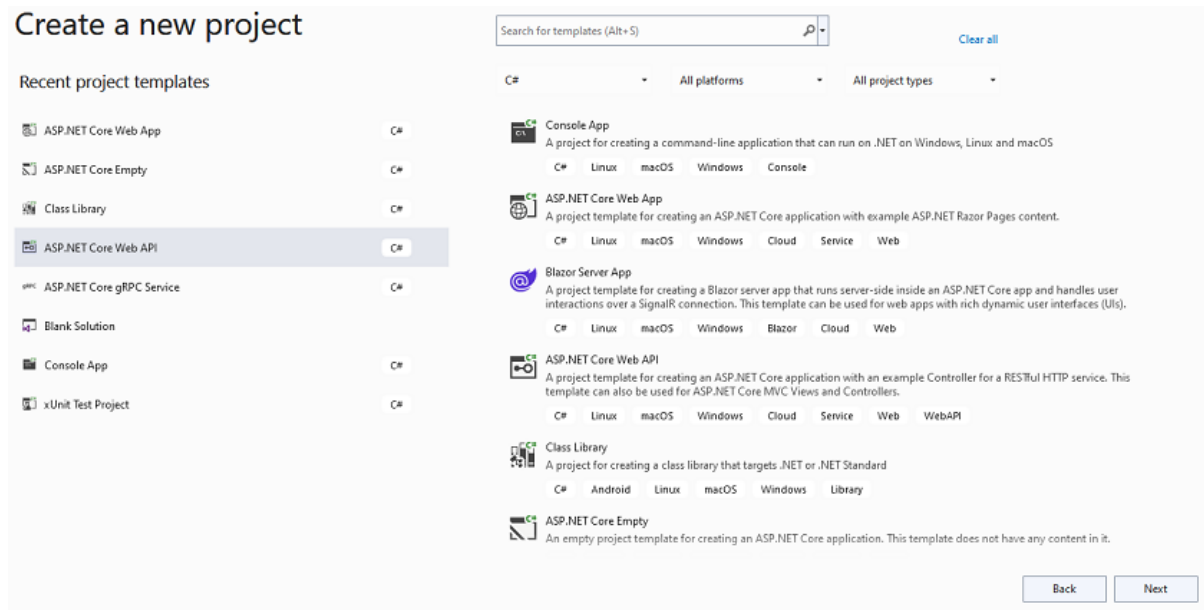


Step by Step Implementation of .NET Core Application

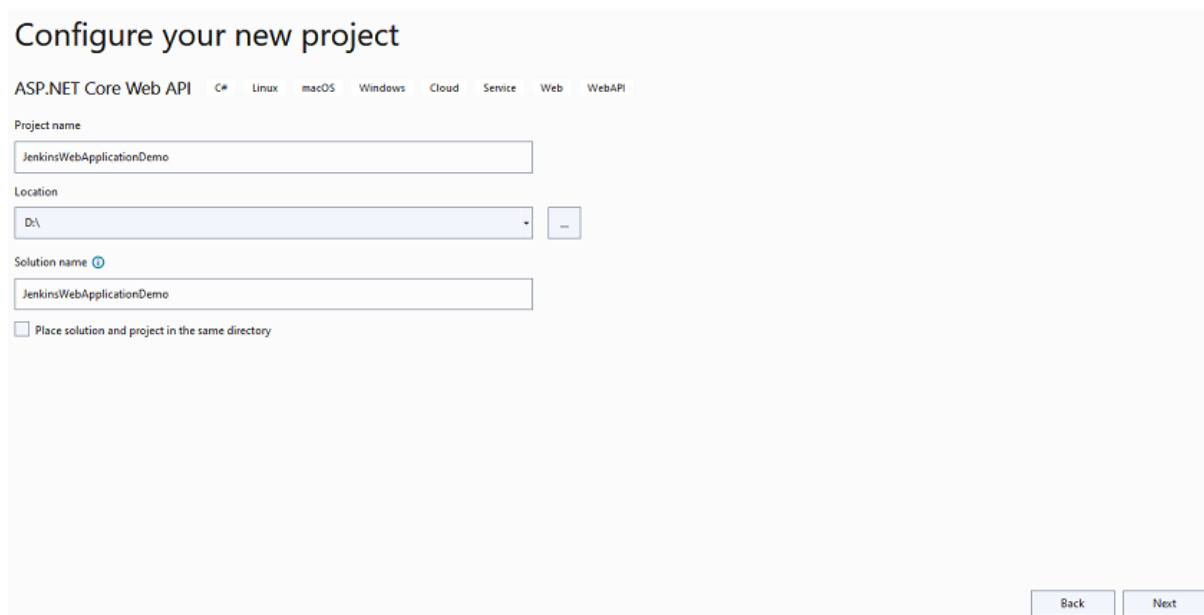
Step 1

Create a new Web API



Step 2

Configure the application



Step 3

Provide additional information

Additional information

ASP.NET Core Web API C# Linux macOS Windows Cloud Service Web WebAPI

Framework ⓘ
[.NET 6.0 (Long-term support)]

Authentication type ⓘ
[None]

☐ Configure for HTTPS ⓘ
☐ Enable Docker ⓘ

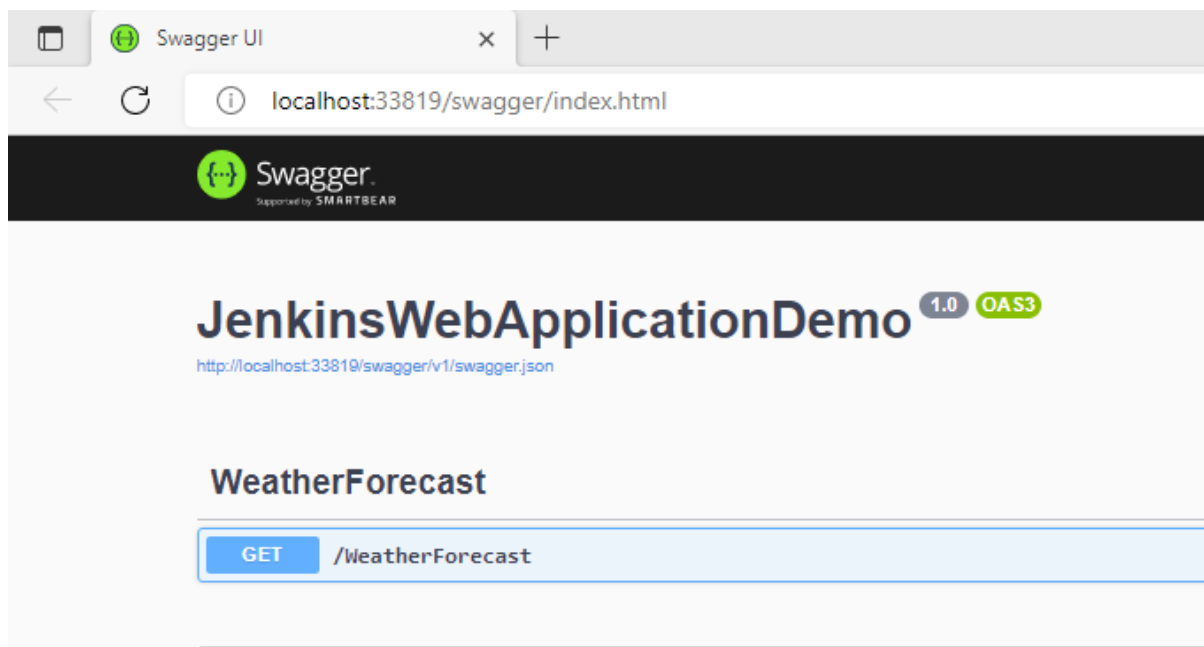
Docker OS ⓘ
[Linux]

☒ Use controllers (unchecked to use minimal APIs) ⓘ
☒ Enable OpenAPI support ⓘ
☐ Do not use top-level statements ⓘ

Back Create

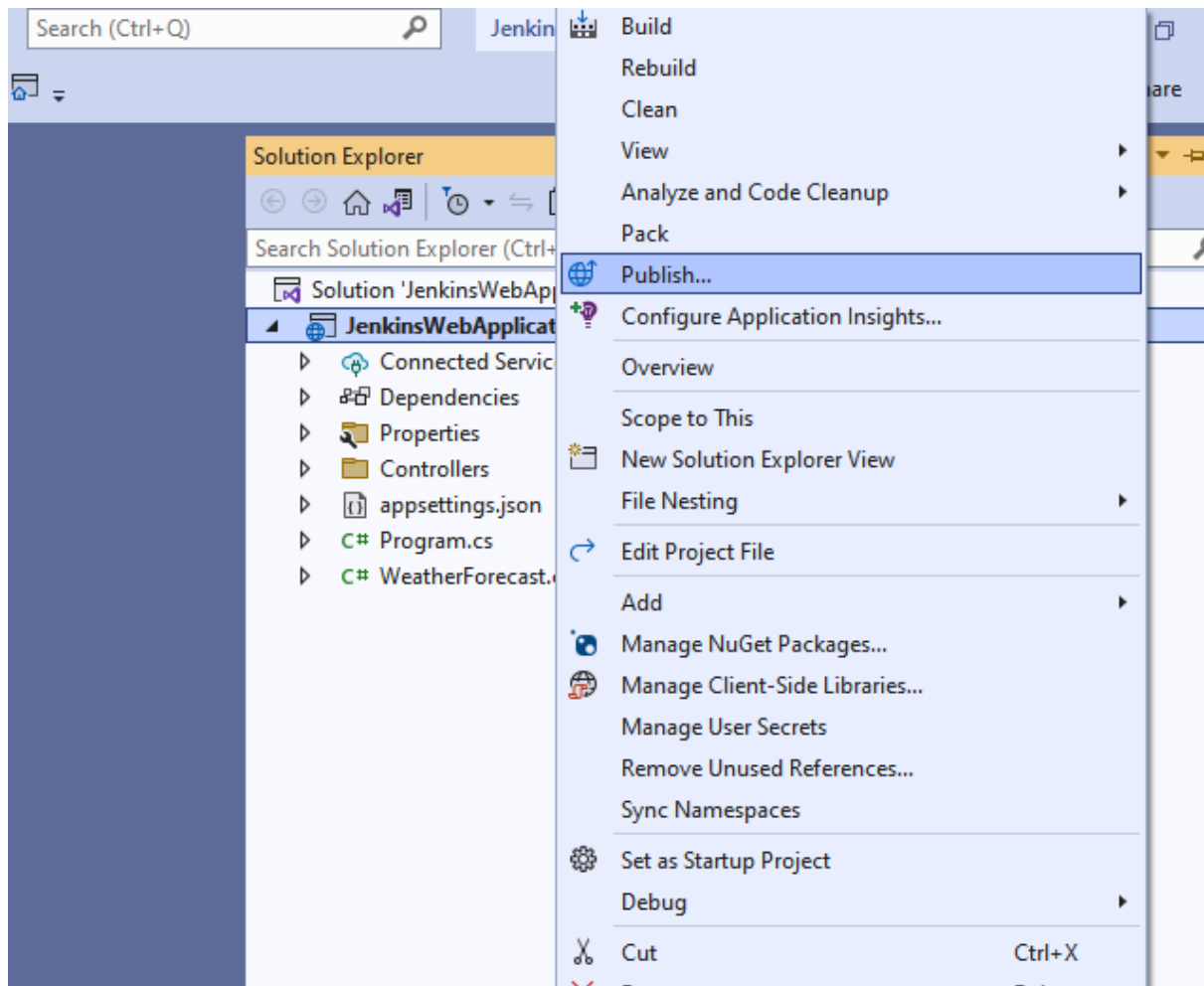
Step 4

Finally, run the application



Step 5

Now we are going to publish this code



Step 6

Select folder

Publish

Where are you publishing today?

Target

**Azure**

Publish your application to the Microsoft cloud

**Docker Container Registry**

Publish your application to any supported Container Registry that works with Docker images

**Folder**

Publish your application to a local folder or file share

**FTP/FTPS Server**

Publish your application to an FTP/FTPS server

**Web Server (IIS)**

Publish your application to IIS using Web Deploy or Web Deploy Package

**Import Profile**

Import your publish settings to deploy your app

Back

Next

Finish

Cancel

Step 7

It will take default publish path

Publish

Provide the path to a local or network folder

Target

Location

Folder location

bin\Release\net6.0\publish\

Browse...

For local folders you can provide either a full path or a relative path to the project, for example:

- publish\ (relative path)
- C:\Users\Username\Documents (full path)

For network folders you have to use \\ and then either the computer name or IP address, for example:

- \\server1\fileshare1
- \\192.168.1.17\fileshare1

Back

Next

Finish

Cancel

Step 8

Finally, publish

FolderProfile.pubxml

Folder

Publish

+ New More actions

Ready to publish.

Settings

| | |
|-----------------------|-----------------------------|
| Target location | bin\Release\net6.0\publish\ |
| Delete existing files | False |
| Configuration | Release |
| Target Runtime | Portable |

Show all settings

Step 9

Go to the property section of project inside solution and edit FolderProfile.pubxml file and change the web publish method to Package



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <!--
3 https://go.microsoft.com/fwlink/?LinkID=208121.
4 -->
5 <Project>
6   <PropertyGroup>
7     <DeleteExistingFiles>False</DeleteExistingFiles>
8     <ExcludeApp_Data>False</ExcludeApp_Data>
9     <LaunchSiteAfterPublish>True</LaunchSiteAfterPublish>
10    <LastUsedBuildConfiguration>Release</LastUsedBuildConfiguration>
11    <LastUsedPlatform>Any CPU</LastUsedPlatform>
12    <PublishProvider>FileSystem</PublishProvider>
13    <PublishUrl>bin\Release\net6.0\publish\</PublishUrl>
14    <WebPublishMethod>Package</WebPublishMethod>
15  </PropertyGroup>
16 </Project>
```

Step 10

We also need Microsoft Web Deploy, and most of time it will come with Visual Studio

Step 11

Later on, we just create an empty xUnit Test Case project just for demo purposes inside the same solution

Step 11

Run your project and confirm all things are working fine on your system before committing to git.

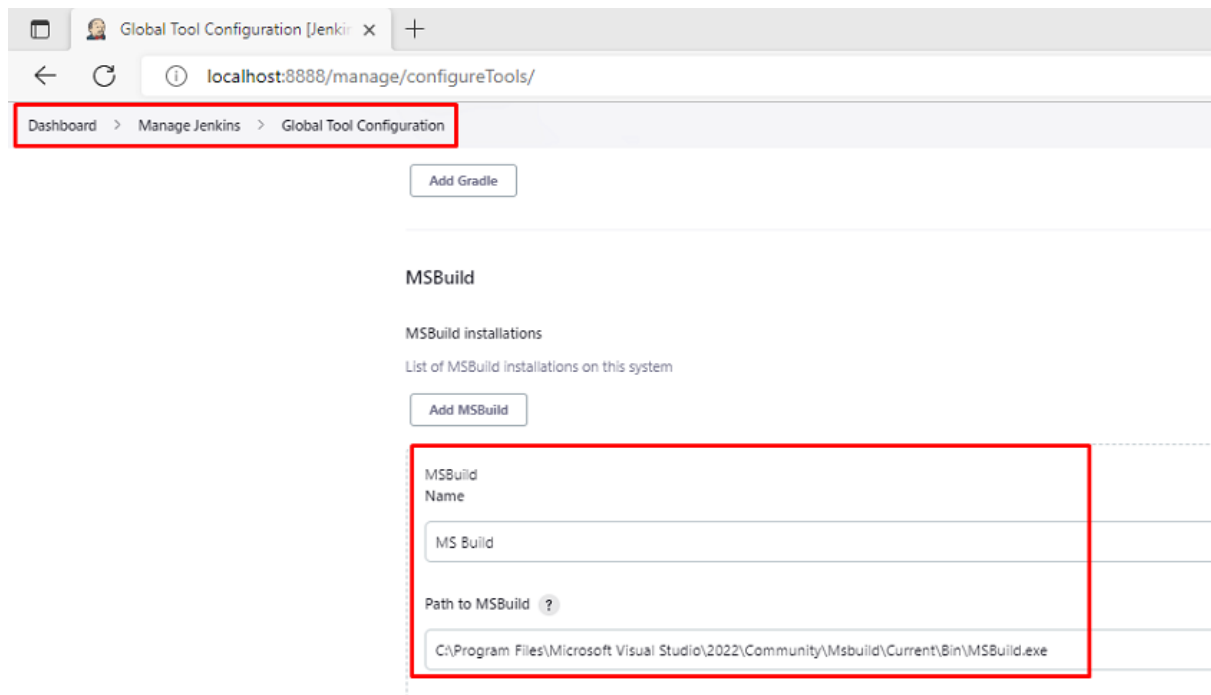
Step 12

Next, create a git repository and push your code into that

Jenkins Pipeline and Configuration

Step 1

First, we are going to add MS Build Path and Configuration for that go to the Global Tool Configuration inside the Manage Jenkins



The screenshot shows the Jenkins web interface at `localhost:8888/manage/configureTools/`. The breadcrumb navigation at the top is `Dashboard > Manage Jenkins > Global Tool Configuration`, which is highlighted with a red box. Below the navigation, there is an `Add Gradle` button. The main section is titled `MSBuild` and contains the heading `MSBuild installations` with the subtitle `List of MSBuild installations on this system`. Below this is an `Add MSBuild` button. A form for adding a new MSBuild installation is highlighted with a red box. It includes a `Name` field with the value `MS Build`, a `Path to MSBuild` field with the value `C:\Program Files\Microsoft Visual Studio\2022\Community\MSbuild\Current\Bin\MSBuild.exe`, and a help icon next to the path label.

Global Tool Configuration [Jenkins] x +

localhost:8888/manage/configureTools/

Dashboard > Manage Jenkins > Global Tool Configuration

Add Gradle

MSBuild

MSBuild installations

List of MSBuild installations on this system

Add MSBuild

MSBuild

Name

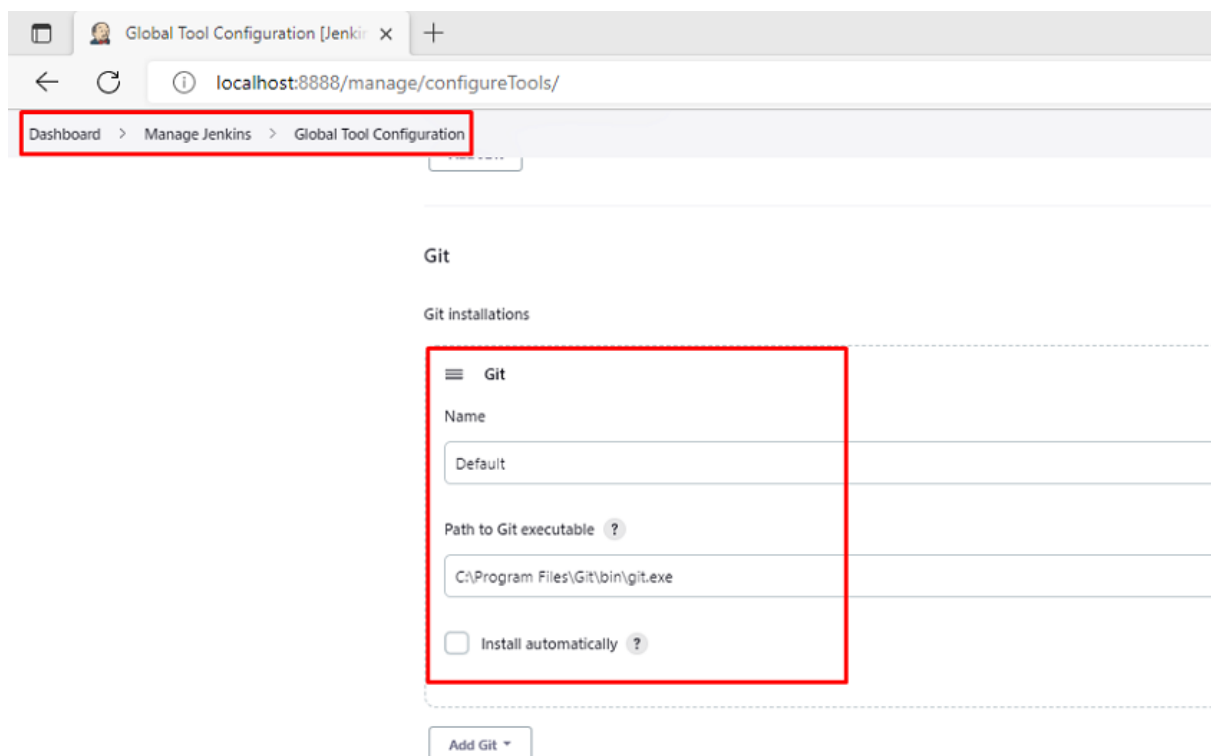
MS Build

Path to MSBuild ?

C:\Program Files\Microsoft Visual Studio\2022\Community\MSbuild\Current\Bin\MSBuild.exe

Step 2

Next, we add the git file path inside the global tool configuration



The screenshot shows the Jenkins web interface at `localhost:8888/manage/configureTools/`. The breadcrumb navigation at the top is `Dashboard > Manage Jenkins > Global Tool Configuration`, which is highlighted with a red box. Below the navigation, there is an `Add Gradle` button. The main section is titled `Git` and contains the heading `Git installations`. A form for adding a new Git installation is highlighted with a red box. It includes a `Name` field with the value `Default`, a `Path to Git executable` field with the value `C:\Program Files\Git\bin\git.exe`, and an `Install automatically` checkbox which is unchecked. There are help icons next to the path label and the checkbox label. Below the form is an `Add Git` button.

Global Tool Configuration [Jenkins] x +

localhost:8888/manage/configureTools/

Dashboard > Manage Jenkins > Global Tool Configuration

Add Gradle

Git

Git installations

Git

Name

Default

Path to Git executable ?

C:\Program Files\Git\bin\git.exe

☐ Install automatically ?

Add Git

Step 3

Now, we are going to add git credentials in Manage Credentials inside Manage Jenkins click and **global credentials(unrestricted)** add and your git credentials like username and password it will create credential id automatically once you click on save and that id you need to put inside pipeline configuration.

Step 4

Open Jenkins Dashboard and click on New Item to create a Pipeline

The pipeline is basically the set of steps that are going to execute before creating the build that includes unit test, integration test case, and many more as per need

New Item [Jenkins] × +

localhost:8888/view/all/newJob

Jenkins Search (CTRL+K)

Dashboard > All >

Enter an item name

TestPipeline

=> Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) for organizing complex activities that do not easily fit in free-style job type.

Step 5

Provide general information like a description

TestPipeline Config [Jenkins] × +

← ↻ ⓘ localhost:8888/job/TestPipeline/configure

Jenkins

Dashboard > TestPipeline >

Configuration

- General
- Advanced Project Options
- Pipeline

General

Description

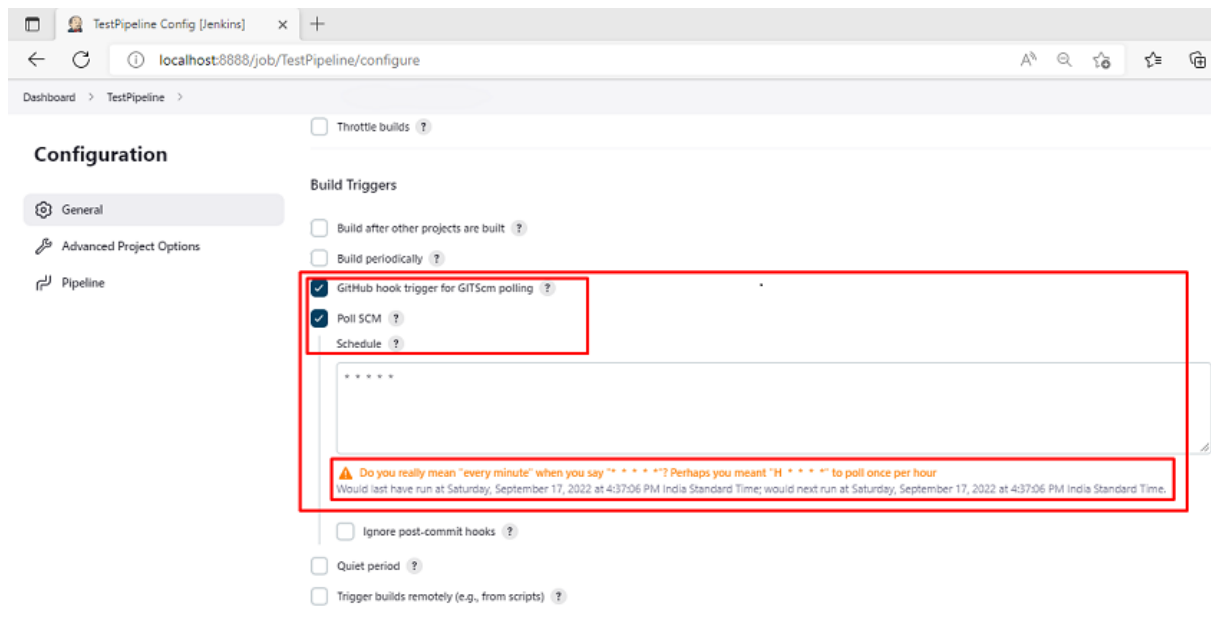
TestPipeline

[Plain text] [Preview](#)

- ☐ Discard old builds ?
- ☐ Do not allow concurrent builds
- ☐ Do not allow the pipeline to resume if the controller restarts
- ☐ GitHub project
- ☐ Pipeline speed/durability override ?
- ☐ Preserve stashes from completed builds ?
- ☐ This project is parameterized ?
- ☐ Throttle builds ?

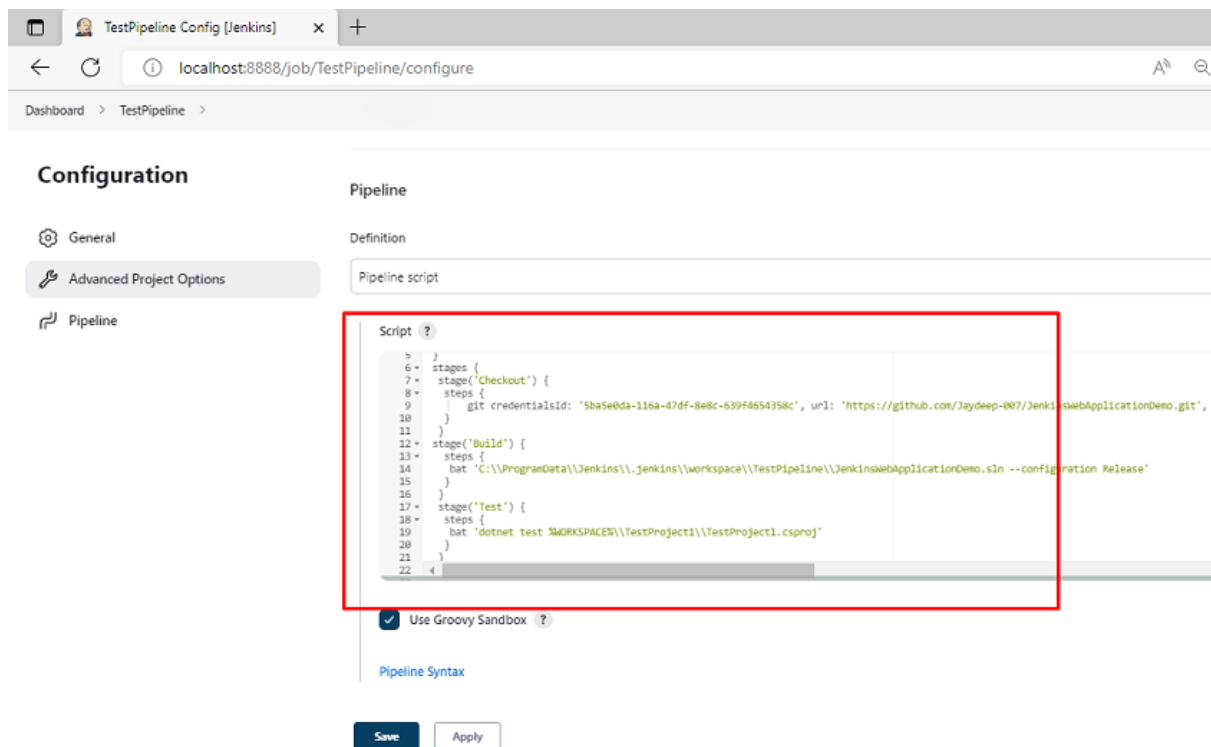
Step 6

Next, click on a few checkboxes which help us to automatically trigger the pipeline whenever we commit code inside git repositories and put five start separated by spaces that is us to trigger pipeline instantly whenever the developers commit the code



Step 7

Add the following pipeline script which is basically a set of steps that are going to executes before creating the build (Note- Please make sure all your paths are corrects otherwise it will throw errors in the build process)



```

pipeline { agent any environment { dotnet = 'C:\\Program
Files\\dotnet\\dotnet.exe' } stages { stage('Checkout Stage')
{ steps { git credentialsId:
'5ba5e0da-116a-47df-8e8c-639f4654358c', url:
'https://github.com/Jaydeep-007/JenkinsWebApplicationDemo.git'
, branch: 'main' } } stage('Build Stage') { steps { bat
'C:\\ProgramData\\Jenkins\\.jenkins\\workspace\\TestPipeline\\
JenkinsWebApplicationDemo.sln --configuration Release' } }
stage('Test Stage') { steps { bat 'dotnet test
%WORKSPACE%\\TestProject1\\TestProject1.csproj' } }
stage("Release Stage") { steps { bat 'dotnet build
%WORKSPACE%\\JenkinsWebApplicationDemo.sln /p:PublishProfile="
%WORKSPACE%\\JenkinsWebApplicationDemo\\Properties\\PublishPro
files\\FolderProfile.pubxml" /p:Platform="Any CPU"
/p:DeployOnBuild=true /m' } } stage('Deploy Stage') { steps {
//Deploy application on IIS bat 'net stop "w3svc"' bat
'"C:\\Program Files (x86)\\IIS\\Microsoft Web Deploy
V3\\msdeploy.exe" -verb:sync
-source:package="%WORKSPACE%\\JenkinsWebApplicationDemo\\bin\\
Debug\\net6.0\\JenkinsWebApplicationDemo.zip" -dest:auto
-setParam:"IIS Web Application Name"="Demo.Web"
-skip:objectName=filePath,absolutePath=".\\\\"PackagDemoeTmp\\"
\\Web.config$" -enableRule:DoNotDelete -allowUntrusted=true'
bat 'net start "w3svc"' } } } }

```

JavaScriptCopy

Step 8

Open the IIS and create empty website and point to the any empty folder in that our pipeline will publish and deploy our code after build is getting completed

Add Website ? X

Site name: Demo.Web Application pool: Demo.Web Select...

Content Directory

Physical path: D:\IISDemo ...

Pass-through authentication

Connect as... Test Settings...

Binding

Type: http IP address: All Unassigned Port: 5152

Host name:

Example: www.contoso.com or marketing.contoso.com

☒ Start Website immediately

OK Cancel

Step 9

After that go to the dashboard and run your pipeline

The screenshot shows the Jenkins Dashboard interface. The top navigation bar includes the Jenkins logo and a search bar. The left sidebar contains various navigation links: New Item, People, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins, and My Views. The main content area displays a table of builds. The 'TestPipeline' build is highlighted, and a context menu is open over it, showing options such as 'Build Now', 'Configure', 'Delete Pipeline', 'Full Stage View', 'Rename', 'Pipeline Syntax', 'GitHub Hook Log', and 'Git Polling Log'. The 'Build Queue' section on the left indicates 'No builds in the queue'. The 'Build Executor Status' section shows two executors in an 'Idle' state.

| S | W | Name | Last Success |
|---|----|--------------|----------------|
| ✓ | ⚙️ | TestPipeline | 9 min 3 sec #1 |

- </> Changes
- ▶ Build Now
- ⚙️ Configure
- 🗑 Delete Pipeline
- 🔍 Full Stage View
- ✏ Rename
- ❓ Pipeline Syntax
- 📄 GitHub Hook Log
- 📄 Git Polling Log

Step 10

Here you can see the build is started



Dashboard >

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

All

+

S W Name ↓



TestPipeline

Icon: S M L

Build Queue



No builds in the queue.

Build Executor Status



1 Idle

2

TestPipeline

#2

(Release)



TestPipeline #2 Console [Jenkins] x +

localhost:8888/job/TestPipeline/2/console

Jenkins Search (CTRL+K)

Dashboard > TestPipeline > #2

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#2'

Git Build Data

Restart from Stage

Replay

Pipeline Steps

Workspaces

Previous Build

Console Output

```
Started by user Jaydeep Patil
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\TestPipeline
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Checkout)
[Pipeline] git
The recommended git tool is: NONE
using credential 5b5ae0da-116a-47df-8e8c-639f4654358c
> C:\Program Files\Git\bin\git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\TestPipeline
Fetching changes from the remote Git repository
> C:\Program Files\Git\bin\git.exe config remote.origin.url https://github.com/Jaydeep-007/JenkinsWebApplicationDemo.git
Fetching upstream changes from https://github.com/Jaydeep-007/JenkinsWebApplicationDemo.git
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> git --version # 'git version 2.36.1.windows.1'
using GIT_ASKPASS to set credentials
> C:\Program Files\Git\bin\git.exe fetch --tags --force --progress -- https://github.com/Jaydeep-007/JenkinsWebAp
+timeout=10
```

TestPipeline #2 [Jenkins] x +

localhost:8888/job/TestPipeline/2/

Jenkins

Dashboard > TestPipeline > #2

Status

Changes

Console Output

Edit Build Information

Delete build '#2'

Git Build Data

Restart from Stage

Replay

Pipeline Steps

Workspaces

Previous Build

Build #2 (Sep 17, 2022, 4:44:44 PM)

Started by user [Jaydeep Patil](#)

git

Revision: 01698a033a68b90ba7dea88066d4c1ddad9c6fb2
Repository: <https://github.com/Jaydeep-007/JenkinsWebApplicationDemo.git>

- refs/remotes/origin/main

Now whenever you change add commit the code then build is triggered and the pipeline is going to create a new build for us and deploy it inside the IIS. This process is continuously running. Also, if the build failed due to some test cases and

something like that then you can also configure the email using SMTP protocol inside Jenkins that helps us build

Step 11

Finally, Open IIS and you can now access your application with the latest build

Step 12

Browse the URL from IIS and you can see the application is in running mode

