1. laboratorijska vježba - zadatak

Cilj prve laboratorijske vježbe je priprema okoline za izvedbu vježbe te upoznavanje s Azure portalom.

U ovoj laboratorijskoj vježbi ćete:

- koristiti Visual Studio Code za rad s Azure servisima
- kreirati jednostavnu funkciju u Pythonu i Go-u
- deployati funkciju koristeći Azure Functions
- deployati funkciju koristeći Azure Kubernetes Service

Napomena: Sve resurse koje kreirate na Azure Cloudu potrebno je imenovati prema sljedećoj konvenciji: <ime_tima>-<tip_resursa>-<naziv_aplikacije> (npr. Tim1-rg-ferlab bi bila resource group od tima Tim1). Za <tip_resursa> koristite kratice (resource group = rg, azure kubernetes service = aks itd.). U slučaju da nazivi pojedinih resursa dopuštaju samo mala slova i brojeve, izostavite "-".

Prijava na platformu (Azure portal)

Na Azure portal se prijavite klikom na Sign in opciju u gornjem desnom kutu na sljedećem linku: https://azure.microsoft.com/en-us/.

Nakon uspješne prijave, kliknite na ikonu Vašeg računa u gornjem desnom kutu te odaberite opciju Azure Portal.

Upoznavanje sa korisničkim sučeljem platforme (Azure portal)

Upoznajte se ukratko s Azure portalom.

Pogledajte listu dostupnih Azure servisa, kako kreirati novi resurs, gdje se nalazi Cloud Shell, popis korisnih poveznica itd.

Visual Studio Code, GoLang and AKS extensions

Kako biste mogli lakše izvršiti zadatke u ovoj, ali i sljedećim laboratorijskim vježbama, potrebno je instalirati određene alate.

1. Instalirajte Visual Studio Code: https://code.visualstudio.com/download.
2. Instalirajte GoLang: https://go.dev/dl/
3. Unutar Visual Studio Code-a, instalirajte sljedeće ekstenzije:
    a. Azure Account
    b. Azure Functions
    c. Azure Kubernetes Service
    d. Go
4. Instalirajte Azure Functions Core Tools: https://docs.microsoft.com/en-us/azure/azure-functions/functions-run-local?tabs=v4%2Cwindows%2Ccsharp%2Cportal%2Cbash#install-the-azure-functions-core-tools

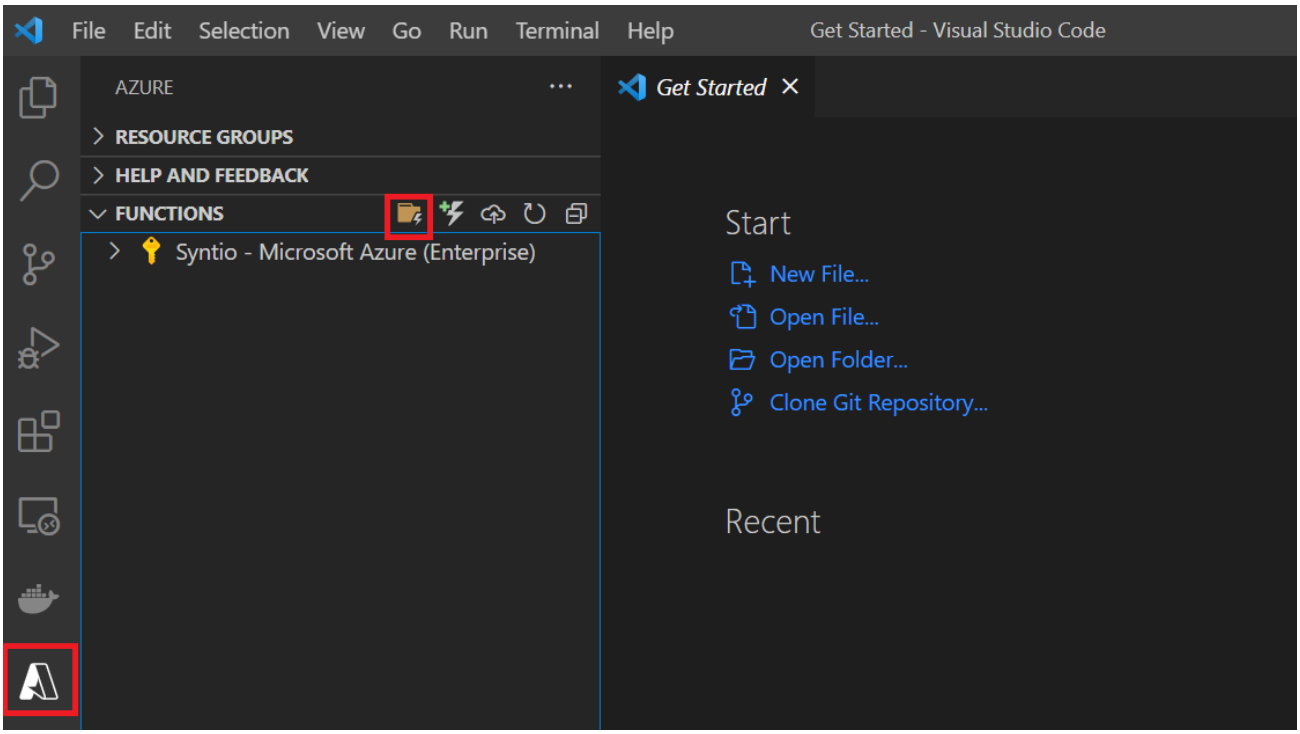
Prva Azure funkcija u Pythonu (Hello world http response)

Vaš je zadatak kreirati svoju prvu Azure funkciju koristeći Visual Studio Code i programski jezik Python.

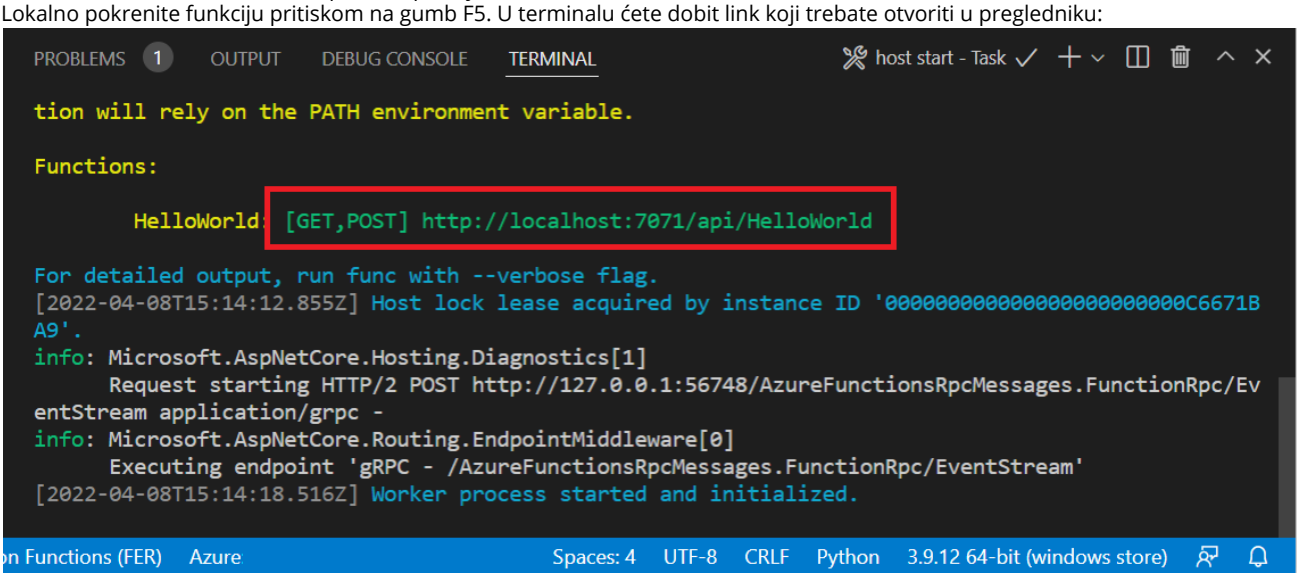
Pretpostavke:

- Imate instaliran Visual Studio Code i Azure Functions ekstenziju
- Imate linstaliran Azure Functions Core Tools
- Imate aktivnu pretplatu na Azure Cloud-u

1. U Visual Studio Code-u kliknite na ikonicu Azure portala. Ako prvi put koristite Azure u Visual Studio Code-u, vjerojatno će vas tražiti da se prijavite, pa tako i napravite.
2. Kreirajte svoj lokalni projekt, klikom na ikonu mape (prikazano na slici 1.). Zatim odaberite mapu na računalu u kojoj ćete razvijati svoju funkciju (preporuka je da to bude prazna mapa).



3. U nastavku, odaberite sljedeće opcije:
    a. programski jezik - Python (preporuka: odaberite verziju 3.9.)
    b. template - HTTP Trigger
    c. naziv funkcije - HelloWorld
    d. način autorizacije - Anonymus (ovaj način autorizacije omogućava bilo kome da poziva vašu funkciju)
4. Nakon završetka kreiranja projekta, otvorit će vam se **"**__init__.py**"** datoteka u kojoj ćete napisati svoju funkciju. Izbrišite automatski izgenerirani sadržaj main funkcije i zamijenite svojim kodom - napišite odgovor koji će funkcija vratiti na okidač (eng. trigger) "Hello World!". Nakon završetka spremite promjene.
5. Lokalno pokrenite funkciju pritiskom na gumb F5. U terminalu ćete dobit link koji trebate otvoriti u pregledniku:



Provjerite ispisuje li Vam se na ekranu željeni tekst.

Deploy i prvo izvršavanje funkcije na platformi

Nakon što ste napisali funkciju u Pythonu te ju pokrenuli lokalno, sljedeći korak je deploy funkcije na Azure Cloud platformu.

Postoji više mogućnosti za deploy same funkcije na platformu. U slučaju da ste instalirali sve potrebne ekstenzije opisane u prethodnim koracima za VS Code, možete koristiti VS Code za sami deployment:

1. Odaberite Azure ikonu na VS Code, zatim odaberite **Deploy to function app...** gumb
2. Odaberite direktorij u kojem se nalazi vaša funkcija

3. Odaberite opciju Create new Function App te upišite tražene parametre

Nakon što je deploy uspješno završen, otiđite na Azure portal kako biste vidjeli svoju novokreiranu funkciju.

Konfiguracija managed docker clustera (AKS)

Sljedeći korak je konfiguracija clustera AKS (Azure Kubernetes Service).

1. Pronađite **Resource groups** te kreirajte novu grupu resursa. Pri imenovanju slijedite konvenciju <ime_tima>-<tip_resursa>-<naziv_aplikacije> (npr. Tim1-rg-ferlab)
2. Pozicionirajte se u kreiranu grupu i odaberite opciju Create te potražite AKS. Ponovno odaberite Create kako biste kreirali novi cluster.
3. Nadopunite sljedeće podatke: naziv pretplate, naziv vaše resursne grupe te ime clustera: <ime_tima>-<tip_resursa>-<naziv_aplikacije> (npr. Tim1-aks-ferlab)
4. Odaberite Cluster preset configuration **Dev/Test ($)**
5. Pronađite izbornik Integrations kako biste povezali cluster s Azure Container Registry (ACR), koji će vam trebati u idućem dijelu laboratorijske vježbe
    a. Create new* U slučaju da ne postoji Container Registry na koji ćete spremati svoje Docker image, možete kreirati novi prilikom kreiranja samog clustera
6. Odabrati opciju “Review + create”

Prebaciti funkciju u GoLang i u kontejner, deploy u kontejneru

Python funkciju koju ste pisali u prethodnim koracima, potrebno je prepisati u programski jezik Go. Nakon što ste prepisali funkciju u Go, potrebno ju je kompajlirati u izvršnu .exe datoteku. U slučaju da želite testirati funkciju lokalno, potrebno je izmijeniti *host.json* datoteku:

- U customHandler treba dodati atribut `"enableForwardingHttpRequest"` i postaviti ga na `true`.
- `"defaultExecutablePath"` treba postaviti na <ime_izvršne_datoteke>.exe

Kako biste ju mogli deployati na kreirani AKS cluster, potrebno ju je staviti u Docker kontejner. U slučaju da nemate Docker Desktop, slijedite upute na sljedećoj poveznici: https://docs.docker.com/desktop/windows/install/ .

Kreiranje kontejnera

Prvi korak je kreiranje *dockerfile* datoteke. Kako biste izgenerirali *dockerfile* pomoću VS Code, pokrenite sljedeću naredbu: `func init --worker-runtime custom --docker`.

Nadopunite *dockerfile* tako da izgleda ovako :
Napomena: pripazite na strukturu svog projekta kako bi mogli ispravno napraviti docker image. Provjerite gdje vam se u strukturi projekta nalaze datoteke spomenute u *dockerfile*-u te prepravite putanje ako je potrebno

```
FROM mcr.microsoft.com/azure-functions/dotnet:3.0-appservice
ENV AzureWebJobsScriptRoot=/home/site/wwwroot \
    AzureFunctionsJobHost__Logging__Console__IsEnabled=true

COPY --from=installer-env ["/home/site/wwwroot", "/home/site/wwwroot"]

FROM golang:1.16-alpine
WORKDIR /app/HelloWorldGo

COPY . .

RUN go build -o ./out/HelloWorld .

# This container exposes port 8080 to the outside world
EXPOSE 8080

# Run the executable
CMD ["./out/HelloWorld"]
```

Kreirajte Docker image te iz kreiranog image-a pokrenite Docker kontejner kako biste provjerili da sve ispravno radi.

Podizanje kontejnera na Cloud

Prvi korak je stavljanje kreiranog image-a na ACR (Azure Container Registry) koji ste kreirali prilikom kreiranja samog clustera. Ukoliko niste kreirali ACR, možete ga kreirati u ovome koraku i naknadno integrirati s postojećim clusterom.

Stavljanje image-a na ACR može se napraviti unutar VS Codea. Odaberite Docker ikonu u VS Code te pronađete odgovarajući image. Desnim klikom otvara se izbornik u kojem je potrebno odabrati opciju **Push** te odabrati ACR u koji želite dodati image.

Kako biste mogli izvrtiti naredbu za podizanje kontejnera na Kubernetes - spojite se na AKS cluster unutar terminala u VS Code:

```
az login
az aks get-credentials –-resource-group <ime_resource_group> --name
<ime_clustera>
```

Kako biste mogli kreirati *Kubernetes pod*, koji će sadržavati vaš kontejner s Go funkcijom, potrebno je kreirati Kubernetes resurs koji se naziva *deployment.* Za kreiranje navedenog resursa potrebno je kreirati manifest .yaml datoteku sa sljedećim sadržajem:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp
spec:
  selector:
    matchLabels:
      app: myapp
  template:
    metadata:
      labels:
        app: myapp
    spec:
      containers:
      - name: myapp
        image: <Image>
        resources:
          limits:
            memory: "128Mi"
            cpu: "500m"
        ports:
        - containerPort: <Port>
```

Potrebno je izmijeniti ime aplikacije, ime imagea i port.

Kako bi se omogućio pristup aplikaciji preko ip adrese, u manifest datoteku je potrebno dodati i dio koji će kreirati *Service* resurs:

```
apiVersion: v1
kind: Service
metadata:
  name: myapp
spec:
  selector:
    app: myapp
  ports:
  - port: <Port>
    targetPort: <Target Port>
  type: LoadBalancer
```

Nakon što je datoteka spremna, možete krenuti s podizanjem vaše aplikacije na Kubernetes:

```
kubectl apply -f <ime_manifest_datoteke>.yaml
```

Nakon što se naredba izvršila, provjerite i testirajte radi li sve kako treba. U vašem AKS clusteru trebali biste vidjeti kreirane resurse.

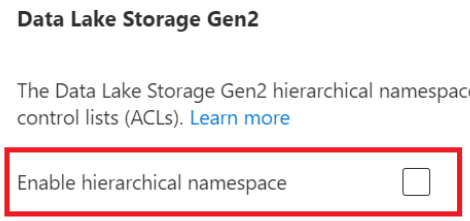
Azure AD i dodjela prava za grupni rad

U ovome koraku ćete se upoznati s Azure Active Directory te dodjeliti prava drugim studentima kako bi svi radili u istoj okolini:

1. Pronaći "Azure Active Directory"
2. Pod Manage kliknuti na **Groups**
3. Odabrati **All groups > New group**
4. U novom **New Group** prozoru odabrati vrstu grupe, ime i opis
5. Uključiti opciju Azure AD roles can be assigned to the group
6. Ispod te opcije nalaze se dijelovi za dodavanje članova i voditelja grupe. Također, treba dodati prava grupi. U ovom koraku možete dodati i pojedina prava unutar grupe
7. Nakon što ste odabrali članove i voditelja grupe kliknite na gumb create na dnu stranice

Konfiguracija spremišta podataka u oblaku (Azure data lake gen2)

1. Unutar Azure Portala otvorite Storage Accounts i odaberite opciju Create.
2. Ispunite tražene informacije na sljedeći način:
    a. Subscription - vaša aktivna pretplata na Azure-u
    b. Resource group - vaša resursna grupa koju ste kreirali ranije u sklopu ove laboratorijske vježbe
    c. Storage account name - <naziv_tima><tip_resursa><naziv_aplikacije> (npr. tim1saferlab)
    d. Region - West Europe
    e. Performance - Standard
    f. Redundancy - GRS
3. Kliknite na sljedeći korak. Pronađite sekciju Data Lake Storage Gen2. Odaberite opciju "Enable hierarchical namespace".



4. Na kraju odaberite opciju Create na dnu ekrana.

Napomena: Na kraju vježbe isključite Azure Kubernetes Cluster koji ste kreirali. Kako biste to napravili pronađite spomenuti resurs te kliknite na tipku "stop"