Week 5

Pipeline Automation with Azure DevOps

Objective

The primary goal of Week 5 is to automate the stock analysis process using Azure DevOps pipelines. By the end of this stage, the system should be able to automatically trigger a Python script daily to evaluate inventory levels and generate a reorder list without any manual effort. This marks the final step of integrating automation into the inventory management system.

Capstone Tasks

1. Automate Daily Stock Check

Instead of manually running the stock-checking script every day, this task involves configuring an automated pipeline in Azure DevOps that runs on a scheduled basis (e.g., daily at 6 AM). This ensures timely and consistent execution of inventory checks.

2. Trigger Python Script via Pipeline

The pipeline will execute a Python script (written in previous weeks) which:

- Loads stock movement or summary data
- Evaluates current stock levels against predefined reorder thresholds
- Identifies products that are low in stock or need urgent restocking

This script can use data from CSV files, databases (MySQL/MongoDB), or data lakes depending on how the earlier stages were implemented.

3. Export Reorder List

After processing the stock data, the script generates a CSV file that includes a list of all products whose stock levels have fallen below their reorder thresholds. This file can be stored, shared, or integrated with external systems for alerts or reporting.

Pipeline Implementation (Theoretical Overview)

A basic Azure DevOps pipeline for this scenario consists of the following components:

- Trigger: Set to run daily using a cron schedule or on specific events (like a code commit or manual trigger).
- Environment: Provisioned with Python (usually via hosted agents like ubuntu-latest).
- Steps:
    - Install or set up Python environment
    - Run the Python script (check_stock.py)
    - Optionally publish or archive the output CSV file

Example YAML Structure (for understanding only)

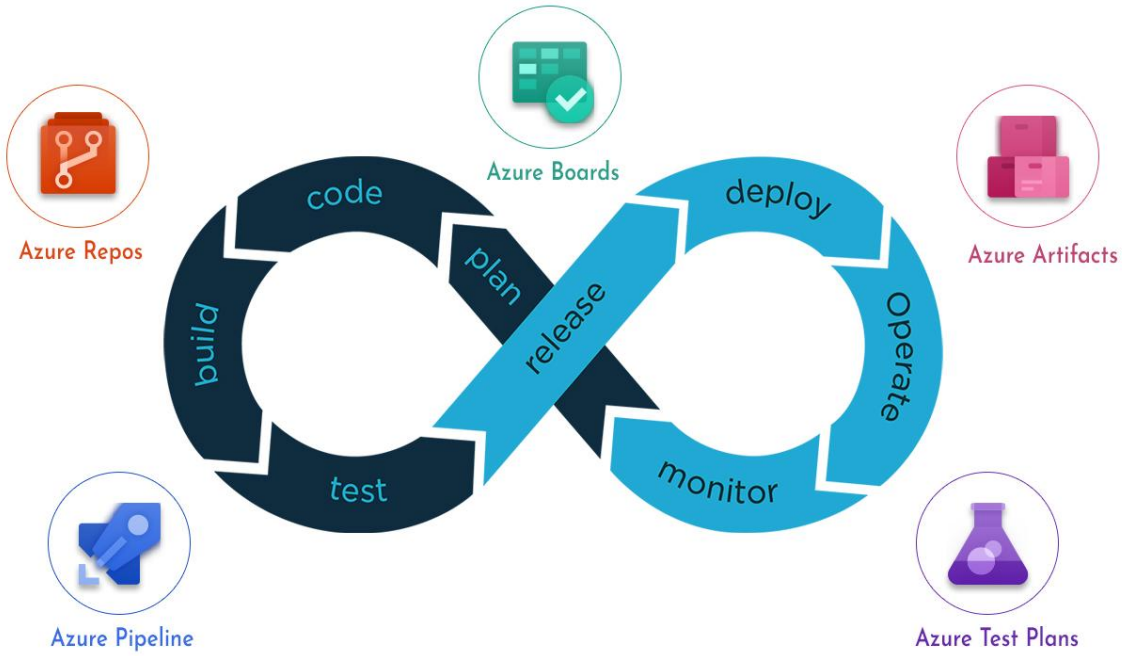
```yaml
trigger:
  schedule:
    - cron: "0 6 * * *"
      displayName: Daily 6AM Check
      branches:
        include: [main]

pool:
  vmImage: 'ubuntu-latest'

steps:
  - task: UsePythonVersion@0
    inputs:
      versionSpec: '3.x'
  - script: python check_stock.py
    displayName: 'Run Stock Checker Script'
```

Output and Result

The output of this pipeline is a CSV file (e.g., reorder_list.csv) that contains:

- Product ID
- Product Name
- Current Stock Level
- Reorder Threshold
- Flag indicating if reorder is needed

Azure DevOps LifeCycle



Benefits of This Automation

- Consistency: Runs the same logic every day, ensuring no stock check is skipped
- Speed: Quickly identifies low-stock products without waiting for human action
- Scalability: Works across thousands of products or warehouses
- Integration Ready: Output can be plugged into dashboards, email alerts, or ERP systems

Week	Focus	Outcome
Week 1	Database Design	Tables in MySQL, audit logs in MongoDB
Week 2	Data Processing	Clean stock data and flag low stock using Python
Week 3	Big Data Analysis	Warehouse-level stock insights using PySpark
Week 4	ETL Integration	Merged inventory view in Databricks (or Colab)
Week 5	Automation	Daily stock analysis pipeline in Azure DevOps