Discovery Days Hack – Student's Guide

Deploying and Operating a Microservices Application on Azure

Challenge Set 0: Pre-requisites - Ready, Set, GO!

Challenges:

- Make sure that you have joined the Teams group for this hack. The first person on your team at your table should create a new channel in this team with your team name.
- Login to the Azure Portal
- In a separate window, start the Azure Cloud Shell and list your subscriptions with the Azure CLI
    - Pick either Powershell or Bash based on your preference!
    - We will be running the Azure CLI, if you want to run it on your local machine just make sure you have the latest version.
- We are going to need the Subscription ID (guid format) of the subscription we will be using.
- **Tip**: We will be using the id many times in this hack. You may want to set a variable to hold it and use it in your future shell commands.
- **Tip:** There are 2 different ways to open the Azure Cloud Shell!

Challenge Set 1: First Thing is First: A Resource Group

Challenges:

- Figure out which Azure data center is closest to you. You will be using it for this hack.
- Put the “scripting name” for the Azure Data Center that the Azure CLI uses in a variable called **loc**
    - **Hint**: South Central US scripting name is ***southcentralus***
- In your shell, create a Resource Group in that data center
- **Tip:** Create a 6 to 8 letter label that is unique to you that can be used to compose names for Azure resources, since several items we are creating today will have public DNS names that must be globally unique.
- **Tip**: Again, you might want to put the resource group name in a shell variable.

Challenge Set 2: Always need App Insights

Challenges:

- In your shell, deploy a public ARM Template to deploy an App Insights resource to have the various applications log to
    - **ARM Template URL:** **aka.ms/wth-microservices-template**
    - **Two Parameters:**
        - **name:** Name of the App Insights Resource to Create
        - **regionId:** Scripting Name of the region to Provision in
- Put the InstrumentationKey of the App Insights resource you just created in a variable called ***appInsightsKey***

Challenge Set 3: Get some Data

Challenges:

- In your shell, create a CosmosDB Account
    - **NOTE**: When creating large resources, be patient for it to end.
- Copy the PrimaryMasterKey of the CosmosDB Account you just created as we will be needing it later
    - **Tip**: Once again, think about using a shell variable for this.

Challenge Set 4: Deploy some containers to ACI

Challenges:

- There are three different containers you are going to deploy as Azure Container Instances (ie: az container)
    - Data API**: microservicesdiscovery/travel-data-service**
    - Itinerary API**: microservicesdiscovery/travel-itinerary-service**
    - DataLoader utility (to setup and seed Cosmos DB)**: microservicesdiscovery/travel-dataloader**
- All of these containers will need 3 environment variables defined:
    - **DataAccountName**: Name of the Cosmos DB Account
    - **DataAccountPassword**: Primary Key of the Cosmos DB Account
    - **ApplicationInsights__InstrumentationKey**: Instrumentation Key of the App Insights Resource
- The Data API and Itinerary API containers, both need to specify a DNS Name, so they are easily addressable by the Web Site.
- After you get the Data API deployed, use the Azure CLI to query the deployed container for its full qualified domain name (FQDN) and store it in a variable called: **dataServiceUri**
- After you get the Itinerary API deployed, use the Azure CLI to query the deployed container for the FQDN and store it in a variable called: **itineraryServiceUri**
- Verify the Data Service by browsing the URL: **http://$dataServiceUri/api/airport** it will return a list of Airports
- Verify the Itinerary Service by browsing the URL: **http://$itineraryServiceUri/api/itinerary/AAA** it will return a 204.
    - **NOTE:** It should return a 404, but to get a positive test response, it returns a 204 (which represents “no content”) to verify the service is up and running.
- After you get the DataLoader deployed, you can see what it actually did by viewing its logs.
    - Use the Azure CLI to print out the logs for the Data Loader container.
    - Try this with the other containers as well just for fun.

Challenge Set 5: Deploy the Web Site

Challenges:

- In your shell, create a Standard Linux App Service Plan
- In your shell, create a Web App and set the **microservicesdiscovery/travel-web** as the container image for the Web App
- The following Application Settings need to be added:
    - **DataAccountName**: Name of the Cosmos DB Account
    - **DataAccountPassword**: Primary Key of the Cosmos DB Account
    - **ApplicationInsights__InstrumentationKey**: Instrumentation Key of the App Insights Resource
    - **DataServiceUrl:** The URL to the Data Service, only over HTTP
    - **ItineraryServiceUrl:** The URL to the Itinerary Service, only over HTTP
- Verify that you can browse to the URL of the App Service