**Running ASP Dot net Applications on Jenkins Agents**

**Step 1:** The tools to install in Windows slave machine

**MSbuild tool:** download the Microsoft MSBuild from this link - <https://www.microsoft.com/en-us/download/details.aspx?id=48159>

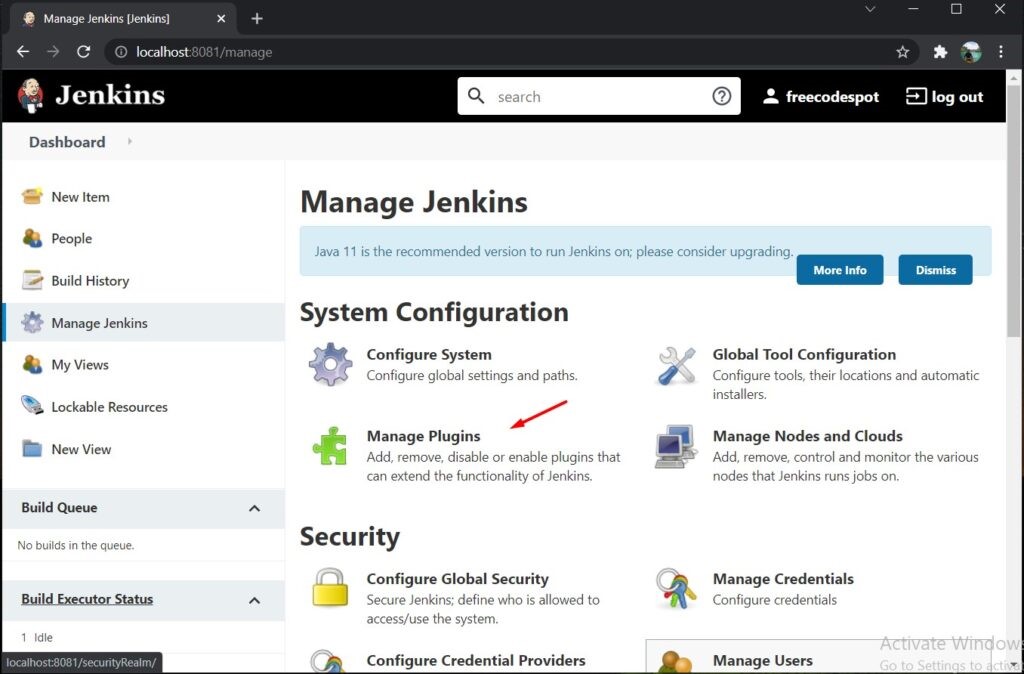
**Nuget packet manager:** Download Nuget Application from this link -  <https://www.nuget.org/downloads>

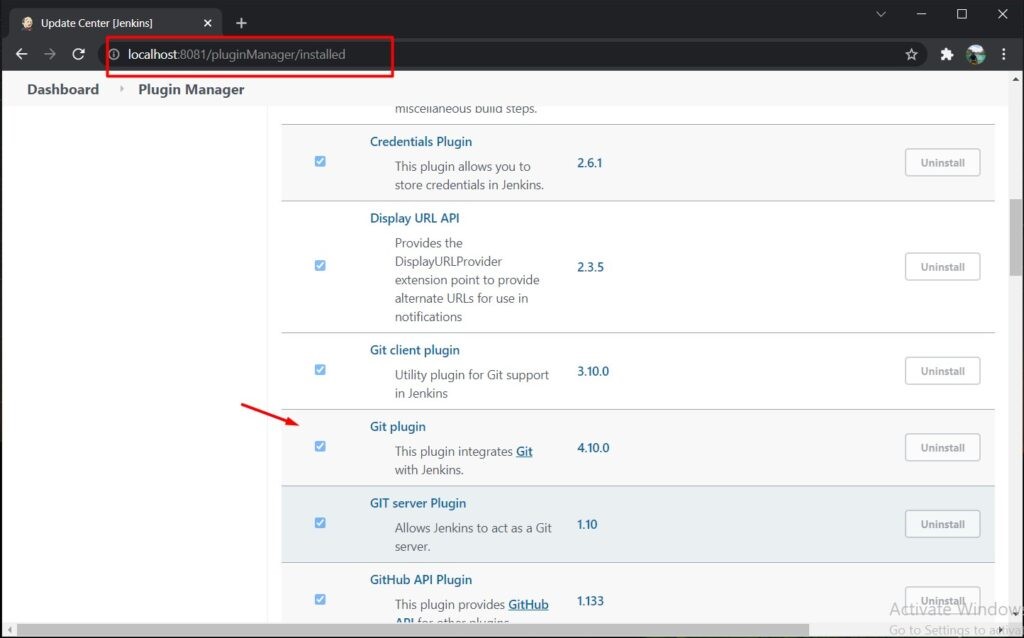
**Step 2:** To run ASP Dot net application on Jenkins agent we need to install the following plugins on Master Jenkins server:

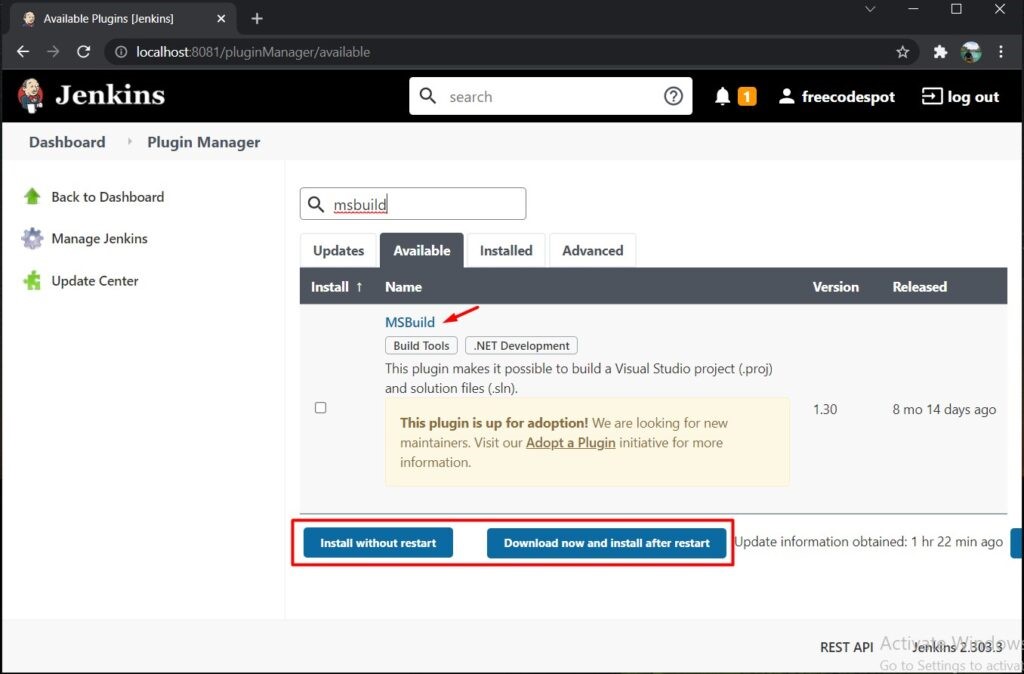
* Git Plugin
* MSBuild Plugin
* MSTest Plugin

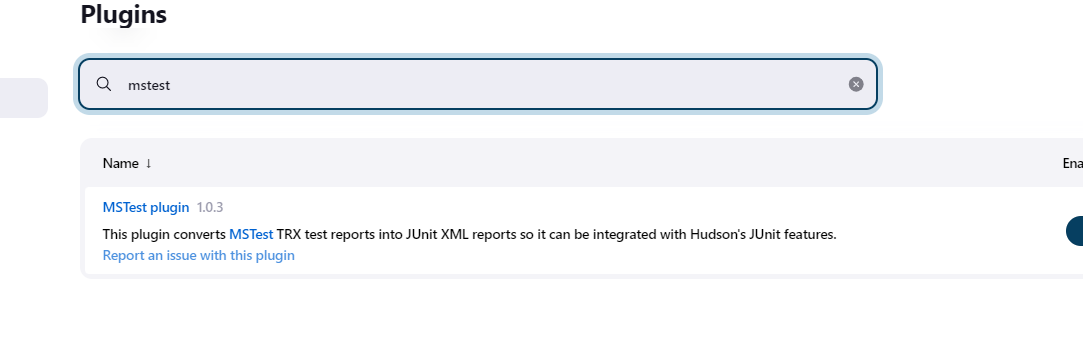
To install these plugins in Jenkins, go to Manage Jenkins ->Manage Plugins->Available Plugins

Here search for Git plugin, MSBuild and MSTest plugins and install them as shown below.





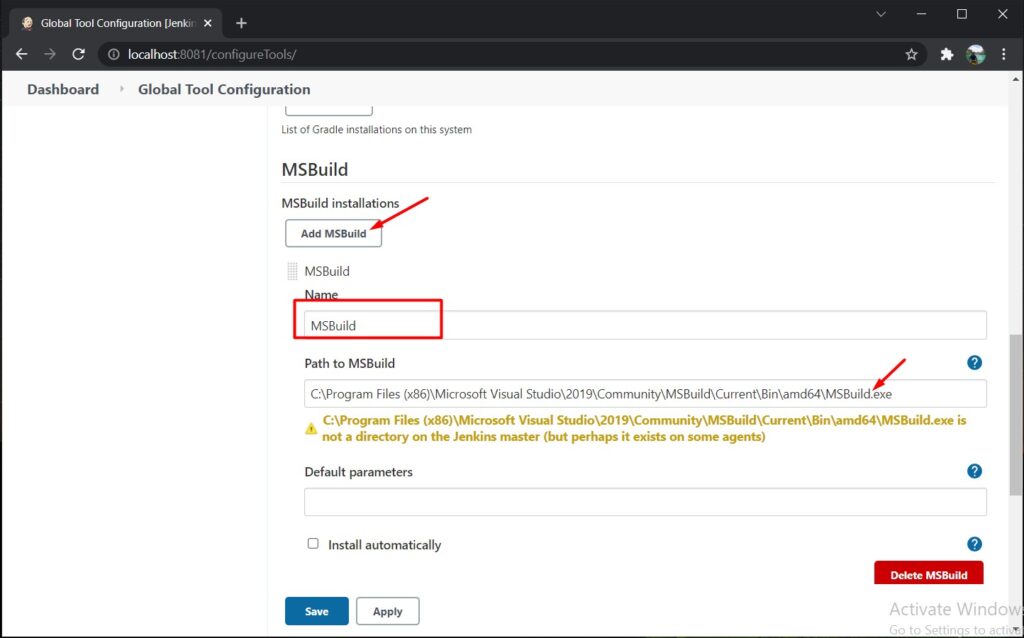


****

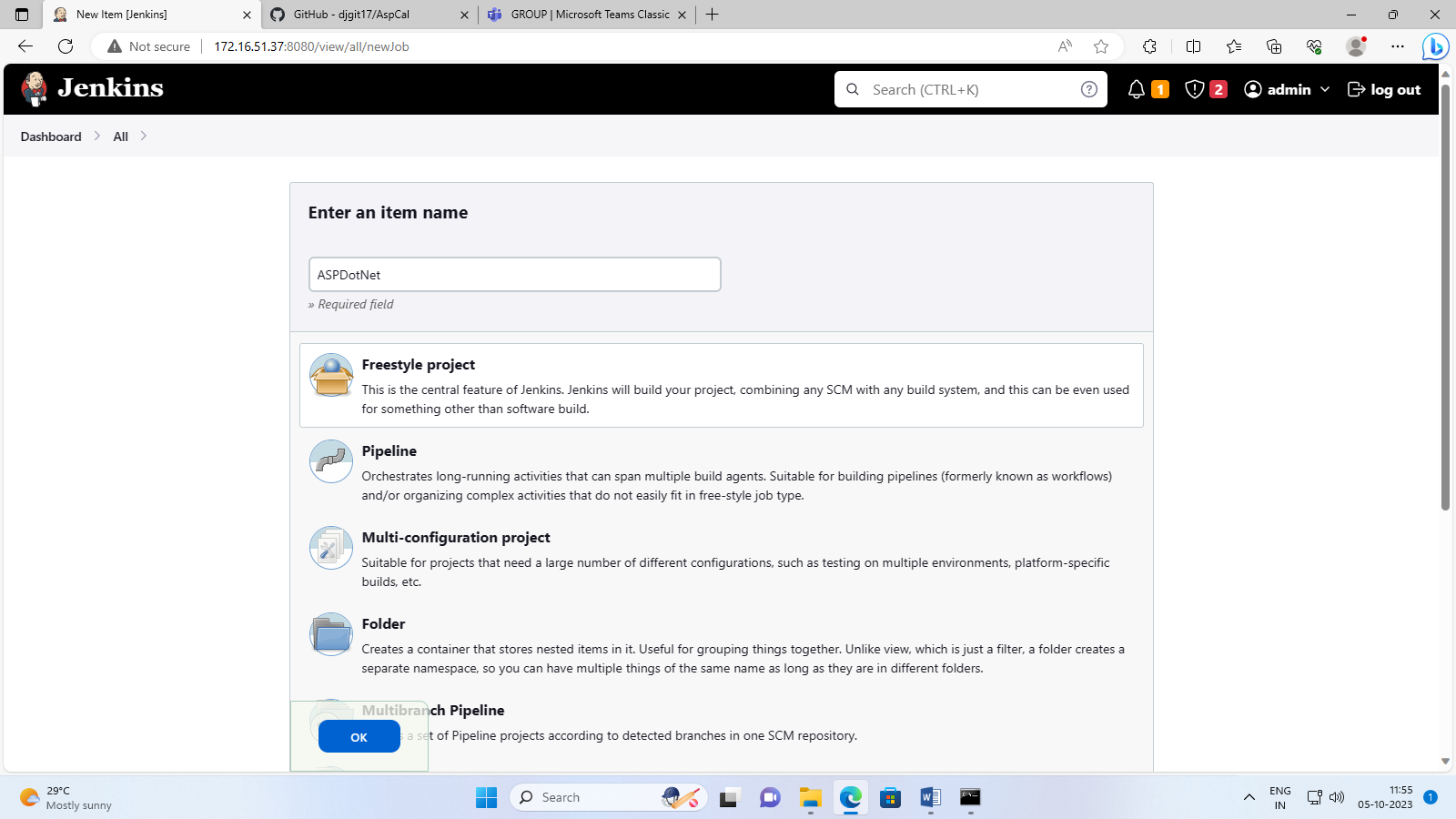
**Step 3:** Locate the MSBuild.exe file in your windows slave machine to add it to global tools in Jenkins

After installing MSBuild plugin added its own configuration options to the Jenkins **Global Tools Configuration**. Navigate to **Manage Jenkins** » **Global Tool Configuration** then scroll down to the **MSBuild**Option.

Now, add **MSBuild**installation, name it MSBuild, give path of MSBuild.exe that you located. See the image below.

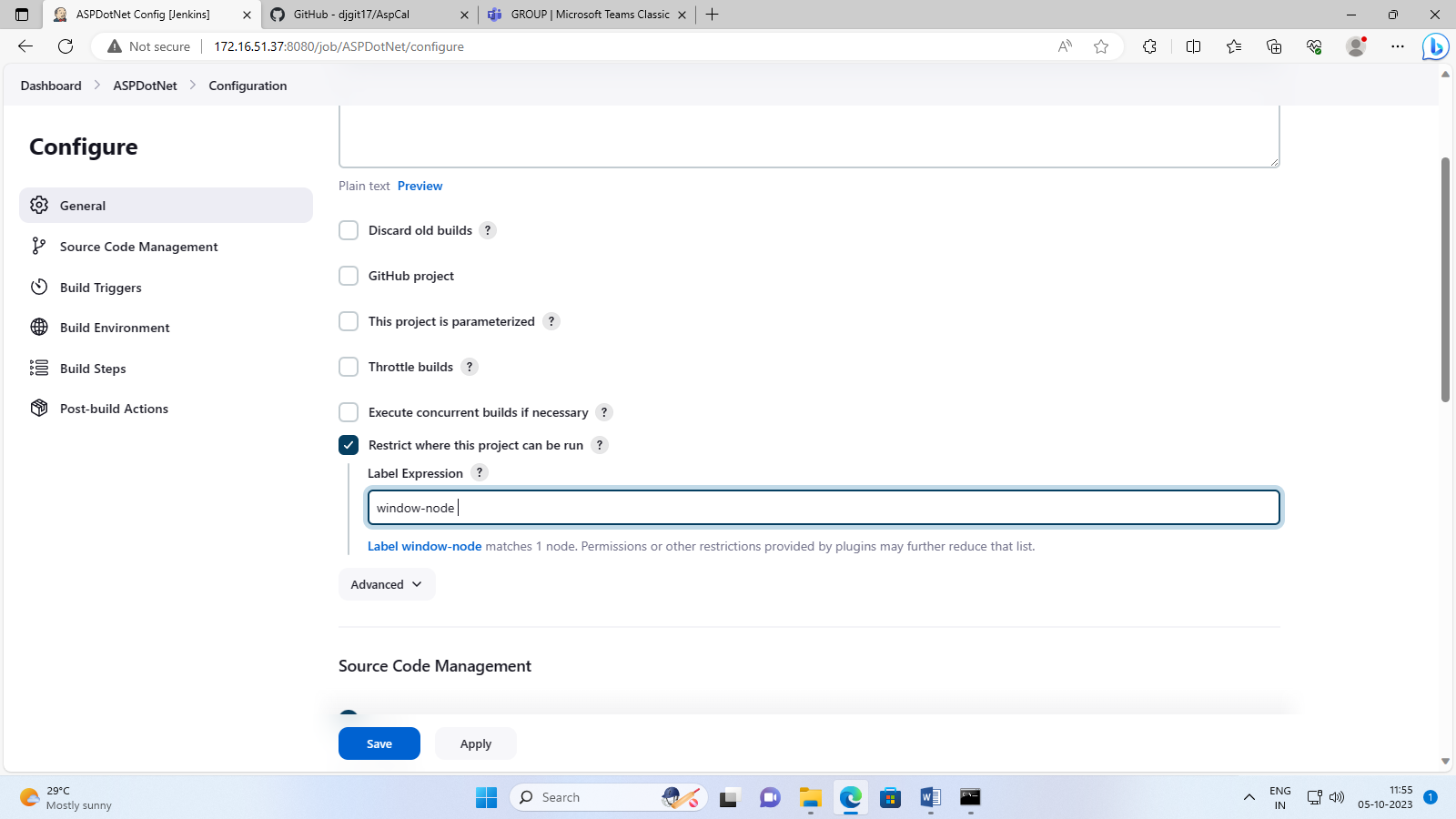


**Step 4:** Now from the Jenkins main Dashboard, select new item to create a new project and create a new freestyle project named **ASPDotnet**.

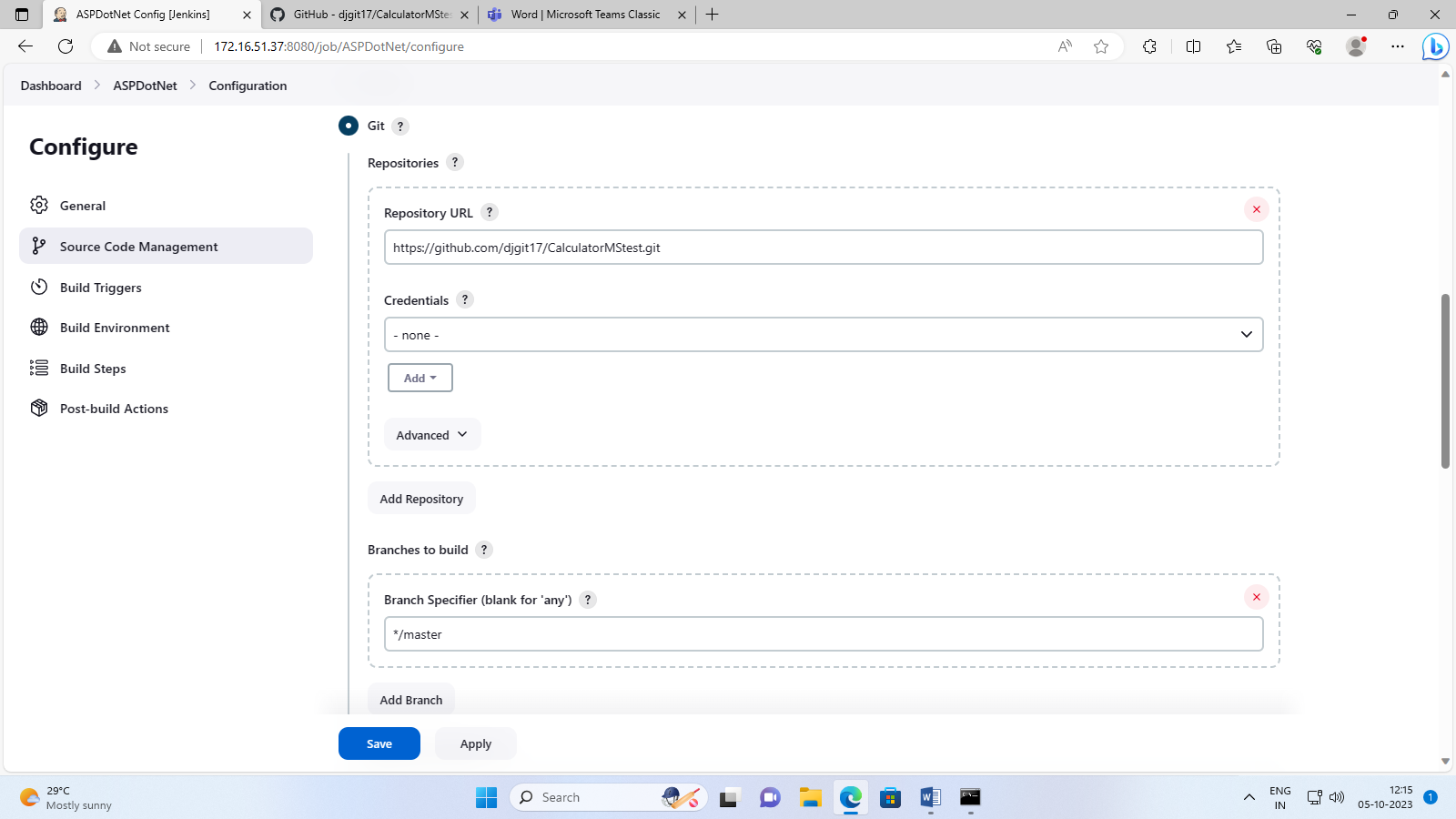


Make sure that Windows slave is connected to Jenkins master.

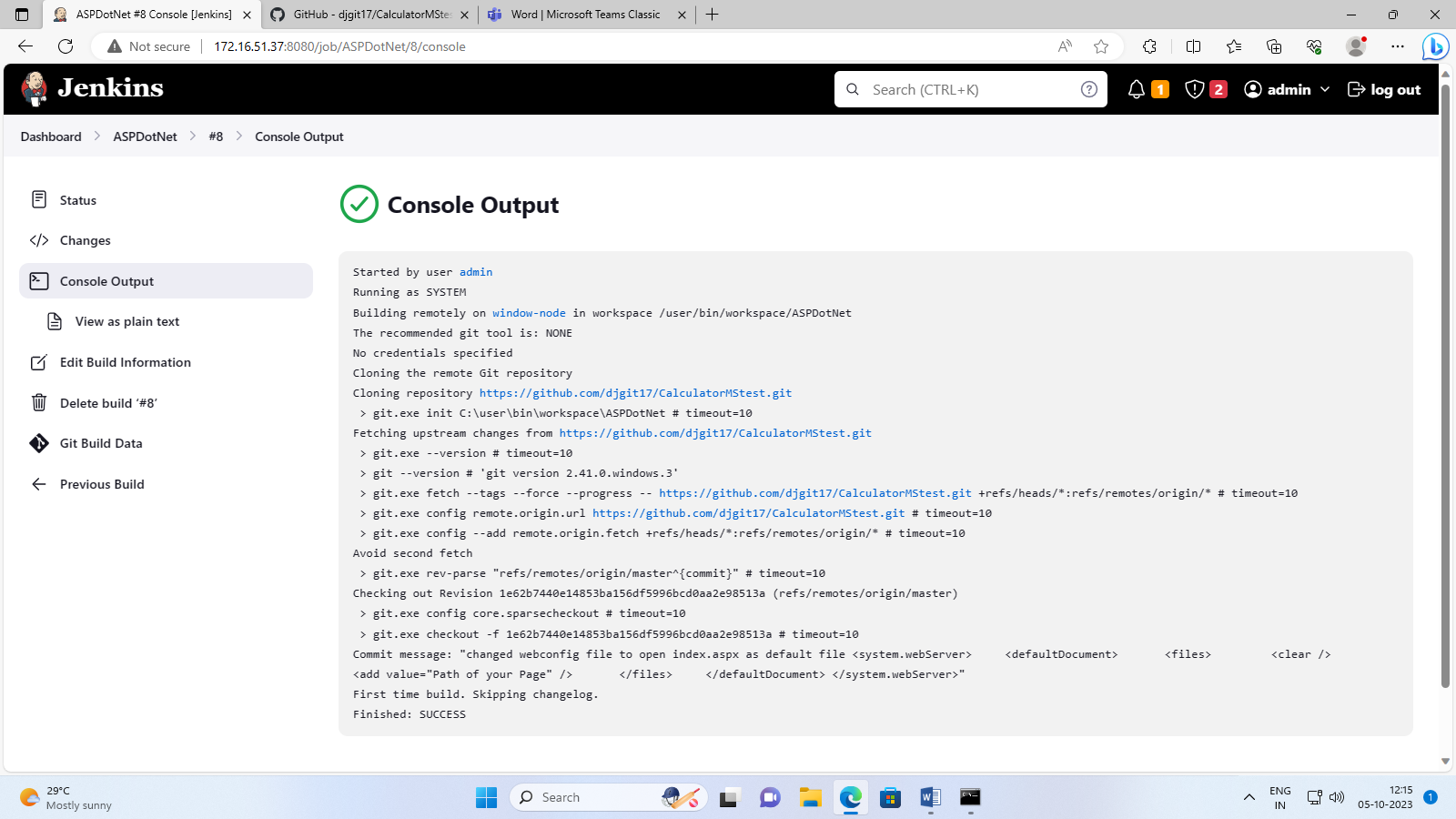
Now configure this project, select **ASPDotnet** project go to configure, in that check **restrict where project can be run**, here give the name you have given to your windows slave.



Next in **Source Code Management** select **Git** and input the repository URL you want to deploy i.e <https://github.com/djgit17/CalculatorMStest.git> Specify the branch as present in the repository,Here it is master, then click the **Save** button.



From your project dashboard, click the **Build Now**tab this will execute a clone command to clone the repository we added a while ago. Just wait for the process to finish, until you see a success message.



You can find the output location of the cloned repository from the **Console Output** (building remotely in workspace) as shown above.

**Step 4:** Next locate where your nuget.exe file is. Then add the NuGet Restore command to our Build configuration. We restore the solution file (.sln file) that is present in workspace where we cloned the repository. To do that, follow the step below.

Re-open project **ASPDotnet** and select **Configure** scroll toBuild then **Add Build Step.**Add **Execute Windows batch command.** And write command: the path where nuget.exe file is present, with path of solution file, here it is **C:\Users\MSIS\Documents\nuget.exe restore AspCalMStest.sln**

Add one more build step.Build a Visual Studio project or a solution using MSBuild.

Select the MSBuild configuration that we created from the previous steps. Here we named it **MSBuild.** and MSBuild Build file is .csproj file that is present in workspace where we cloned the repository. Here it is **C:\Users\MSIS\slave\workspace\ASPDotnet\ASPCalMSTest\ ASPCalMSTest.csproj**

On the **command-line argument** add the command below:

/p:DeployOnBuild=true

/p:DeployDefaultTarget=WebPublish

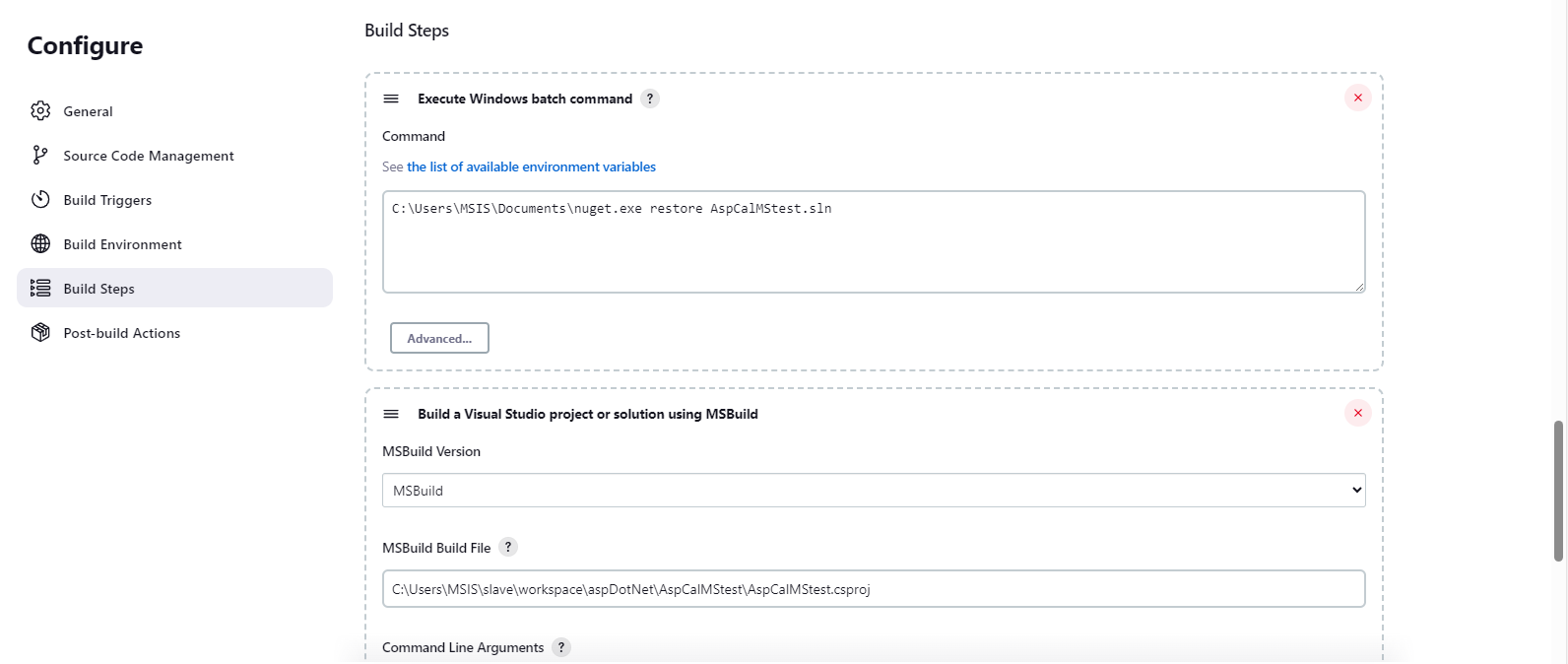
/p:WebPublishMethod=FileSystem

/p:SkipInvalidConfigurations=true

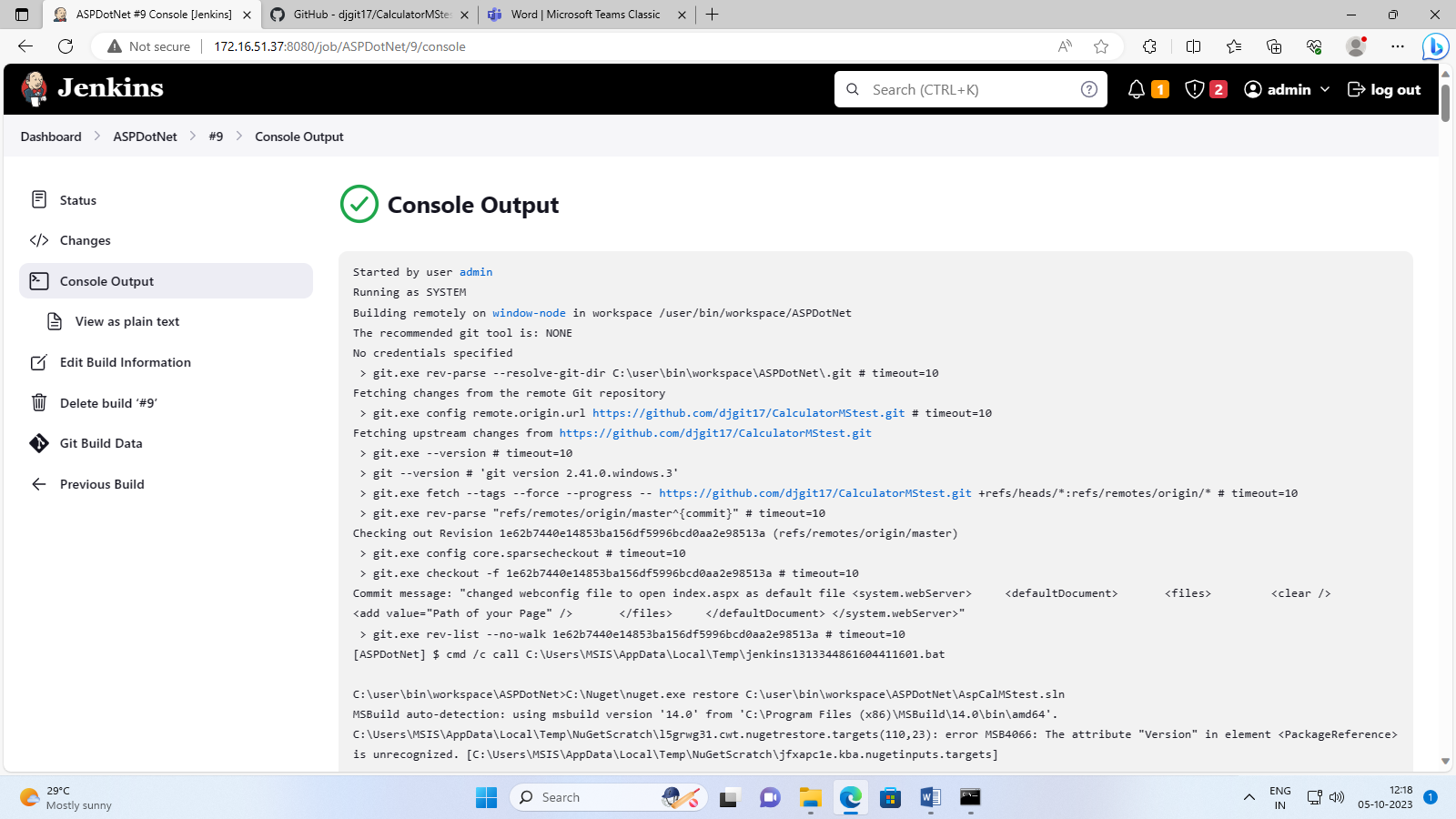
/p:Configuration=Release

/p:DeleteExistingFiles=True

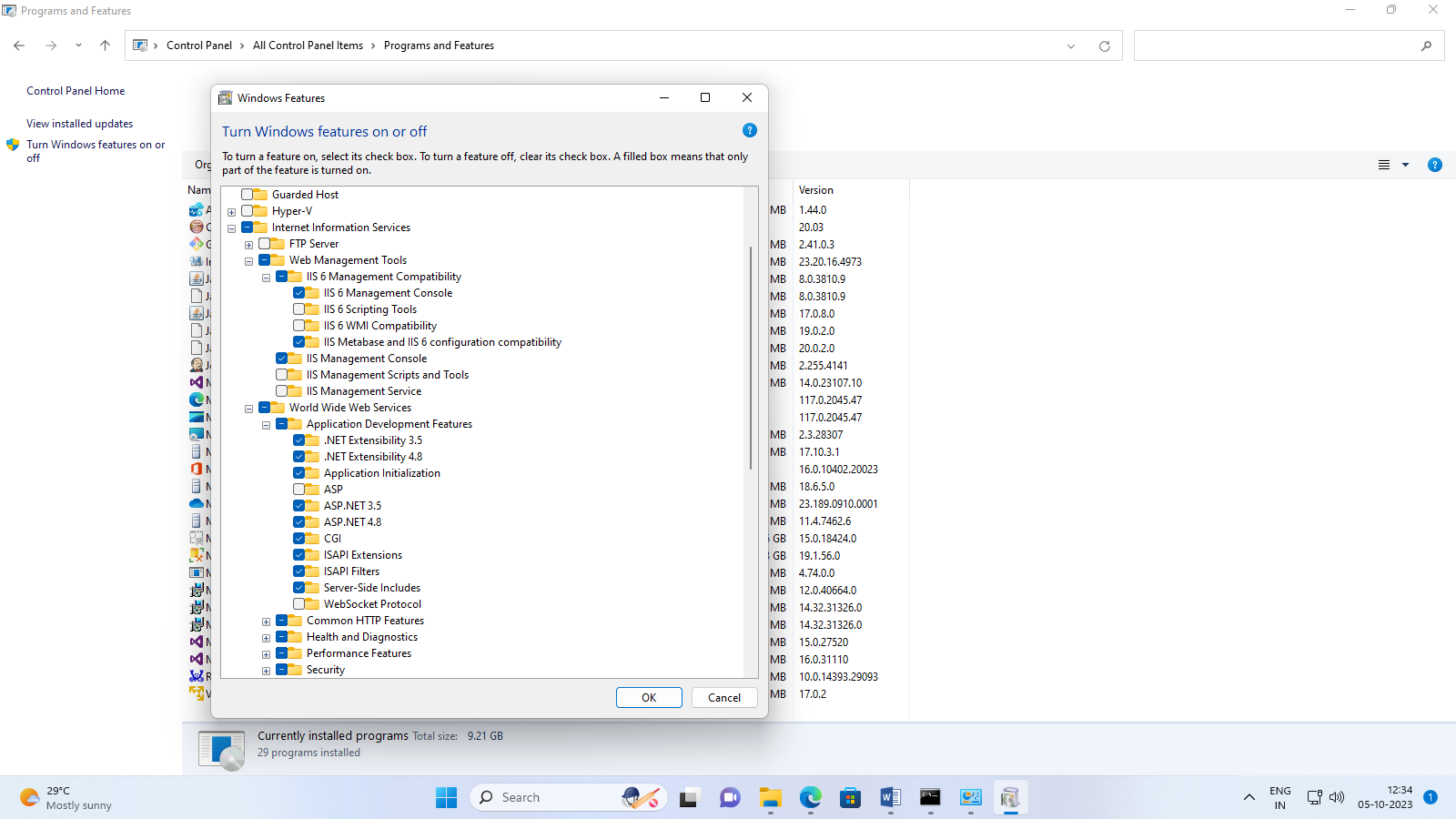
/p:publishUrl=c:\inetpub\wwwroot



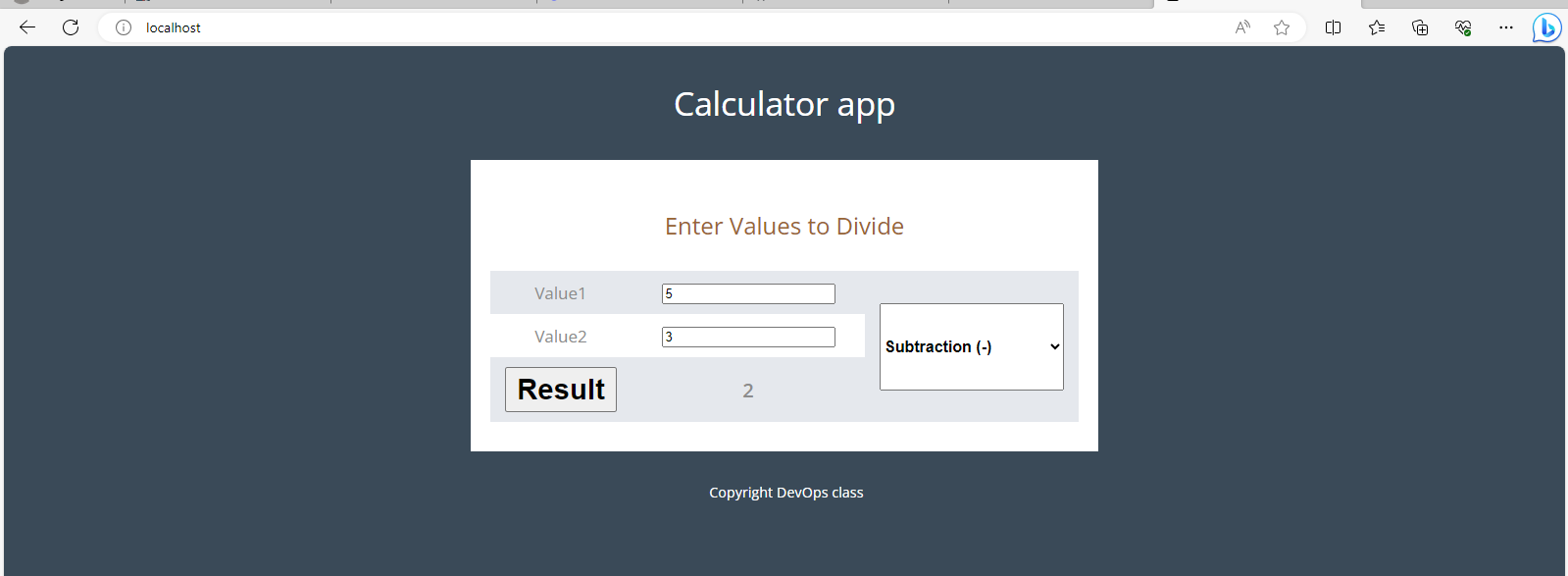
Click on save and then build the project to get success.



To view this application deployed we have to enable IIS server in windows slave machine. To enable the server go Control panel->Programs->Turn windows features on or off and all check all the services as shown below and apply.

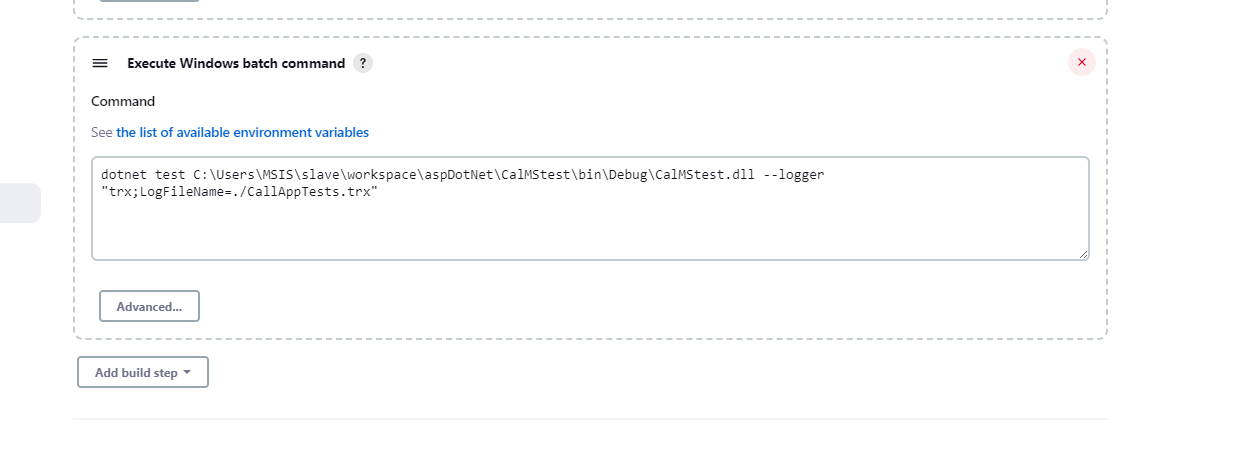


Then in your windows slave open browser and browse localhost to get deployed application as shown below.

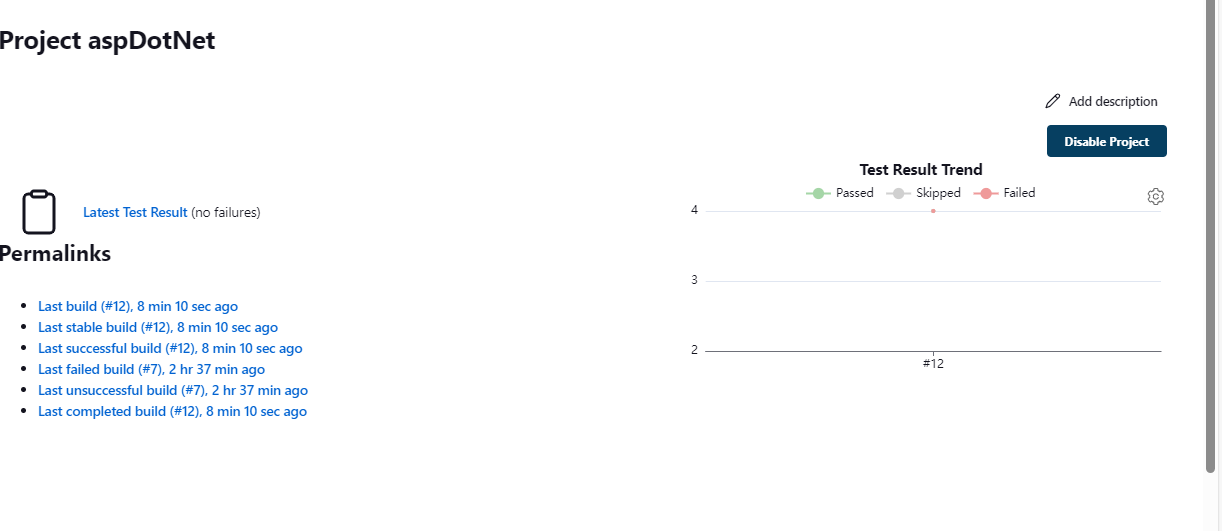
****

**Step 5:** To check test cases,

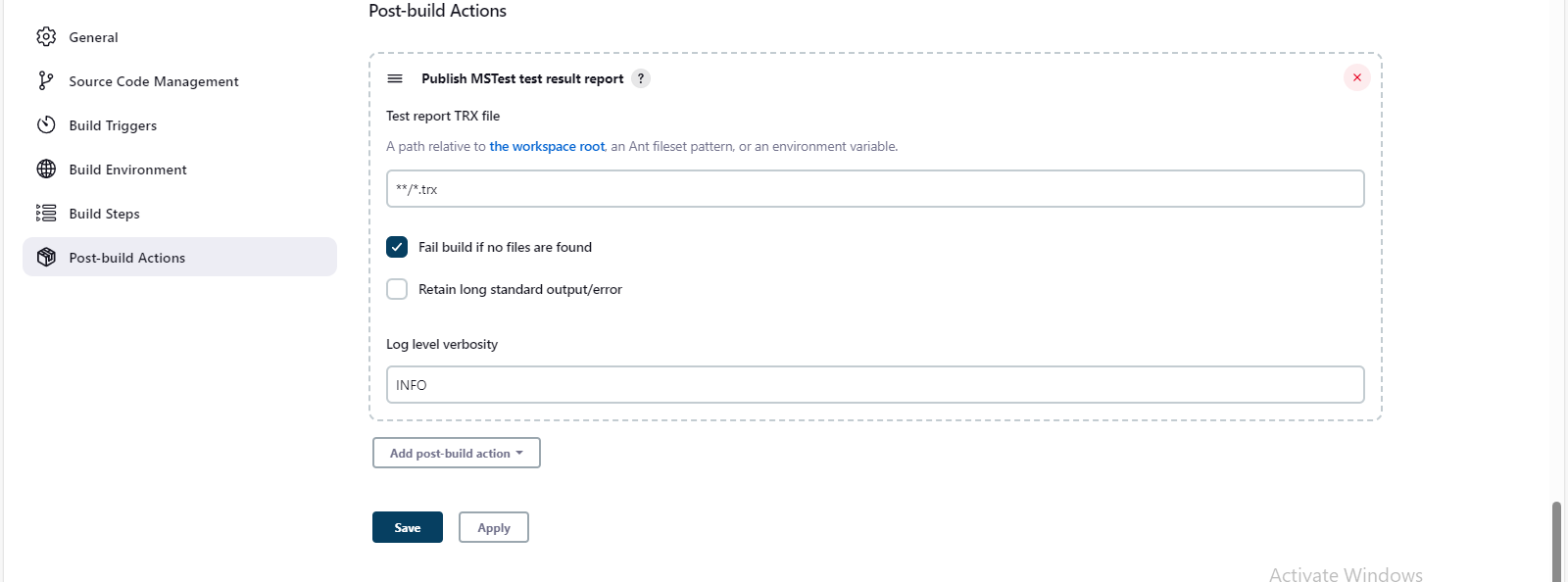
Re-open project **ASPDotnet** and select **Configure** scroll toBuild then **Add Build Step.**Add **Execute Windows batch command** and the command as shown in image. Then build project.

****

**Step 6:** To view the test result trend as shown below, add a post build action.

****

i.e., Re-open project **ASPDotnet** and select **Configure** scroll toPost-build Actions then **add a post-build action.**Add **Publish MSTest test result report,** give Test report TRX file as **\*\*.\*trx**, then build project. You will get the graphical view as shown above.

****