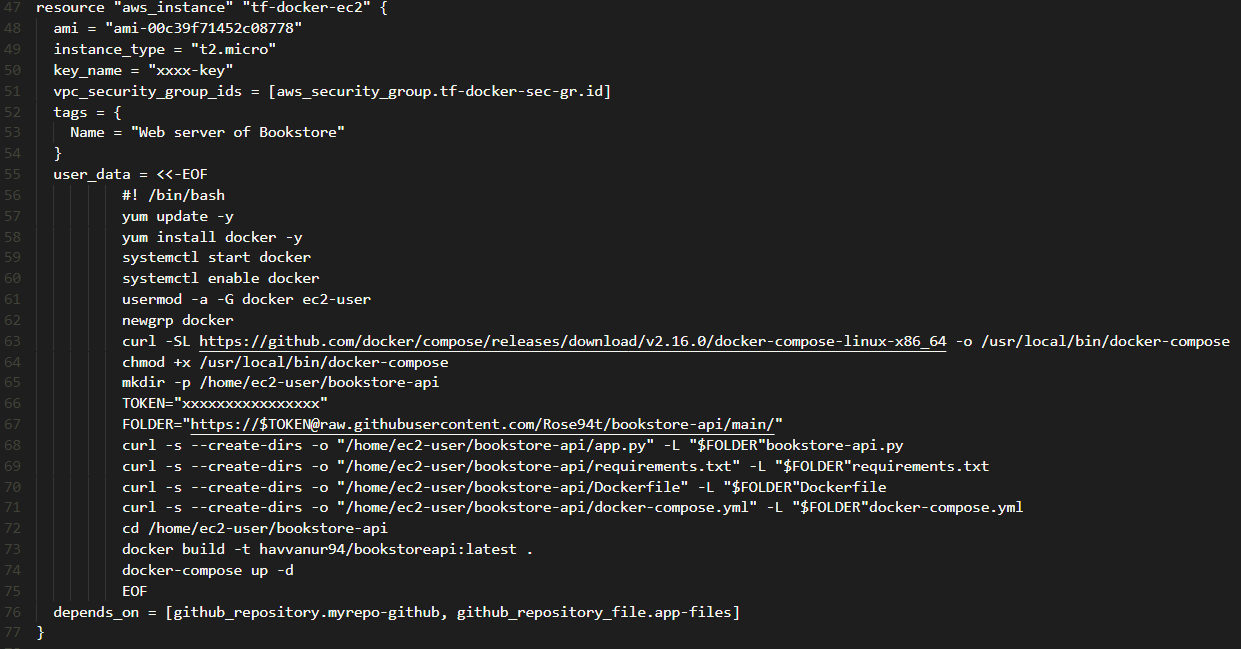
* **İnternette aws provider terraform a gelerek providerı alalım. Aynı şekilde github provider terraform a gelerek providerı alalım. Arka arkaya terraform bloğu yazılı bu şekilde olmamalı github ın terraform kısmını sildim.**
* **AWS provider kısmına region bilgilerini girdim eğer ki lokalde veya ec2 da çalışıyorsam admin yetkisi veya terraform dersinde hazırladığın rolü atamam lazım ya da access\_key ve secret\_key bilgilerini region nun altına girmeliyim..Farklı prıfillerin var ise profile bilgisini de girmen gerek. bu bilgiler aws hesabımda işlem yapmam için gerekli.**
* **Github hesabımda işlem yapabilmem için token bilgilerimi girmem gerek.**
* **Github repository için resource oluşturuyorum. Yukarıda oluşturduğum dosyaları buraya pushlayacağım isim uniq olmalı , commitlemek için auto\_init diyorum. Private diyorum.**
* **Brancıhını belirliyorum bunların hepsini terraform github dokümantasyonundan alıyorum.**
* **Repository=github\_repository.myrepo.name diyorum burada ki name bookstore-api oluyor.**
* **Yukarıda oluşturduğumuz dosyaları gönderme resource nu hazırlayacağız bunu terraform githuba gelerek hazırlayacağım github\_repository\_file kısmına geliyorum ve oradan bilgileri alıyorum ya da terraformda manuel yazabilirim.**



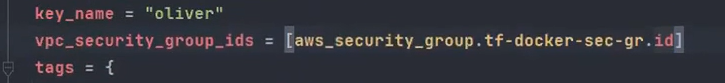
* **Burada geçen content i nasıl doldurabilirim dosyalardan bilgileri çekmesi için file deyip dosya yollarını (pathlerini) belirleyeceğim. Ondan önce variable belirleyeceğim bookstore-api.py, requirements.txt, dockerfile ve dockercomposeyml şeklinde yazacağım..**
* **For\_each toset(var.files) ile dizini set e çevir diyeceğim ve var ile variable files deki bilgileri tek ek al diyeceğim ve content file(each.value) dosyanın içini content olarak tek tek al diyeceğim. Aynı şekilde file yede each.value dersem github repoda da aynı sıra ile ve isim ile içeriğe aynı isimi verecek ve repository belirtiyorum branch belirtiyorum commit mesajı ekliyorum son olarak overwrite\_on\_create true diyorum böylelikle sonra dosyayı değiştirebilirim.**



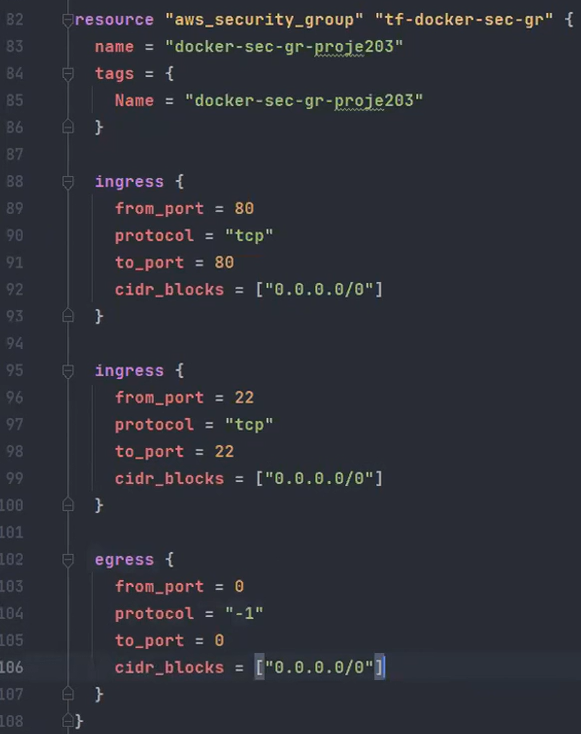
* **İnstance resource nu oluşuruyoruz.Key name kendine göre düzenlemelisin. Burada user datayıda giriyoruz sistemi update ediyoruz ardından docker kuruyoruz ve sistemi start ve enable ediyoruz. Ec2 userı docker grubuna ekliyorum –a –G ile ve newgrp docker ile.. Ardından docker-compose yüklüyorum. Mkdir –p il dosya oluşturuyorum dosyalarımı göndereceğim dosya dizini oluşturdum. (-p O satırdaki klasörlerin hepsini sırayla oluştur demek). Token değişkeni oluşturuyorum uzun bir satır oluşmasın diye. Folder değişkeni oluşturuyorum: Githubın raw url sini alarak curl komutu ile dosyaları çekebiliyorum.. raw url sini main e kadar yazıyoruz main altında ki kısmı curl ile çek diyoruz.**
* **Folder değişkenini uzun uzun girmemek için yukarıda ki oluşturduğum folder değişkenini şimdi curl komutunda kullanacağım.**
* **Curl deki –create-dirs belirtilen şekilde dizin oluştur demek -o küçük o farklı bir isim girebiliyorsun büyük o aldığın dosyayı aynı isimde çekebilirsin.. –L github repoda bulnan dizini (lokaldeki dizini) temsil eder. curl komutu diyorki –L ile folderın altında bulunan dosya her ne ise –o ile ismini app olarak değiştir diyor.**
* **Değişiklik yapmam gereken bir yer var ben bookstore-api.py yi app olarak değiştirdim Dockerfilede de değiştirmem lazım app.py olarak...**
* **Cd ile bokstore-api ye giriyorum githubdan gelecek olan dosyalar orada olacak.**
* **İmage i buid edeceğim –t ile (tag leyerek) senin yukarıda oluşturduğun imajın ne ise onu girmelisin.**
* **Docker copmose up komutunu giriyorum ve –d ile detech modda çalışsın diyorum**
* **Son olarak yapmam gereken bir şey var şuan oluşturduğun instance resource bloğunun github ile herhangi bir bağlantısı var mı yok hata almamam için ilk github ın oluşması lazım ondan dolayo depends\_on kısmına github repository oluşsun ve file ler de oluşsun githubda ki ben curl ile çekebileyim file leri)**
* **End of file (EOF) çoklu satırlarda komutun bittiğini ifade eder.**



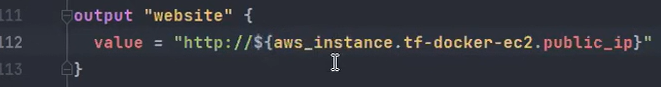
* **Son resource plan sec grup resource bloğunu oluşuracağız...**
* **Bu arada yukarıdaki VPC sec grup ids kısmını doldurmayı unutmuyoruz!!**



* **İngress de 80 ve 22 portunu açalım cıdr bloğunu her yere aç dıyoruz 0.0.0.0 ve egress yani outb. Bunada her yere ulaşabilelim diyoruz 0 tüm portlar -1 tüm protokoller demek anywhere demek için ise cıdr bloğu 0.0.0.0 olmalı..**



* **Her şey tamam birde http url verecek olan output oluşturalım . Terraform da string içinde referans yapıyorsam $ (dolar) ve süslü parantez {} içinde yapıyorum. Aws instance ın public ip sini al diyorum , PUBLİC DNS de olurdu.**



**Terraform init terraform apply komutlarını koşalım... githuba gidip bakabilirsin repo oluşmuş mu dosyalar gelmiş mi? ardından oluşan instance içine girebilirsin dosyalar gelmiş mi? . Conainerlar oluşmuş mu?**

**Ve database cont içine girip oluşan klasör ve dosyaları görebilirsin nasıl**

**Docker ps –a**

**docker exec -it myapp ash (uygulama ya da data içeren container'a bağlanalım)**

**apk add --no-cache mysql-client**

**mysql -u clarusway -p -h database (mysql -uroot -p bookstore\_db)**

**password : Clarusway\_1**

**SHOW DATABASES;**

**USE bookstore\_db**

**SHOW TABLES;**

**SELECT \* FROM books;**

**- Site geldiğinde kitap yazar ismi ekleme yükleme silme işlemleri için bu komutları terminalde girebilirsin.**

**curl komutunda kendi public ip mizi girmeliyiz. Browsera ip nin yanına/books ekleyip görebilirsin..**

**curl --request POST \**

**--url 'http://ec2-3-83-15-204.compute-1.amazonaws.com/books' \**

**--header 'content-type: application/json' \**

**--data '{"author":"Paulo Coelho", "title":"The Alchemist", "is\_sold":true}'**

**curl --request POST \**

**--url 'http://ec2-3-83-15-204.compute-1.amazonaws.com/books' \**

**--header 'content-type: application/json' \**

**--data '{"author":"Gabriel Garcia Marquez", "title":"One Hundred Years of Solitude", "is\_sold":true}'**

**curl --request POST \**

**--url 'http://ec2-3-83-15-204.compute-1.amazonaws.com/books' \**

**--header 'content-type: application/json' \**

**--data '{"author":"Harper Lee", "title":"To Kill a Mockingbird", "is\_sold":true}'**

**curl --request PUT \**

**--url 'http://ec2-3-83-15-204.compute-1.amazonaws.com/books/3' \**

**--header 'content-type: application/json' \**

**--data '{"author":"Jose Rodrigues dos Santos", "title":"A Formula de Deus", "is\_sold":true}'**

**curl --request DELETE \**

**--url 'http://ec2-3-83-15-204.compute-1.amazonaws.com/books/2' \**

**--header 'content-type: application/json'**

**TERRAFORM DESTROY demeyi unutmayalım...**

**EC2 da çalıştığımız dosyaları locale taşımak için**

**EC2 da koşmamız gereken komut**

**zip -r squash.zip /home/ec2-user/Project (çalıştığım klasörü bulunduğum dizine zipledim bulunduğum dizin ec2-user)**

**Localde key.pem in olduğu dizinde koşmamız gereken komut**

**scp -i xxxx-key.pem ec2-user@ec2-54-236-53-147.compute-1.amazonaws.com:/home/ec2-user/squash.zip c:/Users/ACER/OneDrive/Masaüstü**