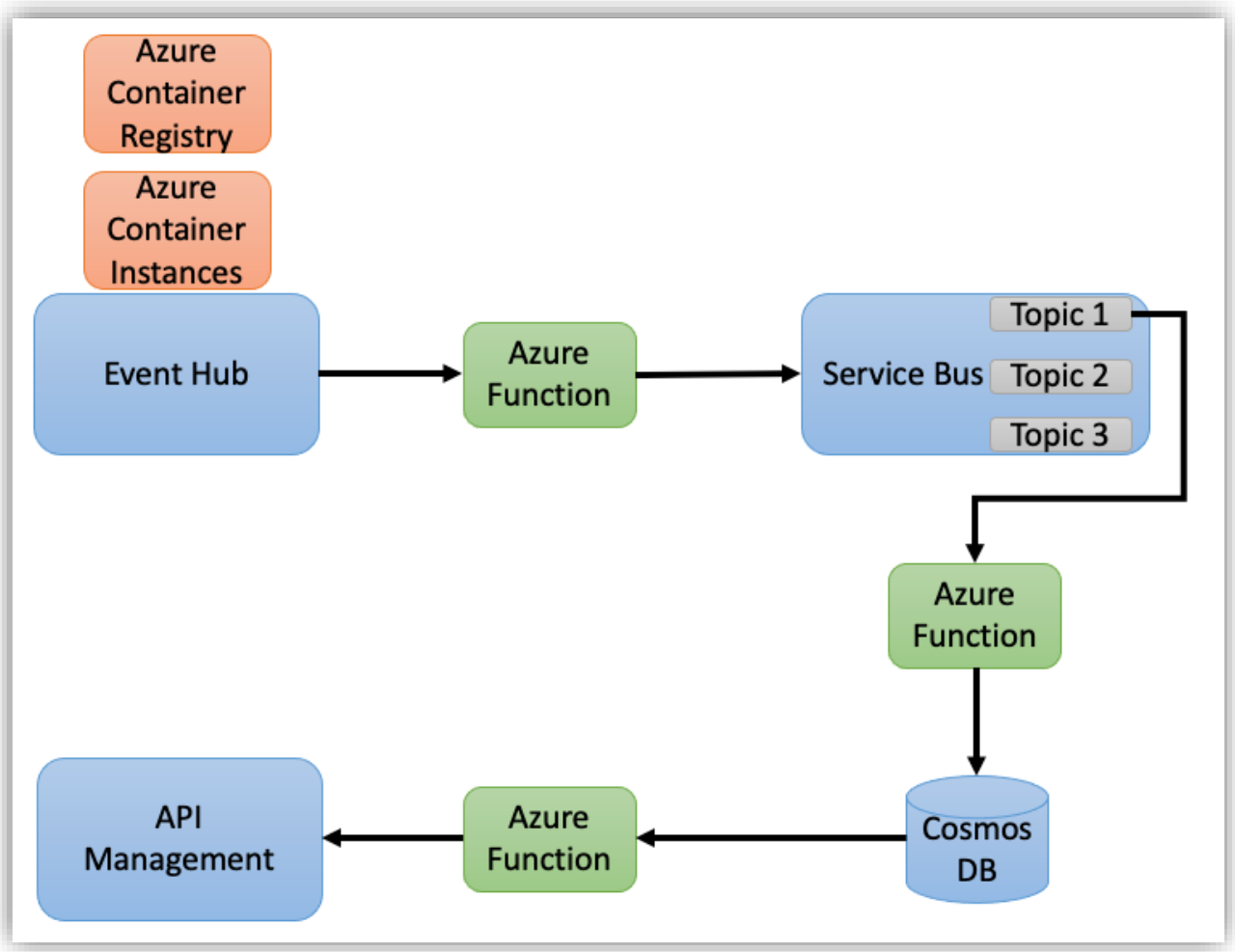
Chmurowisko

PwC Hackathon

Building Enterprise Serverless Lambda Architecture

LAB Overview

This lab introduces you to containers, Event Hub, Azure Functions, Service Bus, Cosmos DB and API Management functionality in Azure.



Task 1: Create an Event Hub namespace

In this section you will learn how to create an Event Hub namespace from Azure Portal.

1. On the left **Hub** menu click on **Create a resource.**
2. On the **New** blade click on **Internet of Things** and select **Event Hubs.**
3. On the **Create namespace** blade provide the following configurations:

    - **Name:** event-hub-hackathonz2X
    - **Pricing Tier:** Standard
    - **Enable Kafka:** None
    - **Make this namespace zero redundant:** None
    - **Subscription:** XXXXXX
    - **Resource group:** Your Resource Group
    - **Location:** West Europe
    - **Throughput Unit:** 1
    - **Enable Auto-inflate:** None

When, you finish click on button **Create.**

4. Next step the Azure Event Hub namespace will be deployed.

Task 2: Add Event Hubs to namespace

In this section you will learn how to add a new instance to namespace of created Azure Event Hub from previous task.

1. After successful deployment of Azure Event Hub, go to created service **event-hub-hackathonz2X**.
   You can search the service using field search **Search resources, services and docs** which is located at the top of Azure Portal.
2. On the **Event Hubs Namespace** page click on the **Event Hub** button marked with a plus sign.
3. On the **Create Event Hub** blade provide the following configurations:

    - **Name**: dataBroker
    - **Partition Count**: 2
    - **Message Retention**: 1
    - **Capture**: Off

When you finish, click on button **Create.**

4. Next step the Event Hub will be added to namespace.

Task 3: Add SAS Policy key to Event Hub

In this section you will learn how to add SAS Policy key to instance of Event Hub from Azure Portal.

1. On the main **Overview** page of **Event Hubs Namespace**, click on **Event Hubs** from the left menu.
2. On the **Event Hubs** page click on created instance of Event Hub – **dataBroker**.
3. On the instance of Event Hub – **dataBroker** page click on **Shared access policies** from the left menu.
4. On the **Shared access policies** page click on **Add** button.
5. On the **Add SAS Policy** blade provide the following configurations:

    - **Policy name:** sender-application
    - **Mange:** <leave-empty>
    - **Send:** Select
    - **Listen:** <leave-empty>

    At the end click on **Create.**
6. After creating SAS Policy key click on new created policy **sender-application** and copy to the clipboard value from **Connection string-primary key** or **Connection string-secondary key.** You will need this key at a later Task.

Task 4: Add consumer groups to Event Hub

In this section you will learn how to add consumer group to instance of Event Hub from Azure Portal.

1. On the main **Overview** page of **Event Hubs Namespace**, click on **Event Hubs** from the left menu.
2. On the **Event Hubs** page click on created instance of Event Hub - **dataBroker**.
3. On the instance of Event Hub – **dataBroker** page click on **Consumer groups** from the left menu.
4. On the **Consumer Group** page click on button **Consumer Group** marked with plus assign and add group named:

    - eventhub

Task 5: Create Docker Image

In this section you will learn how to create Docker Image based on NodeJS application which is simulated simple metrics to Azure Event Hub.

1. Go to https://github.com/cloudstateu/hackathon/blob/main/Zadanie2/ContainerApp.zip and download this file by clicking **Download**.
2. Unzip this file on your computer.
3. You will see folder with 3 files.
4. Open **index2.js** in some editor and replace in code **EVENTHUB_CONNECTION_STRING** value with value from **TASK 3 point 6.**

You should have this effect (it is an example):

```
const client = EventHubClient.createFromConnectionString("Endpoint=sb://event-hub-namespace-chmuromaniak2.servicebus.windows.net/;SharedAccessKeyName=sender-application;SharedAccessKey=xlz02dy7wu6UAUkAGNr0wXJ4ZzTakGmaZwhGK9pcmUo=;EntityPath=databroker", "databroker");
```

5. Open console in unzipped folder.
6. Build image running in command line: **docker build –t containerapp .**

Task 6: Create image repository service: ACR and push image.

In this section you will learn how to create Azure Container Registry and how to push image to it.

1. Open Azure Portal.
2. Click **Create Resource** on left menu.
3. Click **Containers** -> **Azure Container Registry.**
4. On the Create container registry page provide the following configurations:
   - **Registry name:** hackathonz2Xregistry
   - **Resource Group:**
     - **Use existing:** <Your Resource Group>
   - **Location:** West Europe
   - **Admin user:** Enabled
   - **SKU:** Standard
   - Next click on **Create.**
5. Go to deployed resource and on left menu click **Access keys.**
6. Copy username, password and login server.

   Make sure you open a Console not a PowerShell ☺ Otherwise it will not work!
7. Open console and type **docker login –p passwordCopiedFromPoint6 –u userNameCopiedFromPoint6 loginServerFromPoint6**

8. Tag image with repository by typing in console: **docker tag containerapp loginServerFromPoint6/containerapp**
9. Push image to repository by typing in console: **docker push loginServerFromPoint6/containerapp**
10. After push process is finished, go to portal to Azure Container Registry.
11. Open your ACR.
12. On left menu, in Services section click Repositories.
13. Check if your image is on the list.

Task 7: Create ACI

In this section you will learn how to create Azure Container Instances using image from repository which was made in previous task.

1. Open Azure Portal.
2. Click **Create Resource** on left menu.
3. Click **Containers** -> **Azure Container Instances.**
4. On the Create container instance page provide the following configurations:
   - **Resource Group:**
     - **Use existing:** <Your Resource Group>
   - **Container name:** continstances-X
   - **Location:** West Europe
   - **Image source:** Azure Container Registry
     - **Registry:** <Name of your Azure Container Registry from Task 6>
     - **Image:** containerapp
     - **Image tag:** latest
     - **OS type:** Linux
     - **Size:** 1 vcpu, 1.5 GiB memory, 0 gpus

   In **Networking** section:
   - **Networking type:** public
   - **DNS name label:** dnsname-X-container
   - Next click on **Review + create** and then **Create**

Task 8: Create Service Bus with 3 topics

In this section you will learn how to create Service Bus and 3 topics.

1. Open Azure Portal.
2. Click **Create Resource** on left menu.
3. Click **Integration → Service Bus.**
4. On the Create container instance page provide the following configurations:
   - **Resource Group:**
     - **Use existing:** <Your Resource Group>
   - **Namespace name:** service-X-bus
   - **Location:** West Europe
   - **Pricing Tier:** Standard
   - Next click on **Review + create** and then **Create**

5. After creating process is finished, go to your Service Bus page. You can do this by clicking on button **Go to resource.**
6. On left menu click **Topics** and then click **Topic** marked with plus assign.
7. On the Create topic window provide the following configurations:
   - **Name:** topic-X-1
   - **Max topic size:** 1 GB
   - **Message time to live:** 7 Days
   - **Disable auto-delete on idle topic**
   - **Disable duplicate detection**
   - **Enable portioning**
8. Create two more topics but change the name of topic (**topic-X-2** and **topic -X-3**)
9. For each topic you have to create subscription. To do this click **Topics** on the left menu and then choose one name of created topic.
   Then click **Subscription** marked with plus assign.
10. On the Create subscription page provide the following configurations:
    - **Name:** s1-x-sb
    - **Max delivery count:** 10
    - Click **Create.**

    Repeat this for other two topics and create for them subscription (**s2-x-sb** and **s3-x-sb**)

Task 9: Create AF between Event Hub and Service Bus

In this section you will learn how to create Azure Functions which will allow to connect Event Hub and Service Bus.

1. Open Azure Portal and Click **Create Resource** on left menu.
2. In Search the Marketplace write **Function App** then click **Create**.
3. On the Create Function App page provide the following configurations:
   - **Resource Group:**
     - **Use existing:** <Your Resource Group>
   - **Function App name:** function-x-app
   - **Publish:** Code
   - **Runtime stack:** .NET
   - **Version:** 3.1
   - **Region:** West Europe
   - Next click on **Review + Create** and then **Create.**
4. **Azure Event Hub trigger.**
5. On the Template details section provide the following configurations:
   - **New Function:** EventHubTrigger1
   - **Event Hub connection:** Click New and then choose **Event Hub** then choose your Event Hub namespace, then choose databroker and then sender-application. Press **OK.**
   - **Event Hub name:** databroker
   - **Consumer group:** eventhub
   - Click **Add.**
6. After the creation process is finished, go to your function and then click **Integration** on the left menu.
7. Click on **Trigger → Azure Event Hubs (events)**
8. On the Edit Trigger window check configuration:
   - **Binding Type:** Azure Event Hubs
   - **Name of event parameter:** events
   - **Event Hub connection:** <choose this with sender-application>
   - **Event Hub name:** databroker
   - **Consumer group:** eventhub
   - Click **Save**
9. Click on **Outputs → Add output**
10. On the Create output window check configuration:
    - **Binding Type:** Azure Service Bus
    - **Message type:** Service Bus Topic
    - **Service Bus Connection:** <choose this: service-x-bus-RootManageShared…>
    - **Name of event parameter:** outXputSbMsg1
    - **Topic name:** topic-x-1
    - Click **OK.**
    - Repeat this for other two topics. As the result you will have 3 outputs.

11. Go to **Code + Test** on the left menu and paste code copied from file on github:

    https://github.com/cloudstateu/hackathon/blob/main/Zadanie2/EventHubTrigger.cs

12. Click **Save.** You will see new window **Logs** with communicate: Connected! and other logs showing data came to Event Hub. This data is sending to Service Bus topics.

Task 10: Create Azure Cosmos DB

In this section you will learn how to create Azure DB Account and database.

1. Open Azure Portal.
2. Click **Create Resource** on left menu.
3. Click **Databases → Azure Cosmos DB.**
4. On the Create Azure Cosmos DB Account page provide the following configurations:
   - **Resource Group:**
     - **Use existing:** <Your Resource Group>
   - **Instance name:** cosmos-x-db
   - **API:** Core (SQL)
   - **Notebooks (Preview):** Off
   - **Location:** West Europe
   - **Capacity mode: Serverless (preview)**
   - **Account Type:** Non-Production
   - **Availability Zones:** Disable
   - Next click on **Review + create** and then **Create**

5. After the creation process is finished, go to your Azure Cosmos DB account. You can do this by clicking on button **Go to resource.**
6. On left menu click **Data Explorer** and then click **New Container.**
7. On the Add Container window provide the following configurations:
   - **Datebase id:**
     - **Create new:** databaseid
   - **Container id:** Containerid
   - **Partition key:** /chmura

Task 11: Create Azure Functions for topic in Service Bus

In this section you will learn how to create Azure functions for topics in Service Bus and also save data to Cosmos DB.

1. On the Function App (created in task 9) page click **Functions** on the left menu and then click **Add** marked with plus assign.
2. In **Select a template** chooses **Azure Service Bus Topic trigger.**
3. On the Template details section provide the following configurations:
   - **New Function:** ServiceBusTopicTrigger1
   - **Service Bus connection:** Click New and then choose **Service Bus** then choose your Service Bus, then choose RootManageSharedAccessKey. Press **OK.**
   - **Topic name:** topic-X-1
   - **Subscription name:** s1-x-sb
   - Click **Add.**
   - click **Integration** on the left menu.
   - Click on **Trigger → Azure Event Hubs (events)**
4. Click **Integration** on the left menu.
5. Click on **Trigger → Azure Service Bus** and then set **parameter name** → mySbMsg
6. Click **Save**
7. Click on **Outputs → Add output**
8. On the Create output window check configuration:
   - **Binding Type:** Azure Cosmos DB
   - **Name of event parameter:** outputDocument
   - **Database name:** databaseid
   - **Collection name:** Containerid
   - **Cosmos DB connection** → Click **New** → Azure Cosmos DB Account → choose your name of Cosmos DB → Click **OK**
   - Click **OK.**

9. Go to **Code + Test** on the left menu and paste code copied from file on github:

   https://github.com/cloudstateu/hackathon/tree/main/Zadanie2/ServiceBusTopicTriggerTrigger.cs

10. Click **Save.**

Task 12: Create Azure Functions for reading data from Cosmos DB

In this section you will learn how to create Azure functions (Rest API) for reading data from Cosmos DB.

1. Create new Function App (Task 9 → point 1, 2, 3) but in **Function App name** write: function-app-x-db
2. Download functionapp.zip from https://github.com/cloudstateu/hackathon/tree/main/Zadanie2/functionapp.zip
   You have to press **Download** button.
3. Then open Terminal on your computer and write this command:

   az login

   Now you have to login to your Microsoft Account. After that write this command:

   az functionapp deployment source config-zip -g <YOUR_RESOURCE_GROUP_NAME> -n \ <YOUR_FUNCTION_APP_NAME in point 1> --src <YOUR_PATH_TO_ZIP_FILE in point 2>

   Press **Enter** and wait a moment.
4. Then go to Azure Portal → go to your Function App created in point 1 → choose **Configuration**
5. Click **New connection string** marked with plus assign.
6. On the Add/Edit connection string window provide the following configurations:
   - **Name:** cosmodb_conn_string
   - **Value:** Open new tab in your browser with Azure Portal. Then go to your Cosmos DB account → click **Keys** on the left menu → copy PRIMARY CONNECTION STRING and go to previous tab in your browser → paste it to Value.
   - **Type:** MySQL
     Press **OK**
7. Click **Save** and then **Continue.**

Task 13: Create API Management

In this section you will learn how to create API Management and test your function.

1. Open Azure Portal.
2. Click **Create Resource** on left menu.
3. Click **Web → API Management.**
4. On the Create API Management page provide the following configurations:
   - **Subscription:** <choose your only subscription>
   - **Resource Group:**
     - **Use existing:** <Your Resource Group>
   - **Region:** West Europe
   - **Resource name:** apihackX
   - **Organization name:** hackathon
   - **Administrator name:** <write your mail>
   - **Pricing tier:** Consumption
   - Next click on **Review + create** and then **Create**
5. Creating process may take a while.
6. Go to crated API Management and click **APIs** on the left menu.
7. Press **Function App.**
8. On the Create from Function App window click **Browse.** Then click **Configure required settings** and then choose your Function App name created in Task 12. Click **Select** and then click **Select** again.
9. You will come back to Create from Function App window with filled fields. Click **Create.**
10. Now you are going to check your API. Click **APIs** on the left menu → in All APIs click **name of your function app** → click **Test** → click **GET** → copy **Request URL** and open new tab in your browser and paste URL that you copied. You should see your data from your Cosmos DB.