Raport Lab 2: Azure ML API (Docker, ACR, ECS)

Mouad GUEDAD

Year : 2023/2024 Major : QFM

Instructor:
Pr. Fahd KALLOUBI

Introduction

In this extensive exploration within the laboratory setting, we undertake a journey to confront a practical challenge faced by a shipping company situated in the vibrant port of Turku, Finland. As data scientists, our goal is to harness the formidable combination of Azure DevOps, MLflow, and the Azure Machine Learning SDK to craft an advanced solution. The issue at hand revolves around the complex interaction between weather conditions and logistics in the port environment, where 90 percent of Finland's imported goods traverse via cargo ships.

To address this challenge, we strategically utilize Microsoft Azure, a leading cloud service, and MLflow, an open-source tool. This fusion of cloud technology and open-source innovation underscores our dedication to exploring synergies across diverse platforms. Throughout this lab, we delve into the intricacies of integrating open-source tools with robust cloud services, highlighting the potential for a seamless integration that could reshape the landscape of predictive analytics for the maritime industry.

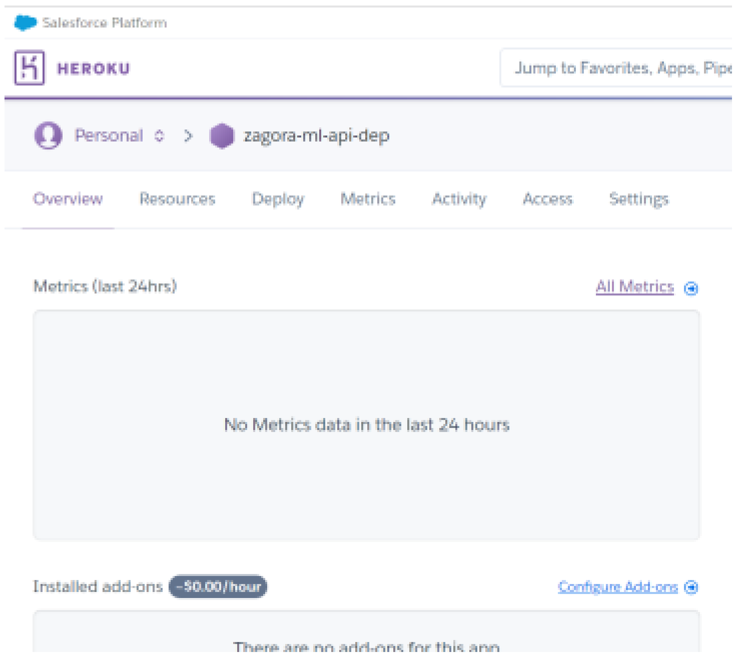
Designing the API and deploying it as a service:

Heroku login and app deployment :

Heroku is a cloud platform that streamlines the deployment, management, and scaling processes for web applications. Following its acquisition by Salesforce, Heroku enables developers to concentrate on constructing their applications without being burdened by the intricacies of the underlying infrastructure. The platform supports multiple programming languages, including Ruby, Python, Java, Node.js, and more, catering to diverse development requirements. Utilizing Heroku, developers can effortlessly deploy their applications using a straightforward "git push" command, while the platform autonomously manages tasks such as provisioning servers, load balancing, and scaling resources based on demand.

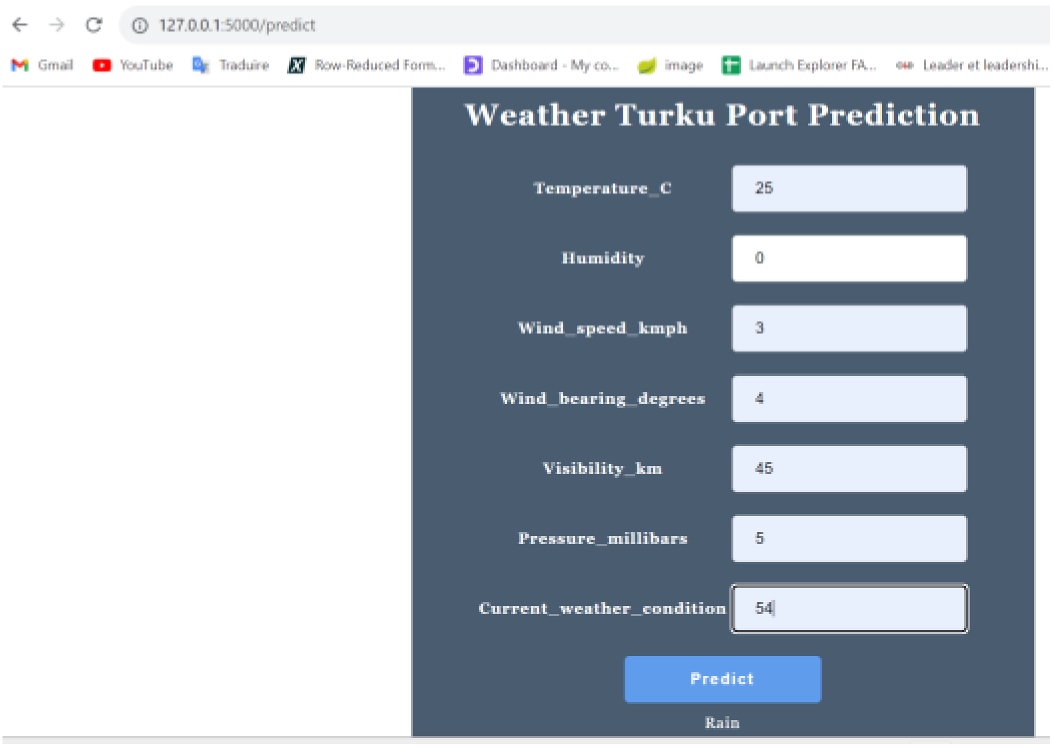
Heroku Login:

After successfully testing our app locally and confirming its functionality, the next step is to log in to Heroku. This allows us to make use of the app later in the deployment process. As illustrated above (referencing a screenshot from the Heroku platform), we have successfully created an app that will be utilized in subsequent steps.



App Consumption:

Similar to Flask, FastAPI also employs the Jinja2 template engine. Consequently, we will develop an application using FastAPI to interact with our service via a user-friendly HTML interface. In the subsequent steps, we have utilized the app through the API, leveraging its capabilities for predictive purposes.



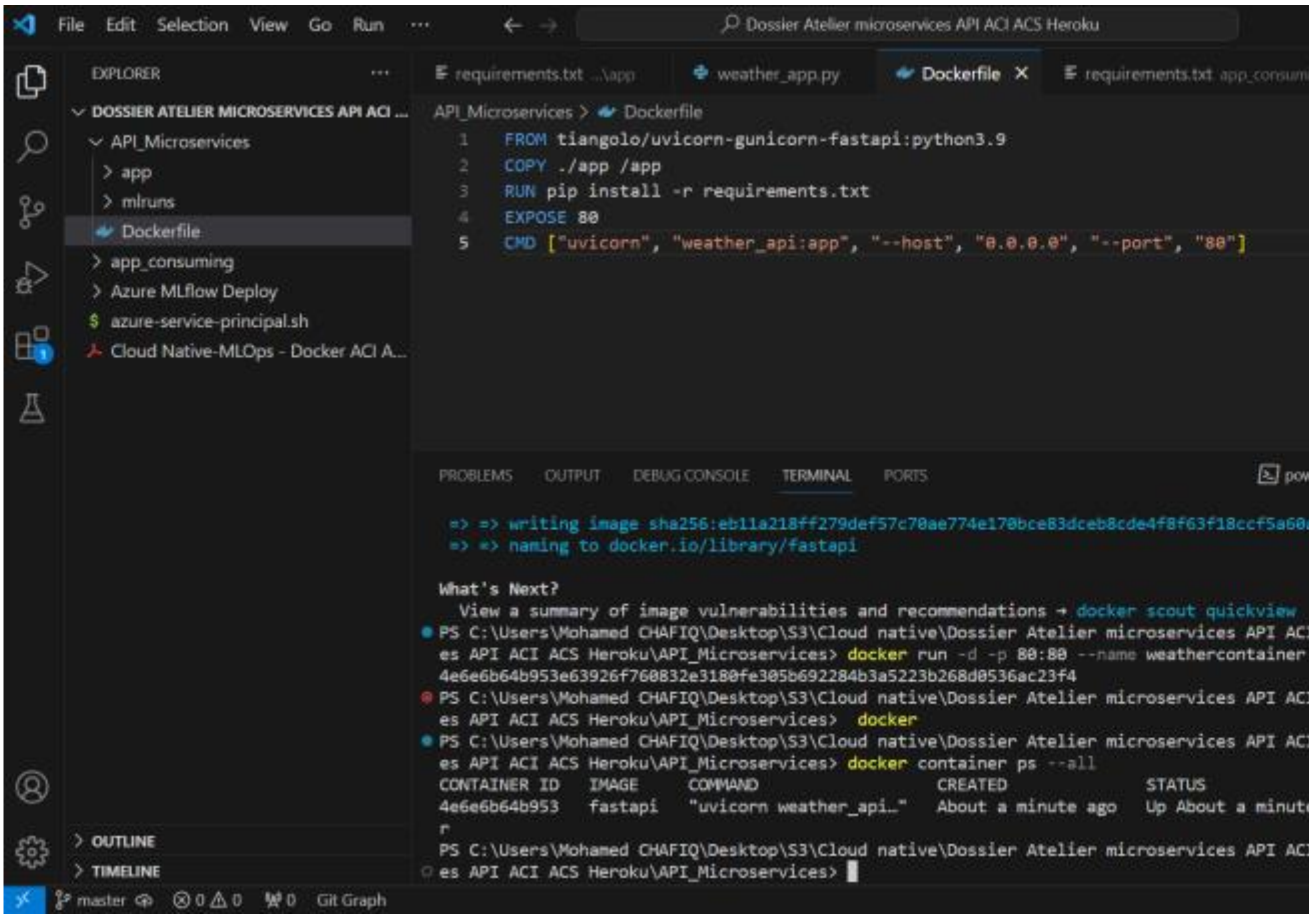
Weather Turku Port Prediction

Temperature_C	25
Humidity	0
Wind_speed_kmph	3
Wind_bearing_degrees	4
Visibility_km	45
Pressure_millibars	5
Current_weather_condition	54

Rain

Deploying ML API as Microservice:

In this section, titled "Deploy ML API as a microservice," we delve into the deployment process of Machine Learning APIs as microservices. This involves exploring the steps and considerations necessary to transform machine learning models into scalable, accessible, and independent microservices, allowing seamless integration into diverse applications and systems. Furthermore, the section addresses the standardization of FastAPI service packaging using Docker. This approach empowers us to deploy the Docker image or container on the deployment target of our choice, thereby improving the overall efficiency and portability of the deployed ML API microservice.


File Edit Selection View Go Run ···
Dossier Atelier microservices API ACI ACS Heroku
EXPLORER
DOSSIER ATELIER MICROSERVICES API ACI ...
API_Microservices
app
mlruns
Dockerfile
app_consuming
Azure MLflow Deploy
azure-service-principal.sh
Cloud Native-MLOps - Docker ACI A...
OUTLINE
TIMELINE
requirements.txt ...\app
weather_app.py
Dockerfile
requirements.txt app_consum
API_Microservices > Dockerfile
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
master ⊗ 0 ⚠ 0 ⚡0 Git Graph

```
FROM tiangolo/uvicorn-gunicorn-fastapi:python3.9
COPY ./app /app
RUN pip install -r requirements.txt
EXPOSE 80
CMD ["uvicorn", "weather_api:app", "--host", "0.0.0.0", "--port", "80"]
```

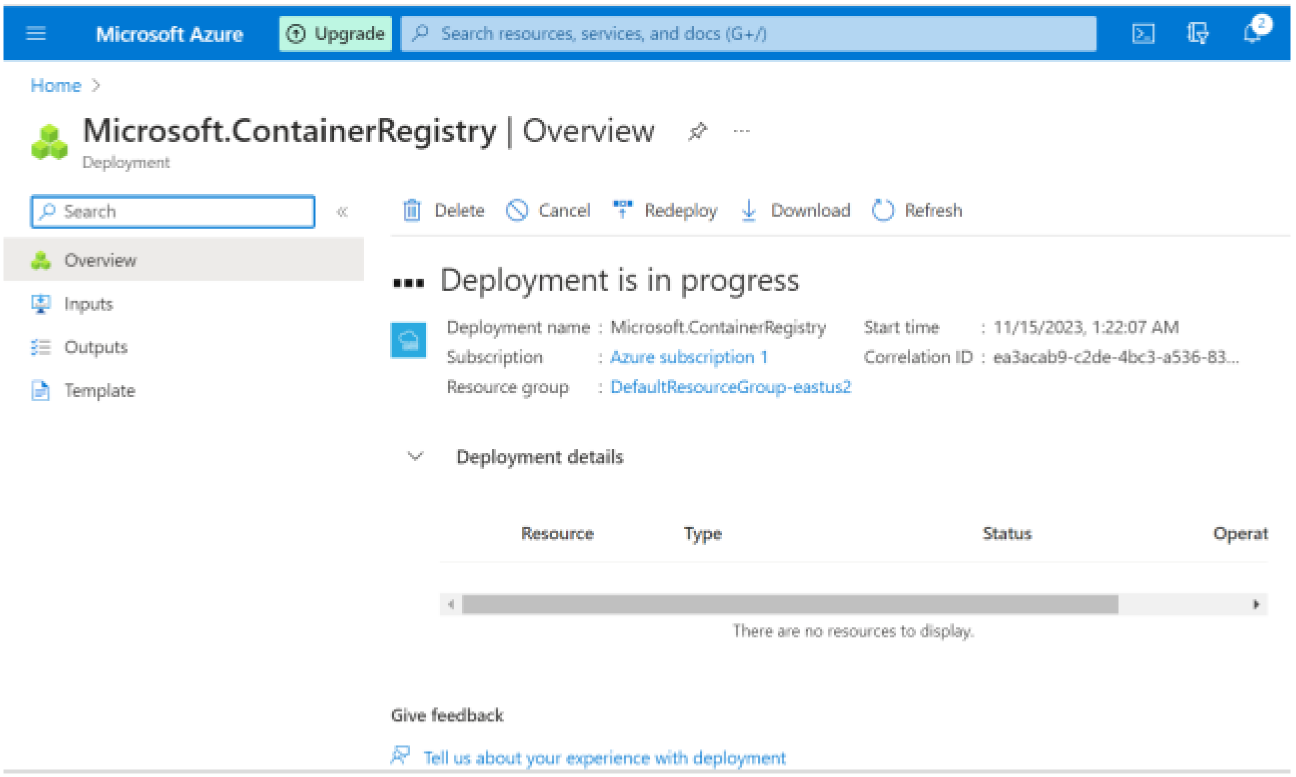
```
=> => writing image sha256:eb11a218ff279def57c70ae774e170bce83dceb8cde4f8f63f18ccf5a60
=> => naming to docker.io/library/fastapi

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\Mohamed CHAFIQ\Desktop\S3\Cloud native\Dossier Atelier microservices API AC
es API ACI ACS Heroku\API_Microservices> docker run -d -p 80:80 --name weathercontainer
4e6e6b64b953e63926f760832e3180fe305b692284b3a5223b268d0536ac23f4
PS C:\Users\Mohamed CHAFIQ\Desktop\S3\Cloud native\Dossier Atelier microservices API AC
es API ACI ACS Heroku\API_Microservices>  docker
PS C:\Users\Mohamed CHAFIQ\Desktop\S3\Cloud native\Dossier Atelier microservices API AC
es API ACI ACS Heroku\API_Microservices> docker container ps --all
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS
4e6e6b64b953   fastapi   "uvicorn weather_api…"   About a minute ago   Up About a minute
r
PS C:\Users\Mohamed CHAFIQ\Desktop\S3\Cloud native\Dossier Atelier microservices API AC
es API ACI ACS Heroku\API_Microservices>
```

Deploying Our Container on Azure:

In the domain of deploying applications in cloud environments, Azure Container Registry (ACR) stands out as a crucial tool for efficiently handling and distributing containerized applications. This section delves into the intricacies of deploying applications on Azure Registry, a fully managed Docker container registry service provided by Microsoft Azure. As we explore the deployment process, we'll navigate through the steps involved in pushing container images to Azure Registry, ensuring smooth integration with Azure services, and optimizing the overall performance and scalability of containerized applications within the Azure ecosystem. This introduction sets the stage for a comprehensive understanding of deploying and managing containerized solutions on Azure Registry.

ACR Manipulation

Creation of an Azure Container Registry instance :



Microsoft.ContainerRegistry | Overview

Deployment

Deployment is in progress

Deployment name : Microsoft.ContainerRegistry
Subscription : Azure subscription 1
Resource group : DefaultResourceGroup-eastus2
Start time : 11/15/2023, 1:22:07 AM
Correlation ID : ea3acab9-c2de-4bc3-a536-83...

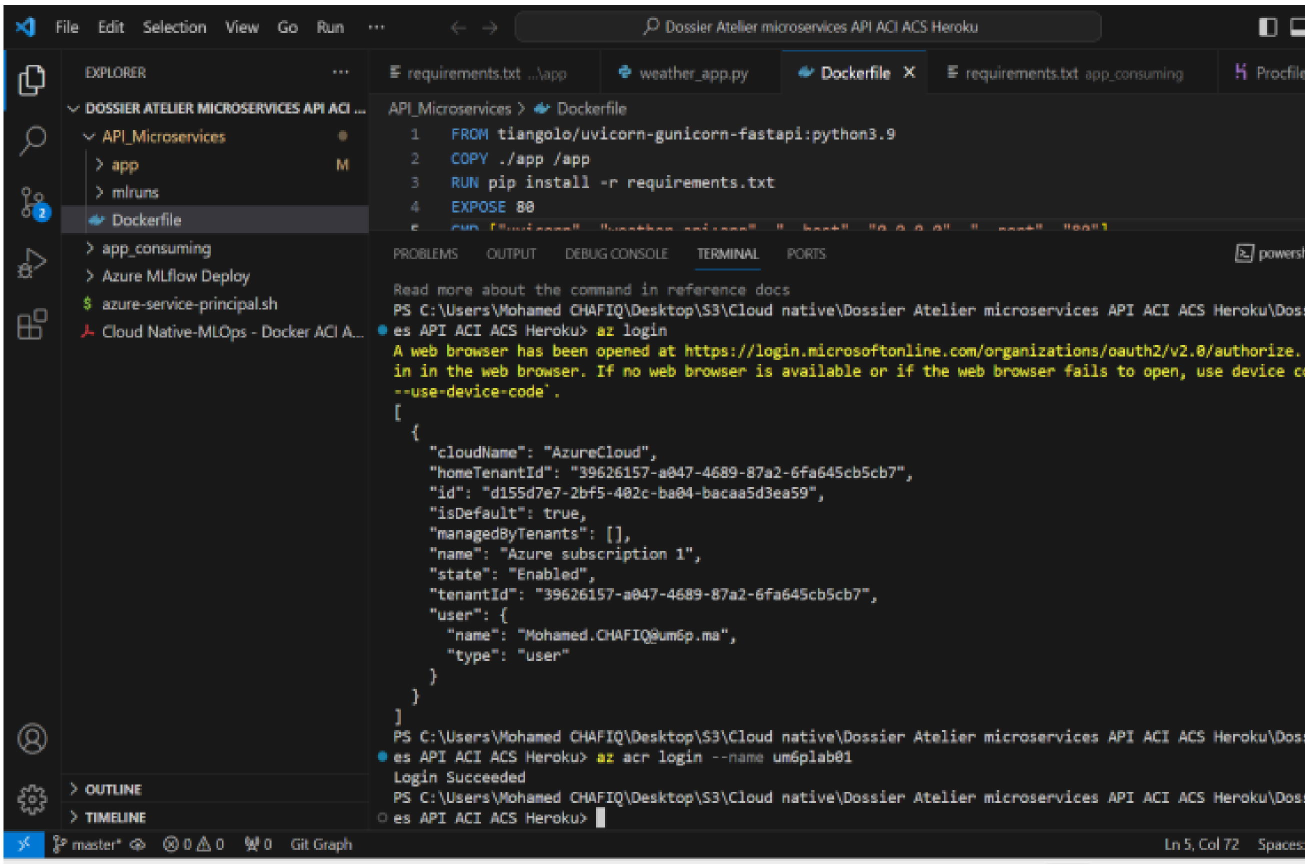
Deployment details

Resource	Type	Status	Operat

There are no resources to display.

Give feedback

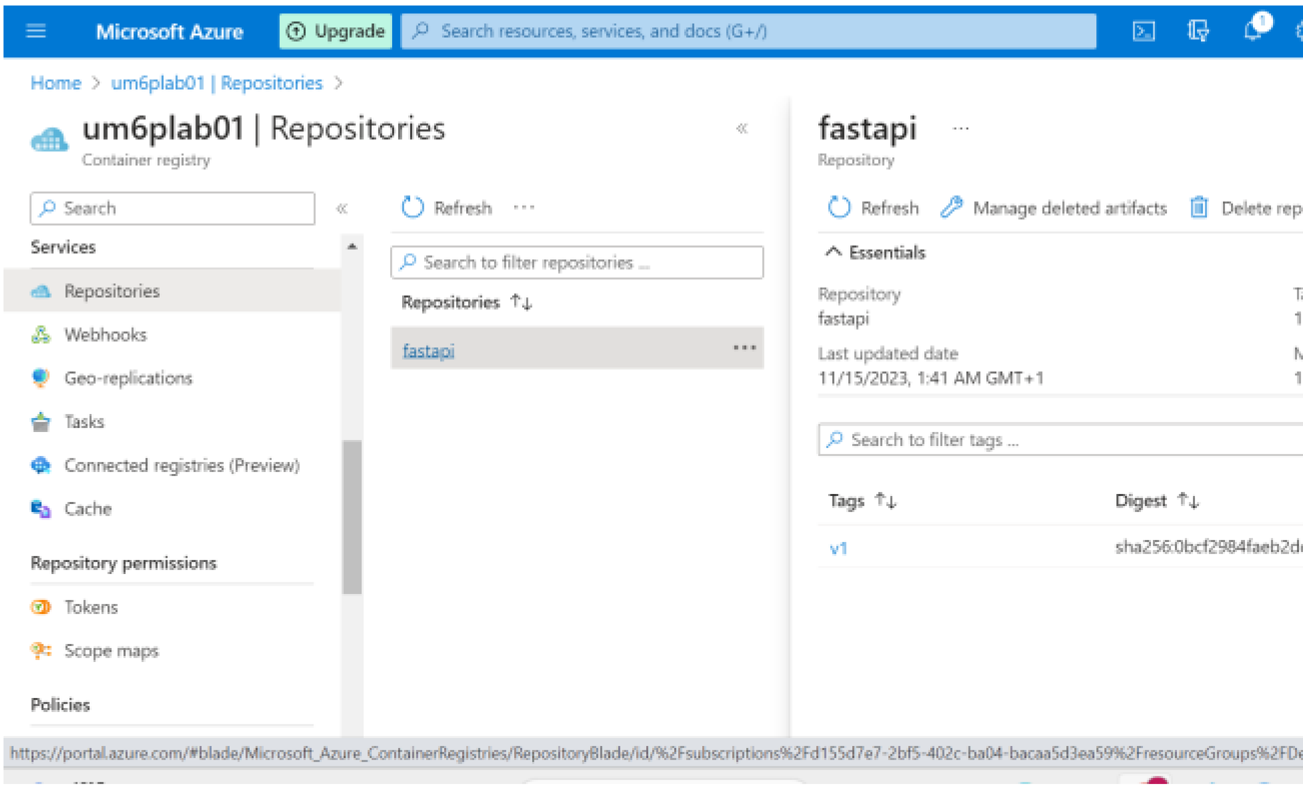
Tell us about your experience with deployment



```
FROM tiangolo/uvicorn-gunicorn-fastapi:python3.9
COPY ./app /app
RUN pip install -r requirements.txt
EXPOSE 80
```

```
Read more about the command in reference docs
PS C:\Users\Mohamed CHAFIQ\Desktop\S3\Cloud native\Dossier Atelier microservices API ACI ACS Heroku\Dos
es API ACI ACS Heroku> az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize.
in in the web browser. If no web browser is available or if the web browser fails to open, use device c
--use-device-code`.
[
  {
    "cloudName": "AzureCloud",
    "homeTenantId": "39626157-a047-4689-87a2-6fa645cb5cb7",
    "id": "d155d7e7-2bf5-402c-ba04-bacaa5d3ea59",
    "isDefault": true,
    "managedByTenants": [],
    "name": "Azure subscription 1",
    "state": "Enabled",
    "tenantId": "39626157-a047-4689-87a2-6fa645cb5cb7",
    "user": {
      "name": "Mohamed.CHAFIQ@um6p.ma",
      "type": "user"
    }
  }
]
PS C:\Users\Mohamed CHAFIQ\Desktop\S3\Cloud native\Dossier Atelier microservices API ACI ACS Heroku\Dos
es API ACI ACS Heroku> az acr login --name um6plab01
Login Succeeded
PS C:\Users\Mohamed CHAFIQ\Desktop\S3\Cloud native\Dossier Atelier microservices API ACI ACS Heroku\Dos
es API ACI ACS Heroku>
```

Tagging the image for ACR :


Microsoft Azure | Upgrade | Search resources, services, and docs (G+/)

Home > um6plab01 | Repositories >

um6plab01 | Repositories
Container registry

Search

Services
- Repositories
- Webhooks
- Geo-replications
- Tasks
- Connected registries (Preview)
- Cache

Repository permissions
- Tokens
- Scope maps

Policies

Refresh ···

Search to filter repositories ...

Repositories ↑↓
fastapi

fastapi ···
Repository

Refresh | Manage deleted artifacts | Delete rep

Essentials

Repository
fastapi

Last updated date
11/15/2023, 1:41 AM GMT+1

Search to filter tags ...

Tags ↑↓	Digest ↑↓
v1	sha256:0bcf2984faeb2d

https://portal.azure.com/#blade/Microsoft_Azure_ContainerRegistries/RepositoryBlade/id/%2Fsubscriptions%2Fd155d7e7-2bf5-402c-ba04-bacaa5d3ea59%2FresourceGroups%2FDe

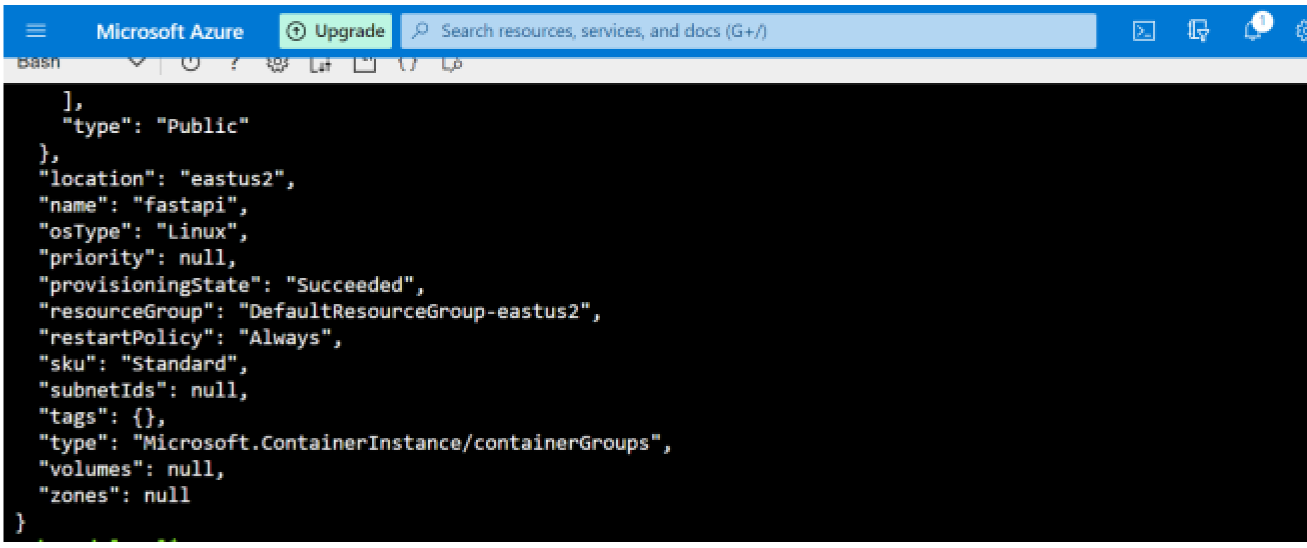
Container deployment :

Extracting Image from ACR to ACI:

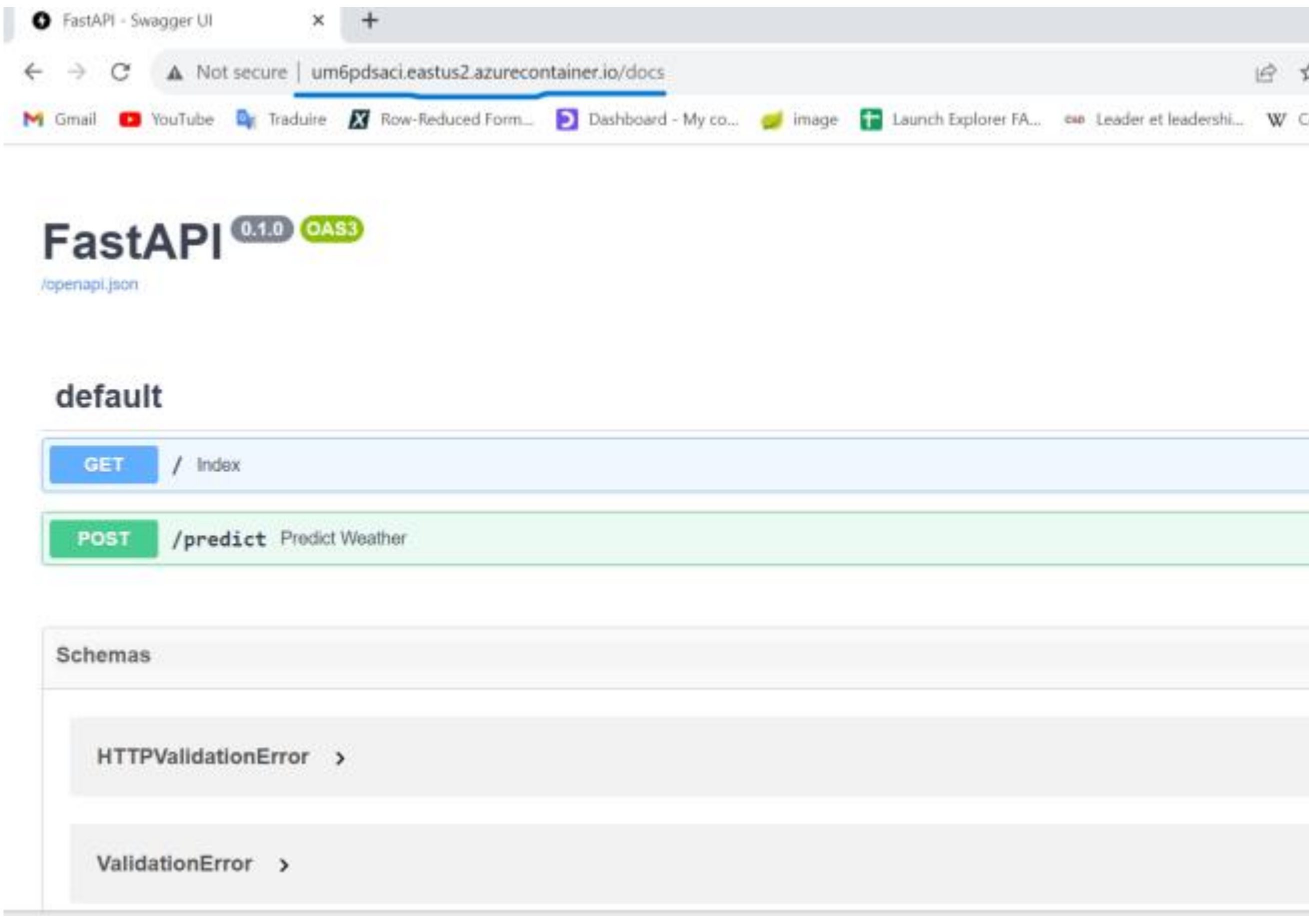
To streamline the extraction of our image from Azure Container Registry (ACR) to Azure Container Instances (ACI), it is essential to create and configure an Azure Active Directory (AD) service principal with the necessary extraction permissions on your registry. This service principal acts as the authentication mechanism for securely accessing your private registry.

Following the creation and configuration of the Azure AD service principal, the next step involves initiating a container in Azure Container Instances. This container pulls its image from your private registry, utilizing the previously established service principal for authentication. The process ensures a secure and authorized transfer of the container image from ACR to ACI.

The accompanying image visually represents the successful deployment of the container on ACI after the extraction process. This demonstrates the seamless integration and execution of the containerized solution within Azure Container Instances.



```
    ],
    "type": "Public"
  },
  "location": "eastus2",
  "name": "fastapi",
  "osType": "Linux",
  "priority": null,
  "provisioningState": "Succeeded",
  "resourceGroup": "DefaultResourceGroup-eastus2",
  "restartPolicy": "Always",
  "sku": "Standard",
  "subnetIds": null,
  "tags": {},
  "type": "Microsoft.ContainerInstance/containerGroups",
  "volumes": null,
  "zones": null
}
```



FastAPI 0.1.0 OAS3

/openapi.json

default

GET / Index

POST /predict Predict Weather

Schemas

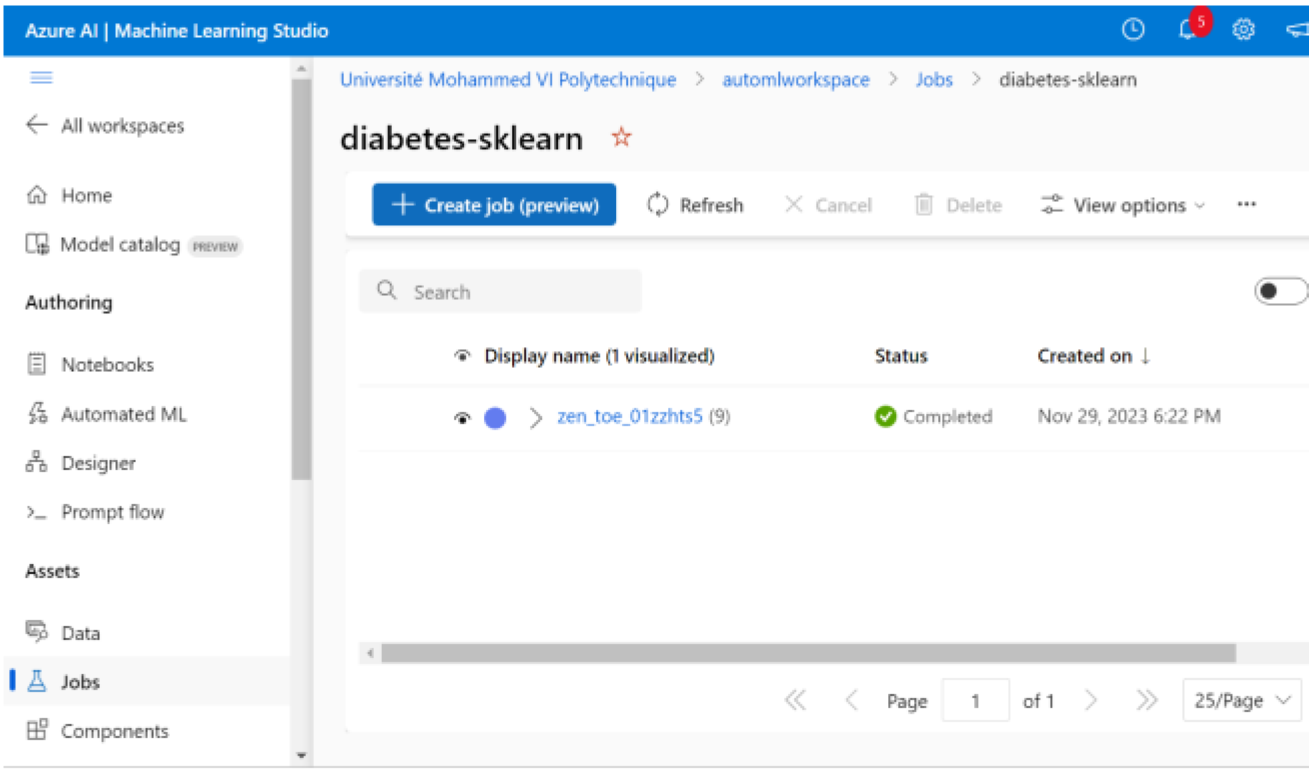
HTTPValidationError >

ValidationError >

Deploying an ML Model on (ACI) using Azure ML and MLflow:

Deploying a machine learning (ML) model on Azure Container Instances (ACI) using Azure ML and MLflow is a seamless process that harnesses the strengths of both platforms. Azure ML provides a comprehensive cloud-based solution for the entire ML model lifecycle, while MLflow simplifies experiment tracking and model packaging. The integration of Azure ML with MLflow facilitates model training, registration, and deployment on ACI.

The process begins with model training and experiment tracking in MLflow. Subsequently, the model is registered in Azure ML, enabling straightforward deployment onto ACI through Docker containers. Azure ML takes care of the underlying infrastructure, ensuring scalability and reliability. This integrated approach not only streamlines the deployment process but also allows for efficient monitoring and testing. Consequently, it proves to be a robust solution for operationalizing ML models in real-world scenarios on the Azure cloud

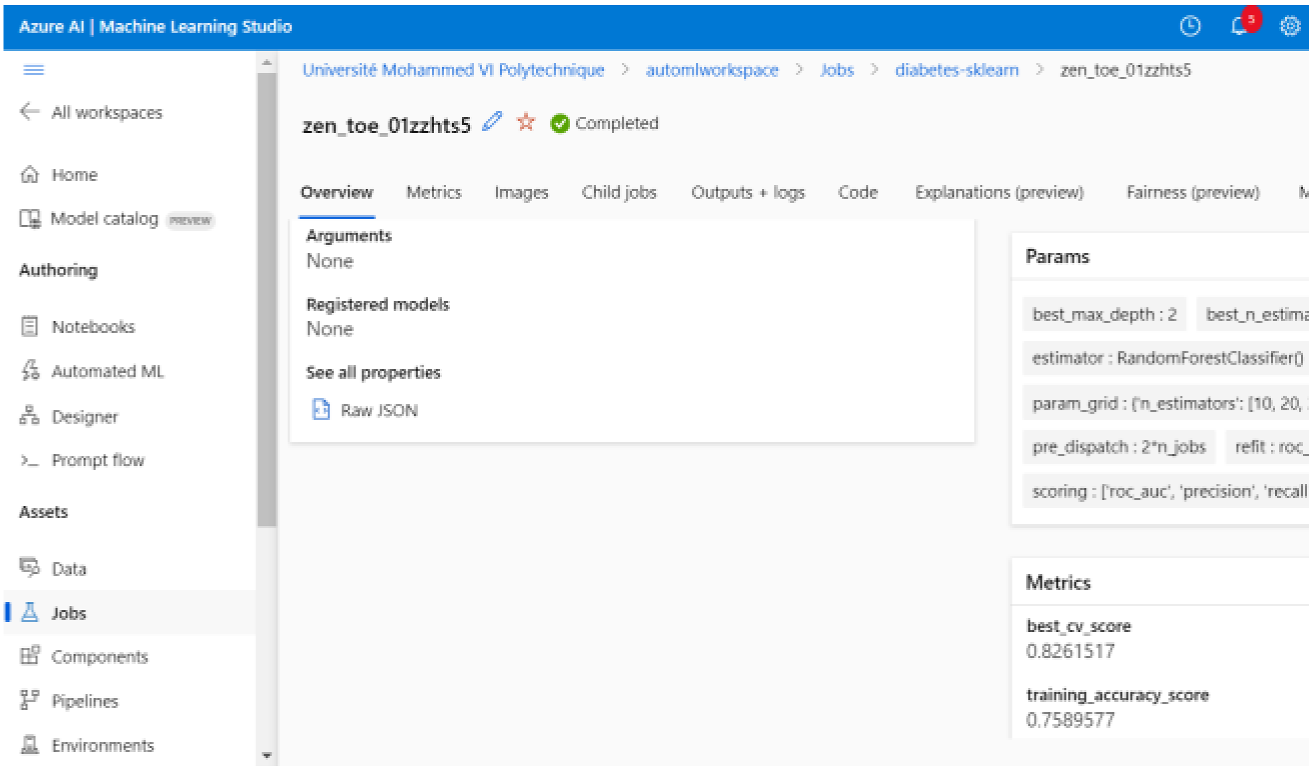


Université Mohammed VI Polytechnique > automlworkspace > Jobs > diabetes-sklearn

diabetes-sklearn

Display name (1 visualized)	Status	Created on ↓
zen_toe_01zzhts5 (9)	Completed	Nov 29, 2023 6:22 PM

As shown above, the MLflow experiment is created on azure and ready for further explorations.



Arguments
None

Registered models
None

See all properties

Raw JSON

Params

best_max_depth : 2
best_n_estima
estimator : RandomForestClassifier()
param_grid : {'n_estimators': [10, 20,
pre_dispatch : 2*n_jobs
refit : roc_
scoring : ['roc_auc', 'precision', 'recall

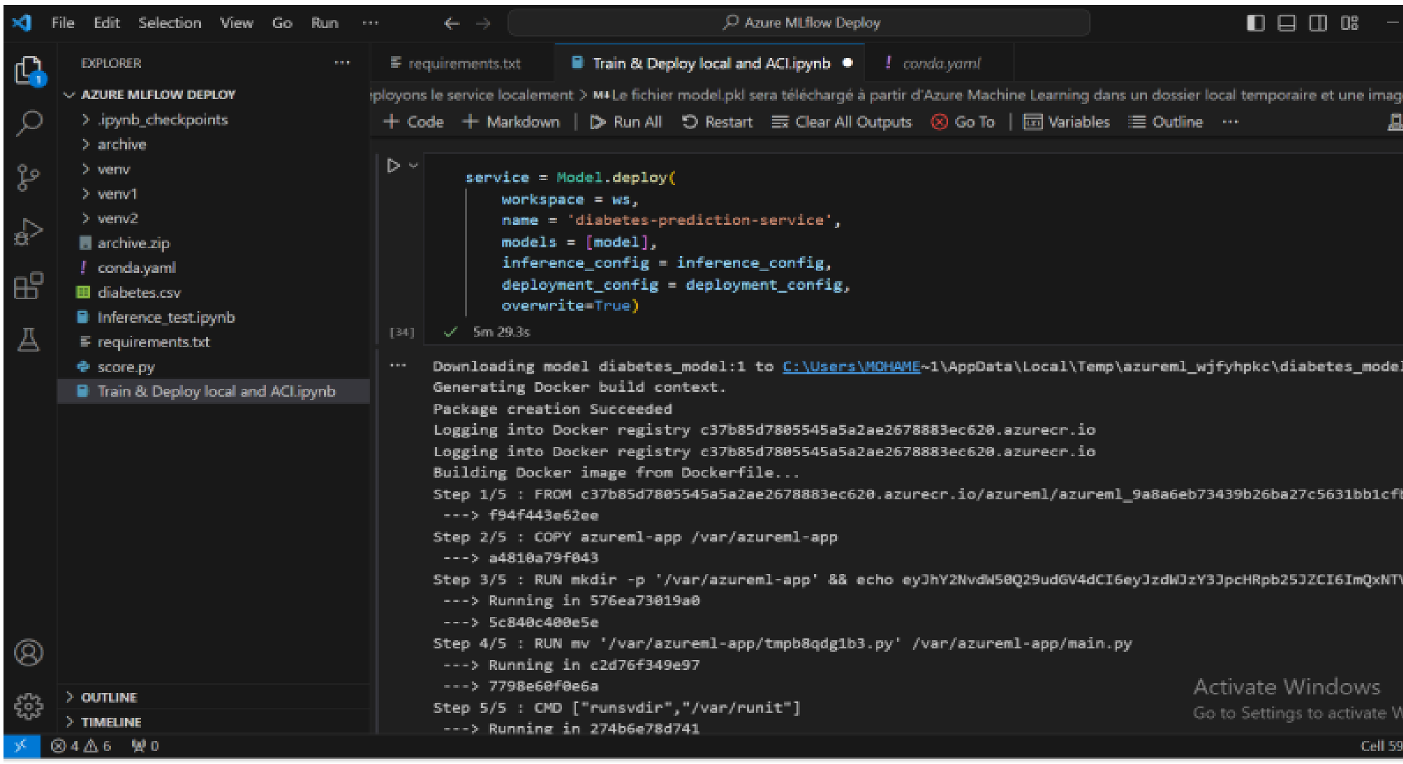
Metrics

best_cv_score
0.8261517

training_accuracy_score
0.7589577

Above, we observe the model artifacts, which encompass the fine-tuned parameters.

Below, we proceed with the local deployment of the model :



ployons le service localement > Le fichier model.pkl sera téléchargé à partir d'Azure Machine Learning dans un dossier local temporaire et une imag

```
service = Model.deploy(
    workspace = ws,
    name = 'diabetes-prediction-service',
    models = [model],
    inference_config = inference_config,
    deployment_config = deployment_config,
    overwrite=True)
```

[34] ✓ 5m 29.3s

```
Downloading model diabetes_model:1 to C:\Users\MOHAME~1\AppData\Local\Temp\azureml_wjfyhpkc\diabetes_model
Generating Docker build context.
Package creation Succeeded
Logging into Docker registry c37b85d7805545a5a2ae2678883ec620.azurecr.io
Logging into Docker registry c37b85d7805545a5a2ae2678883ec620.azurecr.io
Building Docker image from Dockerfile...
Step 1/5 : FROM c37b85d7805545a5a2ae2678883ec620.azurecr.io/azureml/azureml_9a8a6eb73439b26ba27c5631bb1cf
 ---> f94f443e62ee
Step 2/5 : COPY azureml-app /var/azureml-app
 ---> a4810a79f043
Step 3/5 : RUN mkdir -p '/var/azureml-app' && echo eyJhY2NvdW50Q29udGV4dCI6eyJzdWJzY3JpcHRpb25JZCI6ImQxNT
 ---> Running in 576ea73019a0
 ---> 5c840c400e5e
Step 4/5 : RUN mv '/var/azureml-app/tmpb8qdg1b3.py' /var/azureml-app/main.py
 ---> Running in c2d76f349e97
 ---> 7798e60f0e6a
Step 5/5 : CMD ["runsvdir","/var/runit"]
 ---> Running in 274b6e78d741
```

⊗ 4 ⚠ 6 0

Conclusion :

In conclusion, deploying machine learning models on Azure Container Instances (ACI) through the integration of Azure ML and MLflow represents a powerful and efficient strategy for operationalizing models within the Azure cloud environment. This approach seamlessly combines the comprehensive capabilities of Azure ML with the simplicity of MLflow in terms of experiment tracking and model packaging. As a result, organizations can streamline the entire model lifecycle, spanning from training to deployment.

The integration of these tools facilitates straightforward model registration, Docker container creation, and deployment on ACI, with Azure ML managing the complexities of the underlying infrastructure. This unified solution not only improves deployment efficiency but also enables effective monitoring and testing, ensuring the reliability and scalability of machine learning models in real-world applications. As organizations increasingly adopt cloud-based solutions, the collaboration between Azure ML and MLflow provides a robust framework for deploying and managing machine learning models in Azure, contributing to the success of data-driven initiatives.