MERN, MINIKUBE VE UBUNTU SERVER

ŞÜKRÜ ÇİRİŞ

ARÇELİK A.Ş.

1. Giriş:
   MERN uygulamanın kodunu yazdım. Kodu ve ilerdeki kullanılacak dosyaları buradaki repodan bulabilirsiniz: https://github.com/SUKRUCIRIS/MERN-Minikube-Ubuntu.
   Ubuntu Server kurulumunu yap. Bedava Ngrok hesabı aç. Ngrok kullanarak CI/CD için Github Actions üstünde ssh yapıcaz ve minikube üstünde çalıştırdığımız web uygulamasını dış ağa açacağız.
2. Ubuntu Server Minikube kurulumu:
   "sudo apt install docker.io"
   Komutu ile docker indir.
   "sudo apt install -y curl wget apt-transport-https"
   "curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64"
   "sudo install minikube-linux-amd64 /usr/local/bin/minikube"
   Komutları ile minikube kur.
   "curl -LO https://storage.googleapis.com/kubernetes-release/release/`curl -s https://storage.googleapis.com/kubernetes-release/release/stable.txt`/bin/linux/amd64/kubectl"
   "chmod +x kubectl"
   "sudo mv kubectl /usr/local/bin/"
   Komutları kubectl kur.
3. Ubuntu Server SSH ayarlamaları:
   Github actions da root olarak ssh ile girmek gerekiyor diğer türlü yetki sorunları çıkıyor. SSH ile root olarak login olabilmek için birkaç ayarlama gerekiyor. Root olarak login olabilmek için sshd_config dosyasına iki satır eklemek gerekiyor. "nano /etc/ssh/sshd_config" komutu ile o dosyayı açtım ve "PermitRootLogin yes" ve "AllowUsers root" satırlarını ekledim. Daha sonra "service sshd restart" komutunu çalıştırdım.
4. Ubuntu Server Ngrok kurulumu:
   "curl -s https://ngrok-agent.s3.amazonaws.com/ngrok.asc | sudo tee /etc/apt/trusted.gpg.d/ngrok.asc >/dev/null && echo "deb https://ngrok-agent.s3.amazonaws.com buster main" | sudo tee /etc/apt/sources.list.d/ngrok.list && sudo apt update && sudo apt install ngrok" komutu ile ngrok kur.
   Ngrok hesabını oluştur ve authtoken'ini elde et.
   "ngrok config add-authtoken <your_authtoken>" komutu ile Ubuntu Server'da ngrok hesabını aç.
5. Docker ve kubernetes dosyaları:
   Daha önce yaptığım MERN-GKE uygulamasından birkaç küçük fark var. O yüzden sadece o farkları anlatacağım. MERN-GKE belgesi burada: https://github.com/SUKRUCIRIS/MERN-kubernetes

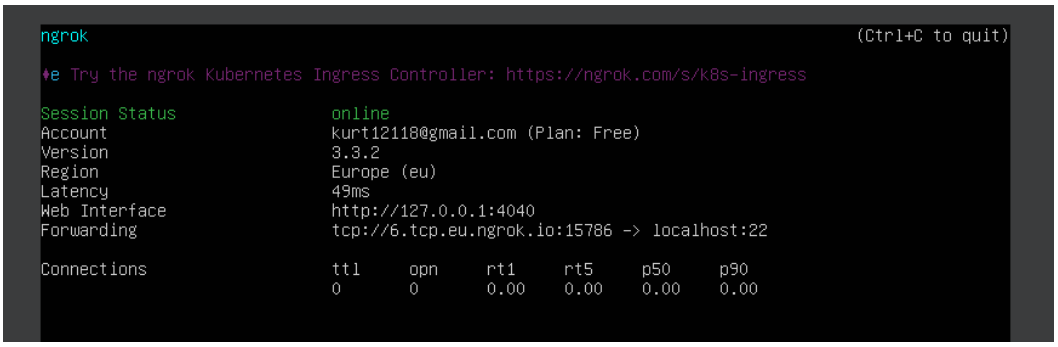
Storage Class dosyasındaki provisioner parametresi değişti:

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
apiVersion: v1
kind: PersistentVolume
metadata:
  name: mongo-pv
  labels:
    type: local
spec:
  storageClassName: mongo-storage
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"

```

GKE dynamic volume provisioning sağlıyordu ama minikube'de olmadı. O yüzden manuel olarak volume oluşturma dosyası da yazdım. En az Volume claim'de claim ettiğiniz boyut kadar volume oluşturmak gerek.

6. CI/CD:
   Öncelikle Ubuntu server'da "ngrok tcp 22" çalıştırılarak uzaktan ssh yapılabilir hale gelir. SSH varsayılan olarak 22 portunda çalışır.



Buradaki host ve port ismini github actions'da kullanacağız.
Github actions dosyası:
GitHub Actions ile github reposuna her push yaptığımda o repodan image'ları oluşturup dockerhub'a push ediyor. Daha sonra server'da onları canlıya alıyor. Public repolar için tamamen bedava. On birinci satırda yazdığım üzere Ubuntu kullanıyorum. Her step'te uses veya run kullanılmalı. run ile bildiğimiz komut çalışıyor, uses ile github'ın hazır bulundurduğu komut kümeleri. On üçüncü satırdaki github reposundaki dosyaları kullanabilmemizi sağlıyor. On dördüncü satırda dockerhub'a login oluyorum, kullanıcı adı ve token kullanarak. Bunları public repodaki bir dosyaya yazmak tehlikeli olacağından github secrets kullandım. Reponun ayarlarından secret ekleyerek burada kullanabilirsiniz. Yirminci satırda SSH yaptım gerekli bilgilerle, burada secrets kullanmadım çünkü server olarak bir sanal bilgisayar kullanıyorum ve bedava ngrok'u her çalıştırdığımda host ve port değişiyor. Yirmi yedinci satırda repoyu server'a indiriyorum. Yirmi sekizinci satırda indirdiğim dosyanın içine giriyorum. Yirmi dokuzuncu satırda minikube halihazırda çalışıyorsa kapatıyorum.

Otuzuncu satırda minikube'u başlatıyorum. Otuz birinci satırda yaratacağım kaynaklar zaten varsa siliyorum. Otuz ikinci satırda onları yaratıyorum. Otuz üçüncü satırda dosyanın dışına çıkıyorum. Otuz dördüncü satırda dosyayı siliyorum.

```yaml
name: Docker Image CI-CD

on:
  push:
    branches: ["master"]
  pull_request:
    branches: ["master"]

jobs:
  runMultipleCommands:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - uses: docker/login-action@v2
        with:
          username: ${{ secrets.DOCKERHUB_USERNAME }}
          password: ${{ secrets.DOCKERHUB_TOKEN }}
      - run: docker compose build
      - run: docker compose push
      - uses: appleboy/ssh-action@v1.0.0
        with:
          host: 6.tcp.eu.ngrok.io
          username: root
          password: sukru
          port: 15786
          script: |
            git clone https://github.com/SUKRUCIRIS/MERN-Minikube-Ubuntu
            cd MERN-Minikube-Ubuntu
            minikube delete
            minikube start --driver=docker --force
            kubectl delete -R -f ./kubernetes-configs/ --ignore-not-found=true
            kubectl create -R -f ./kubernetes-configs/
            cd ..
            rm MERN-Minikube-Ubuntu -r

```

7. Canlıya alma:
   Daha sonra ssh için kullandığımız ngrok bağlantısını kapatıyoruz. "kubectl get all" komutu ile tüm podların hazır olup olmadığını kontrol et. Eğer hazırlarsa "minikube service client" komutu ile frontend'in çalıştığı url'yi gör. Daha sonra "ngrok http <url>" komutu ile uygulamanı dış ağa aç.

```
root@sukru:/home/sukru# minikube service client
|-----------|--------|-------------|--------------------------|
| NAMESPACE |  NAME  | TARGET PORT |           URL            |
|-----------|--------|-------------|--------------------------|
| default   | client |          80 | http://192.168.58.2:30135 |
|-----------|--------|-------------|--------------------------|
* Opening service default/client in default browser...
  http://192.168.58.2:30135
root@sukru:/home/sukru# ngrok http 192.168.58.2:30135
```

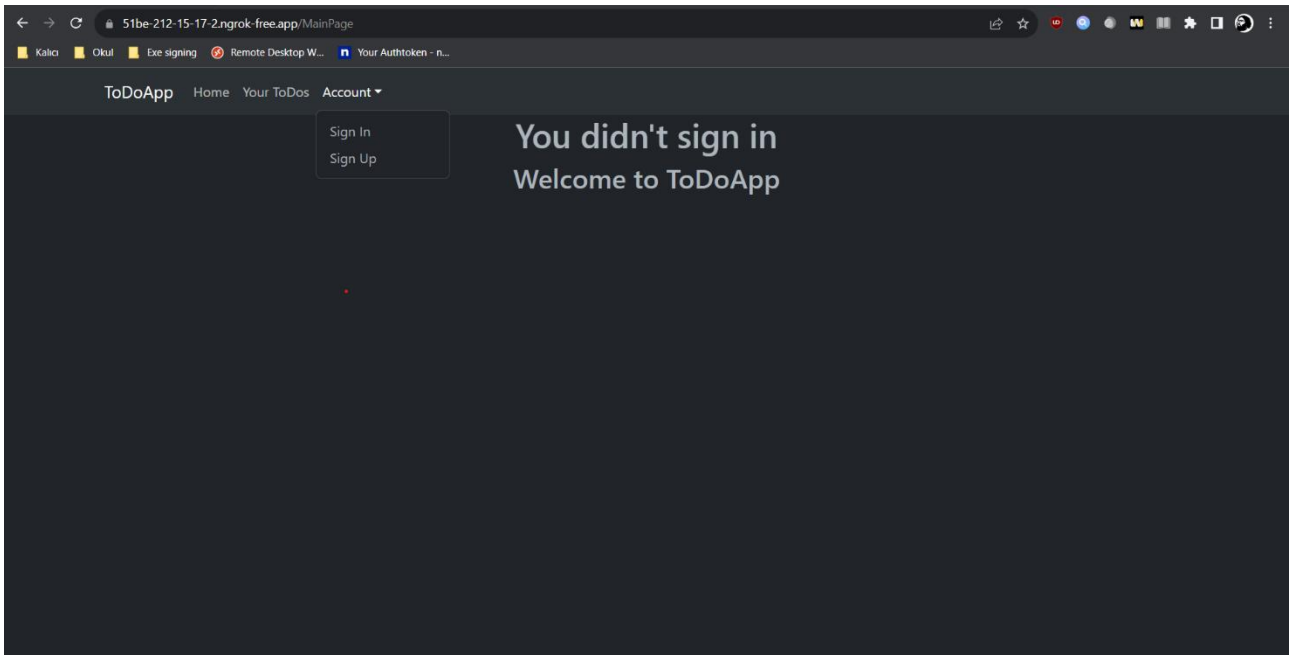
```
ngrok                                                                    (Ctrl+C to quit)

♦e Try the ngrok Kubernetes Ingress Controller: https://ngrok.com/s/k8s-ingress

Session Status                online
Account                       kurt12118@gmail.com (Plan: Free)
Version                       3.3.2
Region                        Europe (eu)
Latency                       53ms
Web Interface                 http://127.0.0.1:4040
Forwarding                    https://51be-212-15-17-2.ngrok-free.app -> http://192.168.58.2:30135

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Forwarding'de belirtilen url'ye gittiğinde uygulamanın çalıştığını görebilirsin.



8. Debug etme:
   "kubectl get all" komutu ile servislerin, pod'ların durumunu görebilirsin. Eğer birinde sorun varsa "kubectl logs <pod_ismi>" komutu ile pod'un loglarını görüp debug edebilirsin. "kubectl delete <isim>" komutu ile servisleri, deployment'ları veya pod gibi çalışan şeyleri silebilirsin.