# Solution M5: Azure and DevOps

## Tasks

One possible solution, using the Azure Portal, may include the following steps:

### AKS Task

* Open a browser tab, navigate to <https://portal.azure.com> and login
* Navigate to **Resource Groups** and create a new one, for example **Homework-M5**
* Enter the resource group and click the **+ Add** button

#### AKS

* Either sear for **Kubernetes Service** or click the link in the **Popular** section
* For **Kubernetes cluster name** enter **k8shw**
* Adjust the **Subscription**, **Resource group**, and **Region** accordingly
* Set the **Node size** for example to **B2s**
* Set the **Node count** to 1
* Click **Review + create** and then **Create**
* Return back to home

#### ACR

* Search for **Container registries** and press **Enter**
* Click on the **+ Add** button to create a new one
* Make sure that **Subscription**, **Resource group**, and **Location** are set correctly
* Enter **archw** from **Registry name**
* Change **SKU** to **Basic**
* Click **Review + create** and then **Create**
* Once the resource is created, click on **Go to resource**
* Navigate to **Access keys** under **Settings**
* Enable the **Admin user**
* Copy one of the passwords

#### Dockerfile

* Navigate to an empty folder on your file system
* Create a folder and prepare a set of files or use the provided archive
* For example, create a **Dockerfile** with the following content

FROM nginx

COPY web /usr/share/nginx/html

* Create a **web** subfolder to accommodate the web page files
* Create a simple **index.html** file in the **web** folder with the following content

<html>

    <head>

        <title>AzE: Homework #5</title>

    </head>

    <body>

        <h3>@username</h3>

        <img src="soft-uni.png" alt="SoftUni" width="50%" />

    </body>

</html>

* Download or create a file **soft-uni.png** or another image and adjust the index file
* Keep in mind that you should have a **Docker** installed, running, and accessible
* Build the image with

**docker build -t aze-hw5-app .**

* You can test the app locally, or continue without testing it
* Login to the **Azure Container Registry** you created earlier

**docker login acrhw.azurecr.io**

* The username should be **acrhw** and the password - the one you copied earlier
* Tag the image against the registry

**docker tag aze-hw5-app acrhw.azurecr.io/aze-hw5-app**

* Push the image to the registry

**docker push acrhw.azurecr.io/aze-hw5-app**

#### Manifests

* Create a new folder named **manifests**
* Create a **deployment.yaml** file in the **manifests** folder with the following content

---

apiVersion: apps/v1

kind: Deployment

metadata:

  name: hw5-deployment

  labels:

    app: hw5-deployment

spec:

  replicas: 1

  selector:

    matchLabels:

      app: hw5

  template:

    metadata:

      labels:

        app: hw5

    spec:

      containers:

      - name: hw5

        image: acrhw.azurecr.io/aze-hw5-app

        ports:

        - containerPort: 80

          name: hw5

* Create a **service.yaml** file in the **manifests** folder with the following content

---

apiVersion: v1

kind: Service

metadata:

  name: hw5-service

  labels:

    app: hw5-service

spec:

  type: LoadBalancer

  ports:

  - port: 80

    targetPort: 80

    name: hw5-port

  selector:

    app: hw5

#### Run the application

* While still on the terminal, log on to **Azure Portal** if not done so

**az login**

* Attach the registry to the cluster

**az aks update -n k8shw -g Homework-M5 --attach-acr acrhw**

* Get the credentials to operate with the cluster

**az aks get-credentials --resource-group Homework-M5 --name k8shw**

* Start the application (from the base folder - where your Dockerfile is)

**kubectl apply -R -f .\manifests\**

* Return to the **Azure Portal**
* Navigate to the **Kubernetes** cluster
* Click on **Services and ingresses (preview)** under **Kubernetes resources**
* Find the **hw5-service** item and copy the IP address
* Open a browser tab and navigate to it

#### Stop and remove the application

* Being in the terminal session where you started the application, execute

**kubectl delete -R -f .\manifests\**

### DevOps Task

The way the challenge is set is beyond what we have covered so far

Instead of trying to solve it as per the requirements, we will improvise and will use a sample application

#### AKS cluster

* We will reuse the one created earlier

#### Source code repository

* The sample application will be uploaded to **Azure DevOps Repo** as part of the setup

#### DevOps project

* Return to **Azure Portal**
* Navigate to the home screen
* Search for **DevOps Starter** and press **Enter**
* Click the **+ Add** button
* Select **.NET** and click **Next**
* Select **ASP.NET Core** and click **Next**
* Select **Kubernetes Service** and click **Next**
* Enter **hw5** in the **Project name** field
* Select an existing **Azure DevOps Organization** or enter a name to create new one
* Check the subscription
* Select **Use Existing** in the **Create new or use existing cluster** section
* Select the cluster created earlier (**k8shw**)
* Click on **Additional settings**
* You can accept most of the parameters with their default values
* Switch **Container Registry SKU** to **Basic**
* Change all **Location** fields to match the one of the cluster
* Click **OK**
* Click **Done**
* Once the creation process is complete, click on **Go to resource**
* There you will notice that the **Build** phase is running
* You can explore both phases while they are running
* After a while you will notice that both phases will end with success
* On the **Azure resources** section in the project dashboard will appear a new **Service** named **sampleapp**
* You may click on the link and check the result (a working simple application)
* You can browse the repository and the pipeline
* Explore a bit and try to adjust something to see the result

## A Reminder

Do not forget to remove the resources created during the homework solution practice