

Project:

# CI/CD Pipeline for .NET Application

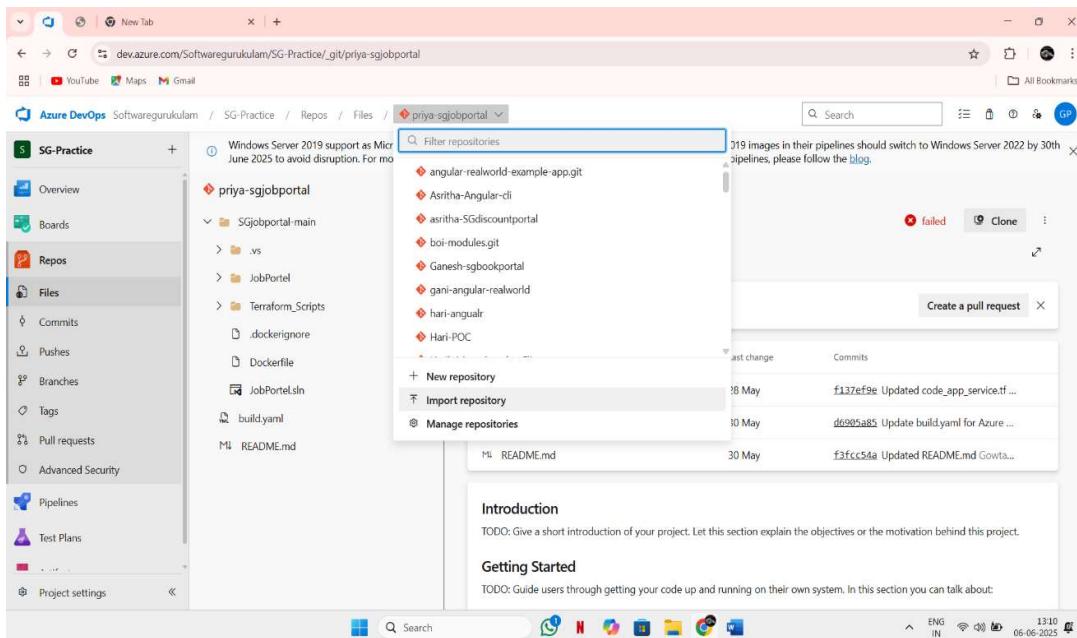
## ✓ Pre-Requisites

- Azure DevOps Project
- Windows VM (Azure or on-prem)
- IIS Installed & configured
- .NET Framework or .NET Core Runtime installed on VM
- Azure DevOps Agent installed on the VM using Deployment Group
- App hosted in Azure Repos

## ✓ Step-by-Step Procedure

### ◆ Step 1: Import a folder containing .NET files into Azure repos

- Import a folder that contains all files related to .NET app into the Azure repos through option Import Repository.



## ◆ Step 2: Create a Deployment Group in Azure DevOps

1. Go to your Azure DevOps Project → **Pipelines > Deployment Groups**
2. Click **Add a deployment group**
  - Name: SG-priyanka-Deploymentgroup
3. Copy the **PowerShell script** shown (Agent installation script)

The screenshot shows the Azure DevOps interface for managing deployment groups. On the left, there's a sidebar with project navigation. The main area is titled 'Deployment groups' and shows a single entry: 'SG-priyanka-Deploymentgroup'. This entry is highlighted with a blue oval. Below the entry, there are tabs for 'Details', 'Targets', 'Save', 'Share', 'Security', and 'Help'. The 'Details' tab is selected. Under 'Deployment group name', the value 'SG-priyanka-Deploymentgroup' is entered. In the 'Description' field, there is a large empty text area. The 'Deployment pool' section shows 'SG-Practice-SG-priyanka-Deploymentgroup' and a 'Manage' button. To the right, the 'Type of target to register:' dropdown is set to 'Windows' and the 'System prerequisites' checkbox is checked. Below that, the 'Registration script (PowerShell)' field contains a long PowerShell script. At the bottom of the page, there's a search bar and some system status indicators.

```
$ErrorActionPreference="Stop";If((!([Security.Principal.WindowsPrincipal] [Security.Principal.WindowsIdentity]::GetCurrent() ).IsInRole([Security.Principal.WindowsBuiltInRole]"Administrator"))){ throw "Run command in an administrator PowerShell prompt";}If($psVersionTable.PSVersion -lt (New-Object System.Version("3.0"))){ throw "The minimum version of Windows PowerShell that is required by the script (3.0) does not match the currently running version of Windows PowerShell." }If((!Test-Path $env:SystemDrive"\agent"));{mkdir $env:SystemDrive'\agent'; cd $env:SystemDrive'\agent'; for($i=1; $i -lt 100; $i++){$destFolder="A\$i";$String=If((!Test-Path ($destFolder))){mkdir $destFolder;}d $destFolder;Break};$agentZip=$PSScriptRoot\agent.zip;$DefaultProxy=[System.Net.WebRequest]::DefaultWebProxy;$securityProtocol=@();$securityProtocol+=[Net.ServicePointManager]::SecurityProtocol;$securityProtocol+=[Net.SecurityProtocolType]::Tls12;[Net.ServicePointManager]::SecurityProtocol=$securityProtocol;$WebClient=[WebClient]::New Object;$Uri="https://download.agent.dev.azure.com/agent/4.255.0/yts-agent-win-x64-4.255.0.zip";If($DefaultProxy -and (-not $DefaultProxy.IsBypassed($Uri))){$WebClient.Proxy=[WebClient]::New Object;$WebClient.Proxy.DefaultProxy.GetProxy($Uri).OriginalString,$true};$WebClient.DownloadFile($Uri, $agentZip);Add-Type -AssemblyName System.IO.Compression.FileSystem;$SystemIOCompressionZipfile=[System.IO.Compression.ZipFile]::ExtractToDirectory($agentZip, "C:\Temp\agent\dagent\agent\agent");$agentDir="C:\Temp\agent\dagent\agent\agent";$agentDir
```

## ◆ Step 3: Set Up VM and Register the Agent

On your **Windows VM**:

1. Open **PowerShell as Administrator**
2. Paste and run the **PowerShell script** from the deployment group page
3. Ensure agent is connected and shown as **online** in Azure DevOps

```
[uri:\file:///C:/Windows/Temp/agent-2.255.0/winx-agent-win-v4.255.0.zip];(Iis24,Wcf,ServiceFabric,Protocol,WebClient,Net
Client,Net,WebClient; Uri=https://download.agent.dev.azure.com/agent/2.255.0/v1/win-x64/4.255.0.zip);(IfDefaultProxy & (!netDefaultProxy || !netDefaultProxy.bypassed($Uri)))(WebClient,Proxy,New-Object Net WebProxy
$UriDefaultProxy,GetProxy($Uri),OriginalString,$true); WebClient.DownloadFile($Uri, $agentZipPath,Add-Type -AssemblyName System.IO.Compression.FileSystem)[System.IO.Compression.ZipFile]::ExtractToDirectory($agentZipPath, "$PWD"); config.cmd --deploymentgroup -deploymentgroupname "SG-shield-deployment group" --agent Sem:[CPU]ULTRAM -runservice -work "work" -url https://dev.azure.com/Sofwareregulation/-/project
name "SG-Practice" -auth PAT -token 2d0c0f4c500d81ca01101900e0f77c0a0f198fb4caaa49ff80aa3d04a95; Remove-Item $agentZipPath

 Directory: C:\agentagent

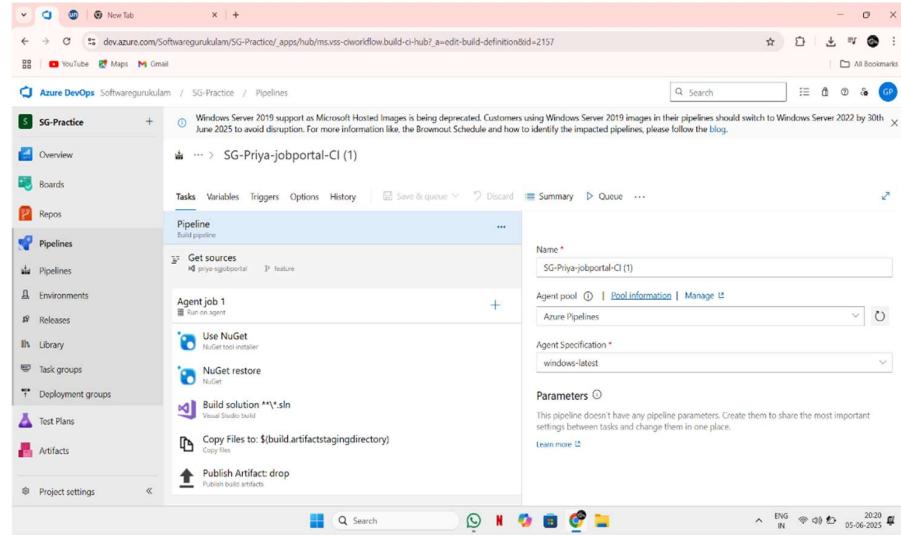
          LastWriteTime           Length Name
-----      (--)              (--)    --
  02-06-2025 09:02 PM        66 agent.v4.255.0           (commit 470b366)

>>> Connect:
Connecting to server ...
>>> Register Agent:
Scanning for tool capabilities.
Connecting to the server.
Enter deployment group tags for agent? (Y/N) (press enter for N) >
successfully added host agent
agent configuration updated
2025-06-02 15:43:54Z: Settings Saved.
Enter enable SERVICE_SID_TYPE_UNRESTRICTED for agent service (Y/N) (press enter for N) >
Enter user account to use for the service (press enter for NT AUTHORITY\SYSTEM) >
Error reported in diagnostic logs. Please examine the log for more details.
  C:\azagent\WSI\diagAgent\agent\bin\35420\utc\log
agent configuration updated
Service vssagent,Softwareregulation,SG-Practice SG-shield-deployment group,SHIELDRA successfully installed
Service vssagent,Softwareregulation,SG-Practice SG-shield-deployment group,SHIELDRA successfully set recovery option
Service vssagent,Softwareregulation,SG-Practice SG-shield-deployment group,SHIELDRA successfully set to delayed auto start
Service vssagent,Softwareregulation,SG-Practice SG-shield-deployment group,SHIELDRA successfully configured
Enter whether to prevent service starting immediately after configuration is finished? (Y/N) (press enter for N) >
Service vssagent,Softwareregulation,SG-Practice SG-shield-deployment group,SHIELDRA started successfully
```

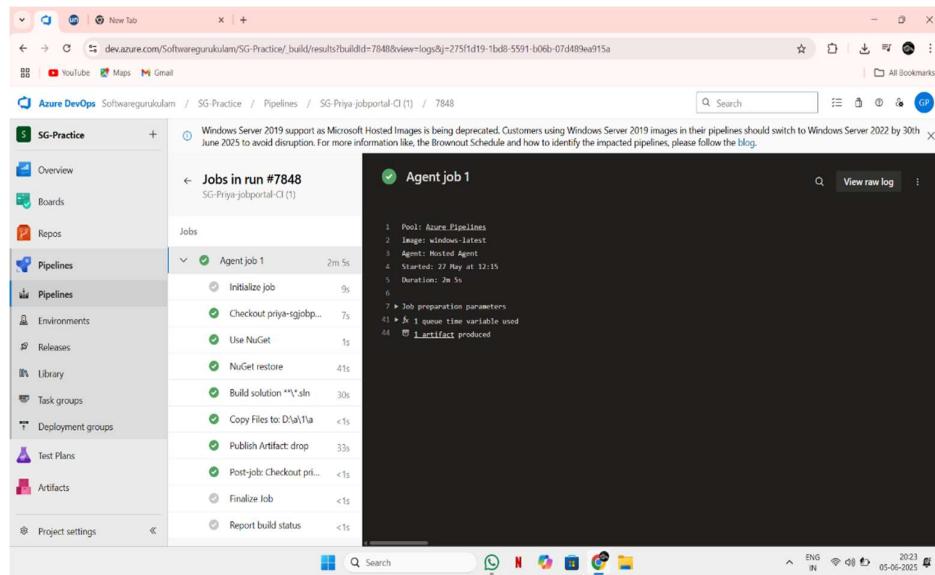
#### ◆ Step 4: Set Up CI Pipeline (Build)

1. Go to **Pipelines > Builds**
  2. Click **New Pipeline** → Use **Classic Editor**
  3. Choose your repo, then select **Empty Job**
  4. In build pipeline select agent specification as **windows-latest**
  5. Change the pipeline name to **SG-Priya-jobportal-CI (1)**
  6. In agent job change agent pool to Azure pipelines and agent specification as **windows-latest**
  7. Add tasks to agent job in order:
    - o **NuGet tool installer**
    - o **NuGet:** give NuGet restore in command
    - o **Visual Studio build:**
      - platform: any cpu
      - configuration: release
    - o **Copy files:**
      - source folder: \$(agent.builddirectory)

- content: \*\* (or any specified type required like \*\*/\*.tf for terraform files)
  - target folder: \$(build.artifactstagingdirectory)
- Publish Build Artifacts:
    - Path to publish: \$(Build.ArtifactStagingDirectory)
    - Artifact name: drop



## 8. Save and Queue build



## 9. Published artifacts:

The screenshot shows the Azure DevOps interface for a project named 'SG-Practice'. The left sidebar is titled 'Pipelines' and includes options like Overview, Boards, Repos, Pipelines, Environments, Releases, Library, Task groups, Deployment groups, Test Plans, Artifacts, and Project settings. The 'Artifacts' section is currently selected. The main area displays a table of published artifacts:

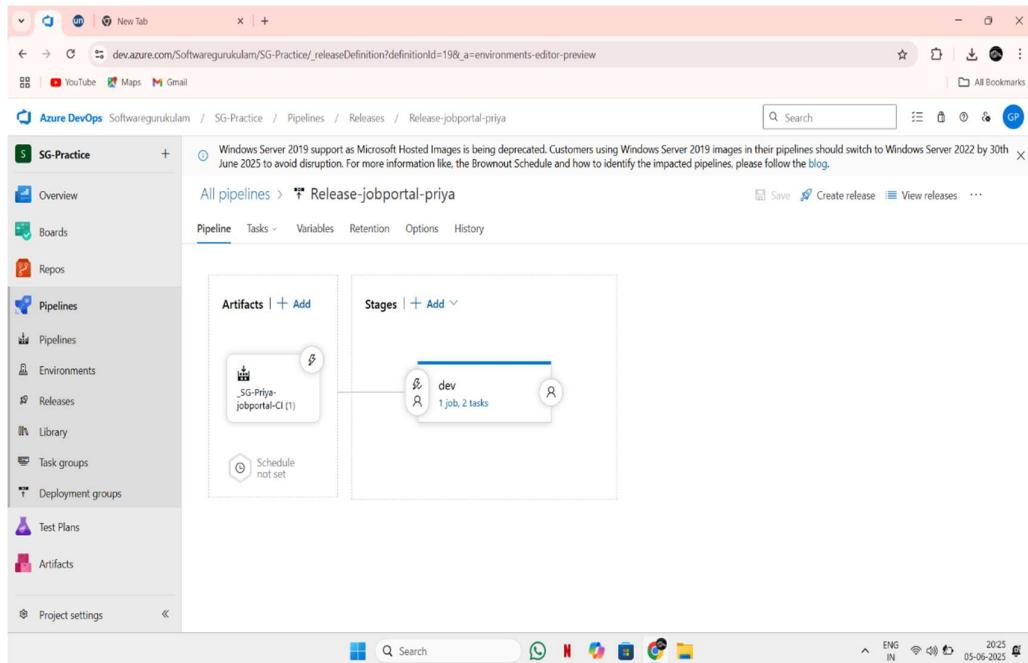
Name	Size
drop	114 MB
s	114 MB
.git	18 MB
README.md	985 B
SGjobportal-main	95 MB
.dockerignore	422 B
.vs	2 MB
Dockerfile	856 B
JobPortal.sln	2 KB
JobPortal	95 MB
Terraform_Scripts	3 KB

### ◆ Step 5: Set Up CD Pipeline (Release) with Deployment Group

1. Go to **Pipelines > Releases**
2. Create a **New Release Pipeline** → Start with **Empty Job**
3. Add an **Artifact** from the build pipeline
4. Enable **CD Trigger** ( icon)

**In the Stage (dev):**

5. Select the template as **IIS website deployment**
6. the stage name as **dev**
7. Inside the stage **dev** it will take the default web site of VM. If you want you can create a custom web site and add bindings in the stage.
8. Under **Agent Job**, change:
  - o **Deployment group** → SG-priyanka-Deploymentgroup



## ◆ Step 6: Add Tasks to Deploy to IIS

Inside the **IIS agent Job**, add these tasks:

### 1. IIS Web App Management

- Task: **IIS Web App Manage**
- **Web site Name:** Priya.web

### 2. IIS Web App Deployment

- Task: **IIS Web App Deploy**
- **Web Site Name:** priya.web (Default Web Site or custom site)
- **Package or Folder:** \$(System.DefaultWorkingDirectory)/\_SG-Priya-jobportal-CI (1)/drop/WebPublish
- Ensure IIS site is pre-created or handled via PowerShell

The screenshot shows the Azure DevOps Pipelines interface. On the left, the navigation bar includes 'Overview', 'Boards', 'Repos', 'Pipelines' (selected), 'Environments', 'Releases', 'Library', 'Task groups', 'Deployment groups', 'Test Plans', and 'Artifacts'. The main area displays the 'All pipelines > Release-jobportal-priya' screen. Under the 'dev' stage, there is an 'IIS Deployment' task. The configuration for this task includes:

- Stage name:** dev
- Parameters:** Run on deployment group
- Configuration type:** IIS Website
- Action:** Create Or Update
- Website name:** priya.web
- Add binding:** http://All Unassigned:81

### 3. Save and Create Release

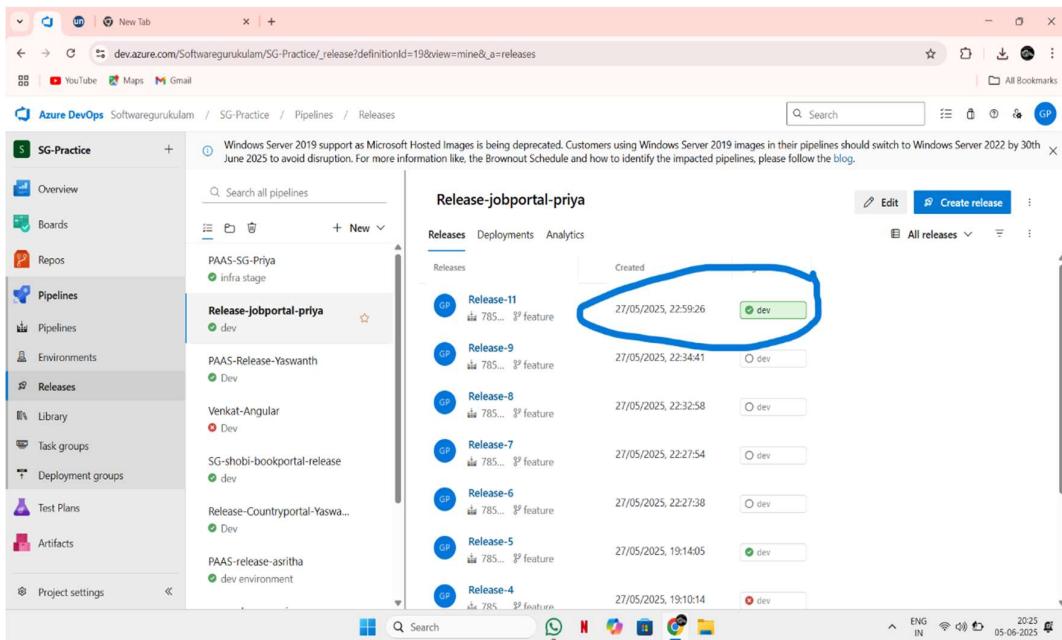
The screenshot shows the Azure DevOps Pipelines interface, specifically the 'Logs' section for the 'Release-jobportal-priya > Release-11 > dev > IIS Deployment' step. The log entry for 'Lokesh-testing' is displayed:

Task	Status	Duration
Initialize job	succeeded	2m 10s
Download artifact - SG-Priya-jobportal-CI (1) - drop	succeeded	20s
IIS Web App Manage	succeeded	2s
IIS Web App Deploy	succeeded	3s
Finalize Job	succeeded	<1s

#### ◆ Step 7: Trigger and Verify Deployment

1. Commit changes to your repo → triggers CI

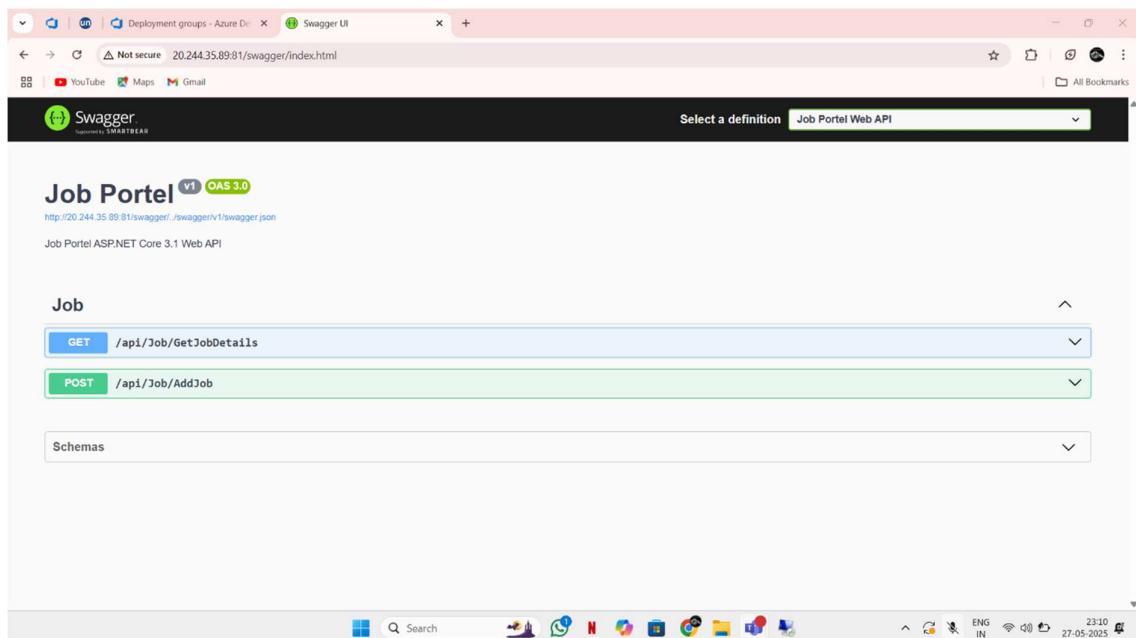
## 2. Release pipeline picks artifact → deploys to IIS



The screenshot shows the Azure DevOps interface for the 'SG-Practice' project. The left sidebar has 'Pipelines' selected. Under 'Releases', there is a list of releases for the 'Release-jobportal-priya' pipeline. The first release, 'Release-11', is highlighted with a blue circle. The table includes columns for 'Created' date and a status indicator.

Release	Created	Status
Release-11	27/05/2025, 22:59:26	dev
Release-9	27/05/2025, 22:34:41	dev
Release-8	27/05/2025, 22:32:58	dev
Release-7	27/05/2025, 22:27:54	dev
Release-6	27/05/2025, 22:27:38	dev
Release-5	27/05/2025, 19:14:05	dev
Release-4	27/05/2025, 19:10:14	dev

## 3. Open your VM's public IP in browser and verify app is running.



The screenshot shows a browser window displaying the Swagger UI for the 'Job Portal' API. The URL is `http://20.244.35.89:81/swagger/index.html`. The interface includes a navigation bar with 'Select a definition' and 'Job Portel Web API'. Below it, the 'Job' section shows two API endpoints: 'GET /api/Job/GetJobDetails' and 'POST /api/Job/AddJob'. At the bottom, there is a 'Schemas' section.

✓ The application is successfully deployed.