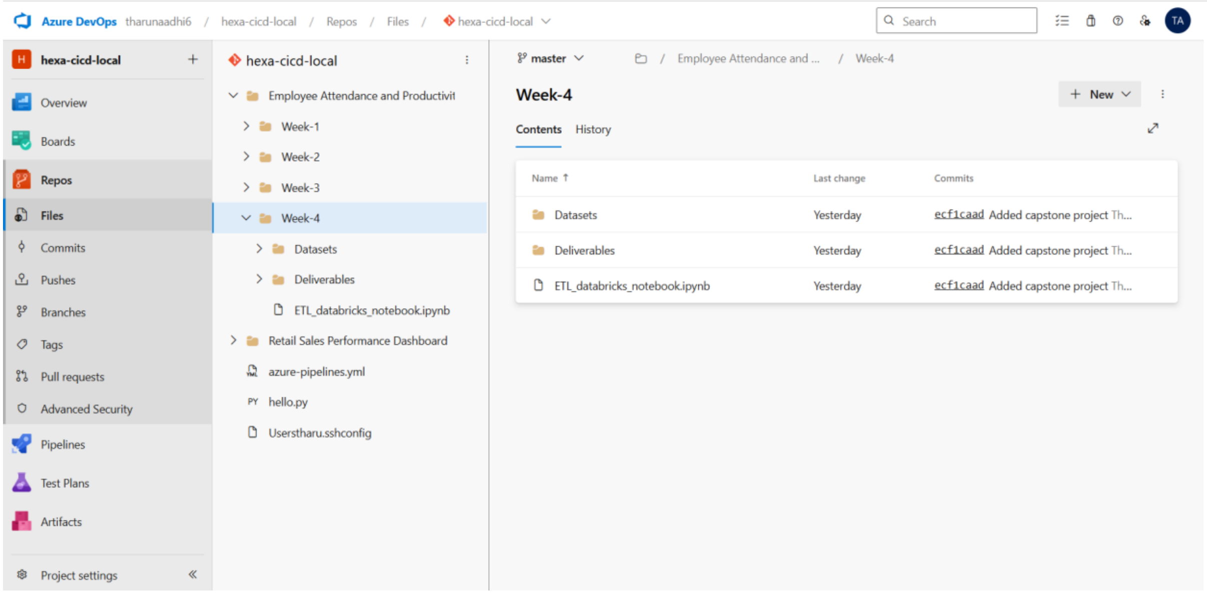
Automation via Azure DevOps

Workflow

Step-1:

Load the folder or databricks script into repo. Here I have loaded week-4 script which does the expected processing.



Azure DevOps tharunaadhi6 / hexa-cicd-local / Repos / Files / hexa-cicd-local

hexa-cicd-local: Overview, Boards, Repos, Files, Commits, Pushes, Branches, Tags, Pull requests, Advanced Security, Pipelines, Test Plans, Artifacts, Project settings

hexa-cicd-local
- Employee Attendance and Productivit
  - Week-1
  - Week-2
  - Week-3
  - Week-4
    - Datasets
    - Deliverables
    - ETL_databricks_notebook.ipynb
- Retail Sales Performance Dashboard
- azure-pipelines.yml
- hello.py
- Userstharu.sshconfig

master / Employee Attendance and ... / Week-4

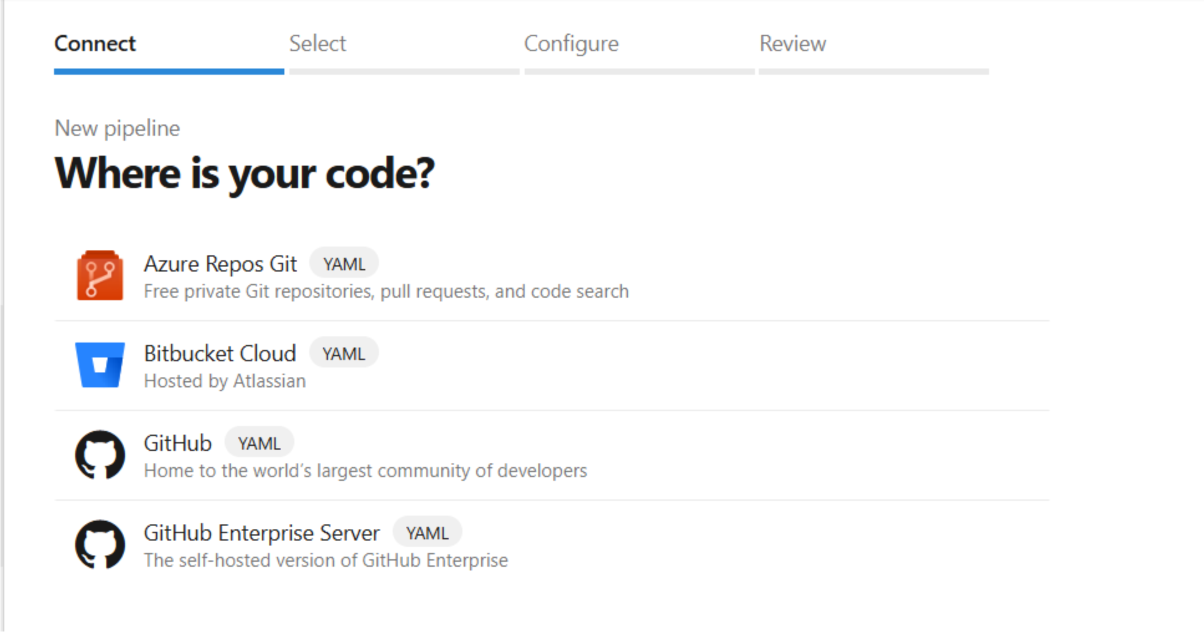
Week-4

Contents History

Name ↑	Last change	Commits
Datasets	Yesterday	ecf1caad Added capstone project Th...
Deliverables	Yesterday	ecf1caad Added capstone project Th...
ETL_databricks_notebook.ipynb	Yesterday	ecf1caad Added capstone project Th...

Step-2:

Select the Version control system(VCS) to continue. In my case it is Azure Repo Git.



Connect Select Configure Review

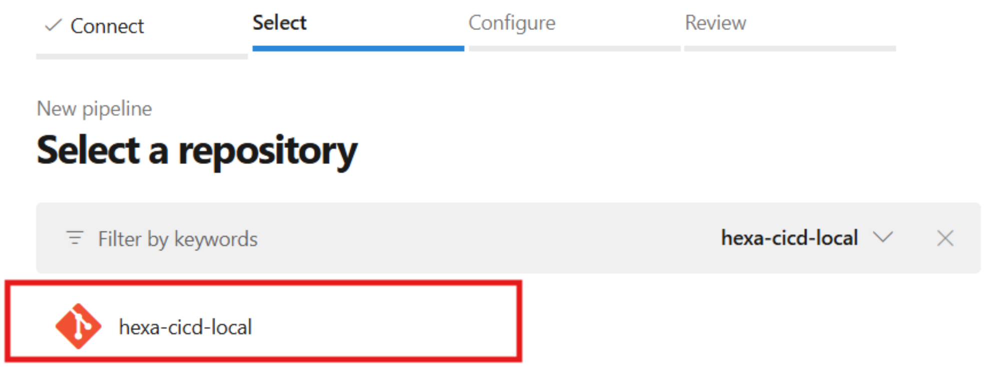
New pipeline

Where is your code?

- Azure Repos Git YAML — Free private Git repositories, pull requests, and code search
- Bitbucket Cloud YAML — Hosted by Atlassian
- GitHub YAML — Home to the world's largest community of developers
- GitHub Enterprise Server YAML — The self-hosted version of GitHub Enterprise

Step-3:

Select the respective repo at which the code, datasets are present. In my case it is present in hexa-cicd-local.

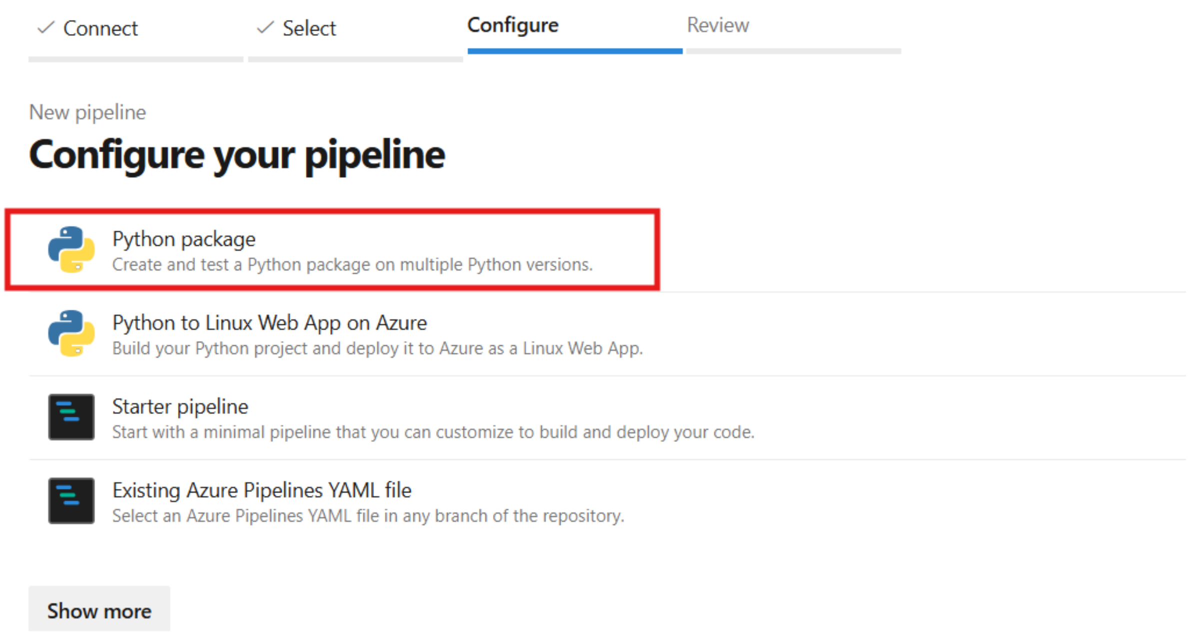


New pipeline

Select a repository

Step-4:

Then select python package to configure the pipeline.



New pipeline

Configure your pipeline

Python package
Create and test a Python package on multiple Python versions.

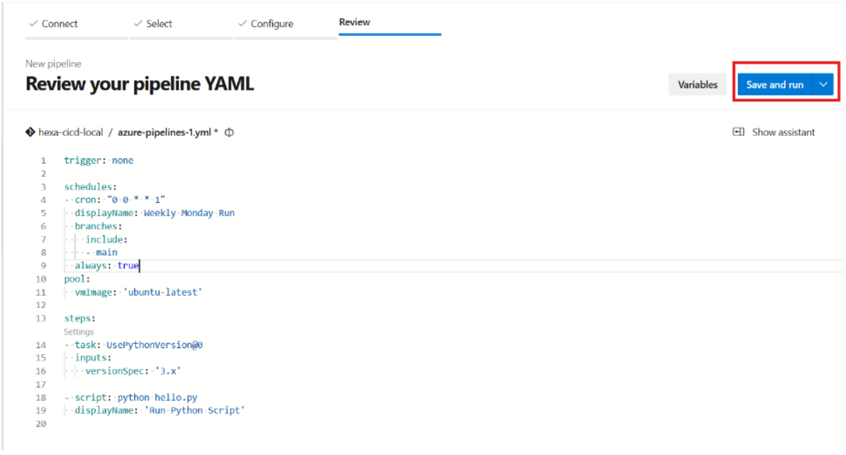
Python to Linux Web App on Azure
Build your Python project and deploy it to Azure as a Linux Web App.

Starter pipeline
Start with a minimal pipeline that you can customize to build and deploy your code.

Existing Azure Pipelines YAML file
Select an Azure Pipelines YAML file in any branch of the repository.

Step-5:

Ensure the YAML file configured properly



New pipeline

Review your pipeline YAML

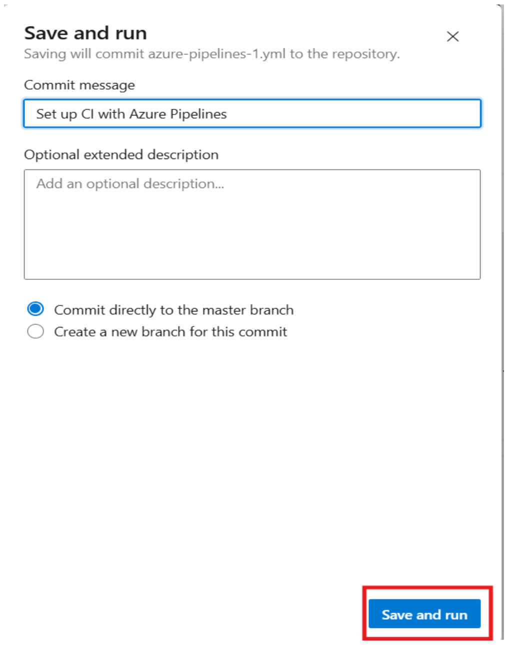
hexa-cicd-local / azure-pipelines-1.yml *

```yaml
trigger: none

schedules:
- cron: "0 0 * * 1"
  displayName: Weekly Monday Run
  branches:
    include:
    - main
  always: true
pool:
  vmImage: 'ubuntu-latest'

steps:
Settings
- task: UsePythonVersion@0
  inputs:
    versionSpec: '3.x'

- script: python hello.py
  displayName: 'Run Python Script'

```

trigger	branch to run
pool	agent to run the tasks
vmImage	virtual machine image
steps	sequence of tasks to perform
task	a predefined tasks from azure

Step-6:

Once all configuration are set then click save and run



Save and run

Saving will commit azure-pipelines-1.yml to the repository.

Commit message

Set up CI with Azure Pipelines

Optional extended description

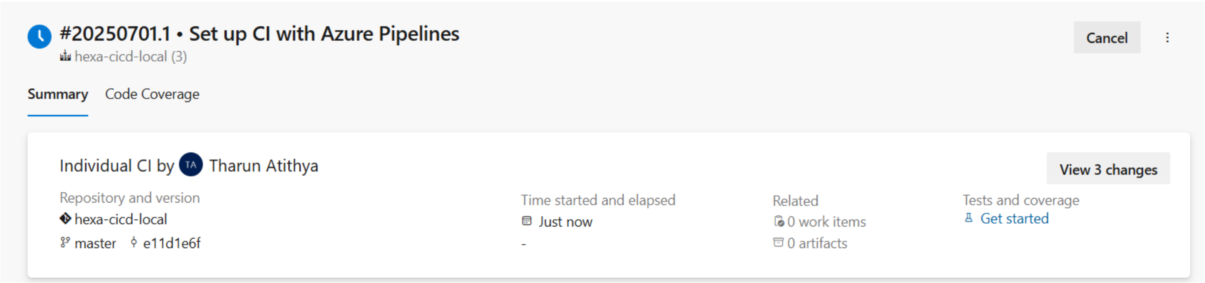
Add an optional description...

● Commit directly to the master branch

○ Create a new branch for this commit

Step-7:

And then we can see the summary of the pipeline and it is scheduled to run with the configured agent.



#20250701.1 • Set up CI with Azure Pipelines

hexa-cicd-local (3)

Individual CI by TA Tharun Atithya

Repository and version
hexa-cicd-local
master e11d1e6f

Time started and elapsed
Just now
-

Related
0 work items
0 artifacts

Tests and coverage
Get started

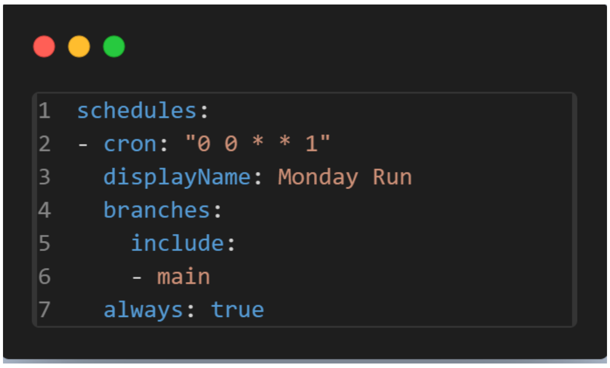
Capstone Tasks

1. Set up a DevOps pipeline to automate weekly processing

Entire workflow defines how to setup a devops pipeline and automate it.

2. Schedule the pipeline to run every Monday

During the YAML configuration in **step-5** we can define the schedule using cron expression

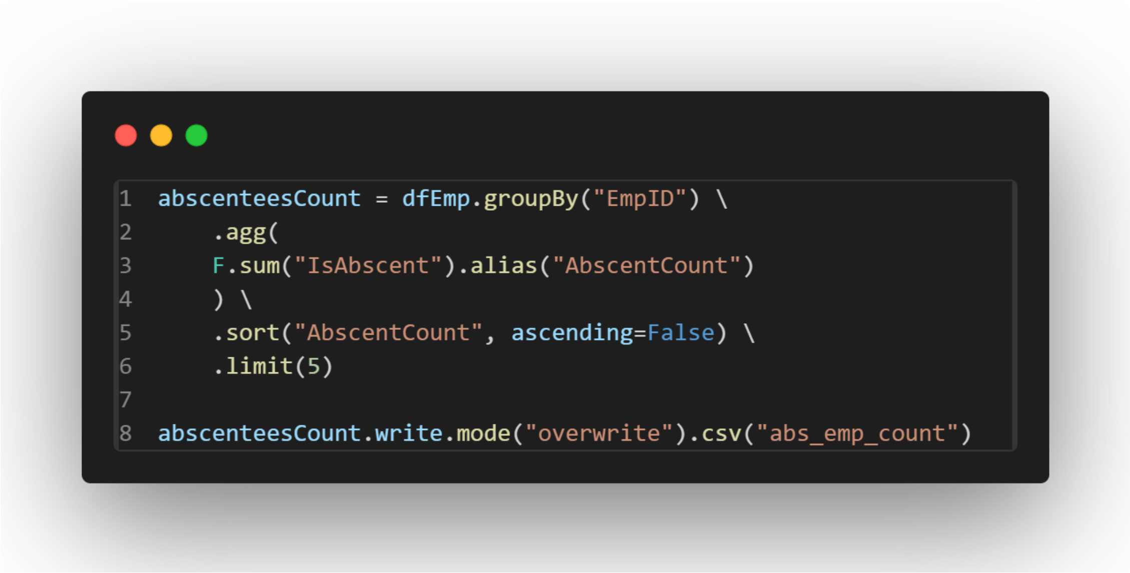


```
schedules:
- cron: "0 0 * * 1"
  displayName: Monday Run
  branches:
    include:
    - main
  always: true
```

3. Output a report with top 5 absentees or lowest performing departments

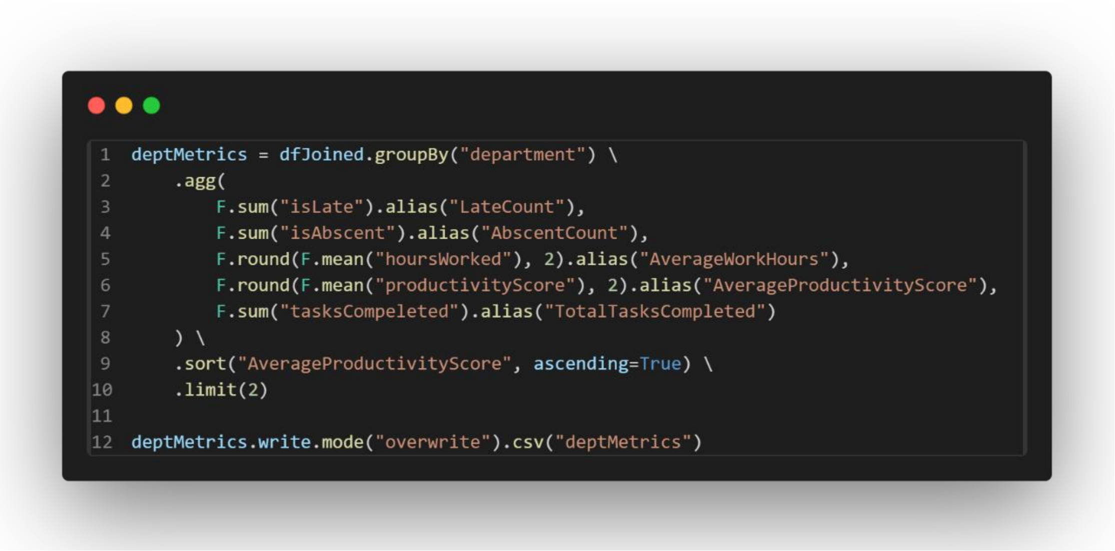
The ETL_databricks_notebook.ipynb has the operations to report the

1. top 5 abscentees



```
abscenteesCount = dfEmp.groupBy("EmpID") \
    .agg(
    F.sum("IsAbscent").alias("AbscentCount")
    ) \
    .sort("AbscentCount", ascending=False) \
    .limit(5)

abscenteesCount.write.mode("overwrite").csv("abs_emp_count")
```

2. lowest performing departments



```
deptMetrics = dfJoined.groupBy("department") \
    .agg(
        F.sum("isLate").alias("LateCount"),
        F.sum("isAbscent").alias("AbscentCount"),
        F.round(F.mean("hoursWorked"), 2).alias("AverageWorkHours"),
        F.round(F.mean("productivityScore"), 2).alias("AverageProductivityScore"),
        F.sum("tasksCompeleted").alias("TotalTasksCompleted")
    ) \
    .sort("AverageProductivityScore", ascending=True) \
    .limit(2)

deptMetrics.write.mode("overwrite").csv("deptMetrics")
```

Deliverables

YAML file and report file of latest attendance metrics is present in /Deliverables folder pushed into github.