**TASK 1**

STEPS TO CREATE A CICD PIPELINE:

1. Create a new project.
2. Clone the project folder using command git clone.
3. Move to that specified folder and open the same in code editor.
4. Now create the required files in the code editor (VS Code)
5. Fetch\_data.py

import json

def fetch():

data = {"students": [

{"id": 1, "name": "Abhinav", "marks": 78},

{"id": 2, "name": "Priya", "marks": 85},

{"id": 3, "name": "Rahul", "marks": 92},

]}

with open("raw\_data.json", "w") as f:

json.dump(data, f)

print("Raw data fetched and saved to raw\_data.json")

if \_\_name\_\_ == "\_\_main\_\_":

fetch()

This code is executed with the help of json library. It uses the sample data’s given in data. Upon execution it opens a new file named raw\_data.json to save these data’s.

1. Process\_data.py

import json

def process():

with open("raw\_data.json", "r") as f:

data = json.load(f)

high\_scorers = [s for s in data["students"] if s["marks"] > 80]

with open("processed\_data.json", "w") as f:

json.dump(high\_scorers, f)

print("Processed data saved to processed\_data.json")

if \_\_name\_\_ == "\_\_main\_\_":

process()

This code is also executed with the help of Json library. It processes the saved raw\_data.json file and finds out the persons with marks>80 and stores these processed data in processed\_data.json file.

1. Azure\_pipelines.yml:

trigger:

- main # Triggers when code is pushed to the main branch

pool:

vmImage: ubuntu-latest

steps:

# **Step 1:** Checkout the repository

- task: Checkout@1

# **Step 2**: Set up Python

- task: UsePythonVersion@0

inputs:

versionSpec: '3.10'

addToPath: true

# **Step 3**: Install dependencies

- script: |

python -m pip install --upgrade pip

pip install -r data\_pipeline/requirements.txt

displayName: 'Install dependencies'

# **Step 4**: Run the data fetcher

- script: |

cd data\_pipeline

python fetch\_data.py

displayName: 'Fetch raw data'

# **Step 5**: Process the data

- script: |

cd data\_pipeline

python process\_data.py

displayName: 'Process data'

# **Step 6**: Publish the processed file as an artifact

- task: PublishBuildArtifacts@1

inputs:

PathtoPublish: 'data\_pipeline/processed\_data.json'

ArtifactName: 'ProcessedData'

publishLocation: 'Container'

**trigger: - main**

* This tells Azure DevOps to **run the pipeline automatically** every time someone pushes commits to the **main** branch.
* If you commit to any other branch, this pipeline won’t start (unless you run it manually).

**pool: vmImage: ubuntu-latest**

* The pipeline will run on a **Microsoft-hosted Ubuntu Linux VM**.
* Think of it as a fresh, temporary machine that Azure gives you just for this run.

**steps:**

This is the list of actions the VM will perform, in order.

**STEP 1 - Checkout the repository**

* Pulls your code from GitHub/Azure Repos onto the VM so the next steps can use your files.
* After this, the VM has your data\_pipeline folder, scripts, and requirements.txt.

**STEP 2 – Set up python**

* Installs Python 3.10 on the VM and adds it to the PATH.
* This makes the python command available for the next steps.

**STEP 3 - Install dependencies**

* Upgrades pip (Python’s package manager).
* Installs all libraries listed in data\_pipeline/requirements.txt (e.g., pandas).
* After this, the environment is ready to run your Python scripts.

**STEP 4 - Run the data fetcher**

* Moves into the data\_pipeline folder.
* Runs fetch\_data.py, which creates raw\_data.json containing sample student data.
* You’ll see a console message like “Raw data fetched and saved to raw\_data.json”.

**STEP 6 - Process the data**

* Again enters data\_pipeline.
* Runs process\_data.py, which reads raw\_data.json, filters high scorers (marks > 80), and writes processed\_data.json.
* You’ll see “Processed data saved to processed\_data.json”.

**STEP 7 - Publish the processed file as an artifact**

* Takes the file data\_pipeline/processed\_data.json and uploads it to the pipeline run as an Artifact named ProcessedData.
* After the run, in Azure DevOps → the specific pipeline run → Artifacts tab, you can download processed\_data.json.

The final project will be of the following structure:

azure-data-pipeline/

├── data\_pipeline/

│ ├── fetch\_data.py

│ ├── process\_data.py

│ └── requirements.txt

└── azure-pipelines.yml