# Capstone Project: Customer Order Insights & Delivery Tracker

# Week 5 – Automation using Azure DevOps

**Tools: Azure DevOps**

**Capstone Tasks:**

* Build an Azure DevOps pipeline to run the full Python analysis
* Log delay summary to a file
* Send a basic success notification (print/log)

## Objective

The goal of this stage is to automate the execution of the Customer Order Insights & Delivery Tracker analysis pipeline using Azure DevOps. The pipeline will run Python scripts, generate logs, and mark completion, ensuring consistency and automation.

## Step 1: Create an Azure DevOps Organization

1. Visit <https://azure.microsoft.com/en-us/products/devops>.
2. Sign in with your Microsoft account.
3. Create a new Organization (unique name).
4. Select the region nearest to you.

* **Build an Azure DevOps pipeline to run the full Python analysis (Step 2 and Step 3)**

## Step 2: Create a Project

1. Inside your organization, click New Project.  
2. Enter details:  
 - Project Name: CustomerOrderInsights  
 - Visibility: Public (recommended to use free CI/CD minutes)  
3. Click Create.

## Step 3: Push Your Project Code to Repository

1. Navigate to Repos in your DevOps project.  
2. Add your Python analysis scripts (orders\_analysis.py, requirements.txt, cleaned CSV files).  
3. Push the code using Git commands:  
 git init  
 git remote add origin <repo-URL>  
 git add .  
 git commit -m 'Customer order analysis project'  
 git push -u origin main

* **Log delay summary to a file (Step 4)**

## Step 4: Create a Pipeline

1. Go to Pipelines → Create Pipeline.  
2. Select your repository.  
3. Choose YAML pipeline option.

* **Send a basic success notification (print/log) (Step 5)**

## Step 5: Write YAML Pipeline

Add a file named azure-pipelines.yml at the root of your repo with the following content:

trigger:  
- main  
  
pool:  
 vmImage: 'ubuntu-latest'  
  
steps:  
- task: UsePythonVersion@0  
 inputs:  
 versionSpec: '3.x'  
 addToPath: true  
  
- script: |  
 python -m pip install --upgrade pip  
 pip install -r requirements.txt  
 python orders\_analysis.py > delay\_summary.log  
 displayName: 'Run Customer Order Analysis'  
  
- script: echo "Pipeline execution completed successfully!"  
 displayName: 'Mark Completion'

## Step 6: Save and Run the Pipeline

1. Commit and push the YAML file to the repository.  
2. The pipeline will start automatically when changes are pushed to the main branch.  
3. Logs will display dependency installation, script execution, and completion status.

## Step 7: Verify Logs

After the pipeline run completes, check the Job Summary:  
- Pipeline Status: Success or Failed  
- Execution logs from orders\_analysis.py  
- Delay summary saved in delay\_summary.log

If failed, review logs to fix errors.

## Step 8: Automate with Scheduled Runs (Optional)

1. Go to Pipeline → Triggers → Scheduled.  
2. Add a CRON schedule (e.g., run daily at 6 PM).  
3. Ensures automatic execution at fixed intervals.

## Benefits of Automating Customer Order Analysis

1. Automated Execution – Runs without manual intervention.  
2. Logs & Monitoring – Delay insights logged for tracking.  
3. Scalable – Can handle larger datasets as business grows.  
4. Consistent Results – Ensures repeatable analysis process.