## **Part 3 – Configure CI/CD Pipeline**

I set up a **CI/CD pipeline** in Azure DevOps to automate building and testing my E-Commerce Application. Below are the steps I followed:

1. **Created a new pipeline**
   * I went to Azure DevOps → Pipelines → Create Pipeline.
   * I connected my pipeline to the repository where my E-Commerce Application code is stored.
2. **Configured YAML file**
   * I wrote an azure-pipelines.yml file inside my project repository.
   * In this file, I defined the pipeline structure.

Example configuration I used:  
  
 trigger:

- main

pool:

vmImage: 'ubuntu-latest'

steps:

- task: UsePythonVersion@0

inputs:

versionSpec: '3.x'

- script: python src/app.py

displayName: 'Run E-Commerce Application Placeholder'

1. **Defined Trigger**
   * I configured the pipeline to trigger automatically whenever changes were pushed to the **main branch**.
2. **Selected Build Agent**
   * I used the **Ubuntu-latest** virtual machine image provided by Microsoft-hosted agents.

1. **Set up Python Environment**
   * I added a step to use **Python 3.x** as the runtime version for the pipeline execution.

1. **Executed Application Script**
   * I configured a script step that runs the application placeholder (src/app.py).
   * This acted as a validation that the pipeline runs without errors.

1. **Validated Pipeline Execution**
   * I saved and ran the pipeline in Azure DevOps.
   * The pipeline succeeded as soon as the script executed successfully with no errors.

Through this setup, I ensured that the pipeline automatically runs whenever new code is pushed, giving me confidence that my application can execute successfully in a CI/CD environment.