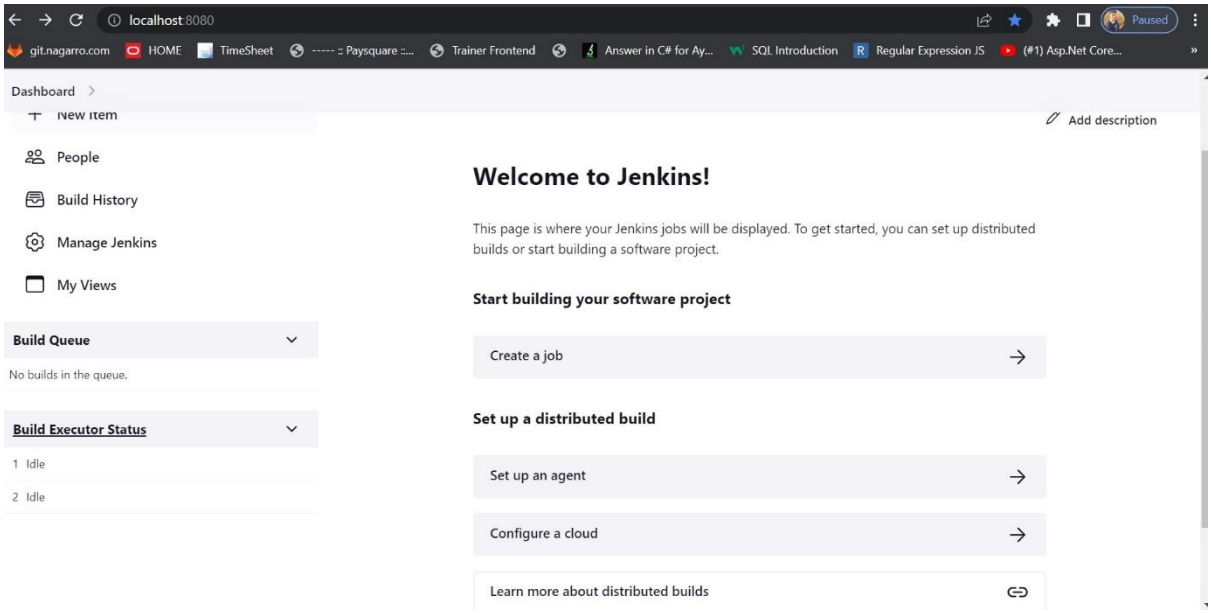
DEVOPS ASSIGNMENT

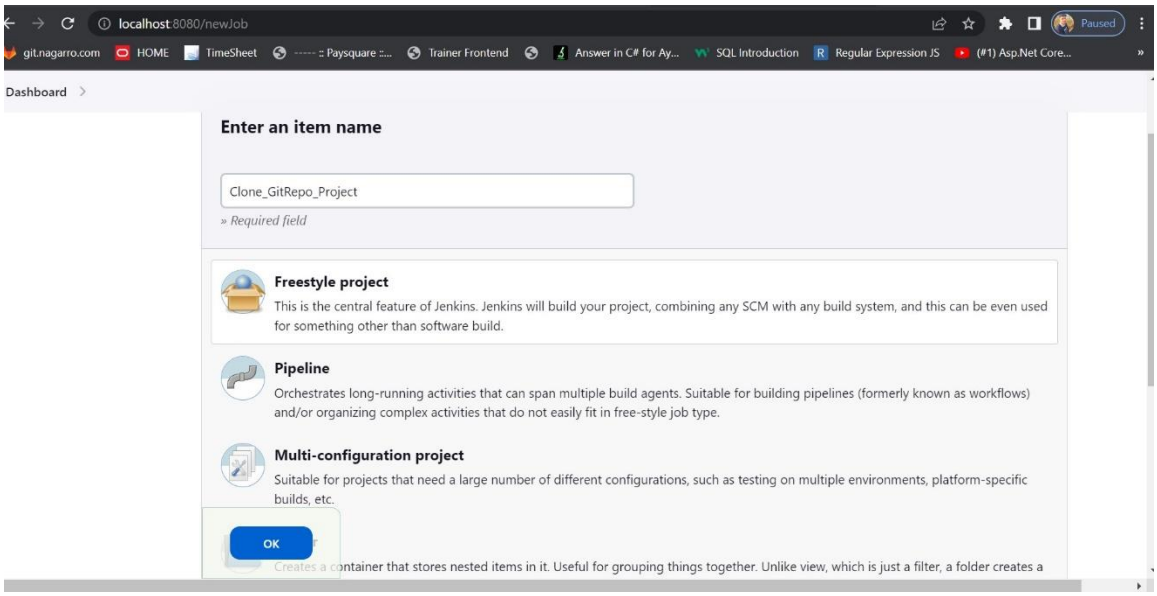
Assignment - Installation and Integration of the build tools:

1. **Create freestyle Jenkins job to download code from Git.**

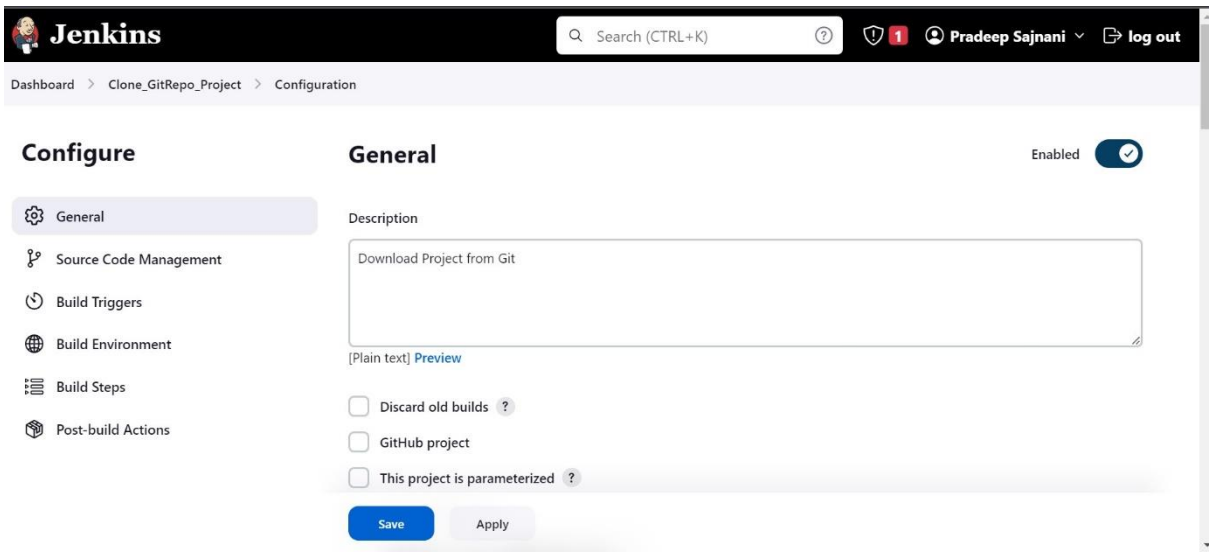
STEP 1: Login into Jenkins which is installed on your machine.

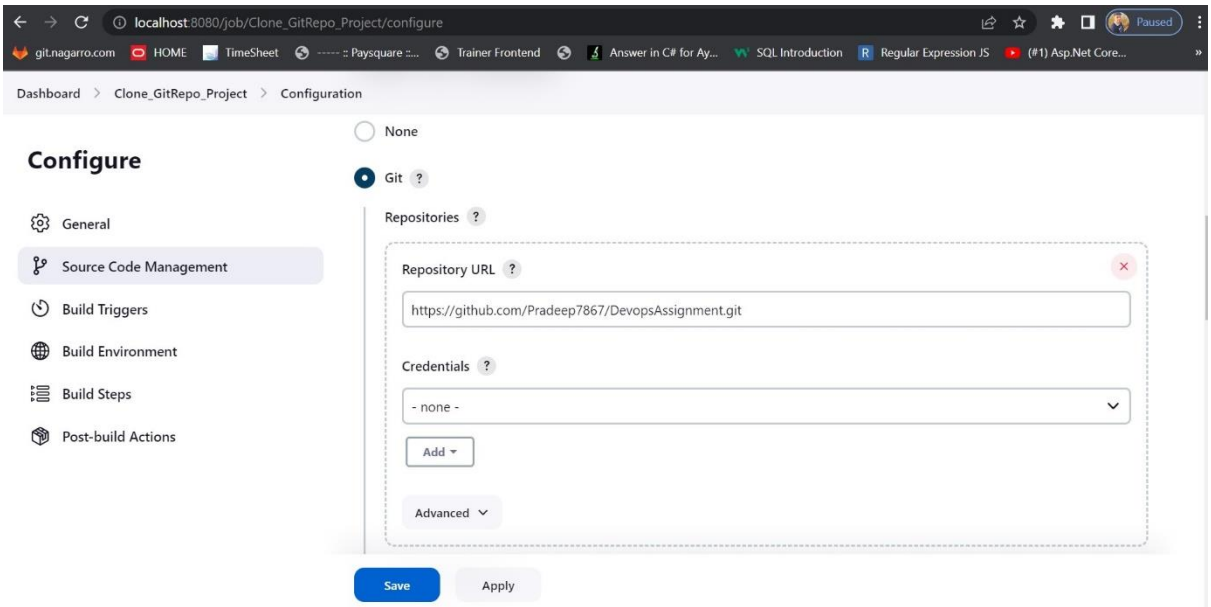


STEP 2: Create a Job or a new item in that create a freestyle project give a name to it eg: **Clone_GitRepo_project**

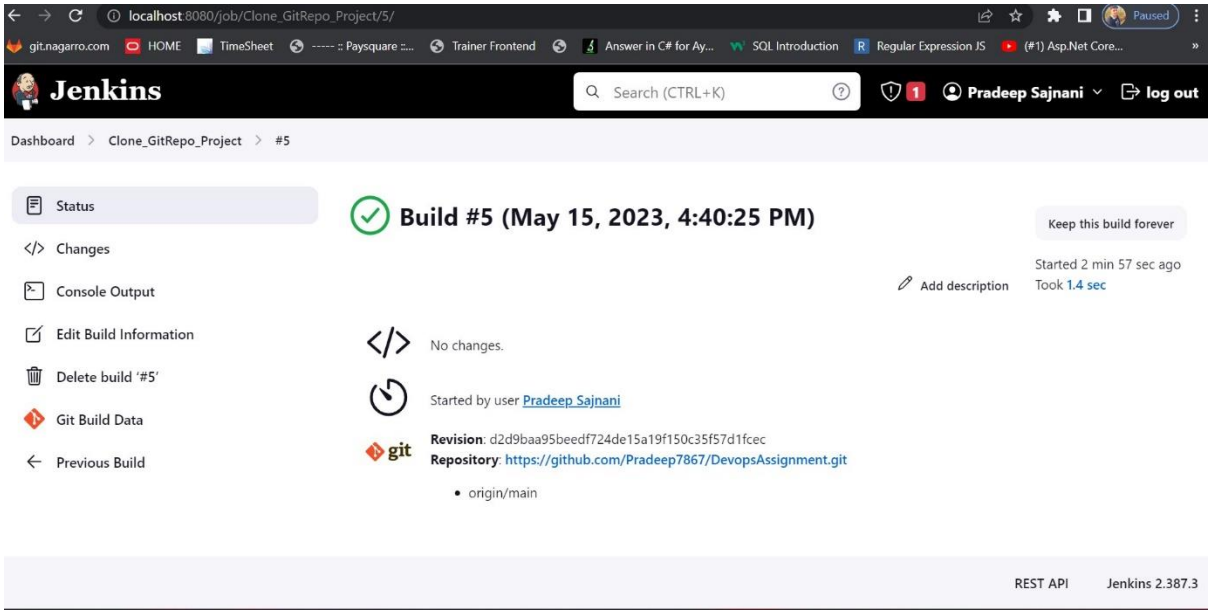


STEP 3: After that Add Configuration and select Git and the add your git repository which you want to clone from the GitHub

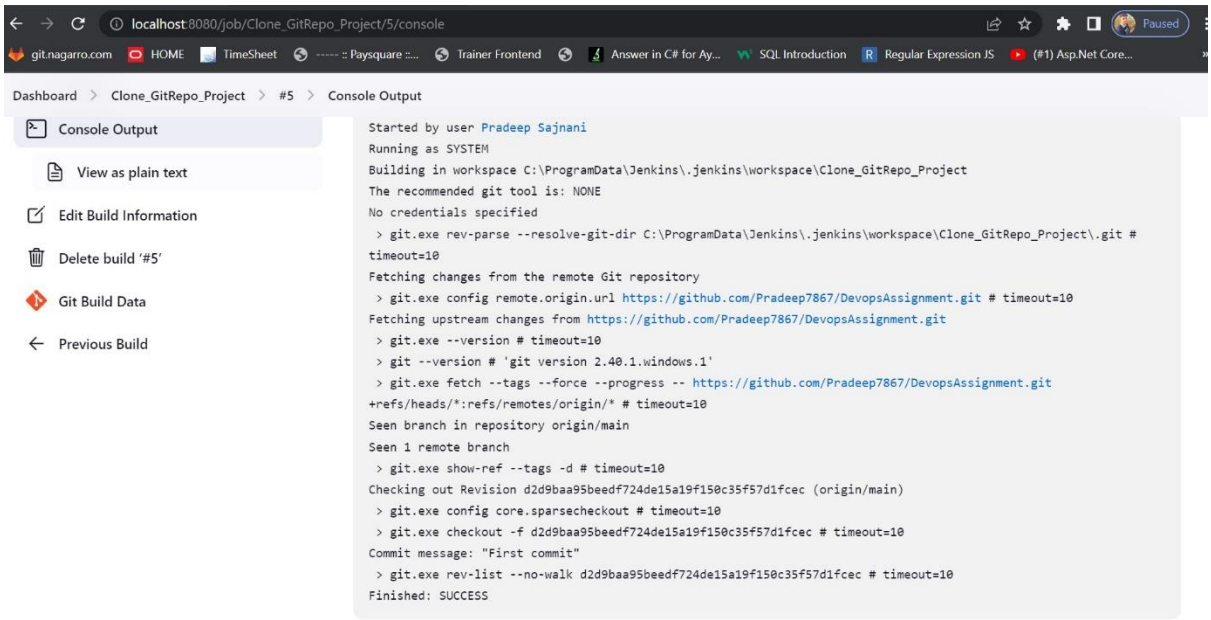




STEP: After adding the configuration click apply & save after that go to the dashboard where you will se your all projects then clickon Build now.



If You want to see the whole process You can see it Under Console Output

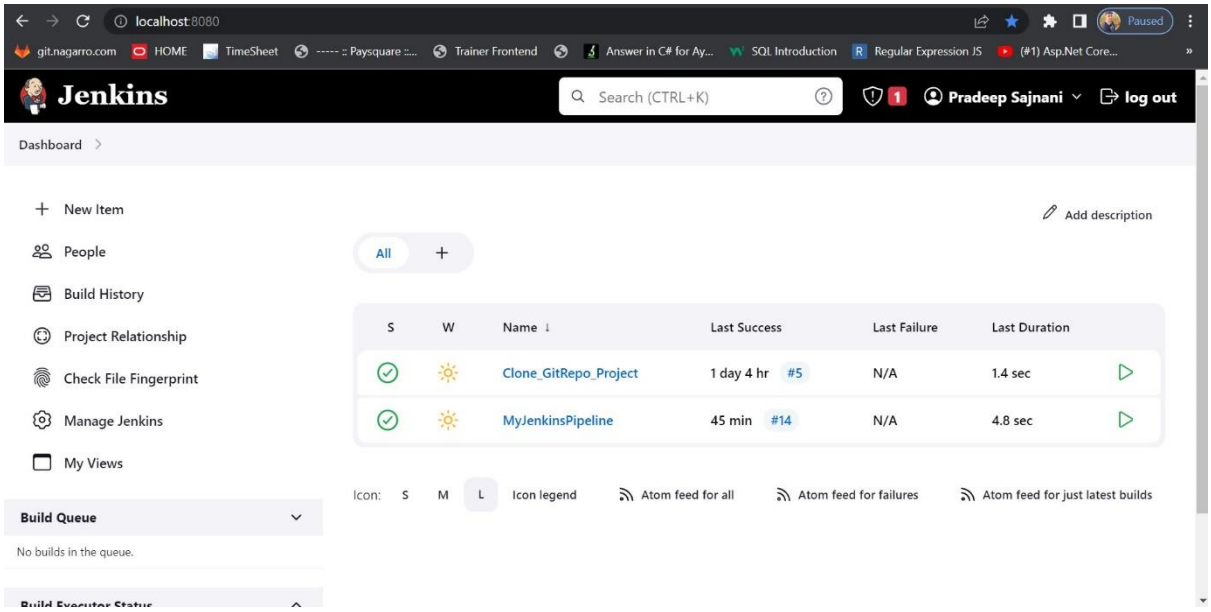


That's It

2. Create pipeline to build and test basic .net project code.

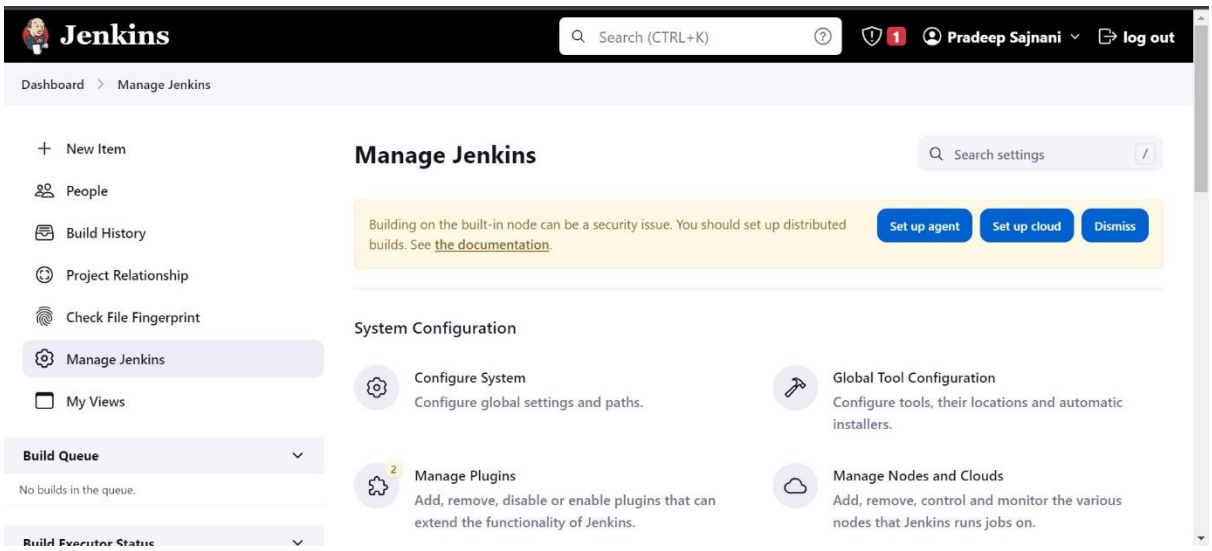
Install and Set up Jenkins:

Install Jenkins on your server or machine by downloading it from the official Jenkins website and following the installation instructions for your operating system. Once installed, access the Jenkins dashboard by opening a web browser and navigating to **http://localhost:8080** (assuming Jenkins is running on your local machine). Follow the on-screen instructions to complete the initial setup, including creating an admin user and installing recommended plugins.



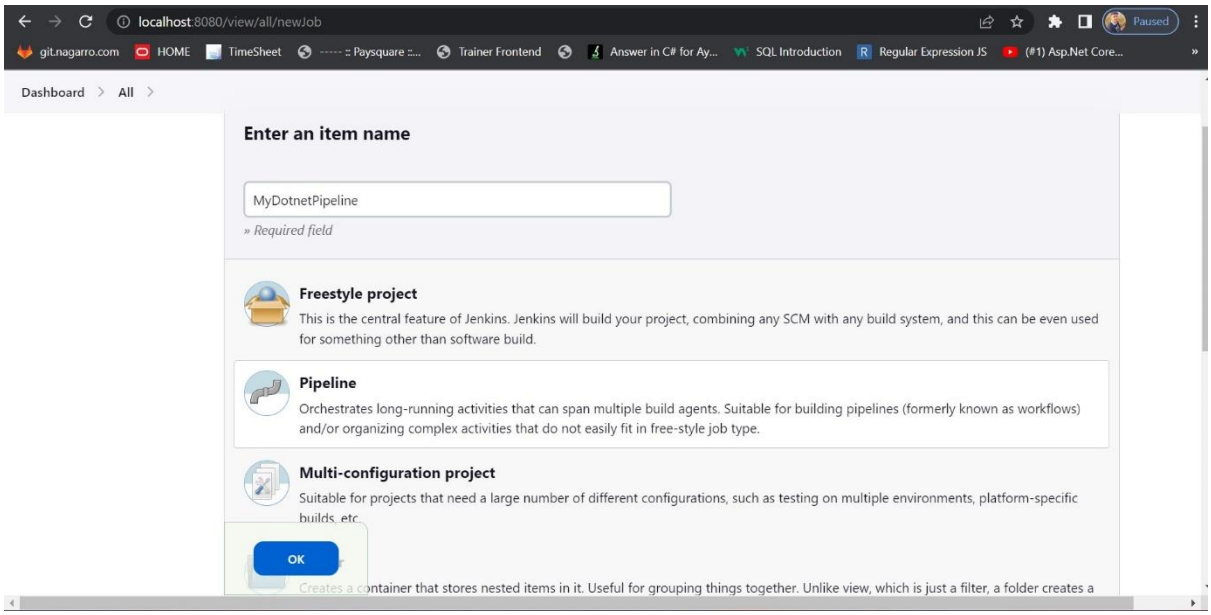
Configure Jenkins Global Tools:

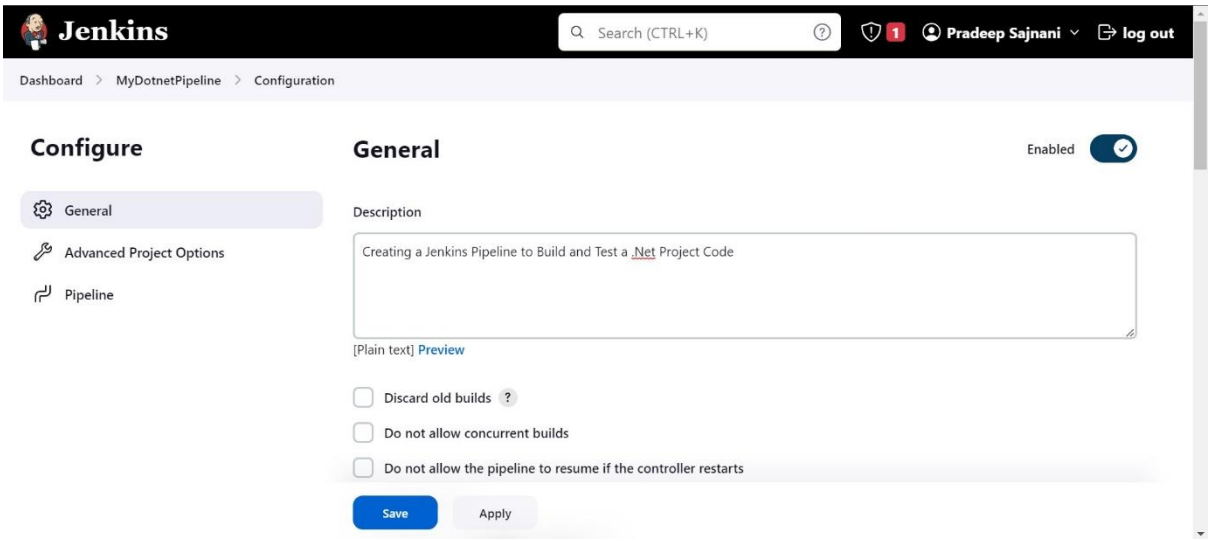
In the Jenkins dashboard, go to "Manage Jenkins" -> "Global Tool Configuration." Scroll down to the "DotNet SDK" section and configure the .NET Core SDK path by specifying the installed SDK location.



Create a New Jenkins Pipeline Job:

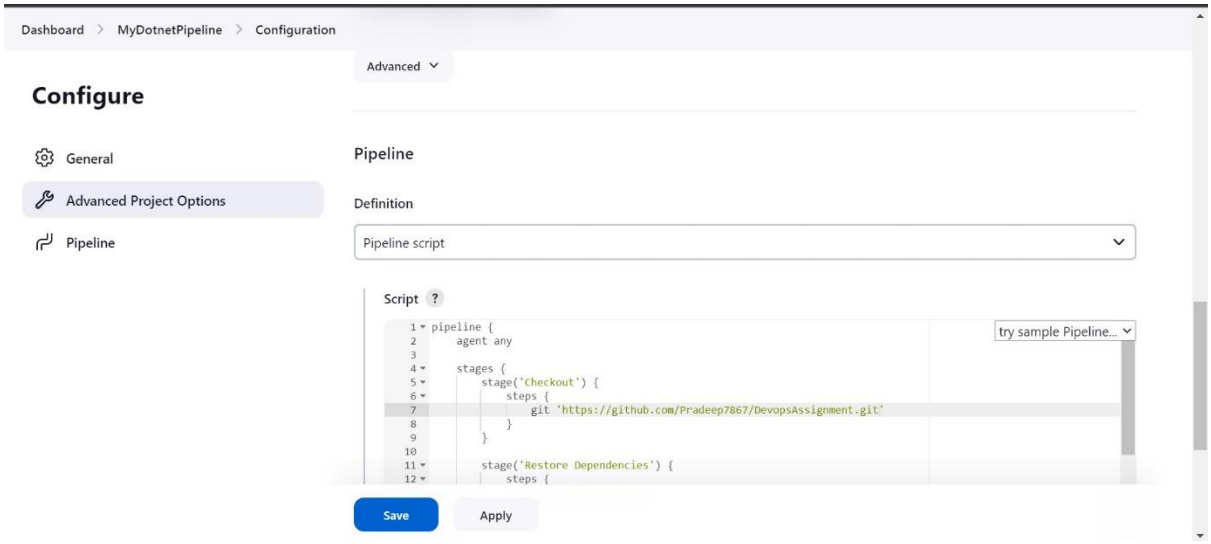
In the Jenkins dashboard, click on "New Item" to create a new Jenkins job. Enter a name for the job (e.g., "MyDotNetPipeline"), select "Pipeline" as the job type, and click "OK" to proceed.





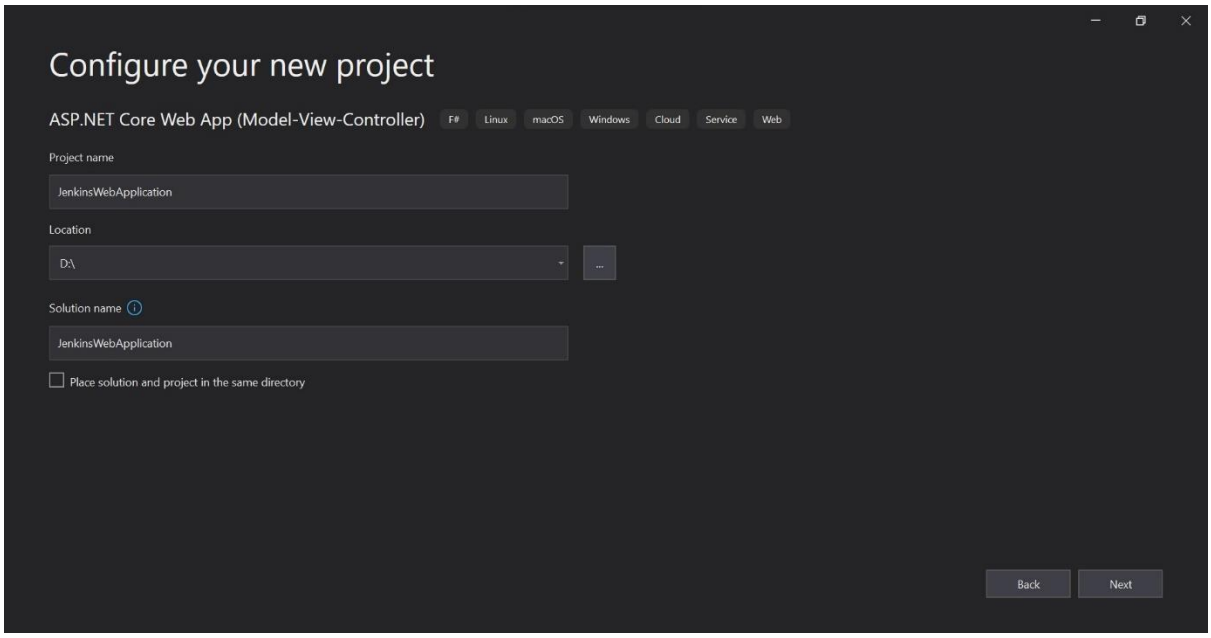
Configure Pipeline Job:

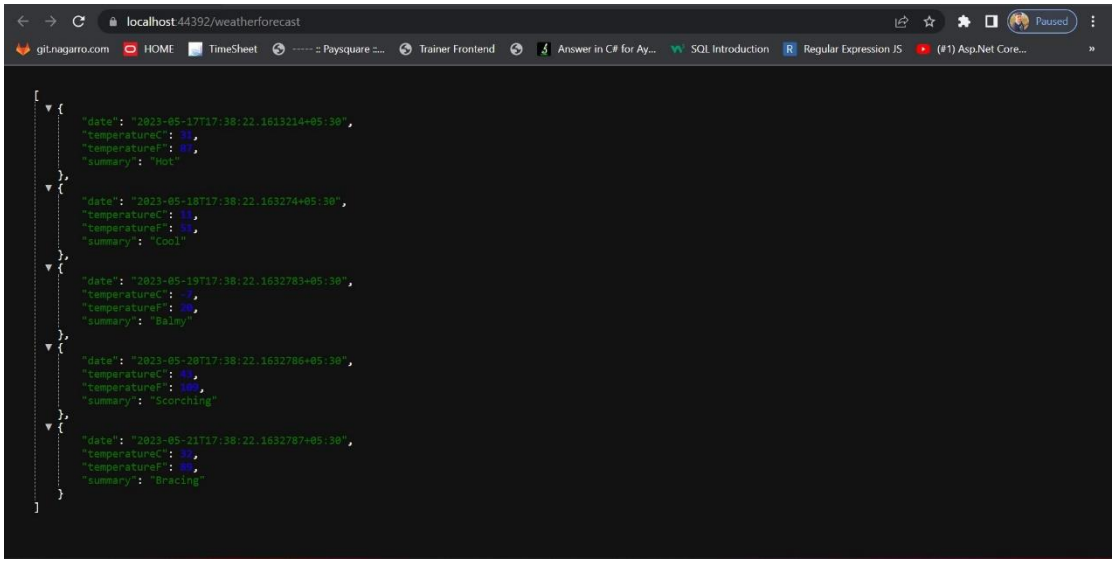
In the pipeline job configuration, scroll down to the "Pipeline" section and select the "Pipeline script" option. Here, you can define the pipeline script using the Jenkinsfile syntax.



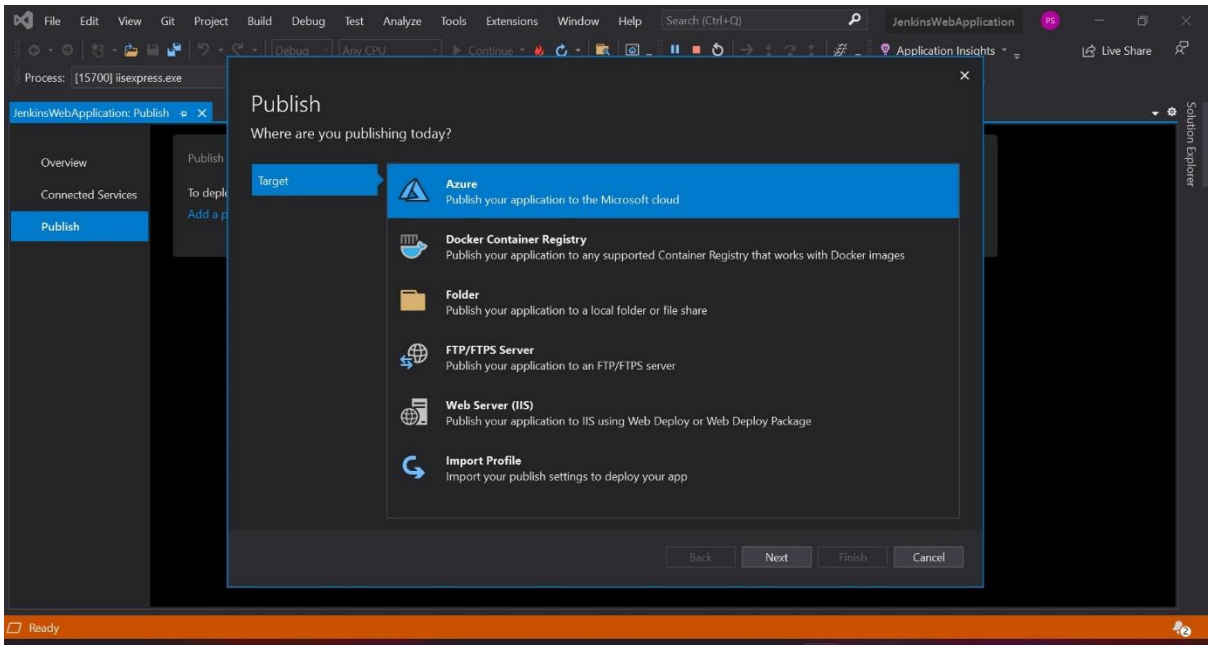
Now save And Apply

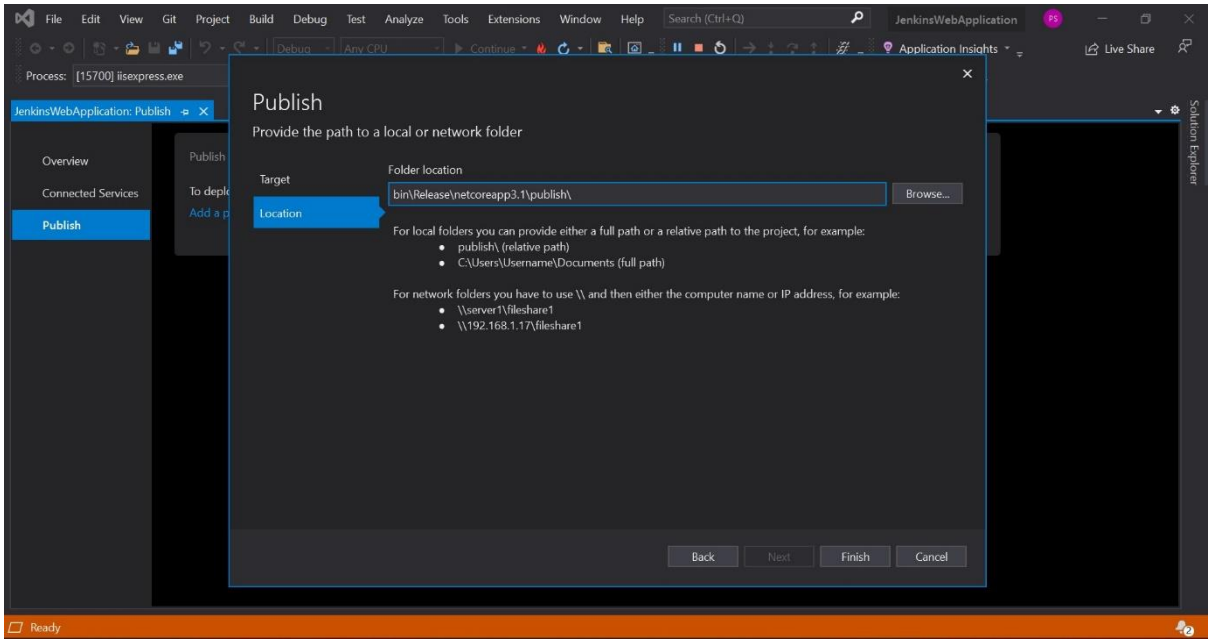
After that create a .Net Web API under Visual Studio 2019 and create a basic API or user default one

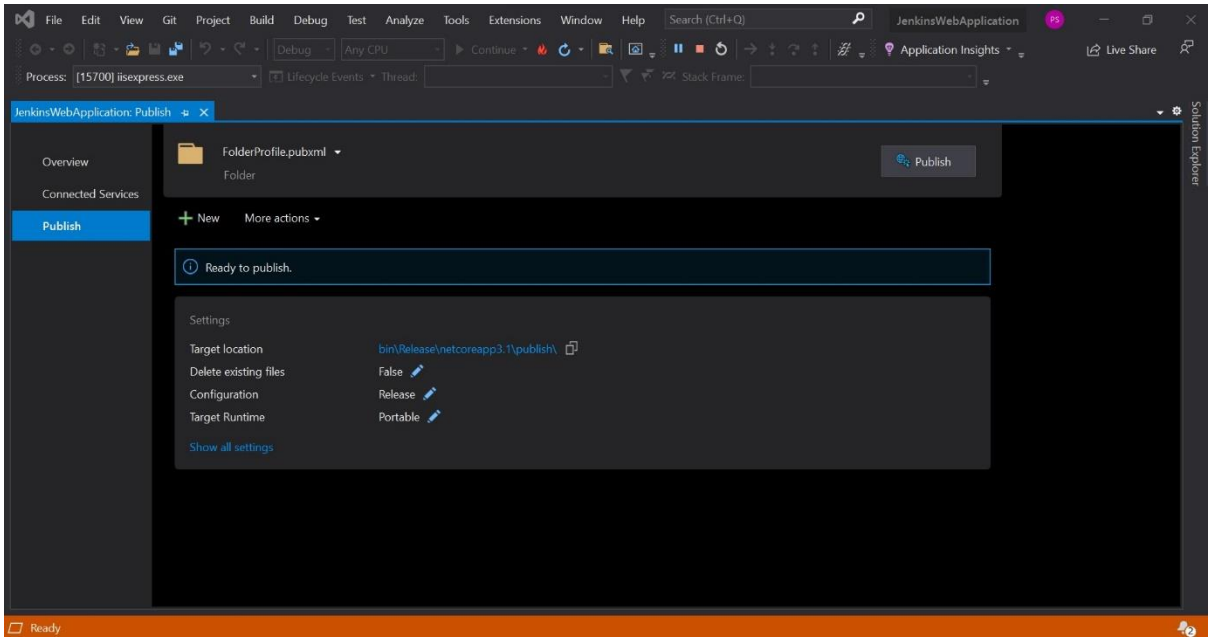




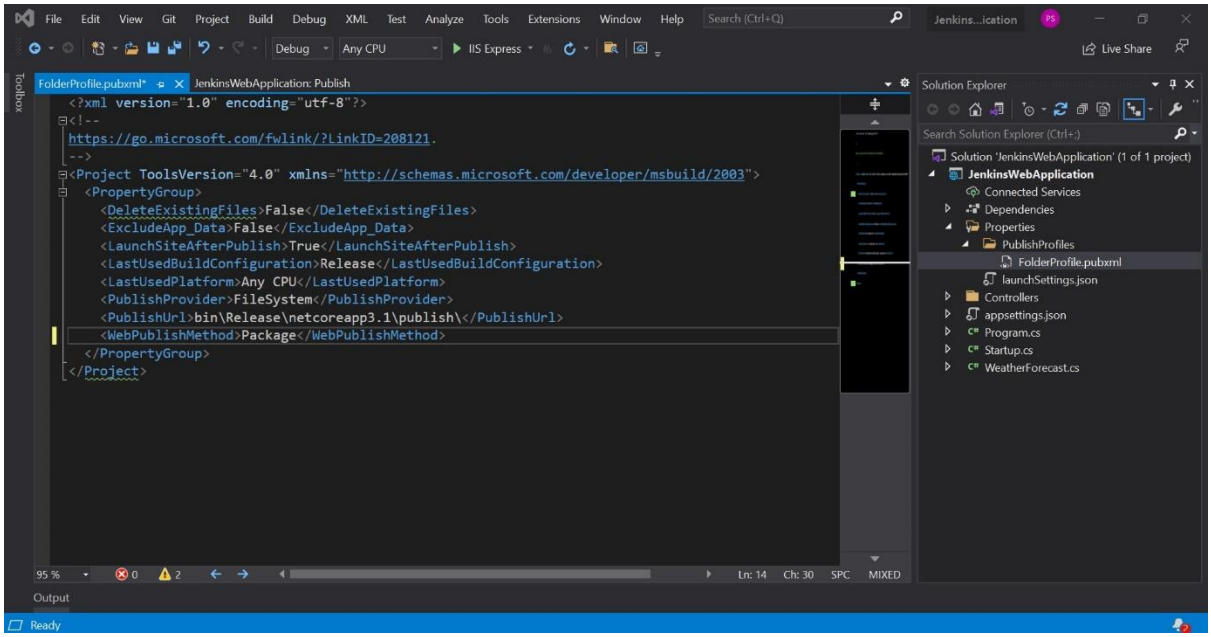
After that Under Solution Explorer right Click on Project Click on publish and in publish select folder then click ok



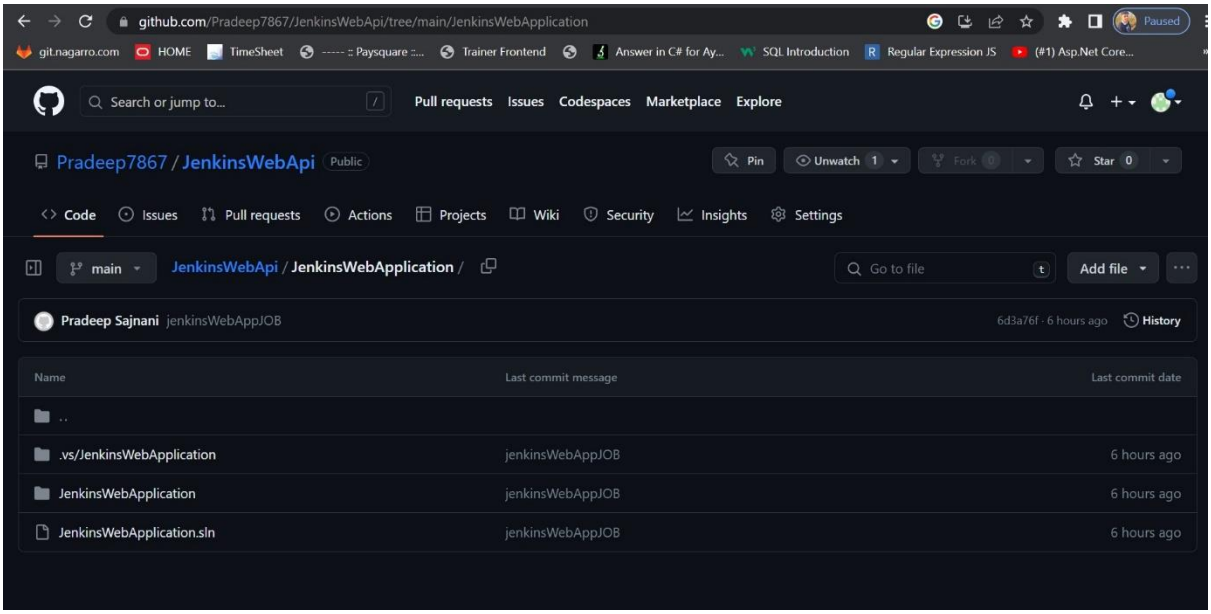




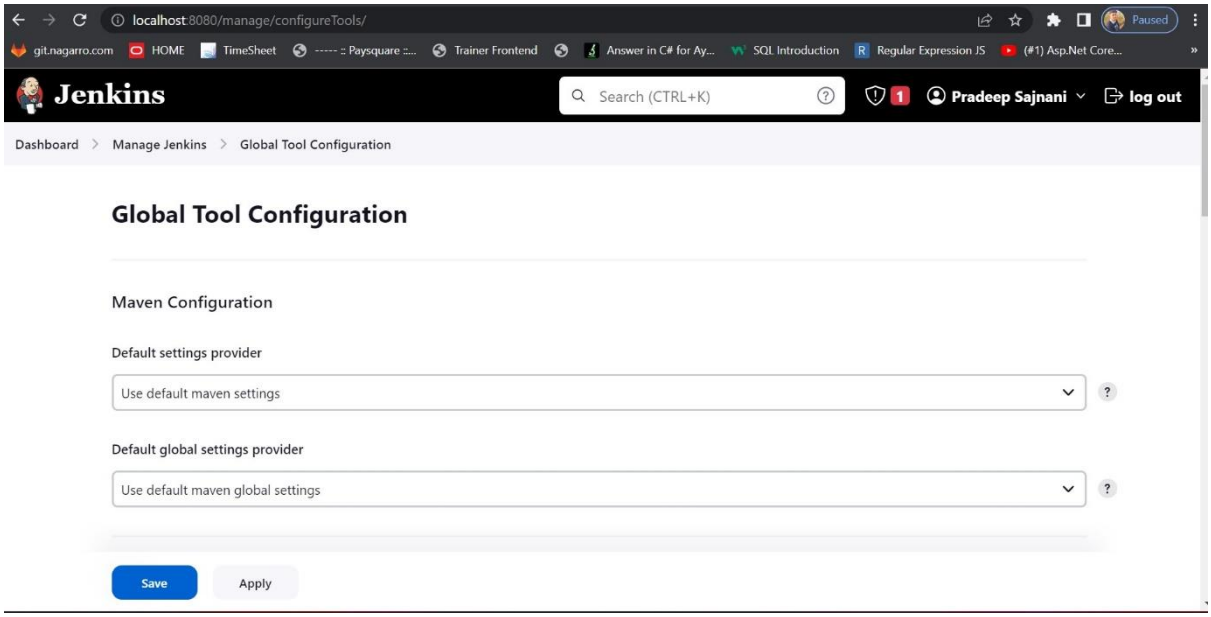
After that you Can Change the webPublicMethod to Package

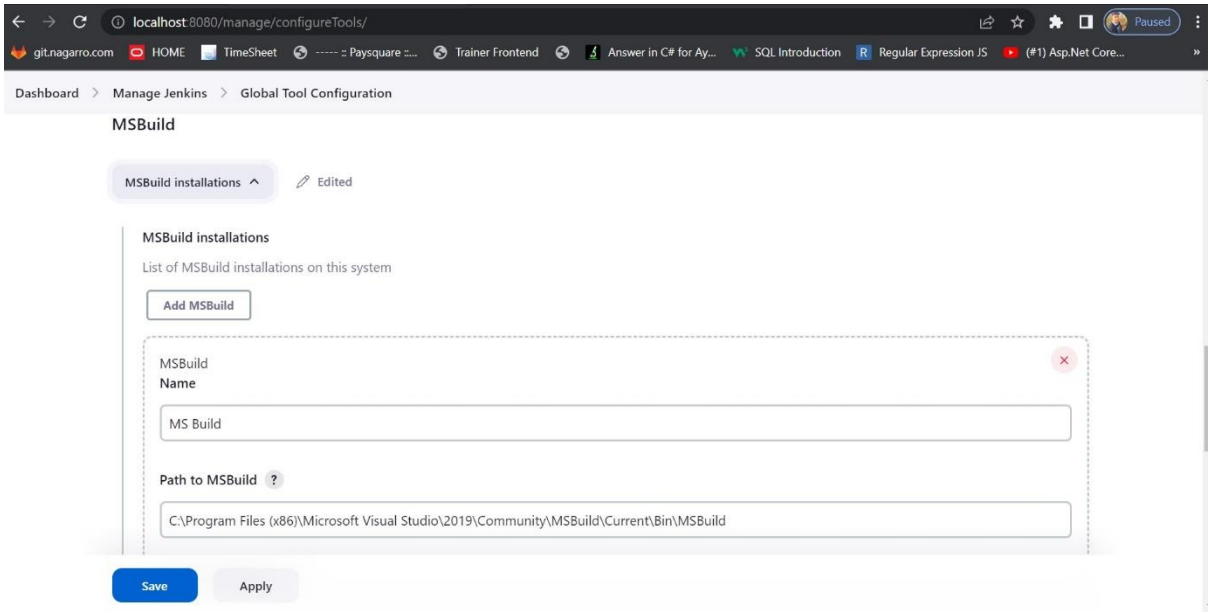


Now You can Commit this Code to Your Repository into your Github repository

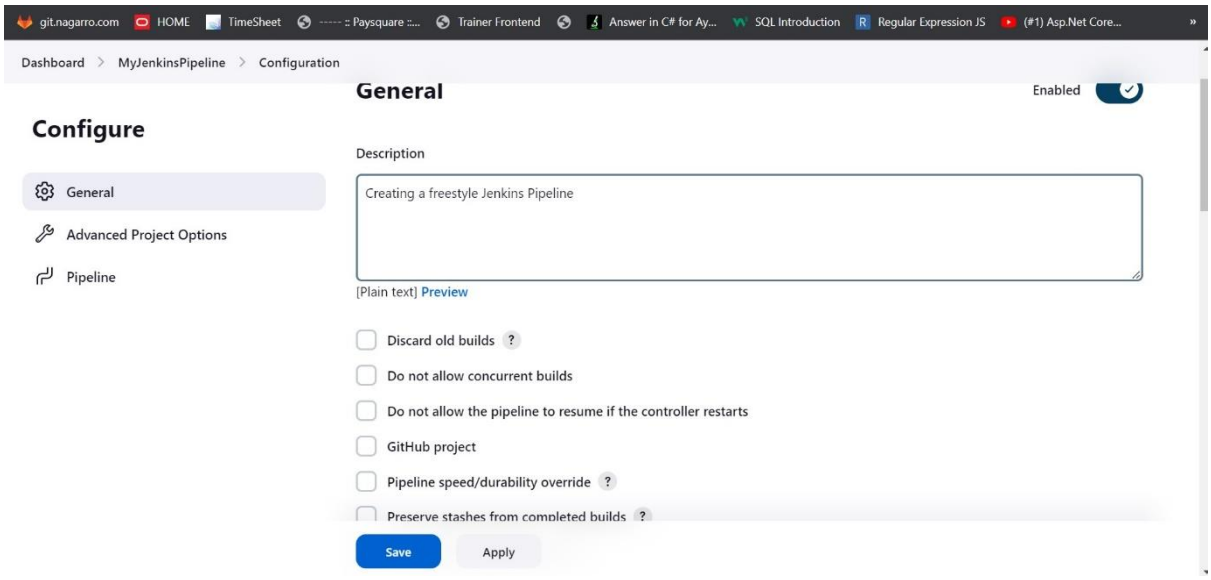


Now Goto Jenkins And in Jenkins go to Global tools Configuration and now in this section add MSBuild Name and Path where MSBuild.exe in install in your System





Now go to previously created Pipeline under Dashboard select your Pipeline and now select **Config now** option to open your Pipeline Configuration

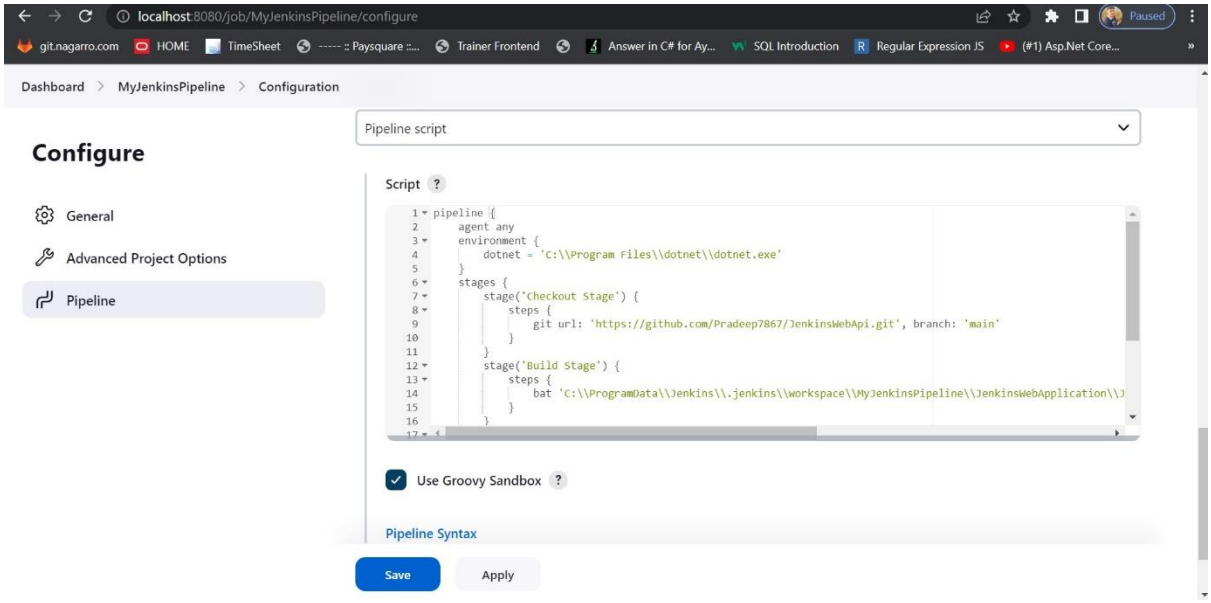


Now Add Script I created this JenkinsFile script

```
pipeline {
    agent any
    environment {
        dotnet = 'C:\\Program Files\\dotnet\\dotnet.exe'
    }
    stages {
        stage('Checkout Stage') {
            steps {
                git url: 'https://github.com/Pradeep7867/JenkinsWebApi.git', branch: 'main'
            }
        }
        stage('Build Stage') {
            steps {
                bat 'C:\\ProgramData\\Jenkins\\.jenkins\\workspace\\MyJenkinsPipeline\\JenkinsWebApplication\\JenkinsWebApplication.sln /p:Configuration
=Release'
            }
        }
        stage('Test Stage') {
            steps {
                bat 'dotnet test C:\\ProgramData\\Jenkins\\
.jenkins\\workspace\\MyJenkinsPipeline\\JenkinsWebApplication\\JenkinsWebApplication\\JenkinsWebApplication.csproj --configuration
Release'
            }
        }


    }
}
```

Now Click on Save & Apply



NOTE:

Checkout Repository:

Use the git command or the Git plugin to clone the repository containing your .NET project code. This step ensures that the latest code is available for building and testing.

Build Project:

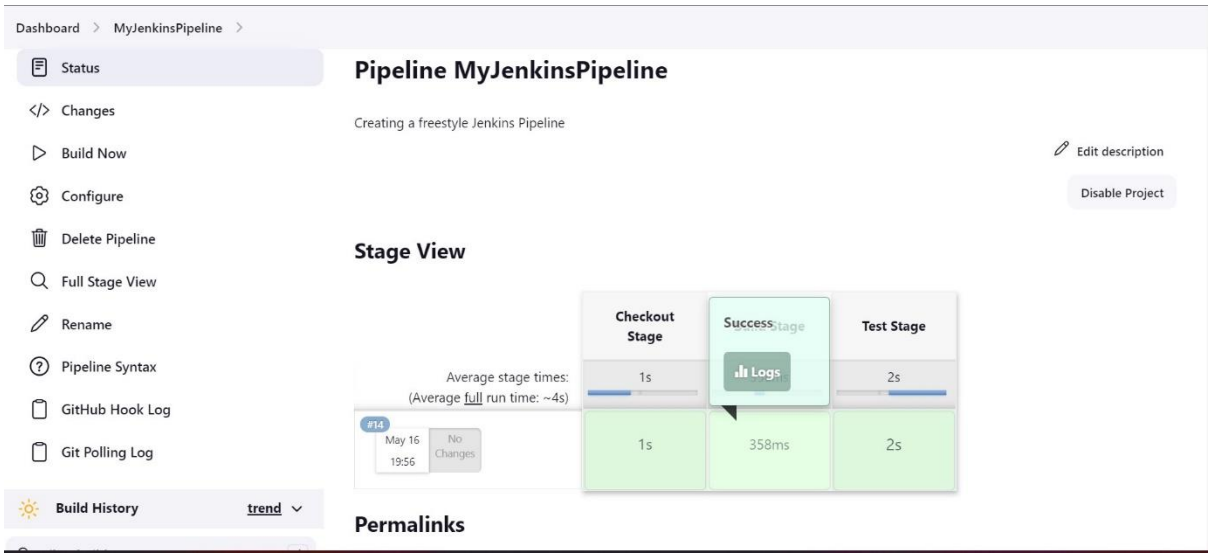
Use the dotnet build command to build the .NET project. This command compiles the code and generates the necessary binaries.

Run Tests:

Use the testing framework of your choice (e.g., NUnit, xUnit) to run the unit tests. You can use the corresponding test runner commands, such as dotnet test with the appropriate options, to execute the tests.

Save and Run Pipeline:

Save the pipeline job configuration and manually trigger the job to verify that it builds and tests the .NET project successfully. You can monitor the console output and test reports in the Jenkins dashboard.



You can Check in Console Ouptut as well as it has Built & test Our .Net Project and also clone the repository in the given Jenkins folder

C:\ProgramData\Jenkins\.jenkins\workspace\MyJenkinsPipeline\JenkinsWebApplication

Dashboard > MyJenkinsPipeline > #14

- Status
- Changes
- Console Output
  - View as plain text
- Edit Build Information
- Delete build '#14'
- Git Build Data
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces

Console Output

```
Started by user Pradeep Sajnani
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\Jenkins\.jenkins\workspace\MyJenkinsPipeline
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Checkout Stage)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
 > git.exe rev-parse --resolve-git-dir C:\ProgramData\Jenkins\.jenkins\workspace\MyJenkinsPipeline\.git #
timeout=10
Fetching changes from the remote Git repository
 > git.exe config remote.origin.url https://github.com/Pradeep7867/JenkinsWebApi.git # timeout=10
Fetching upstream changes from https://github.com/Pradeep7867/JenkinsWebApi.git
 > git.exe --version # timeout=10
 > git --version # 'git version 2.40.1.windows.1'
```

localhost:8080/job/MyJenkinsPipeline/14/console

git.nagarro.com | HOME | TimeSheet | ----- :: Paysquare ::... | Trainer Frontend | Answer in C# for Ay... | SQL Introduction | Regular Expression JS | (#1) Asp.Net Core...

Dashboard > MyJenkinsPipeline > #14

```
C:\ProgramData\Jenkins\.jenkins\workspace\MyJenkinsPipeline>C:\ProgramData\Jenkins\.jenkins\workspace\MyJenkinsPi
peline\JenkinsWebApplication\JenkinsWebApplication.sln /p:Configuration=Release
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test Stage)
[Pipeline] bat

C:\ProgramData\Jenkins\.jenkins\workspace\MyJenkinsPipeline>dotnet test
C:\ProgramData\Jenkins\.jenkins\workspace\MyJenkinsPipeline\JenkinsWebApplication\JenkinsWebApplication\JenkinsWe
bApplication.csproj --configuration Release
  Determining projects to restore...
  Restored
C:\ProgramData\Jenkins\.jenkins\workspace\MyJenkinsPipeline\JenkinsWebApplication\JenkinsWebApplication\JenkinsWe
bApplication.csproj (in 110 ms).
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

This PC > Local Disk (C:) > ProgramData > Jenkins > .jenkins > workspace > MyJenkinsPipeline > JenkinsWebApplication

Name	Date modified	Type	Size
.vs	16-05-2023 18:25	File folder	
JenkinsWebApplication	16-05-2023 18:25	File folder	
JenkinsWebApplication.sln	16-05-2023 18:25	Visual Studio Solution	2 KB

This is a basic outline of the steps involved in creating a Jenkins pipeline for building and testing a .NET project. You can further customize and enhance the pipeline based on your specific requirements, such as deploying the application or integrating additional stages for static code analysis or code coverage.

All the listed questions have been performed by Pradeep Sajnani if need any help pls contact at

pradeepsajnani742@gmail.com

---------- Thank You ----------