

BULUT BİLİŞİM DERSİ

2. ÖDEVİ

ŞEVVAL ÇELİK - B221200057

MUHAMMED ALİ SARIHAN - B231200300



Görev Dağılımı

Bu projede 2 kişilik bir grup oluşturulmuştur. Bazı görevler beraber bazıları ayrı ayrı yapılmıştır.

Muhammed Ali Sarıhan:

- Veritabanı içeren Asp.NET Core uygulamasını konteynerleme ve buluta taşıma (RAM hatası verdi ve buluta taşınmadığı için baştan başladık.)
- Yeni bir WebSitesi ve Dockerfile oluşturma
- WebSitesini Docker'a konteyner olarak yükleme

Şevval Çelik:

- Yeni bir WebSitesi ve Dockerfile oluşturma
- WebSitesini Google Cloud'a taşıma
- Rapor ve Youtube videosunu hazırlama

Uygulama

- Bu ödev kapsamında bulut platformuna yüklemek için Node.js ve Express.js ile oluşturulan bir web çalışması seçilmiştir.
- Bulunduğumuz tarih dolayısıyla tema yılbaşısıdır. 'Yılbaşı Büyüsü 2025' adını koyduğumuz sitenin içeriğinde; yılbaşında yapılacak aktiviteler, alınacak hediyeler, ortaya çıkış ve kutlanma hikayesi sayfaları bulunmaktadır.



Uygulama



- Hediye fikirleri sayfasında sevdiklerinize satın alabileceğiniz veya elinizle yapabileceğiniz hediyeler bulunur.

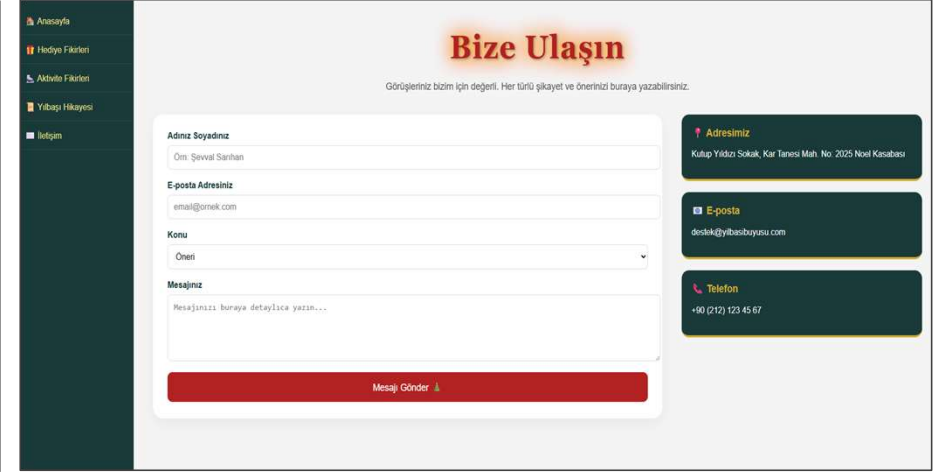


- Aktivite fikirleri sayfasında ise yeni yılı kutlamak ve kaliteli zaman geçirmek için yapılabilecek aktiviteler bulunmaktadır.

Uygulama



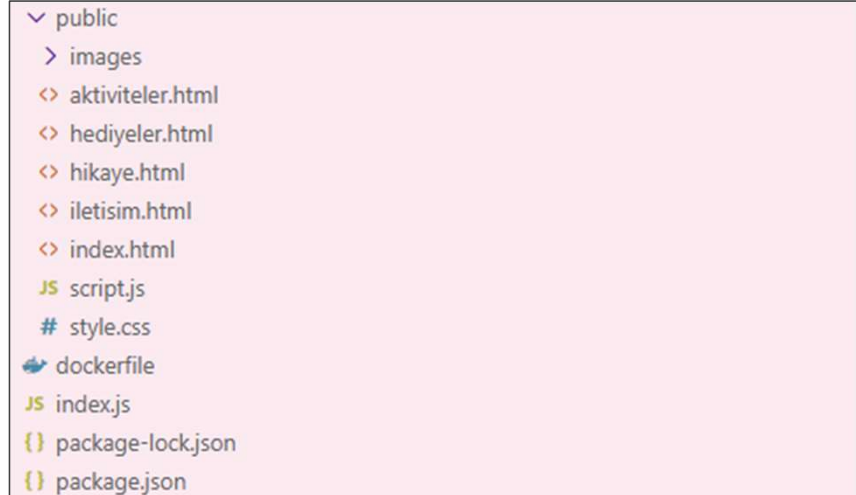
- Yılbaşı Hikayesi sayfasında yılbaşının geçmiş zamandaki izleri, kutlanmaya başlama hikayesi ve yanlış bilinen gerçekler bulunmaktadır.



- Son olarak İletişim sayfasında ise site hakkında yorumunu belirtmek isteyen kişiler öneri, şikayet, teşekkür ya da diğer fikirlerini bize ulaştırabilir.

Uygulama Mimarisi

- Görselde Node.js tabanlı 'bulut-bilişim-odev' klasörünün hiyerarşisi görüntülenebilmektedir.
- Public Klasörü: Web sitesinin frontend bileşenlerini barındırır. Html, Stil ve JavaScript dosyaları ile resimler klasörü bulunmaktadır.
- Dockerfile: Konteyner sistemi için yapılandırma dosyasıdır.
- Package, Package-Lock: Kütüphaneleri ve projenin nasıl başlatılacağını barındırır.
- Index.js: Uygulamayı bir web sunucusuna dönüştürür. Bu çalışmada standart olması için '3000' portu seçilmiştir.



```
▼ public
  > images
  <> aktiviteler.html
  <> hediyeler.html
  <> hikaye.html
  <> iletisim.html
  <> index.html
  JS script.js
  # style.css
  dockerfile
  JS index.js
  {} package-lock.json
  {} package.json
```

Docker Konteyner

- Geliştirilen web uygulamasının taşınabilir, ölçeklenebilir ve platformdan bağımsız bir şekilde çalışabilmesi için Docker Container teknolojisi kullanılmıştır.
- Uygulama klasöründe oluşturulan Dockerfile kullanılarak uygulamanın Docker imajı oluşturulmuştur.
- Oluşturulan imaj Docker Hub'a push edilerek bulut platformlarında erişilebilir hale getirilmiştir.
- Ayrıca arayüzde birkaç defa değişiklik yapılmıştır ve her değişiklikten sonra imaj yeniden build edilmiş ve Docker Hub'a pushlanmıştır.

Docker Image Oluşturma kodu:

➤ `docker build -t sevvalcelik/bulut-bilisim-odev .`

Docker Image Lokal Çalıştırma

➤ `docker run -p 3000:3000 sevvalcelik/bulut-bilisim-odev`

Docker Hub'a Push Etme

➤ `docker login`

➤ `docker push sevvalcelik/bulut-bilisim-odev`

Docker Konteyner

```
PS C:\Users\SEVVAL\bulut-bilisim-odev> docker build -t bulut-bilisim-odev .
[+] Building 145.4s (11/11) FINISHED                                docker:desktop-linux
=> [internal] load build definition from dockerfile                 0.0s
=> => transferring dockerfile: 162B                                0.0s
=> [internal] load metadata for docker.io/library/node:18         1.9s
=> [auth] library/node:pull token for registry-1.docker.io        0.0s
=> [internal] load .dockerignore                                   0.0s
=> => transferring context: 2B                                       0.0s
=> [1/5] FROM docker.io/library/node:18@sha256:c6ae79e38498325db67193d391e6ec1d224d96c693a8a 137.2s
=> resolve docker.io/library/node:18@sha256:c6ae79e38498325db67193d391e6ec1d224d96c693a8a4d 0.0s
=> sha256:461077a72fb7fe40d34a37d6a1958c4d16772d0dd77f572ec50a1fdc41a3754d 446B / 446B 0.4s
=> sha256:c6b30c3f16966552af10ac00521f60355b1fcfd46ac1c20b1038587e28583c 45.68MB / 45.68MB 59.7s
=> sha256:3697be50c98b9d071df4637e1d3491d00e7b9f3a732768c876d82309b3c5a145 1.25MB / 1.25MB 1.9s
=> sha256:cda7f44f2bddcc4bb7514474024b3f3705de00ddb6355a33be5ac7808e5b7125 3.32kB / 3.32kB 0.8s
=> sha256:e23f099911d692f62b851cf49a1e93294288a115f5cd2d014180e4d3684 211.36MB / 211.36MB 132.5s
=> sha256:79b2f47ad4443652b9b5cc81a95ede249fd976310efdbee159f29638783778 64.40MB / 64.40MB 78.0s
=> sha256:37927ed901b1b2608b72796c6881bf645480268eca4ac9a37b9219e050bb4d 24.02MB / 24.02MB 34.5s
=> sha256:3e6b9d1a95114e19f12262a4e8a59ad1d1a10ca7b82108adc0f0605a2002949 48.49MB / 48.49MB 49.8s
=> extracting sha256:3e6b9d1a95114e19f12262a4e8a59ad1d1a10ca7b82108adc0f0605a200294964 0.9s
=> extracting sha256:37927ed901b1b2608b72796c6881bf645480268eca4ac9a37b9219e050bb4d84 0.3s
=> extracting sha256:79b2f47ad4443652b9b5cc81a95ede249fd976310efdbee159f29638783778c0 1.2s
=> extracting sha256:e23f099911d692f62b851cf49a1e93294288a115f5cd2d014180e4d3684d34ab 3.0s
=> extracting sha256:cda7f44f2bddcc4bb7514474024b3f3705de00ddb6355a33be5ac7808e5b7125 0.0s
=> extracting sha256:c6b30c3f16966552af10ac00521f60355b1fcfd46ac1c20b1038587e28583ce7 1.1s
=> extracting sha256:3697be50c98b9d071df4637e1d3491d00e7b9f3a732768c876d82309b3c5a145 0.0s
=> extracting sha256:461077a72fb7fe40d34a37d6a1958c4d16772d0dd77f572ec50a1fdc41a3754d 0.0s
=> [internal] load build context                                    0.2s
=> => transferring context: 2.32MB                                   0.2s
=> [2/5] WORKDIR /app                                             2.2s
=> [3/5] COPY package.json .                                     0.0s
=> [4/5] RUN npm install                                          3.3s
=> [5/5] COPY . .                                                 0.1s
=> exporting to image                                             0.6s
=> => exporting layers                                              0.2s
=> => exporting manifest sha256:9e5e1822c8c49a9b2555ee77b3b11a8be6608256df7607b892521dffb987f 0.0s
=> => exporting config sha256:4c5e51dc79e67422e897e29c3ae8b281eb65f072c6d28b52ab63218fcc861cc6 0.0s
=> => exporting attestation manifest sha256:b11f1ba6b98fda584f5d2371f6a55b3ce3af35c7c21135d20a 0.0s
=> => exporting manifest list sha256:89d09f65cfc70c4264b375250448160b31b4b9dcf662d2172699f9e9b 0.0s
=> => naming to docker.io/library/bulut-bilisim-odev:latest      0.0s
=> => unpacking to docker.io/library/bulut-bilisim-odev:latest   0.3s
PS C:\Users\SEVVAL\bulut-bilisim-odev> docker images
```

IMAGE	ID	DISK USAGE	CONTENT SIZE	EXTRA
bulut-bilisim-odev:latest	89d09f65cfc7	1.59GB	398MB	

```
C:\Users\SEVVAL\bulut-bilisim-odev>docker tag bulut-bilisim-odev sevvalcelik/bulut-bilisim-odev
C:\Users\SEVVAL\bulut-bilisim-odev>docker login
Authenticating with existing credentials... [Username: sevvalcelik]

Info → To login with a different account, run 'docker logout' followed by 'docker login'

Login Succeeded

C:\Users\SEVVAL\bulut-bilisim-odev>docker push sevvalcelik/bulut-bilisim-odev
Using default tag: latest
The push refers to repository [docker.io/sevvalcelik/bulut-bilisim-odev]
3697be50c98b: Pushed
461077a72fb7: Pushed
2b00628fd6e8: Pushed
df871577994c: Pushed
610aa3efa582: Pushed
37927ed901b1: Pushed
79b2f47ad444: Pushed
e23f099911d6: Pushed
d03d68ae3ffa: Pushed
60fdf70711b2: Pushed
cda7f44f2bdd: Pushed
3e6b9d1a9511: Pushed
c6b30c3f1696: Pushed
latest: digest: sha256:89d09f65cfc70c4264b375250448160b31b4b9dcf662d2172699f9e9b557421 size: 856
```


Bulut Platformu

- Bulut platformu seçiminde göz önüne alınanlar:
 - Konteyner tabanlı uygulamaları doğrudan çalıştırabilmesi
 - Otomatik ölçeklenebilirlik sunması
 - Fiyatlandırması
- Bu vesilede ilk tercih Microsoft Azure olmuştur. Ancak uygulamanın Docker Hub'tan Azure ortamına dağıtımı sırasında, Docker Hub tarafından uygulanan pull hızı sınırlamaları ile karşılaşmıştır.
- Bu sınırlamalar ücretsiz kullanım kapsamında gerçekleşmektedir ve dağıtım sürecinde hata alınmasına neden olmuştur.

...

Dağıtım sürüyor

Dağıtım adı : Microsoft.ContainerInstances-20251227224229

Abonelik : Azure subscription 1

Kaynak grubu : [bulut-bilisim-rg](#)

Başlangıç zamanı : 27.12.2025 22:48:28

Bağıntı Kimliği : ddc98957-280c-4468-a72c-d6e0bcc34409

▼

Dağıtım ayrıntıları

Kaynak	Tür	Durum	İşlem ayrıntıları
<div><div></div><div>WorkspaceDeployment-20251227224826</div></div>	<div><div></div><div>Dağıtım</div></div>	Created	İşlem ayrıntıları

! Dağıtım başarısız

'bulut-bilisim-rg' kaynak grubuna dağıtım başarısız oldu. Temel alınan API'den yararlı olabilecek ek ayrıntılar: At least one resource deployment operation failed. Please list deployment operations for details. Please see <https://aka.ms/arm-deployment-operations> for usage details.

- ☐ Hızlı başlangıç görüntüleri
 - ☐ Azure Container Registry
 - ☒ Diğer kayıt defteri
- Docker Hub hizmetinin yakın zamanda Docker görüntüleri için bir çekme hızı sınırı getirdiğini unutmayın. Docker Hub kayıt defterinden bir görüntü belirtirken bu sınır kapsayıcı örneğinizin oluşturulmasını etkileyebilir. [Daha fazla bilgi](#)

Bulut Platformu

- Microsoft Azure üzerinden yapılan dağıtım başarısız olduğu için Google Cloud üzerinden çalışmaya karar verilmiştir.
- Google Cloud seçilmesinin nedeni; Docker imajlarını doğrudan ve hızlı bir şekilde çalıştırabilmesidir. Aynı zamanda ücretsiz denemesi sayesinde fiyatlandırma olmadan kolaylık sağlar.
- Google Cloud, Cloud Run servisi ile sanal makine oluşturulmasına gerek kalmadan otomatik ölçeklenebilir şekilde konteynerler çalıştırılabilmektedir.
- Bu ödevde tercih edilen konteyner mimarisi ile birebir uyumludur.

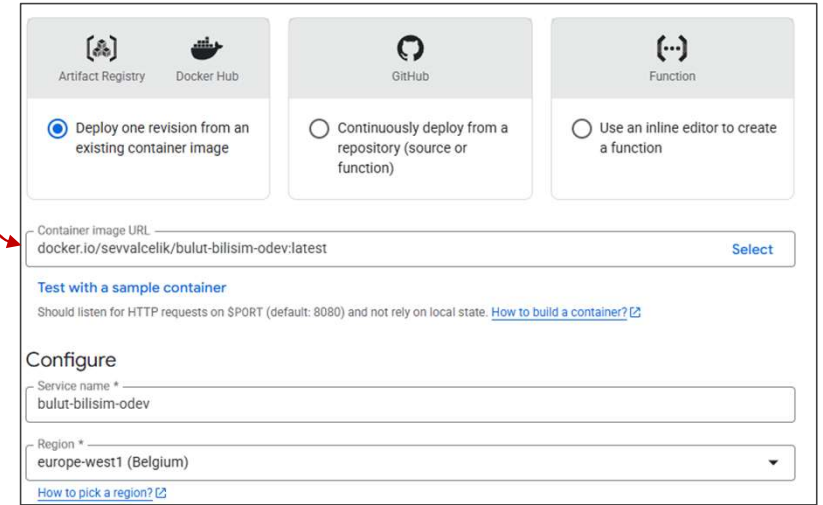


Cloud Run



Dağıtım Süreci

- Bulut platformu seçildikten sonra ilk olarak GCP üzerinde 'bulut-bilişim-odev' adında yeni bir proje tanımlanmıştır. Oluşturulan proje seçilerek altına yeni bir servis oluşturulmuştur.
 - Taşıma servislerini oluştururken Docker Hub'taki konteyner imajını belirtmek gerekmektedir.
 - Bunun için Docker Hub kullanıcı adı, imaj ismi, ve imaj etiketi (tag) bilgilerine sahip olunmalıdır.
- docker.io/sevvalcelik/bulut-bilişim-odev/latest



Artifact Registry Docker Hub GitHub Function

☒ Deploy one revision from an existing container image

☐ Continuously deploy from a repository (source or function)

☐ Use an inline editor to create a function

Container image URL
docker.io/sevvalcelik/bulut-bilisim-odev:latest [Select](#)

[Test with a sample container](#)
Should listen for HTTP requests on \$PORT (default: 8080) and not rely on local state. [How to build a container?](#)

Configure

Service name *
bulut-bilisim-odev

Region *
europe-west1 (Belgium)

[How to pick a region?](#)

Local

My Hub

1.22 GB / 0 Bytes in use

2 images

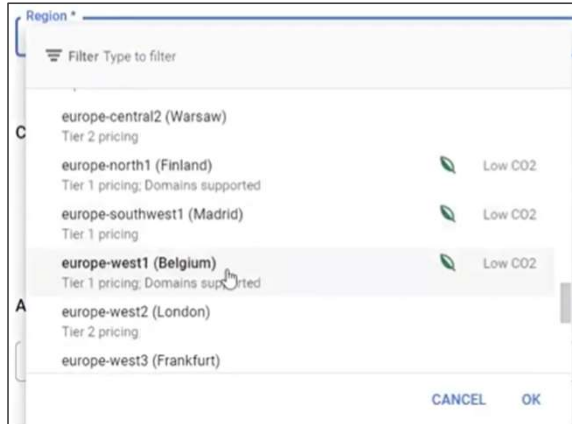
Last refresh: 23 hours ago

Q Search

<input type="checkbox"/>	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	<div><div></div>sevvalcelik/bulut-bilisim-odev</div>	latest	c75a2dbc3372	16 hours ago	1.62 GB	<div><div></div><div></div><div></div></div>
<input type="checkbox"/>	<div><div></div>bulut-bilisim-odev</div>	latest	89d09f65cfc7	21 hours ago	1.58 GB	<div><div></div><div></div><div></div></div>

Dağıtım Süreci

- Docker Hub bilgilerini tanımladıktan sonra dağıtım ayarları düzenlenmiştir.
- Bölge olarak europe-west1 (Belgium) seçilmiştir.
- Herkesin erişebileceği bir WebSite olacağı için herkese açık (public) olarak yapılandırılmıştır.
- Uygulamanın Dockerfile'ında yazan port (3000) dağıtım sürecinde de tanımlanmıştır.
- Bu ayarlar dışındakiler default bırakılmıştır.



Authentication *

- ☒ Allow public access
No authentication checks will be performed.
- ☐ Require authentication
Select between Identity and Access Management (IAM) and/or Identity-Aware Proxy (IAP).

Container image URL
docker.io/sevvalcelik/bulut-bilisim-odev:latest

Container port
3000

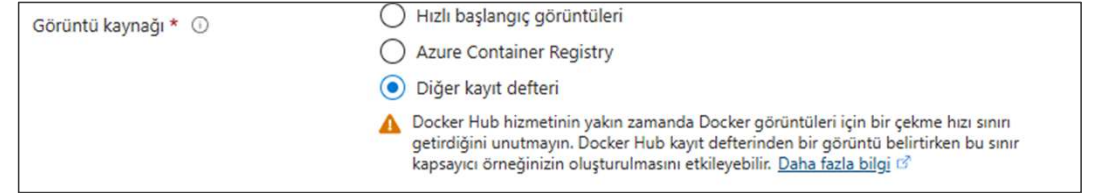
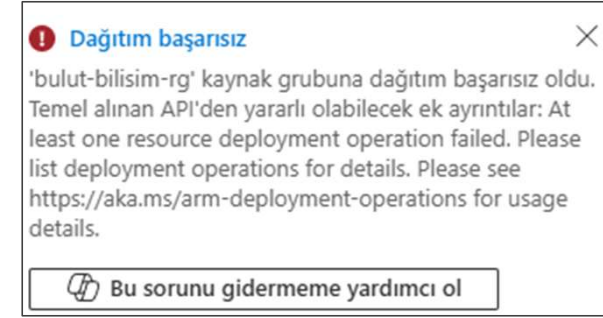
Requests will be sent to the container on this port. We recommend listening on \$PORT instead of this specific number.

Hedefler

- Bu projenin temel hedefi, konteyner tabanlı bir web uygulamasının bulut bilişim ortamında çalıştırılmasını uygulamalı olarak gerçekleştirmektir.
- Bu kapsamda hem Docker teknolojisi hem de modern bir bulut platformu kullanılarak, gerçek hayatta karşılaşılan bir dağıtım sürecinin deneyimlenmesi amaçlanmıştır.
- Proje ile hedeflenenler şunlardır:
 - Node.js tabanlı bir web uygulamasının geliştirilmesi
 - Geliştirilen uygulamanın Docker kullanılarak konteyner haline getirilmesi
 - Docker imajının Docker Hub üzerinde yayınlanması
 - Konteyner tabanlı bir mimari kullanılarak bulut ortamına dağıtım yapılması
 - Google Cloud Platform üzerinde Cloud Run servisi kullanılarak uygulamanın yayına alınması
 - Bulut bilişimde platform bağımsız ve taşınabilir uygulama geliştirme sürecinin öğrenilmesi

Karşılaşılan Sorunlar

- Bu uygulamayı gerçekleştirirken çeşitli sorunlarla karşılaşmıştır. Bunlardan biri:
- Ücretsiz erişimde Microsoft Azure platformunun Docker Hub tarafından sınırlanmasıyla dağıtım yapılmasına izin vermemesidir.
- Bu sorunun çözümü bulunamadığı için Azure platformundan vazgeçilmek durumunda kalınmıştır.
- Google Cloud Platformunda böyle bir sınırlamayla karşılaşılmamıştır.



Karşılaşılan Sorunlar

- Bir diğer sorun ise Cloud Run üzerinde dağıtım ayarlarını yapılandırırken ortaya çıkmıştır.
- GCP, ayarlarda belirtilmediği sürece dağıtım portunu 8080 kabul etmektedir. Fakat bu uygulama Dockerfile'ında port 3000 olarak tanımlanmıştır.
- Sorunun kaynağı teşhis edilene kadar denemeler yapılmış, index.js dosyası değiştirilmiştir. Yeniden push edildikten sonra test edilmiş fakat sorun devam etmiştir.

Eski index.js

```
app.listen(3000, () => {  
  console.log("Server running on port 3000");  
});
```

Yeni index.js

```
// Cloud Run uyumlu port  
const PORT = process.env.PORT || 3000;  
  
app.use(express.static("public"));  
  
app.get("/", (req, res) => {  
  res.sendFile(path.join(__dirname, "public", "index.html"));  
});  
  
app.listen(PORT, () => {  
  console.log(`Server running on port ${PORT}`);  
});
```


Karşılaşılan Sorunlar

- Kaynaklara bakıldığında, GCP'nin dağıtım ayarlarında port belirtilmese de Dockerfile'da yazan portu kabul edeceği yazmaktadır. Bu durum kafa karışıklığı yarattığı için sorunun çözümünü bulmak zorlaşmıştır.
- Uzun uğraşlar sonucunda yeni bir servis oluşturulmuş ve dağıtım ayarlarında port 3000 olarak belirtilmiştir. Böylece port karmaşıklığı ortadan kalkmış ve site bulutta yayınlanmıştır.

```
Deploying container to Cloud Run service [bulut-bilisim] in project [bulut-bilisim-482520] region [eur  
ope-west1]  
X Deploying new service...  
  - Creating Revision...  
    . Routing traffic...  
    ✓ Setting IAM Policy...  
Deployment failed  
ERROR: (gcloud.run.deploy) The user-provided container failed to start and listen on the port defined  
provided by the PORT=8080 environment variable within the allocated timeout. This can happen when the  
container port is misconfigured or if the timeout is too short. The health check timeout can be extend  
ed. Logs for this revision might contain more information.  
  
Logs URL: https://console.cloud.google.com/logs/viewer?project=bulut-bilisim-482520&resource=cloud_run  
_revision/service_name/bulut-bilisim/revision_name/bulut-bilisim-00001-mqb&advancedFilter=resource.typ  
e%3D%22cloud_run_revision%22%0Aresource.labels.service_name%3D%22bulut-bilisim%22%0Aresource.labels.re  
vision_name%3D%22bulut-bilisim-00001-mqb%22  
For more troubleshooting guidance, see https://cloud.google.com/run/docs/troubleshooting#container-fai  
led-to-start
```

Container image URL

docker.io/sevvalcelik/bulut-bilisim-odev:latest

Container port
3000

Requests will be sent to the container on this port. We recommend listening on \$PORT instead of this specific number.

Linkler

➤ Youtube Sunum Linki:

<https://www.youtube.com/watch?v=pQi-2EhALA4>

➤ WebSite Linki:

<https://bulut-bilisim-odev-957308431318.europe-west1.run.app>

➤ Github Linklerimiz:

<https://github.com/clksevval>

<https://github.com/MuhammedSarihan>

DİNLEDİĞİNİZ İÇİN
TEŞEKKÜR EDERİZ!