Module 7: Mini Project-1 Part-A

Patient Survival Prediction using XGBoost

Deployment with AWS ECR and ECS

For this project, you will train a XGBoost model to predict survival of patients with heart failure. After that, you will save the trained model, create a Gradio-based app, build & push docker image to ECR, and deploy the application with ECS. Please refer to the *Demo session - ECR ECS* held on May 4th for this mini-project.

Step 1: Go through the mini-project notebook *M7_NB_MiniProject_1 Patient_Survival_Prediction_using_XGBoost* shared along with this document and understand the problem statement. [1 point]

1.1 Perform model training
1.2 Save the trained model on your system

Step 2: Create Gradio-based application for Patient Survival Prediction within Colab notebook [2 points]

2.1 Load the trained model
2.2 Create Gradio application based on instructions given in the notebook
2.3 Create '*requirements.txt*', and '*app.py*' files on your system

Step 3: Dockerize the Gradio application on Cloud9 [1 point]

3.1 Access your AWS Management Console from LMS - as per steps given at last of this document
3.2 Create a new environment on Cloud9
3.3 Upload the trained model, '*requirements.txt*', and '*app.py*' files on Cloud9 environment
3.4 Create a Dockerfile to dockerize the application
3.5 Build a docker image for survival prediction

- Make sure the image size is below 3GB, if you are using AWS account provided by TS

Step 4: Run docker container on Cloud9 and access the application [1 point]

4.1 Run a new container using the docker image created above

4.2 Access the application via publicIP of EC2 instance associated with the Cloud9 environment

- Make sure the Security Group attached with the EC2 instance has the application port (eg.8001) open for inbound traffic

4.3 Debug and re-create image if the application is giving error

Step 5: Push docker image to ECR [2 points]

5.1 Create a new Private Repository on ECR

5.2 Push the docker image present on Cloud9 environment to your ECR repository

Step 6: Deploy application with ECS [2 points]

6.1 Create a new Cluster (Ignore any namespace warning showing up during the cluster creation)

6.2 Create a Task Definition

- For a docker image of size ~2GB, use 4GB, 1vCPU as memory and cpu requirements respectively

6.3 Create a Service (Ignore any autoscaling warning showing up during the service creation)

6.4 Once the service is deployed, access the application via publicIP associated with the Task running in your service

Step 7: CAUTION ! Clean up [1 point]

7.1 Delete/Terminate/Release the resources created on AWS for this mini-project

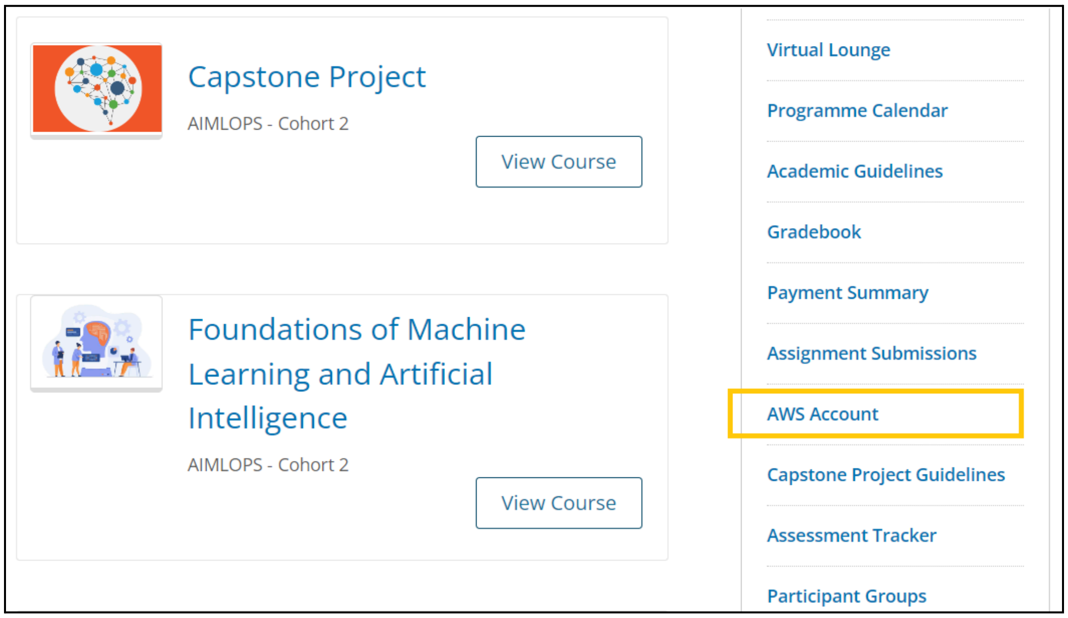
- Cloud9
  - Environment
  - EC2 instance associated
- ECR
  - Images
  - Repository
- ECS
  - Service
  - Cluster
  - Task Definition

Access your AWS Management Console from LMS

Once the AWS accounts are activated, you can access your AWS console through the Quicklink available on LMS.
Follow below steps to access the console:

1. Login to LMS with your credentials.

2. Click on the *AWS Account* quick link available on LMS.



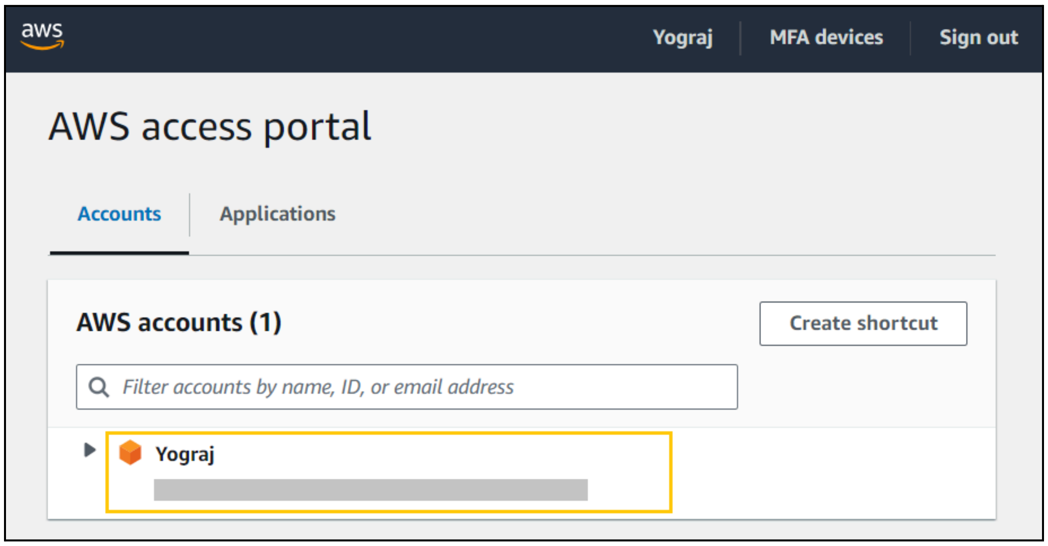
Capstone Project

AIMLOPS - Cohort 2

Foundations of Machine Learning and Artificial Intelligence

AIMLOPS - Cohort 2

3. You will be directed to the page similar to below. Select your Name.



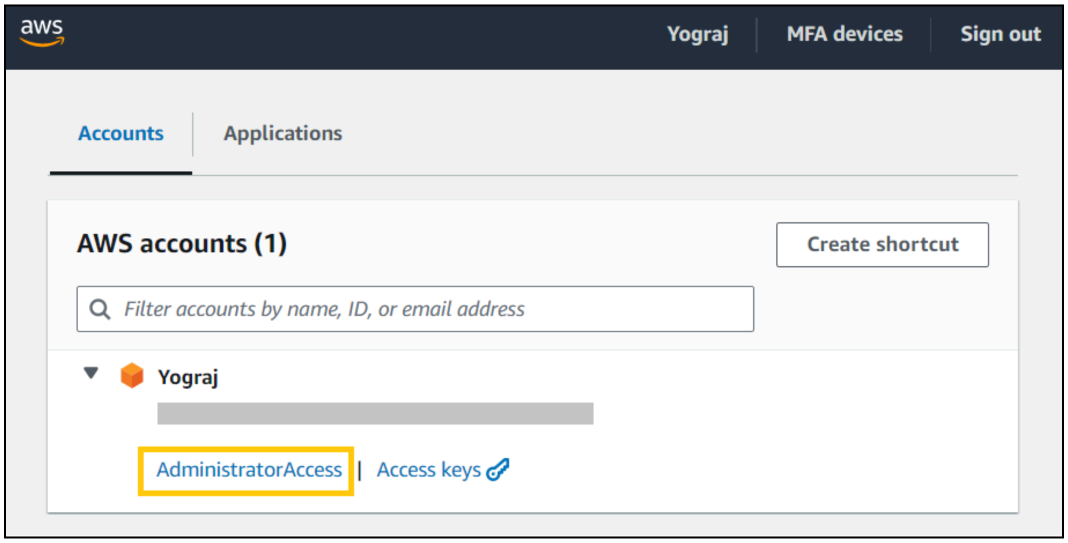
AWS access portal

AWS accounts (1)

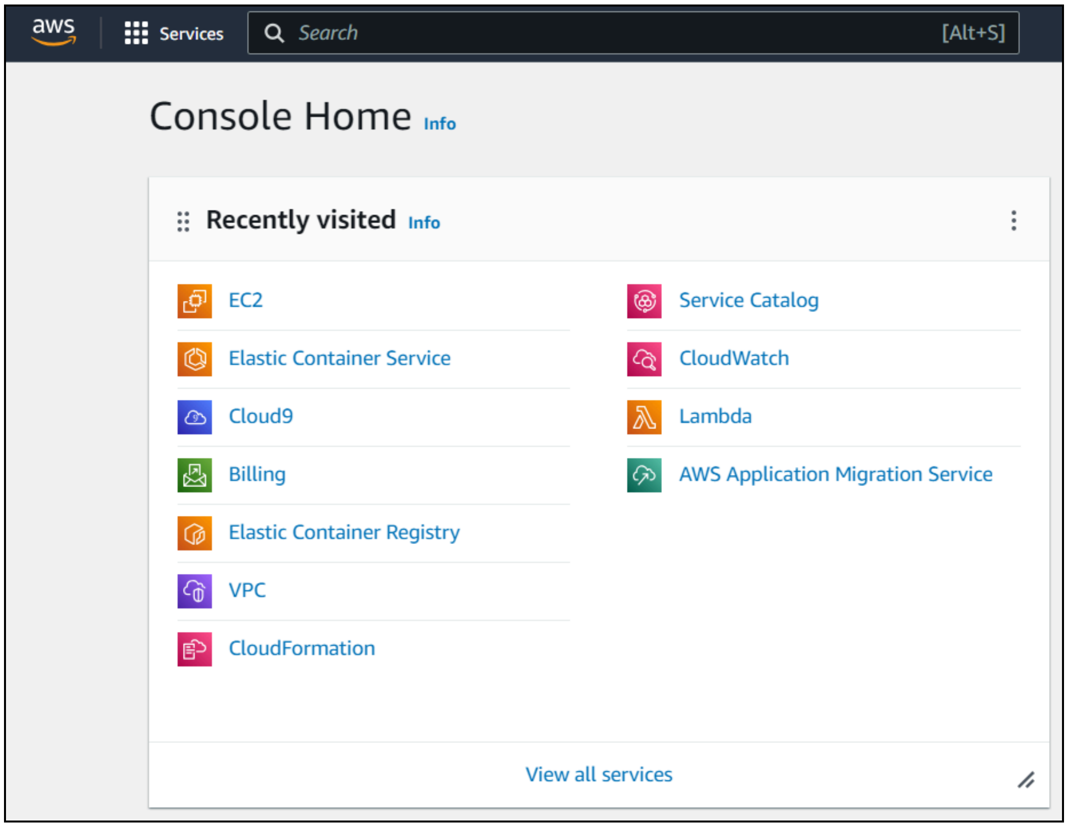
Yograj

[redacted]

4. Select ***AdministratorAccess.***



5. You will be redirected to your AWS Management Console.



6. Regarding usage details, go through the AWS Account Guidelines shared along with this document.