**Introduction to CI/CD and Azure DevOps**

**What is CI/CD?**

**CI: Continuous Integration**

**Definition**:  
Continuous Integration is the process where developers integrate code into a shared repository several times a day. Each integration triggers an automated build and test to catch issues early.

**CI Workflow:**

1. Developer writes code.
2. Code is pushed to the version control system (e.g., Git).
3. A pipeline is triggered.
4. The code is automatically:
   * Built (compiled, packaged, etc.)
   * Tested (unit tests, linting)
   * Validated (static analysis, security scan)

**CI Goals:**

* Reduce integration issues.
* Detect bugs early.
* Improve software quality.

**CD: Continuous Delivery & Continuous Deployment**

**Continuous Delivery**

**Definition**:  
Code that passes CI is automatically prepared for **manual** deployment to staging/production.

**Continuous Deployment**

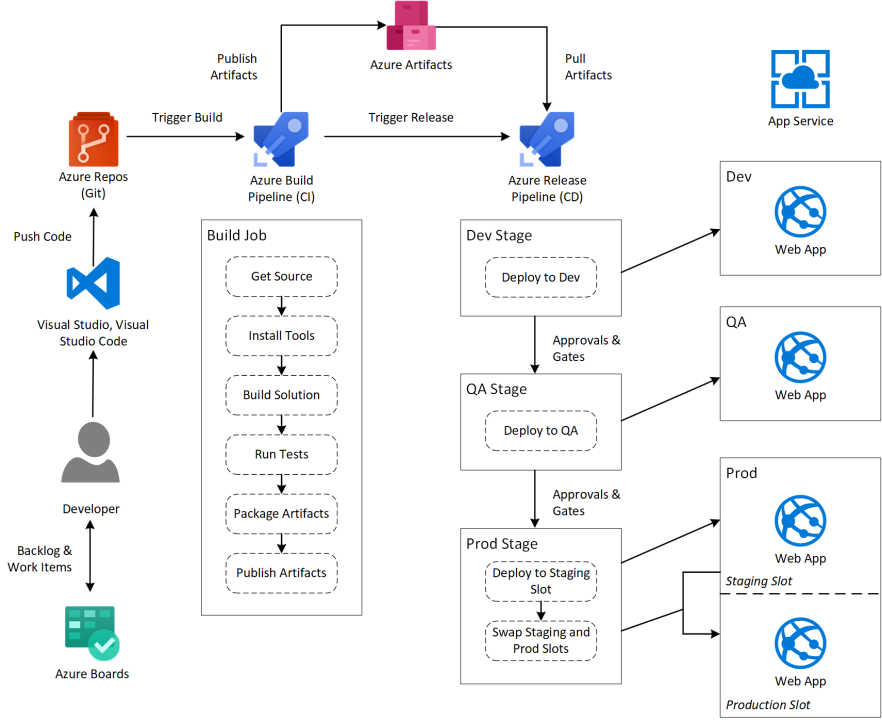
**Definition**:  
Same as Continuous Delivery, but the deployment is **automatic** — no manual approval is required.

**CD Workflow:**

1. After CI, a release pipeline is triggered.
2. Code is deployed to staging or production.
3. Configuration files, secrets, and infrastructure-as-code are applied.
4. Post-deployment testing and alerts are triggered.

**Key Components for CI/CD:**

| **Component** | **Description** |
| --- | --- |
| Repos | Source code management using Git |
| Pipelines | CI/CD automation |
| Artifacts | Package hosting & distribution |
| Environments | Deployment targets like dev, prod |



**Creating a CI/CD Pipeline in Azure DevOps**

**🔧 Step-by-Step CI/CD Pipeline Setup:**

**1. Create a Repository**

* Use Azure Repos to host your code.
* Use Git to push your project into Azure DevOps.

**2. Define CI using azure-pipelines.yml**

trigger:

- main

pool:

vmImage: 'ubuntu-latest'

steps:

- script: echo Hello, CI/CD!

**3. Setup the Pipeline**

* Go to **Pipelines > Create Pipeline**
* Choose your repo > YAML > Select your azure-pipelines.yml file
* Click **Run** – this triggers the CI process.

**Adding CD – Deployment Stage**

- stage: Deploy

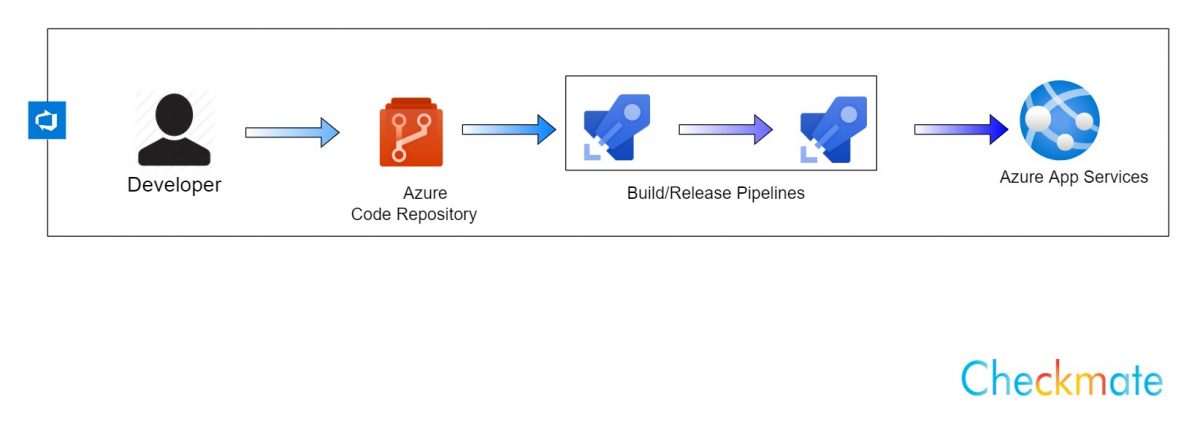
jobs:

- job: DeployJob

steps:

- script: echo Deploying App

Use **Environments** in Azure DevOps to configure where this stage deploys.



**Benefits of CI/CD in Azure DevOps**

| **Benefit** | **Description** |
| --- | --- |
| Automated Testing | Detect errors early |
| Faster Delivery | Rapid deployment to staging/production |
| Traceability | Complete history of code changes and builds |
| Integration with Azure Cloud | Native support for Azure VMs, App Services, etc. |

**Real-Time Use Case: Python App Deployment**

* A Python script is pushed to Azure Repos
* Azure Pipeline runs tests automatically
* If successful, code is deployed to an Azure Web App
* Emails are sent to team after deployment

