3.4 Triggering the Build



This section will guide you to:

* Trigger a build in Jenkins

This guide has four subsections, namely:

3.4.1 Logging into Jenkins dashboard

3.4.2 Setting up a .NET Build Project in Jenkins for continuous integration

3.4.3 Triggering the build in Jenkins

3.4.4 Pushing the code to your GitHub repositories

* Jenkins is installed in your practice lab. Refer to **DotNet Lab guide: Phase 4** for more information.

**Step 3.4.1:** Logging into Jenkins dashboard

* Jenkins is already installed in your practice lab.
* Open <http://localhost:8080> in the default browser.
* Provide username and password and click on **Login.**

**Step 3.4.2:** Setting up a .NET Build Project in Jenkins for continuous integration

* Open Jenkins dashboard page on <http://localhost:8080>.
* Login as admin user.
* From the left bar, click **Manage Jenkins.**
* In the next page click **Manage Plugins.**
* Click on the **Available** tab.
* Check **Team Foundation Server,** **VS Team Services Continuous Deployment** and **MS Build.**
* At the bottom of the page click **Install without Restart.**
* Go back to the Dashboard page.
* In the Dashboard page, click on **New Item** from the left bar.
* Enter name as NET\_BUILD, select **FreeStyle Project** andlick **Ok.**
* This will open the Configuration screen.
* In the **General** tab enter a description of your choice and check **Discard Old builds.**
* In **Log Rotation,** set **Days to Keep Builds** to 3.
* Click **Source Code Management** tab and check **Team Foundation Version Control.**
* Enter the TFVC **Collection URL** eg. <https://team.myprojects.com/>. (You can see this URL in the TFS Admin console. The URL should comprise of the details mentioned. - <http://servername:8080/tfs/thiscollectionname> )
* Enter the **Project Path.** Eg. $/MyTeam.
* In the **Credentials** dropdown select **Manual** andenter your VSTS username and password(This can be set to **Automatic** also).
* Click the **Build Environment** tab.
* Check **Delete workspace before build starts**
* Under **Post Build Actions** click **Add Post Build Action.**
* Choose **Email Notification** and enter the recipient email ids in the form.
* Click **Save.**
* From the left bar, click **Build Now** to build the project. This will generate a build in the Jenkins workspace. Once the **Build Number** is displayed click on it to see the workspace. This will show all the files that have been generated by MSBuild.

**Step 3.4.3:** Triggering the Build in Jenkins

* Open Visual Studio and load the project that is targeted in Jenkins.
* Add a git comment and do a git push.
* This will trigger Jenkins to generate a new build. You have to make sure that the project is unloaded from Visual Studio as otherwise the files will be unavailable for Jenkins to access them.

**Step 3.4.4:** Pushing the code to your GitHub repositories

Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add .

Commit the changes using the following command:

git commit -m “Changes have been committed.”

Push the files to the folder you created initially using the following command:

git push -u origin master