Ci-Cd Pipeline

CI-CD:

CI/CD (Continuous Integration/Continuous Deployment) is a software development practice that involves automating the process of integrating code changes, testing those changes, and then deploying them to production environments. It's a set of principles and practices aimed at improving the efficiency, reliability, and speed of software development and deployment.

Continuous Integration (CI): This is the practice of frequently integrating code changes from multiple developers into a shared repository.

Continuous Deployment (CD): Building on the CI foundation, continuous deployment extends the automation process to include the deployment of code changes to production or staging environments automatically and regularly.

We have implemented ci-cd pipeline in local with GitLab repository (Nop Commerce 4.6) on Windows

Gitlab CI-CD Pipeline (OS: windows):

Pre- Requisites:

- Create a repository with GitLab
- Create and setup GitLab runner
- IIS server setup

Create and setup runner:

Step 1:

- Install GitLab runner on windows
- For register runner with your existing repo in GitLab, go to settings > click on CI/CD > **Runners > New Project Runner > Copy the runner token**

Step 2:

- Run shell in administration Mode and go to the path where gitlab-runner.exe available
- gitlab-runner.exe register
- Enter the coordinator URL for GitLab: https://gitlab.com/

  Enter the GitLab-CI token:

[To get GitLab-CI token, Enter the project and go to the CI/CD setting on GitLab. Go to the runners' section and expand it. copy the token and paste to console then press enter]

1. Use tag: SSH and CI
2. Next select the executor and done with registration
   Need to set execution policy remote. For that run the command below
3. “Set-ExecutionPolicy RemoteSigned-Scope CurrentUser”
4. Then, To run the runner , write the following command on the following directory : gitlab-runner.exe run

gitlab-ci.yml :

Clone Repository and Create a “.**gitlab-ci.yml**” file in the root directory of your Nop Commerce project then define the stages that your pipeline will go through. For a basic Nop Commerce setup, you might have stages like **build**, **test**, and **deploy.**

Build:
In the build stage, you typically specify the commands to restore dependencies and build the Nop Commerce project.

Test:
In the test stage, you can run any automated tests associated with your Nop Commerce application.

Deploy:
In the deploy stage, you can define deployment actions to deploy your Nop Commerce application to a specific environment. This might involve copying files, publishing them to a web server, or deploying them to a container.

Here is our code for deploying with local IIS with GitLab

```yaml
stages:
  - build
  - test
  - deploy

variables:
  BUILD_DIR: "D:\\TeamMahbubVai\\nopcommerce-4.6\\src"
  TEMP_DIR : "D:\\artifacts"
  PUBLISH_DIR: "C:\\inetpub\\wwwroot\\test"
  SITE_NAME: "test"


build_job:
  stage: build
  script:
   - cd $BUILD_DIR
   - dotnet clean --configuration Release
   - dotnet restore
   - dotnet build --no-restore -c Release

test_job:
  stage: test
  script:
    - cd $BUILD_DIR
    - dotnet test

deploy_job:
  stage: deploy
  script:
    - cd $BUILD_DIR
    - dotnet clean --configuration Release
    - dotnet restore
    - dotnet build --no-restore -c Release
    - cd Presentation/Nop.Web/
    - dotnet publish Nop.Web.csproj --no-build -c Release -o $TEMP_DIR /p:CopyRefAssembliesToPublishDirectory=true
    - C:\Windows\System32\inetsrv\appcmd stop site $SITE_NAME
    - 'xcopy /y /s $TEMP_DIR $PUBLISH_DIR'
    - C:\Windows\System32\inetsrv\appcmd start site $SITE_NAME
  only:
    - CI/CD
```

We have taken local dotnet and used to build, test and deploy it using local environment on local IIS server, in case of remote IIS Server we need configure connection with remote IIS, then do the followings :

For GitHub:

```yaml
name: .NET

on:
  push:
    branches: [ "CI/CD" ]
  pull_request:
    branches: [ "CI/CD" ]
```

```yaml
jobs:
  build_test_publish:
    runs-on: windows-latest
    defaults:
      run:
        working-directory: ./src
    steps:
    - uses: actions/checkout@v3

    - name: Setup .NET
      uses: actions/setup-dotnet@v3
      with:
        dotnet-version: 6.0.x

    - name: Restore dependencies
      run: dotnet restore

    - name: Build
      run: dotnet build --no-restore --configuration Release

    - name: Test
      run: dotnet test --no-build --verbosity normal --configuration Release

    - name: Publish
      run: dotnet publish ./Presentation/Nop.Web --no-build --output ./publish -c Release /p:CopyRefAssembliesToPublishDirectory=true

    - name: Zipping
      run: Compress-Archive -Path publish -DestinationPath ./published_with_source.zip -Force

    - name: Upload Artifact
      uses: actions/upload-artifact@v3
      with:
        name: published-artifact
        path: src/published_with_source.zip
        if-no-files-found: error
```

```yaml
deploy:
  needs: build_test_publish
  runs-on: self-hosted
  steps:
  - name: Download Artifact
    uses: actions/download-artifact@v3
    with:
      name: published-artifact
      path: C:\actions-runner\_work\downloaded-published

  - name: UnZipping
    run: Expand-Archive -Path C:\actions-runner\_work\downloaded-published\published_with_source.zip -DestinationPath C:\actions-runner\_work\downloaded-published

  - name: Stop Specific IIS Site
    run: C:\Windows\System32\inetsrv\appcmd stop site "test"

  - name: Publish to IIS
    run: xcopy /s /Y "C:\actions-runner\_work\downloaded-published\publish\*" "C:\Office\Projects\Deploy\test"

  - name: Start IIS
    run: C:\Windows\System32\inetsrv\appcmd start site "test"

  - name: Remove download contents
    run: Remove-Item -Path "C:\actions-runner\_work\downloaded-published" -Recurse -Force
```

Here, build stage and deploy are processing on GitHub.

Here,

- **`build_test_publish` Job:** This job performs the build, test, and publish steps for your nopCommerce project.
- **`runs-on`**: Specifies that the job will run on a Windows environment.
- **`defaults`**: Sets the working directory for subsequent steps to the "src" directory.
- **`steps`**: These are the individual steps that make up the build, test, and publish process:
- **`actions/checkout`**: Checks out your repository code.
- **`actions/setup-dotnet`**: Sets up the .NET environment for the job.
- **`dotnet restore`**: Restores project dependencies.
- **`dotnet build`**: Builds the project in Release configuration.
- **`dotnet test`**: Runs tests in the Release configuration.
- **`dotnet publish`**: Publishes the nopCommerce project to a "publish" directory.

- **`Compress-Archive`**: Creates a ZIP archive containing the published output.
- **`actions/upload-artifact`**: Uploads the ZIP archive as an artifact for later use.
- **`deploy Job:`** This job handles the deployment of the published nopCommerce project to an IIS server.
- **`needs`**: Specifies that this job depends on the successful completion of the previous **`build_test_publish`** job.
- **`runs-on`**: Specifies that the job should run on a self-hosted runner (assuming you have a runner set up).
- **`steps`**: These steps perform the deployment process:
- **`actions/download-artifact`**: Downloads the artifact generated in the previous job.
- **`Expand-Archive`**: Unzips the downloaded archive.
- **`appcmd stop site`**: Stops a specific IIS site named "test".
- **`xcopy`**: Copies the contents of the unzipped publish directory to an IIS site directory.
- **`appcmd start site`**: Starts the IIS site named "test".
- **`Remove-Item`**: Removes the downloaded content to clean up.

`Plane Text Code:`

`.gitlab-ci.yml (Gitlab)`

```
stages:
  - build
  - test
  - deploy

variables:
  BUILD_DIR: "D:\\TeamMahbubVai\\nopcommerce-4.6\\src"
  TEMP_DIR : "D:\\artifacts"
  PUBLISH_DIR: "C:\\inetpub\\wwwroot\\test"
  SITE_NAME: "test"


build_job:
  stage: build
  script:
   - cd $BUILD_DIR
```

```
  - dotnet clean --configuration Release
  - dotnet restore
  - dotnet build --no-restore -c Release

test_job:
  stage: test
  script:
    - cd $BUILD_DIR
    - dotnet test

deploy_job:
  stage: deploy
  script:
    - cd $BUILD_DIR
    - dotnet clean --configuration Release
    - dotnet restore
    - dotnet build --no-restore -c Release
    - cd Presentation/Nop.Web/
    - dotnet publish Nop.Web.csproj --no-build -c Release -o $TEMP_DIR
/p:CopyRefAssembliesToPublishDirectory=true
    - C:\Windows\System32\inetsrv\appcmd stop site $SITE_NAME
    - 'xcopy /y /s $TEMP_DIR $PUBLISH_DIR'
    - C:\Windows\System32\inetsrv\appcmd start site $SITE_NAME
  only:
    - CI/CD
```

Corns: Gitlab Runner is a slow while completing jobs in GitLab than GitHub