**Continuous Integration**

Continuous Integration is a development practice that calls upon development teams to ensure that a build and subsequent testing is conducted for every code change made to a software program. This concept was meant to remove the problem of finding the late occurrences of issues in the build lifecycle. Instead of the developers working in isolation and not integrating enough, continuous integration was introduced to ensure that the code changes and builds were never done in isolation.

Continuous Integration after a code commit, the software is built and tested immediately. In a large project with many developers, commits are made many times during a day. With each commit code is built and tested. If the test is passed, build is tested for deployment. If deployment is a success, the code is pushed to production. This commit, build, test, and deploy is a continuous process and hence the name continuous integration/deployment.

A Continuous Integration Pipeline is a powerful instrument that consists of a set of tools designed to host, monitor, compile and test code, or code changes, like:

* Continuous Integration Server (Jenkins, Bamboo, CruiseControl, TeamCity, and others)
* Source Control Tool (e.g., CVS, SVN, GIT, Mercurial, Perforce, ClearCase and others)
* Build tool (Make, ANT, Maven, Ivy, Gradle, and others)
* Automation testing framework (Selenium, Appium, TestComplete, UFT, and others)
* **Why Continuous Integration?**

Continuous integration has become a very integral part of any software development process. The continuous Integration process helps to answer the following questions for the software development team.

* Do all the software components work together as they should? – Sometimes systems can become so complex that there are multiple interfaces for each component. In such cases, it’s always critical to ensure that all the software components work seamlessly with each other.
* Is the code too complex for integration purposes? – If the continuous integration process keeps on failing, there could be a possibility that the code is just too complex. And this could be a signal to apply proper design patterns to make the code lesser complex and more maintainable.
* Does the code adhere to the established coding standards? – Most of the test cases will always check that the code is adhering to the proper coding standards. By doing an automated test after the automated build, this is a good point to check if the code meets all the desired coding standards.
* How much code is covered by automated tests? – There is no point in testing code if the test cases don’t cover the required functionality of the code. So it’s always a good practice to ensure that the test cases written should cover all the key scenarios of the application.
* Were all the tests successful after the latest change? – If a test fails, then there is no point in proceeding with the deployment of the code, so this is a good point to check if the code is ready to move to the deployment stage or not.

### 1) [Buddy](https://bit.ly/2NJz4mx)

[](https://bit.ly/2NJz4mx)

[Buddy](https://bit.ly/2NJz4mx) is a smart CI/CD tool for web developers designed to lower the entry threshold to DevOps. It uses delivery pipelines to build, test and deploy software. The pipelines are created with over 100 ready-to-use actions that can be arranged in any way – just like you build a house of bricks.

* 15-minute configuration in clear & telling UI/UX
* Lightning-fast deployments based on changesets
* Builds are run in isolated containers with cached dependencies
* Supports all popular languages, frameworks & task managers
* Dedicated roster of Docker/Kubernetes actions
* Integrates with AWS, Google, DigitalOcean, Azure, Shopify, WordPress & more
* Supports parallelism & YAML configuration

### 2) Travis CI:

[](https://github.com/travis-ci/travis-ci)

[Travis](https://github.com/travis-ci/travis-ci) is a popular CI Tool that is free for open source projects. As it is hosted, it does not have to depend on any platform. This CI tool provides supports for many build configuration and languages like Node, PHP, Python, Java, Perl, etc.

**Features:**

* Travis uses the virtual machines to build application
* Notifications via Slack, HipChat, Emails and more
* Allows running parallel tests
* Linux and Mac, and iOS supported
* Easy Setup, no installation required.
* Powerful API and command line tool

**Download Link:** <https://github.com/travis-ci/travis-ci>

### 3) Final builder:

[](https://www.finalbuilder.com/)

[FinalBuilder](https://www.finalbuilder.com/) is Vsoft's build tool. With FinalBuilder there is no need to edit XML, or write scripts. You can define and debug build scripts when it schedules them with windows scheduler, or integrate with Jenkins, Continua CI, etc.

**Features:**

* It presents build process in a logically structured, graphical interface
* It includes try and catch actions for localized error handling
* It provides tight integration with the Windows scheduling service, which allows builds to be scheduled
* FinalBuilder supports more than a dozen version control systems
* It provides support for scripting
* The output from all actions in the build process is directed to the build log.

**Download Link:** <https://www.finalbuilder.com/downloads/finalbuilder>

### 4) CruiseControl:

[https://www.guru99.com/images/2-2017/072817_0526_Top20Contin18.png](http://cruisecontrol.sourceforge.net/)

[CruiseControl](http://cruisecontrol.sourceforge.net/) is both CI tool and an extensible framework. It is used for building a custom continuous build process. It has many plugins for a variety of source controls, build technologies which include email and instant messaging.

**Features:**

* Integration with a many different Source Control systems like vss, csv, svn, git, hg, perforce, clearcase, filesystem, etc.
* It allows building multiple projects on single server
* Integration with other external tools like NAnt, NDepend, NUnit, MSBuild, MBUnit and Visual Studio
* Provide support for Remote Management

**Download link:** <http://cruisecontrol.sourceforge.net/download.html>

### 5) Integrity:

[](http://integrity.github.io/)

[Integrity](http://integrity.github.io/) is a continuous integration server which works only with GitHub. In this CI tool whenever users commit the codes, it builds and runs the code. It also generates the reports and provides notifications to the user.

**Features:**

* This CI tool currently only works with git, but it can easily mirror with other SCM
* This CI tool supports numbers of notification mechanisms like AMQP, Email, HTTP, Amazon SES, Flowdock, Shell, and TCP.
* HTTP Notifier feature sends an HTTP POST request to the specific URL

**Download link:** <http://integrity.github.io/>

### 6) GoCD:

[](https://www.gocd.org/)

[GoCD](https://www.gocd.org/) is an Open source Continuous Integration server. It is used to model and visualize complex workflows with ease. This CI tool allows continuous delivery and provides an intuitive interface for building CD pipelines.

**Features:**

* Supports parallel and sequential execution. Dependencies can be easily configured.
* Deploy any version, anytime
* Visualize end to end workflow in realtime with Value Stream Map.
* Deploy to production securely.
* Handle user authentication and authorization
* Keep orderly configuration
* Tons of plugins to enhance functionality.
* Active community for help and support.

**Download link:** <https://www.gocd.org/download/>

### 7) Urbancode:

[](https://developer.ibm.com/urbancode/)

[IBM UrbanCode](https://developer.ibm.com/urbancode/) Deploy is a CI application. It combines robust visibility, traceability, and auditing feature into a single package.

**Features:**

* Increase frequency of software delivery by automated, repeatable deployment processes
* Reduce deployment failure
* Streamline the deployment of multi-channel apps to all environments whether on-premises or in the cloud
* Enterprise level security and scalability
* Hybrid cloud environment modeling
* Drag-and-drop automation

**Download link:** <https://www.ibm.com/ms-en/marketplace/application-release-automation>

### 8) Autorabit:

[](http://www.autorabit.com/)

[AutoRABIT](http://www.autorabit.com/) is an end-to-end Continuous Delivery Suite to speed up the development process. It streamlines the complete release process. It helps the organization of any size to implement Continuous Integration.

**Features:**

* The tool is specially designed to deploy on Salesforce Platform
* Lean and faster deployments based on changes supporting all the 120+ supported metadata types.
* Fetch changes from Version Control System and deploy them into Sandbox automatically
* Auto-commit changes into Version Control System directly from Sandbox

**Download link:** <http://www.autorabit.com/tag/autorabit-download/>

### 9) CircleCI:

[](https://circleci.com/)

[Circle CI](https://circleci.com/) is a flexible CI tool that runs in any environment like cross-platform mobile app, Python API server or Docker cluster. This tool reduces bugs and improves the quality of the application.

**Features:**

* Allows to select Build Environment
* Supports many languages like Linux, including C++, Javascript, NET, PHP, Python, and Ruby
* Support for Docker lets you configure customized environment
* Automatically cancel any queued or running builds when a newer build is triggered
* It split and balance tests across multiple containers to reduce overall build time
* Forbid non-admins from modifying critical project settings
* Improve Android and iOS store rating by shipping bug-free apps.
* Optimal Caching and Parallelism for fast performance.
* Integration with VCS tools

**Download link:** <https://circleci.com/>

### 10) Buildkite:

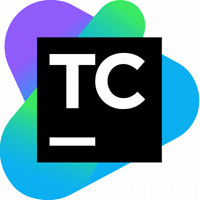
[](https://buildkite.com/)

The [buildkite](https://buildkite.com/) agent is a reliable and cross-platform build runner. This CI tool makes it easy to run automated builds on your infrastructure. It is mainly used for running build jobs, reporting back the status code and output log of the job.

**Features:**

* This CI tool runs on a wide variety of OS and architectures
* It can run code from any version control system
* Allows to run as many build agents as you want on any machine
* It can integrate with the tools like Slack, HipChat, Flowdock, Campfire and more
* Buildkite never sees source code or secret keys
* It offers stable infrastructure **Download link:** <https://buildkite.com/>

### 11) TeamCity

[](https://www.jetbrains.com/teamcity/)

[TeamCity](https://www.jetbrains.com/teamcity/) is a Continuous Integration server which supports many powerful features.

**Features:**

* Extensibility and Customization
* Provides better code quality for any project
* It maintains CI server healthy and stable even when no builds are running
* Configure builds in DSL
* Project level cloud profiles
* Comprehensive VCS integration
* On-the-fly build progress reporting
* Remote run and pre-tested commit

**Download link:** [https://www.jetbrains.com/teamcity/download/#section=windows](https://www.jetbrains.com/teamcity/download/)

### 12) Wercker

[](http://www.wercker.com/)

[Wercker](http://www.wercker.com/) is a CI tool that automates builds and deploys the container. It creates automated pipelines which can be executed through the command line interface.

**Features:**

* Fully integrated with Github & Bitbucket
* Use Wercker CLI for faster local iterations
* Execute builds concurrently to keep your team moving
* Run parallel tests to reduce wait time of your team
* Integrate with 100s of external tools
* Receive system notification in product and by email

**Download link:** <http://www.wercker.com/>

### 13) Bitrise

Bitrise is a Continuous Integration and Delivery Platform as a Service. It offers Mobile Continuous Integration and Delivery for your entire team. It allows integrations with many popular services like Slack, HipChat, HockeyApp, Crashlytics, etc.

**Features:**

* Allows to create and test workflows in your terminal
* You get your apps without the need of manual controls
* Every build runs individually in its own virtual machine, and all data is discarded at the end of the build
* Support for third party beta testing and deployment services
* Support for GitHub Pull Request

**Download link:** [https://github.com/bitrise-io/bitrise#install-and-setup](https://github.com/bitrise-io/bitrise)

### 14) Bamboo

[https://www.guru99.com/images/2-2017/072817_0526_Top20Contin5.png](https://www.atlassian.com/software/bamboo)

[Bamboo](https://www.atlassian.com/software/bamboo) is a continuous integration build server which performs - automatic build, test, and releases in a single place. It works seamlessly with JIRA software and Bitbucket. Bamboo supports many languages and technologies such as CodeDeply, Ducker, Git, SVN, Mercurial, AWS and Amazon S3 buckets.

**Features:**

* Run parallel batch tests
* Setting up Bamboo is pretty simple
* Per-environment permissions feature allows developers and QA to deploy to their environments
* It can trigger builds based on changes detected in the repository, push notifications from Bitbucket
* Available as hosted or on-premise versions
* Facilitates real-time collaboration and integrated with HipChat.
* Built-in Git branching and workflows. It automatically merges the branches.

**Download link:** <https://www.atlassian.com/software/bamboo>

### 15) Strider

[](https://github.com/Strider-CD/strider)

[Strider](https://github.com/Strider-CD/strider) is an open source tool. Its written in Node.JS / JavaScript. It uses MongoDB as a backing store. Hence, MongoDB and Node.js are essential for installing this CI. The tool offers supports for different plugins that modify the database schema & register HTTP routes.

**Features:**

* Strider integrates with many projects like GitHub, BitBucket, Gitlab, etc.
* Allows to add hooks to execute arbitrary build actions
* Build and test your software projects continuously
* Integrates seamlessly with Github
* Publish and subscribe to socket events
* Create and modify Striders user interfaces
* Powerful plugins to customize default functionalities
* Supports Docker

**Download link:** <https://github.com/Strider-CD/strider>

### 16) Gitlab CI

[](https://gitlab.com/)

[GitLab](https://gitlab.com/) CI is a part of GitLab. It is a web application with an API that stores its state in a database. It manages projects and provides a friendly user interface, besides offering the advantage of all the features of GitLab.

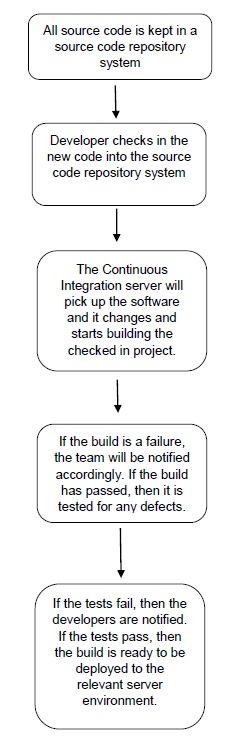
**Features:**

* GitLab Container Registry is a secure registry for Docker images
* GitLab offers a convenient way to change metadata of an issue or merge request without adding slash commands in the comment field
* It provides APIs for most features, so it allows developers to create deeper integrations with the product
* Helps developers to put their idea into production by finding areas of improvement in their development process
* It helps you to keep your information secure with Confidential Issues
* Internal projects in GitLab allow promoting inner sourcing of internal repositories.

**Download link:** <https://about.gitlab.com/installation/>

* **Workflow**

The following image shows a quick workflow of how the entire Continuous Integration workflow works in any software development project. We will look at this in detail in the subsequent chapters.



So, based on the above workflow, this is generally how the continuous integration process works.

* First, a developer commits the code to the version control repository. Meanwhile, the Continuous Integration server on the integration build machine polls source code repository for changes (e.g., every few minutes).
* Soon after a commit occurs, the Continuous Integration server detects that changes have occurred in the version control repository, so the Continuous Integration server retrieves the latest copy of the code from the repository and then executes a build script, which integrates the software
* The Continuous Integration server generates feedback by e-mailing build results to the specified project members.
* Unit tests are then carried out if the build of that project passes. If the tests are successful, the code is ready to be deployed to either the staging or production server.
* The Continuous Integration server continues to poll for changes in the version control repository and the whole process repeats.
* **Source Code Repository**

The source code repository is used for maintaining all the source code and all the changes made to it. The two most popular ones for source code repository management is subversion and Git with Git being the most recent popular system. We will now look at how to get Git installed on the system.

## Installing Git

**Step 1** − The official website for Git is <https://git-scm.com/>. If you click on the link, you will get to the home page of the Git official website as shown in the following screenshot.

**Step 2** − To download Git, just scroll down the screen and go to the Downloads section and click Downloads.

**Step 3** − Click the Windows link and the download for Git will begin automatically.

**Step 4** − Click the downloaded .exe file for Git. In our case, we are using the Git-2.6.1-64-bit.exe file. Click Run which comes appears on the next screen.

**Step 5** − Click the Next button that appears on the following screen.

**Step 6** − Click Next in the following screen to accept the General License agreement.

**Step 7** − Choose the location for your Git installation.

**Step 8** − Click Next to accept the default components that are need to be installed.

**Step 9** − Choose the option of ‘Use Git from the Windows command prompt’ since we are going to be using Git from Windows.

**Step 10** − In the following screen, accept the default setting of ‘Checkout Windows-style, commit Unix-style line endings’ and click Next.

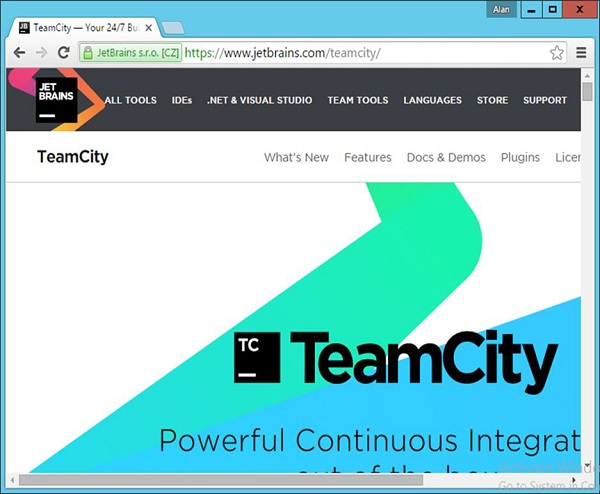
**Step 11** − In the following screen, choose the option of ‘Use Windows default console window’, since we are using Windows as the system for installation of Git.

* **Continuous Integration Server**

The next crucial software required for the entire continuous integration pipeline is the Continuous Integration software itself. Following are the most commonly used Continuous Integration softwares used in the industry −

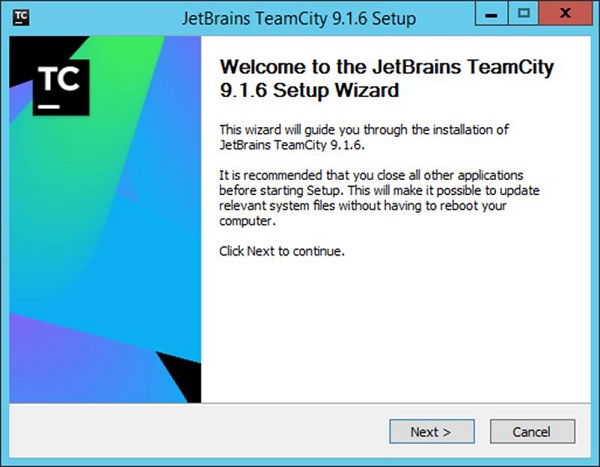
* **Jenkins**− This is an open source Continuous Integration software which is used by a lot of development communities.
* **Jet Brains TeamCity** − This is one of the most popular commercial Continuous Integration software’s available and most companies use this for their Continuous Integration needs.
* **Atlassian Bamboo** − This is another popular Continuous Integration software provided by a company called Atlassian Pvt. Ltd.
* **TeamCity**

**Step 1** − The official website for TeamCity is<https://www.jetbrains.com/teamcity/>. If you click the given link, you will go to the home page of the TeamCity official website as shown in the following screenshot. You can browse the page to download the required software for TeamCity.

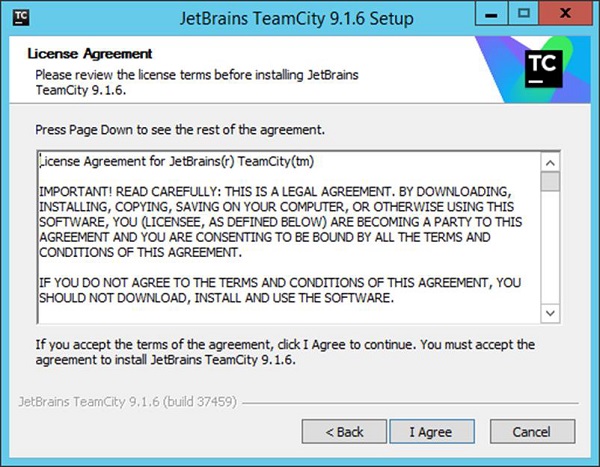


**Step 2** − The downloaded .exe is being used for the purpose of executing **TeamCity-9.1.6.exe**. Double-click the executable and then click Run in the next screen that pops up.

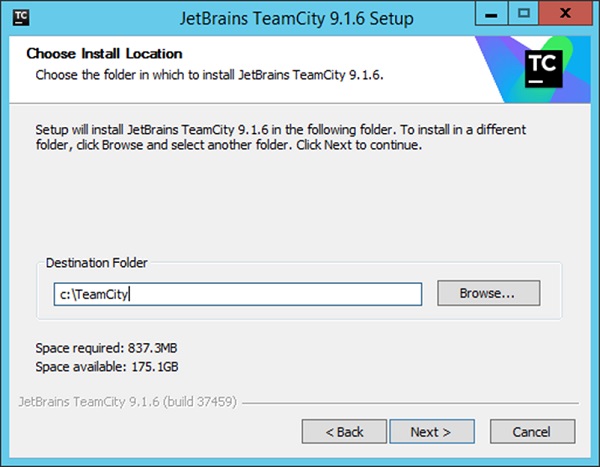
**Step 3** − Click Next to start the setup.



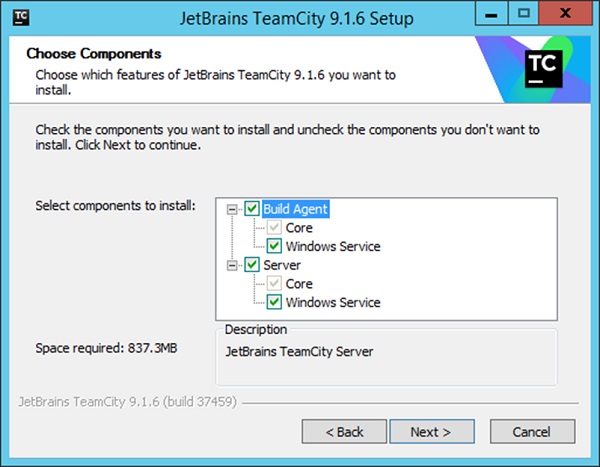
**Step 4** − Click the ‘I Agree’ button to accept the license agreement and proceed with the installation.



**Step 5** − Choose the location for the installation and click Next.

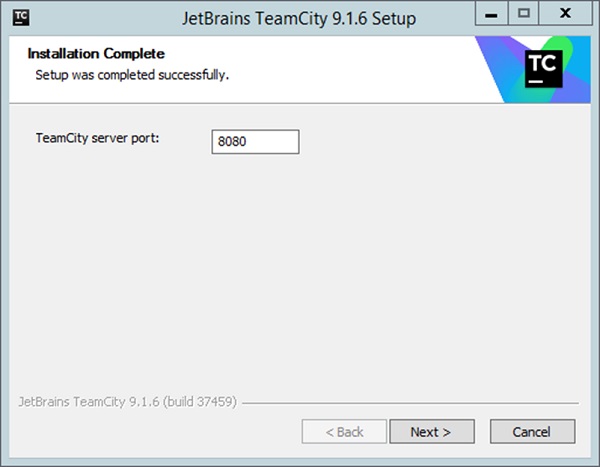


**Step 6** − Choose the default components for the installation and click Next

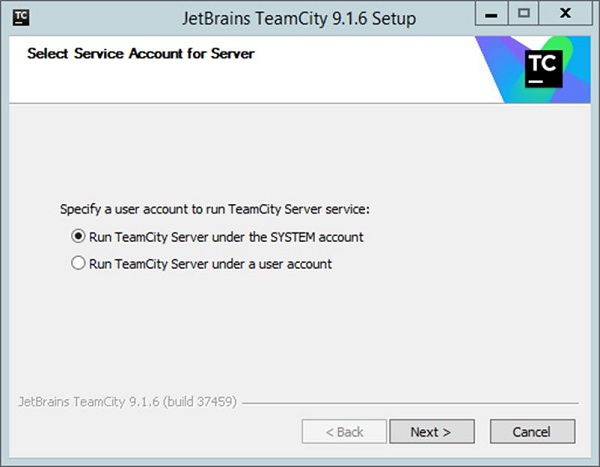


This will start the installation process. Once completed the configuration process will follow.

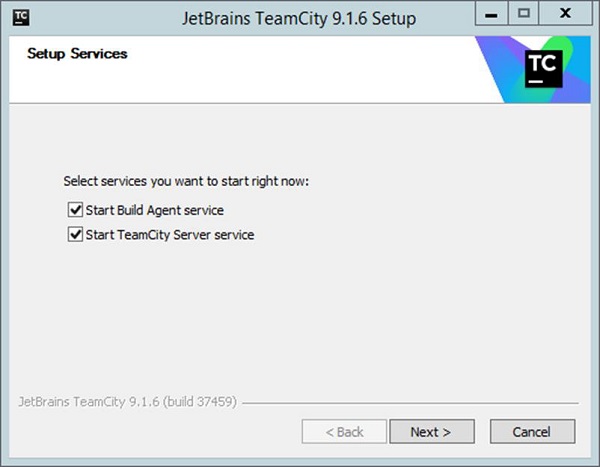
**Step 7** − Choose a port number for the server to run. Best is to use a different port such as **8080**.



**Step 8** − Next it will ask for which account TeamCity needs to run as. Choose the SYSTEM account and Click Next.



**Step 9** − Next it will ask for the services which needs to be started. Accept the default ones and then click Next.

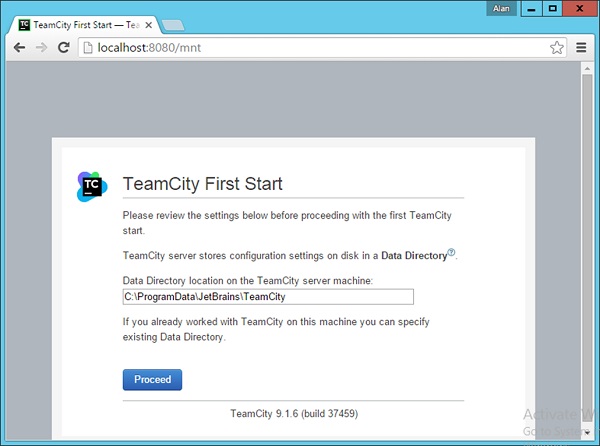


## Configuring TeamCity

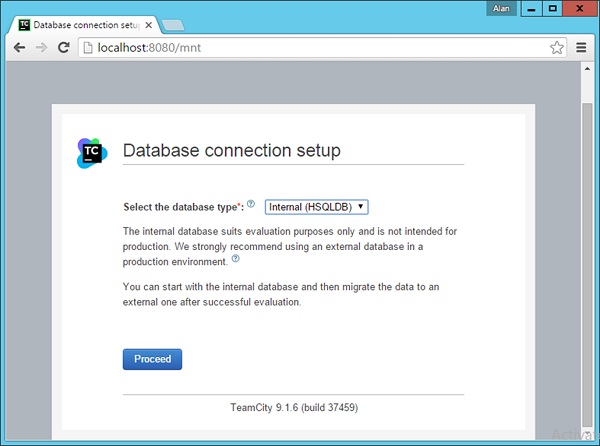
Once the installation is complete, the next step is the configuration of TeamCity. This software can be opened by browsing on the following URL in the browser −

**http://locahost:8080**

**Step 1** − The first step is to provide the location of the builds, which will be carried out by TeamCity. Choose the desired location and click the Proceed button.

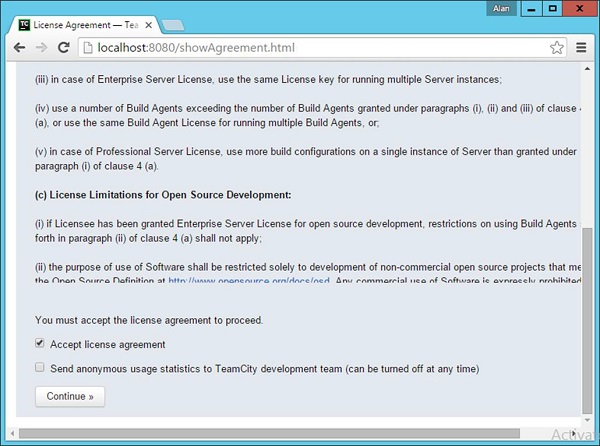


**Step 2** − The next step is to specify the database for storing all the TeamCity artefacts. For the purpose of the tutorial, one can choose the **Internal (HSQLDB)**, which is an internal database that is best suited when using products for testing purposes.

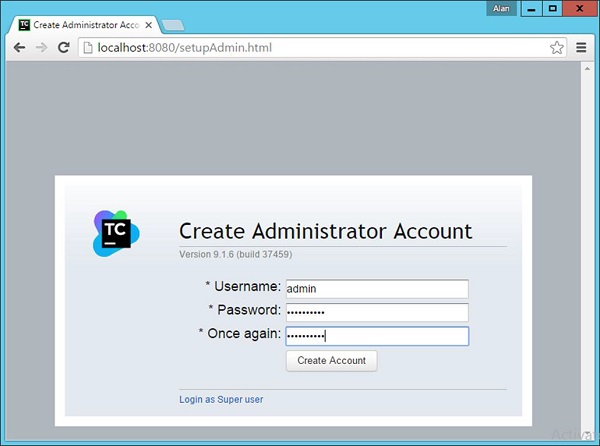


TeamCity will then process all the necessary steps to get it up and running.

**Step 3** − Next you will be requested to Accept the license agreement. Accept the same and click Continue.



**Step 4** − You need to create an administrator account that will be used to log into the TeamCity software. Enter the required details and click the ‘Create Account’ button.



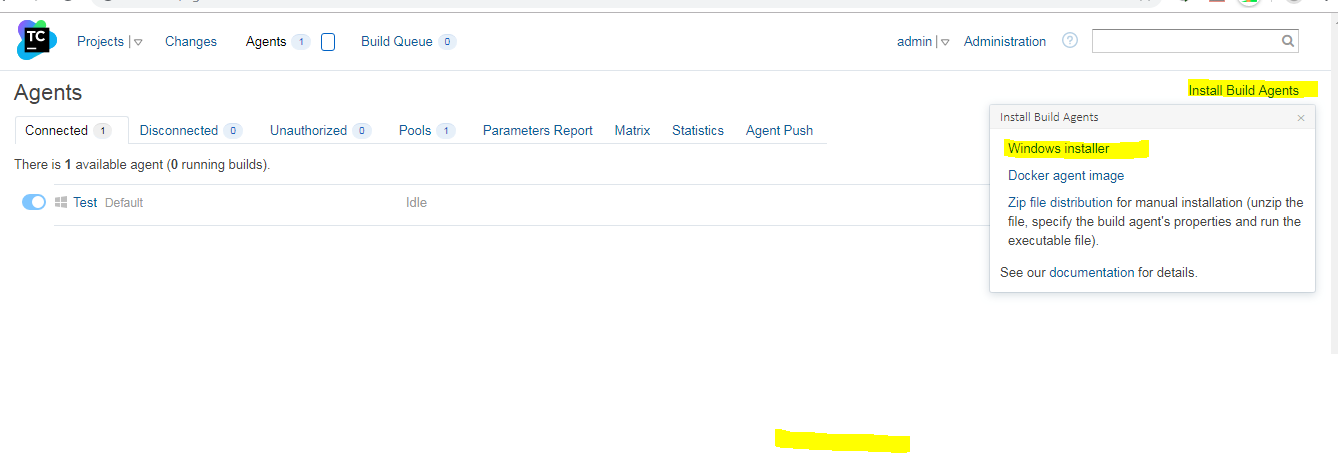
You will now be logged into TeamCity.

* **Setting up TeamCity Build agents**

TeamCity server supports multiple build agents. A build agent is an actual machine or virtual machine where the builds are run and tests are executed. Each build agent should have supporting software required for building the project.

**How to install Build Agent:**

1. Open Teamcity and Click on Install Build Agent on right side

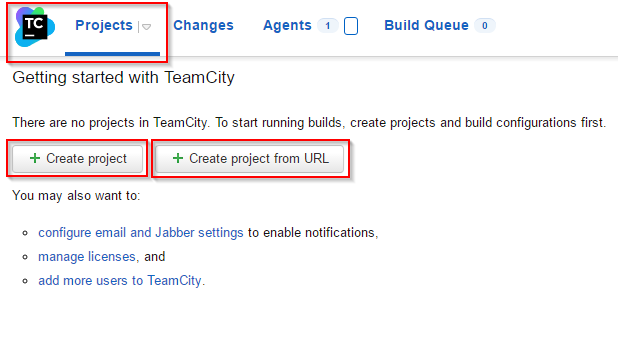


1. Click Windows Installer
2. Goto C:\BuildAgent>bin
3. Type command : agent.bat start.It will open a command prompt
4. Restart Teamcity server

## Setting up the projects in TeamCity

We can have multiple projects in team city. Each project can have multiple builds.

We can create a project from scratch or by using source code url as shown in below image.

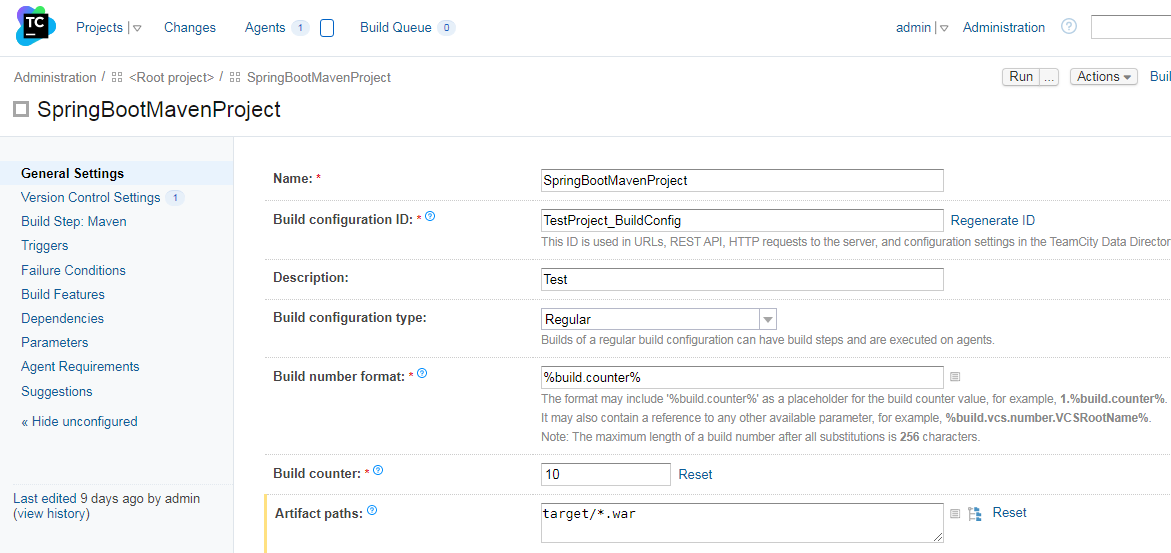
[](https://www.softpost.org/wp-content/uploads/2015/08/New-project-in-TeamCity.png)

**Each project has below settings.**

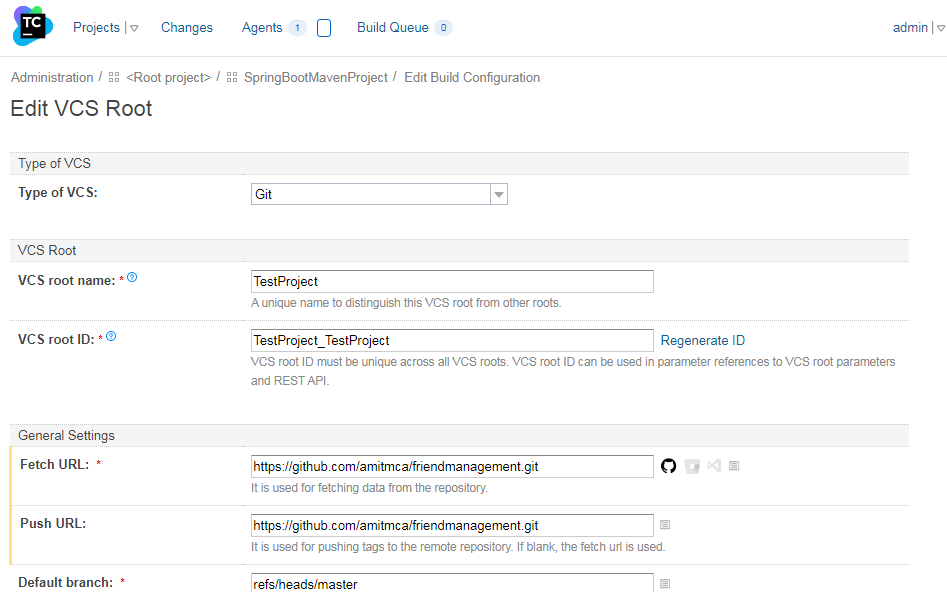
1. General – Name, project id, description, build configuration and templates, sub-projects
2. VCS roots – Here we specify the repository of project source
3. Builds Schedule – here we can configure the schedules for the build
4. SSH Keys – The settings is used to specify the SSH keys used for VCS authentication purposes.
5. Clean up rules – Here we can set up rules to perform clean up tasks like backing up the build output etc.
6. Version settings

* **Teamcity setting Up Java Maven Project from Github :**

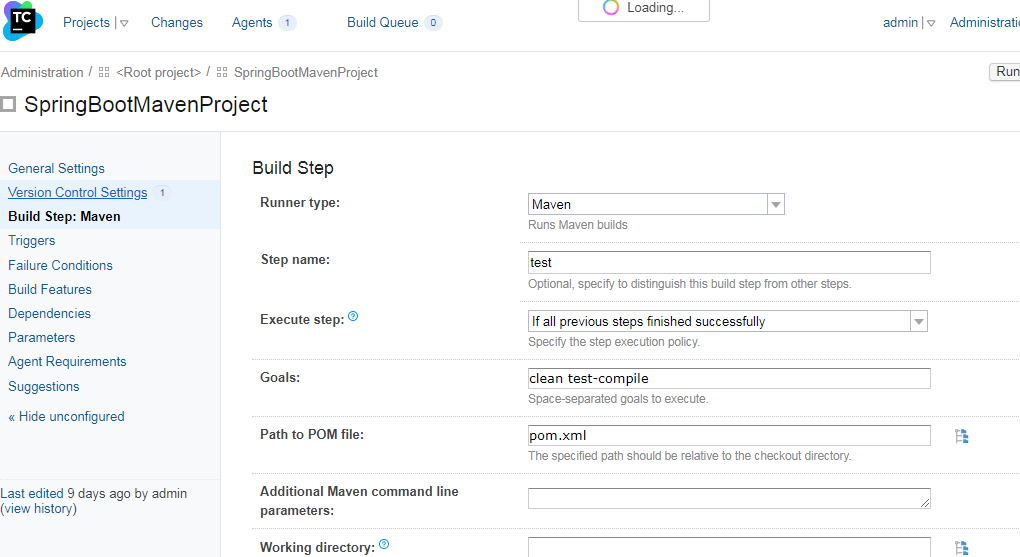
1. **General Settings**



1. **Version Control Settings : Github**



1. **Build Step**

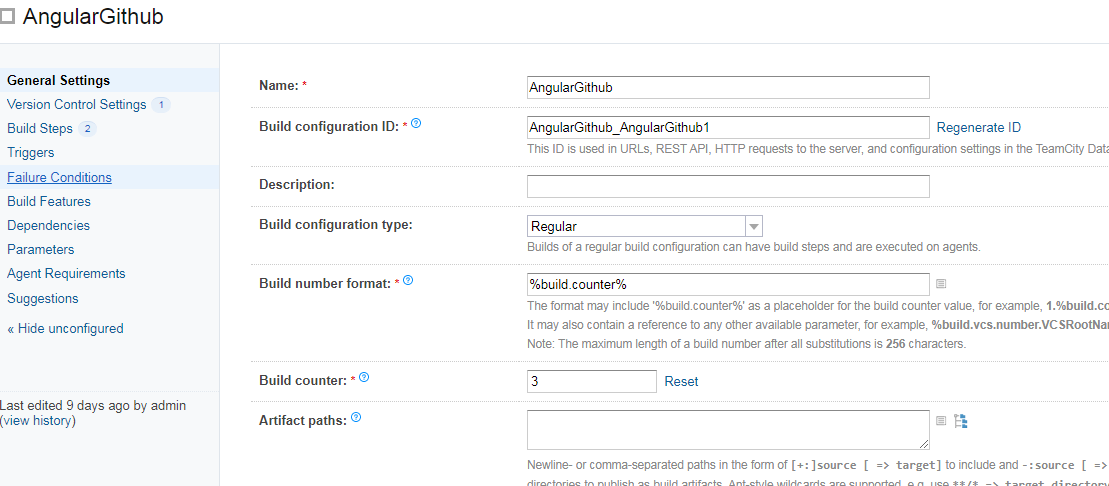


Click Save and Run the Build

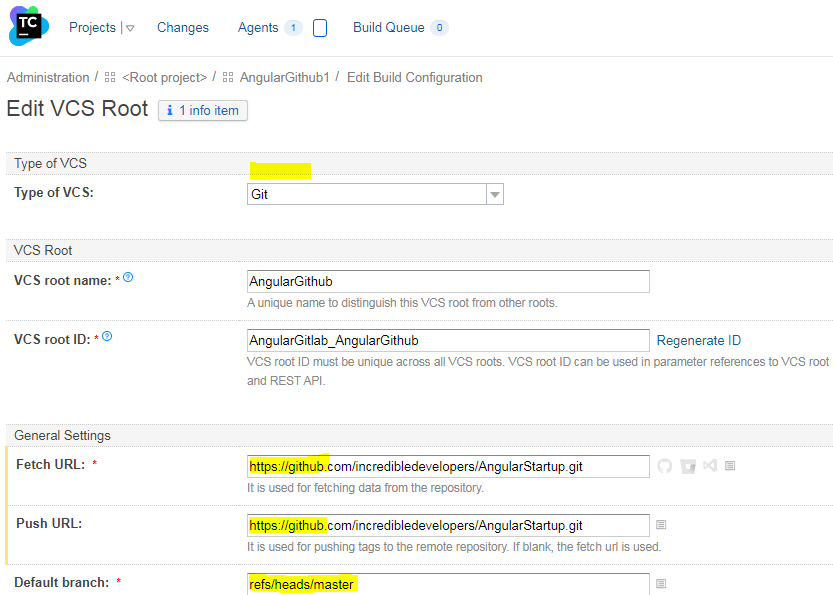


* **Teamcity setting Up Angular Project from Github :**

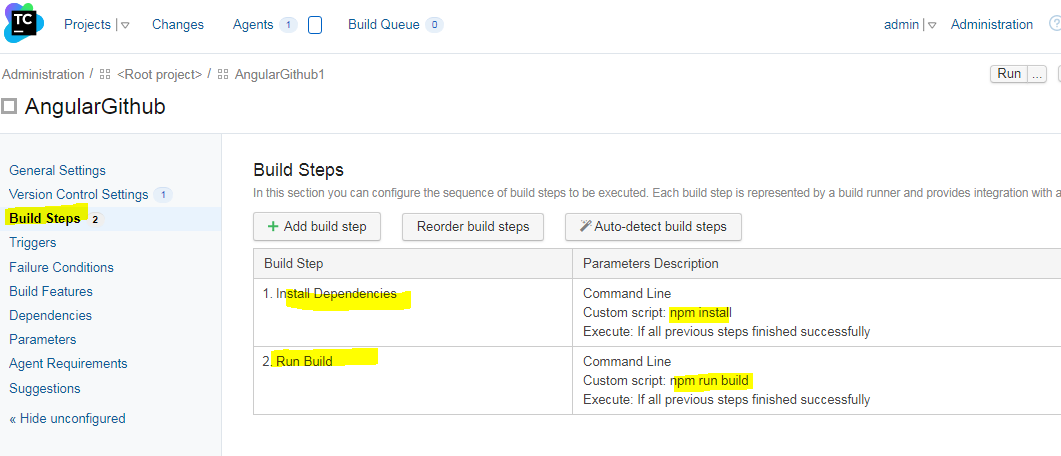
1. **General Settings:**



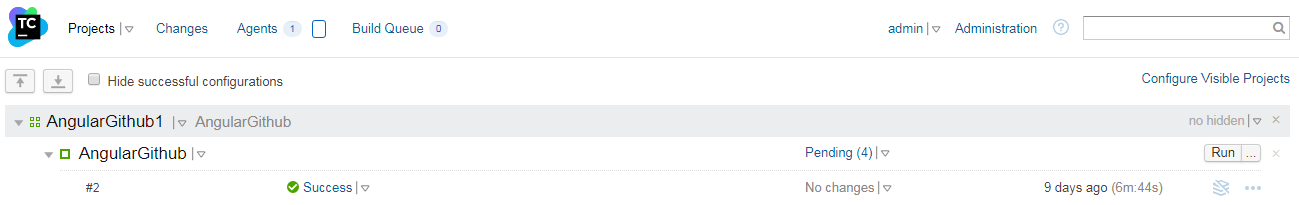
1. **Version Control Settings :**



1. **Build Steps:**

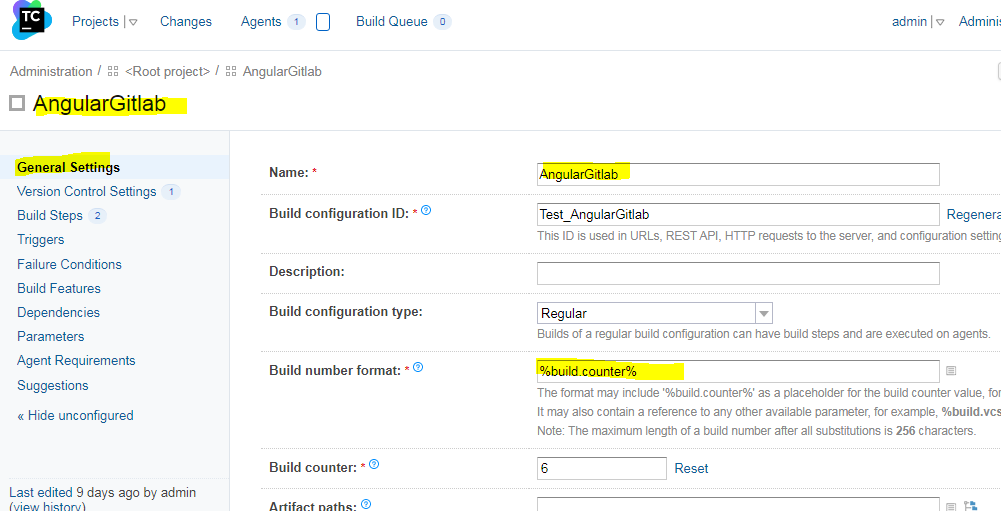


Run the Build

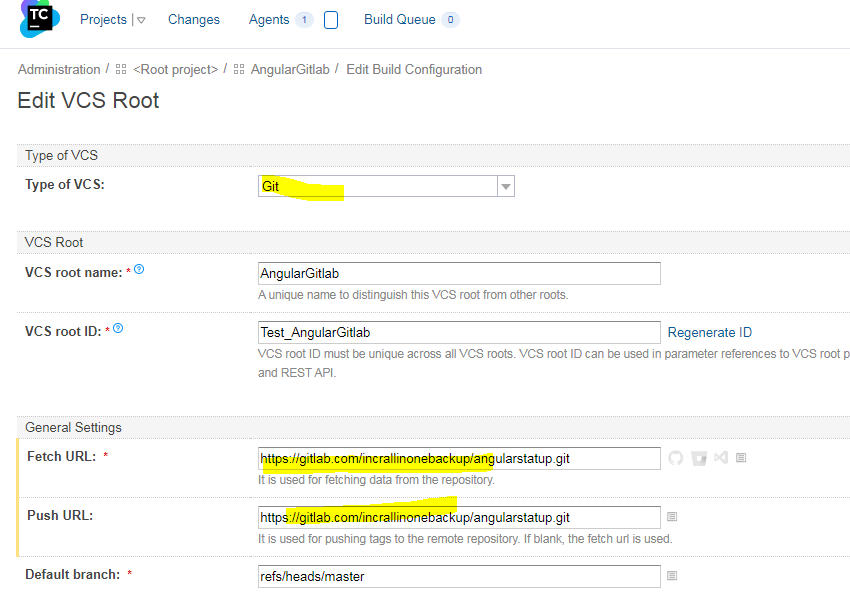


* **Teamcity setting Up Angular Project from GitLab :**

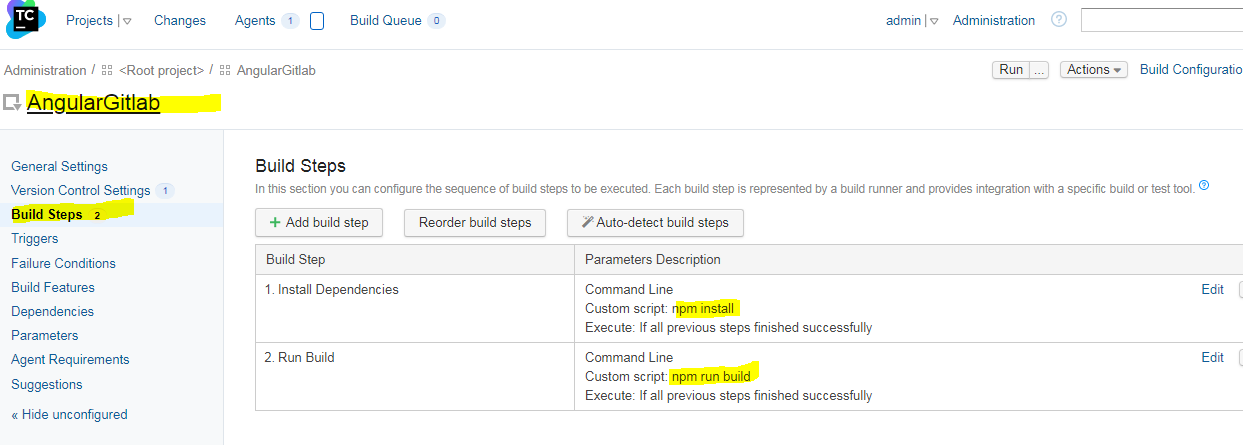
1. **General Settings**

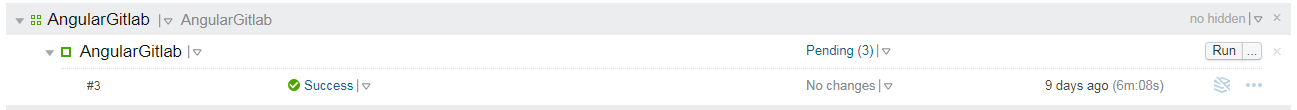


1. **Version Control Settings**

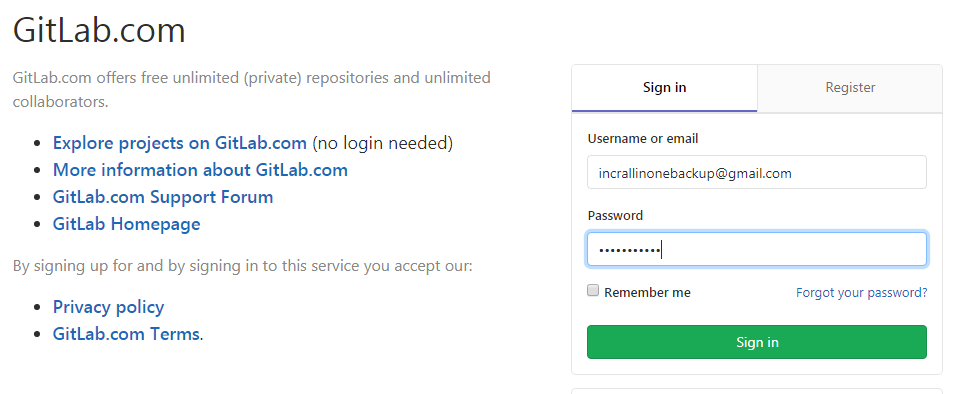


1. **Build Steps**

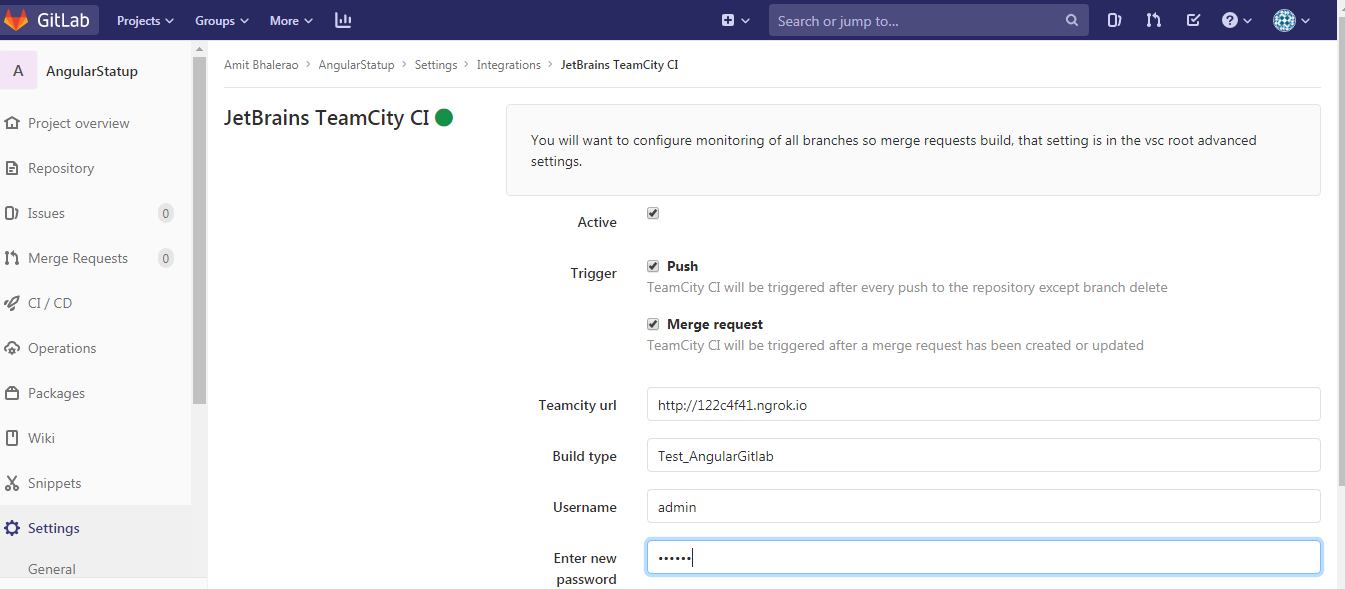




* **Teamcity Angular GitLab Webhook :**
  1. Create Build Configuration using above steps
  2. Login to GitLab



* 1. Select your project and navigate to Settings > Integrations > [**JetBrains TeamCity CI**](https://gitlab.com/incrallinonebackup/angularstatup/-/services/teamcity/edit)

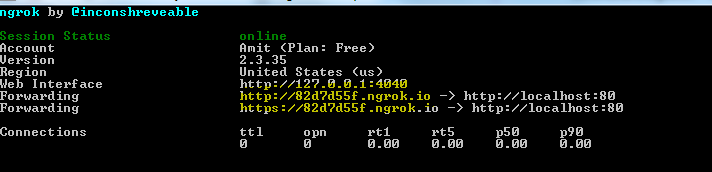


Here the Teamcity url : <http://122c4f41.ngrok.io> is an id generated using **ngrok**(ngrok.com)

ngrok is a cross-platform application that enables developers to expose a local development server to the Internet with minimal effort

Steps to expose your local development server (Teamcity server running on port 80)

* Download and extract ngRok in folder ngrok-stable-windows-amd64
* Open the Command Prompt
* Use command : ngrok.exe http 80



Now your localhost running on Port 80 is exposed to internet and is accessible from http://82d7d55f.ngrok.io

* 1. Click on .Observe that Teamcity build is started automatically.



* **Jenkins**

## What is Jenkins?

**Jenkins** is an open source Continuous Integration server capable of orchestrating a chain of actions that help to achieve the Continuous Integration process (and not only) in an automated fashion.

Jenkins is entirely written in Java. Jenkins is a widely used application around the world that has around 300k installations and growing day by day.

It is a server-based application and requires a web server like Apache Tomcat. The reason Jenkins became so popular is that of its monitoring of repeated tasks which arise during the development of a project. For example, if your team is developing a project, Jenkins will continuously test your project builds and show you the errors in early stages of your development.

By using Jenkins, software companies can accelerate their software development process, as Jenkins can automate build and test at a rapid rate. Jenkins supports the complete development lifecycle of software from building, testing, documenting the software, deploying and other stages of a software development lifecycle.

## Jenkins History

* Kohsuke Kawaguchi, a Java developer, working at SUN Microsystems, was tired of building the code and fixing errors repetitively. In 2004, created an automation server called ***Hudson*** that automates builds and test task.
* In 2011, Oracle who owned Sun Microsystems had a dispute with Hudson open source community, so the forked Hudson and renamed it as **Jenkins**.
* Both Hudson and Jenkins continued to operate independently. But in short span of time, Jenkins acquired a lot of projects and contributors while Hudson remained with only 32 projects. With time, Jenkins became more popular, and Hudson is not maintained anymore.

## Why Continuous Integration with Jenkins?

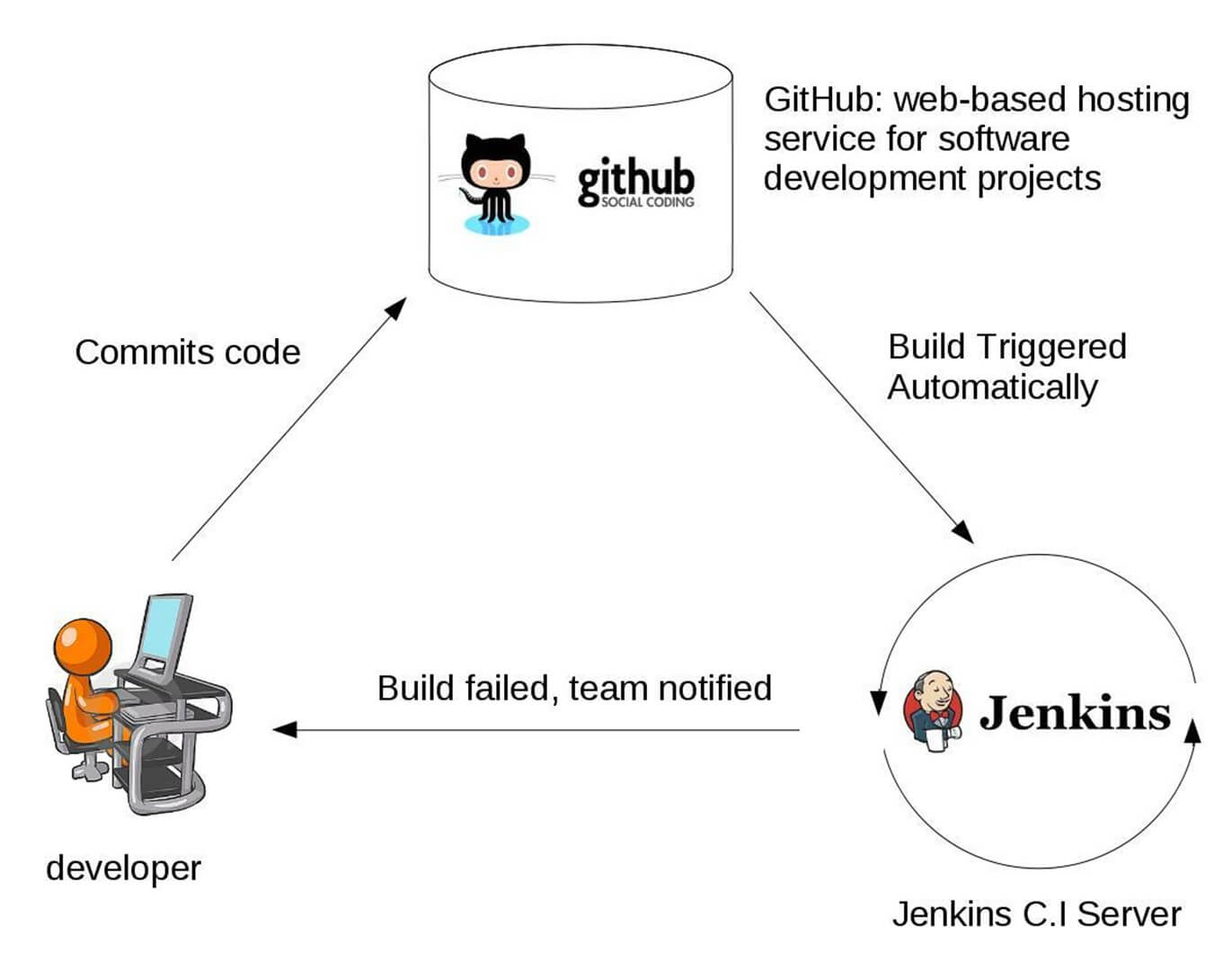
Some people might think that the old-fashioned way of developing the software is the better way. Let us imagine that there are around 10 developers who are working on a shared repository. Some developer completes their task in 25 days while others take 30 days to complete.

|  |  |
| --- | --- |
| **Before Jenkins** | **After Jenkins** |
| Once all Developers had completed their assigned coding tasks, they used to commit their code all at same time. Later, Build is tested and deployed.  Code commit built, and test cycle was very infrequent, and a single build was done after many days. | The code is built and tests as soon as Developer commits code. Jenkin will build and test code many times during the day  If the build is successful, then Jenkins will deploy the source into the test server and notifies the deployment team.  If the build fails, then Jenkins will notify the errors to the developer team. |
| Since the code was built all at once, some developers would need to wait until other developers finish coding to check their build | The code is built immediately after any of the Developer commits. |
| It is not an easy task to isolate, detect, and fix errors for multiple commits. | Since the code is built after each commit of a single developer, it's easy to detect whose code caused the built to fail |
| Code build and test process are entirely manual, so there are a lot of chances for failure. | Automated build and test process saving timing and reducing defects. |
| The code is deployed once all the errors are fixed and tested. | The code is deployed after every successful build and test. |
| Development Cycle is slow | The development cycle is fast. New features are more readily available to users. Increases profits. |

## Real-world case study of Continuous Integration

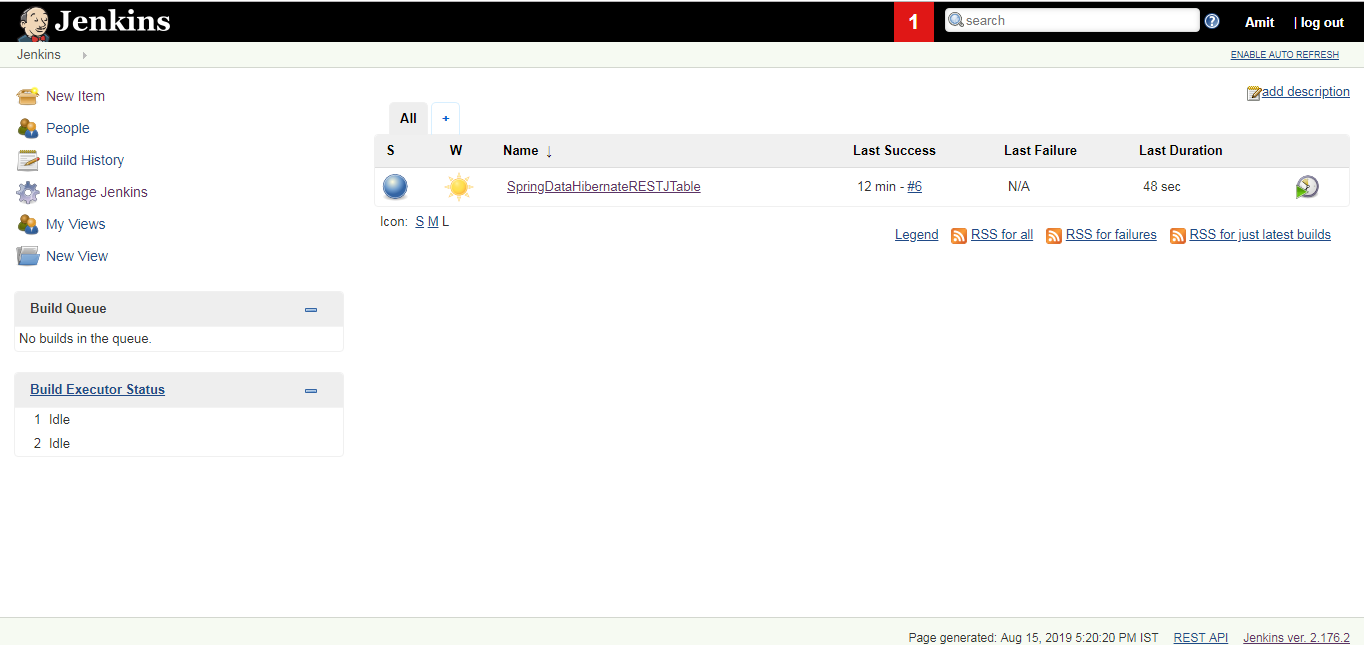
I am sure all of you aware of old phone Nokia. Nokia used to implement a procedure called nightly build. After multiple commits from diverse developers during the day, the software built every night. Since the software was built only once in a day, it's a huge pain to isolate, identify, and fix the errors in a large code base.

Later, they adopted Continuous Integration approach. The software was built and tested as soon as a developer committed code. If any error is detected, the respective developer can quickly fix the defect.

[](https://www.guru99.com/images/1/063018_1012_WhatisJenki1.jpg)

### Jenkins Plugins

By default, Jenkins comes with a limited set of features. If you want to integrate your Jenkins installation with version control tools like Git, then you need to install plugins related to Git. In fact, for integration with tools like Maven, Amazon EC2, you need to install respective plugins in your Jenkins.



## Advantages of using Jenkins

* Jenkins is being managed by the community which is very open. Every month, they hold public meetings and take inputs from the public for the development of Jenkins project.
* So far around 280 tickets are closed, and the project publishes stable release every three months.
* As technology grows so does Jenkins. So far Jenkins has around 320 plugins published in its plugins database. With plugins, Jenkins becomes even more powerful and feature rich.
* Jenkins also supports cloud-based architecture so that you can deploy Jenkins in cloud-based platforms.
* The reason why Jenkins became popular is that it was created by a developer for developers.

## Disadvantages of using Jenkins

Though Jenkins is a very powerful tool, it has its flaws.

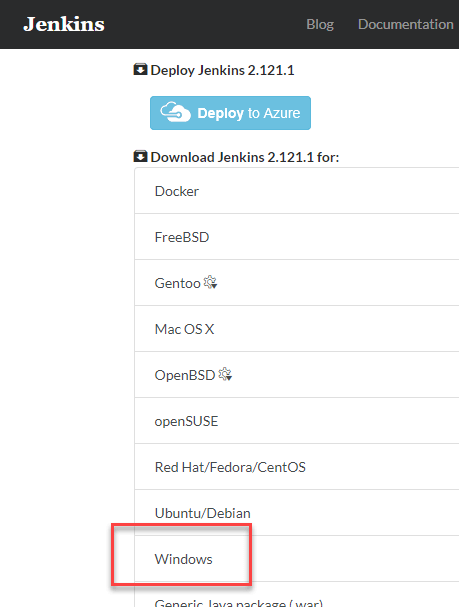
* Its interface is out dated and not user friendly compared to current UI trends.
* Though Jenkins is loved by many developers, it's not that easy to maintain it because Jenkins runs on a server and requires some skills as server administrator to monitor its activity.
* One of the reasons why many people don't implement Jenkins is due to its difficulty in installing and configuring Jenkins.
* Continuous integrations regularly break due to some small setting changes. Continuous integration will be paused and therefore requires some developer attention.

# How to Download & Install Jenkins on Windows

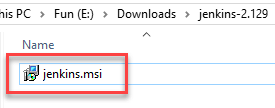
## How to Download Jenkins?

Following steps should be followed so that to install Jenkins successfully:

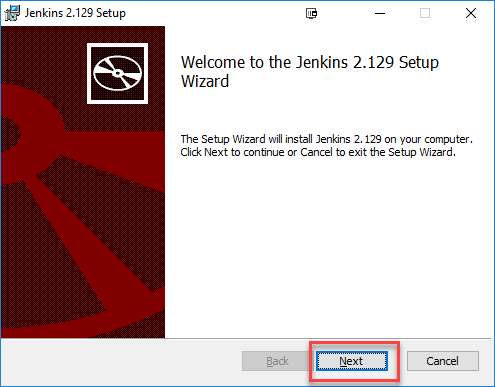
**Step 1)** Got to <https://jenkins.io/download/> and select the platform. In our case Window

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo1.png)

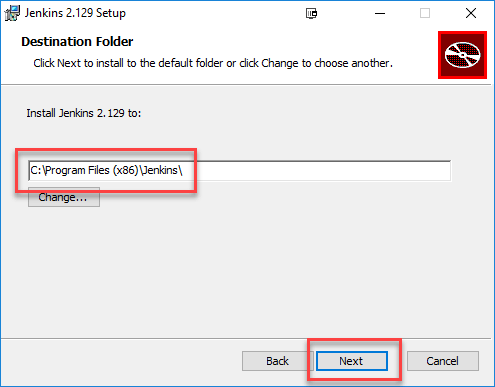
**Step 2)** Go to download location from local computer and unzip the downloaded package. Double-click on unzipped **jenkins.msi**

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo2.png)

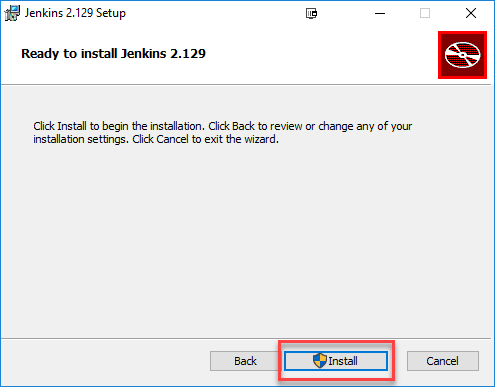
**Step 3)** In the setup screen, click Next.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo3.png)

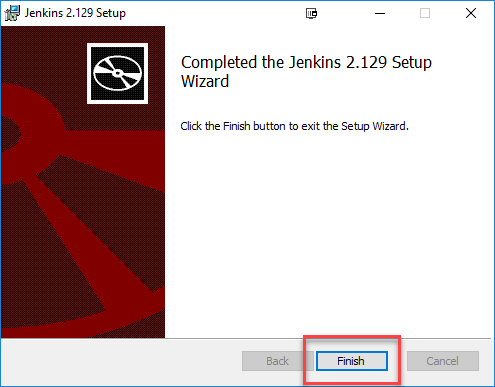
**Step 4)** Choose the location where you want to have the Jenkins instance installed (default location is C:\Program Files (x86)\Jenkins), then click on **Next** button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo4.png)

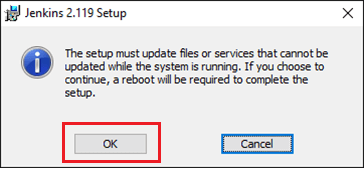
**Step 5)**Click on the Install button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo5.png)

**Step 6)** Once install is complete, click Finish.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo6.png)

**Step 7)** During the installation process an info panel may pop-up to inform the user that for a complete setup, the system should be rebooted at the end of the current installation. Click on OK button when the Info panel is popping-up:

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo7.png)

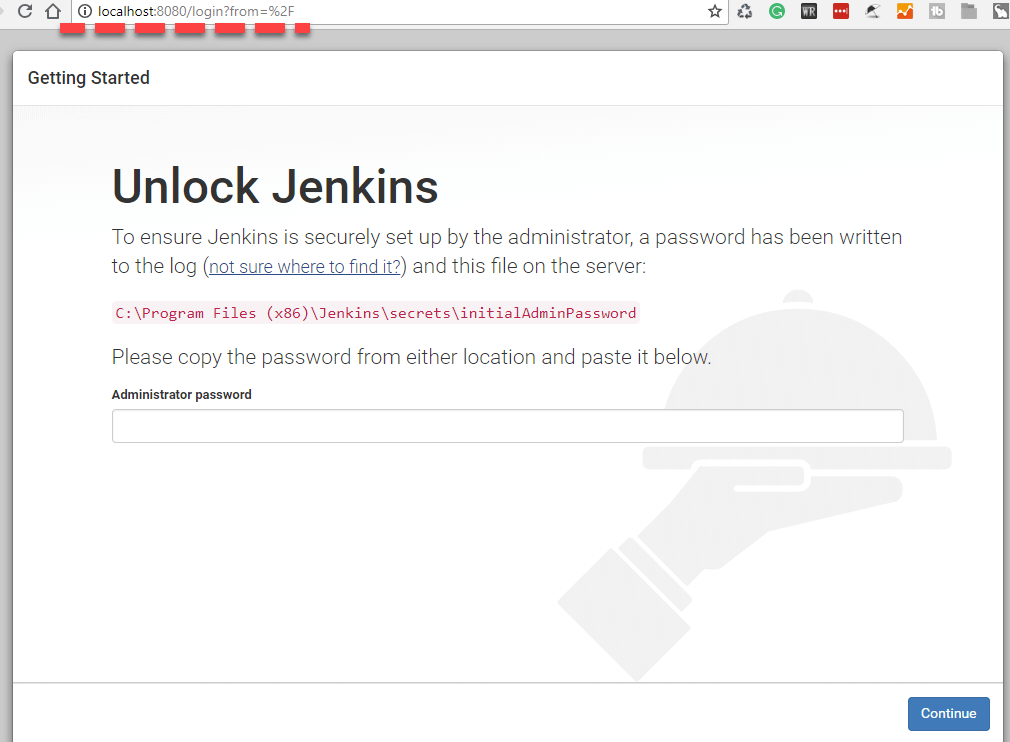
## How to Unblock Jenkins?

After completing the Jenkins installation phase, you should proceed further and start its configuration. Next steps will guide you how you can unblock Jenkins application:

**Step 1)** After completing the Jenkins installation process, a browser tab will pop-up asking for the initial Administrator password. To access Jenkins, you need to go to browse the following path in your web browser.

http://localhost:8080

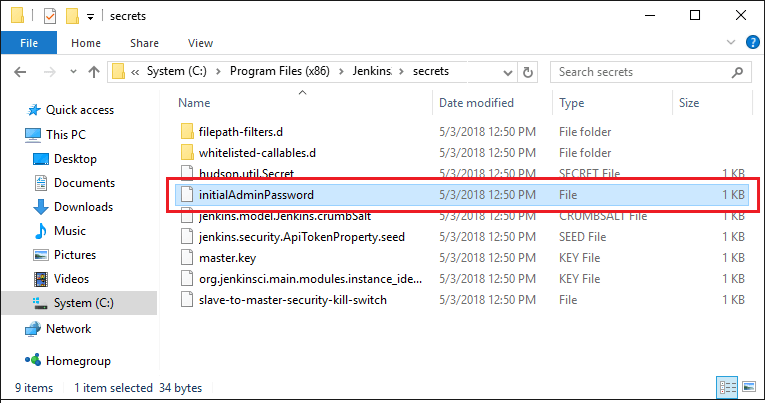
If you can access the above URL, then it confirms that Jenkins is successfully installed in your system.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo8.png)

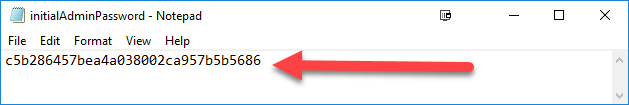
**Step 2)** The initial Administrator password should be found under the Jenkins installation path (set at Step 4 in Jenkins Installation).

For default installation location to C:\Program Files (x86)\Jenkins, a file called **initialAdminPassword** can be found under C:\Program Files (x86)\Jenkins\secrets.

However, If a custom path for Jenkins installation was selected, then you should check that location for **initialAdminPassword** file.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo9.png)

**Step 3)** Open the highlighted file and copy the content of the **initialAdminPassword**file.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo10.png)

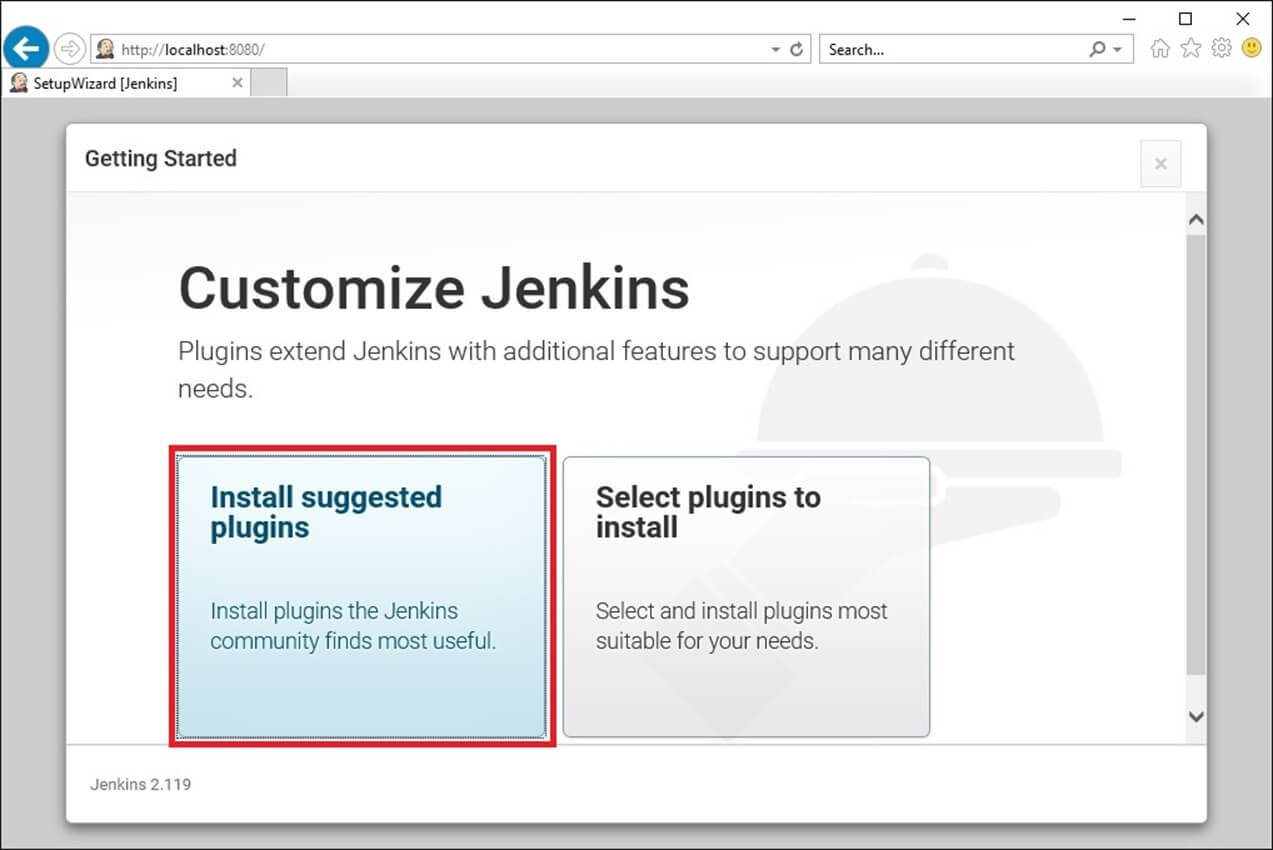
**Step 4)** Paste the password it into browser's pop-up tab (<http://localhost:8080/login?form=%2F>) and click on Continue button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo11.jpg)

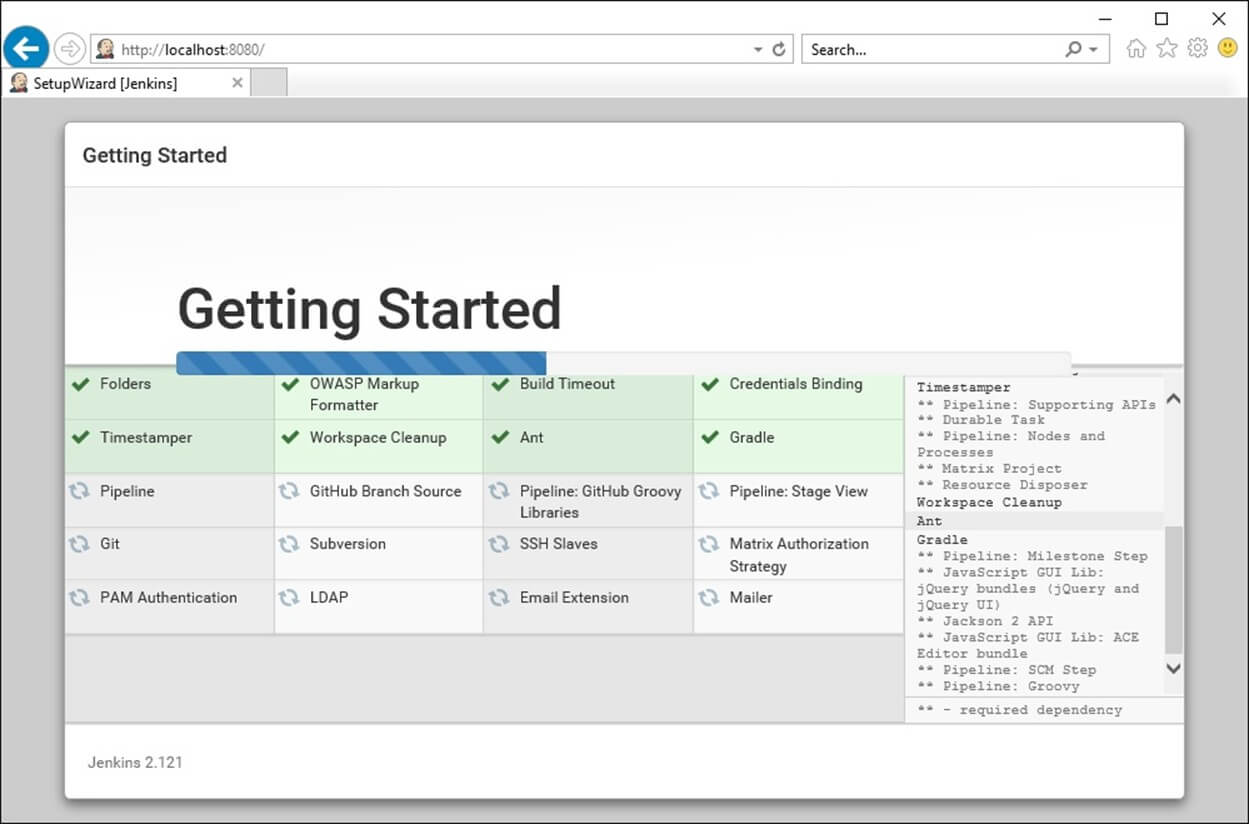
## Customize Jenkins

You can also customize your Jenkins environment by below-given steps:

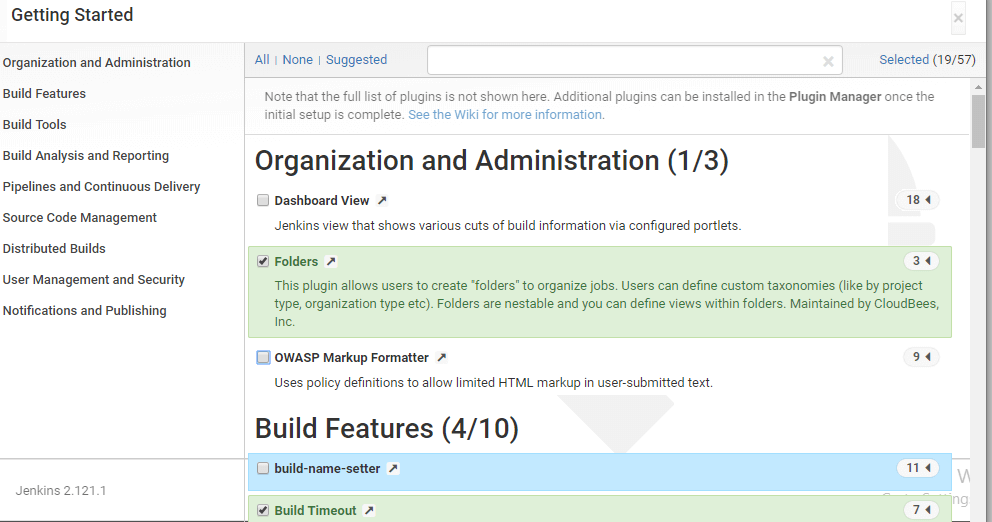
**Step 1)** Click on the "Install suggested plugins button" so Jenkins will retrieve and install the essential plugins

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo12.jpg)

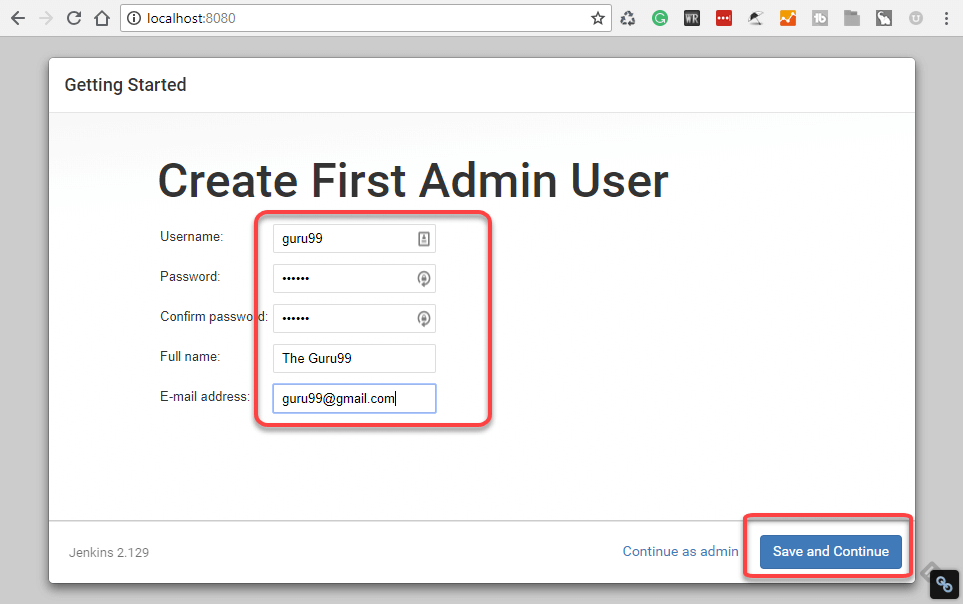
Jenkins will start to download and install all the necessary plugins needed to create new Jenkins Jobs.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo13.jpg)

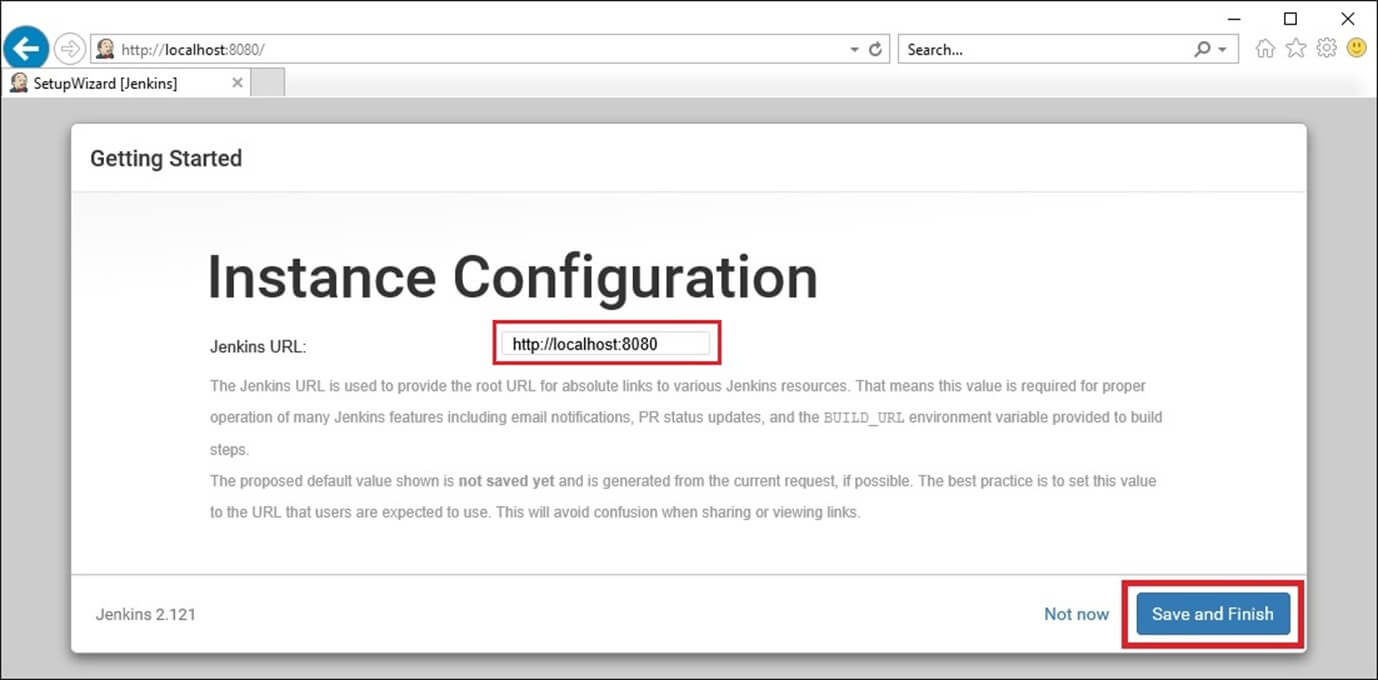
**Note**: You can choose the Option "Select Plugins to Install" and select the plugins you want to install

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo14.png)

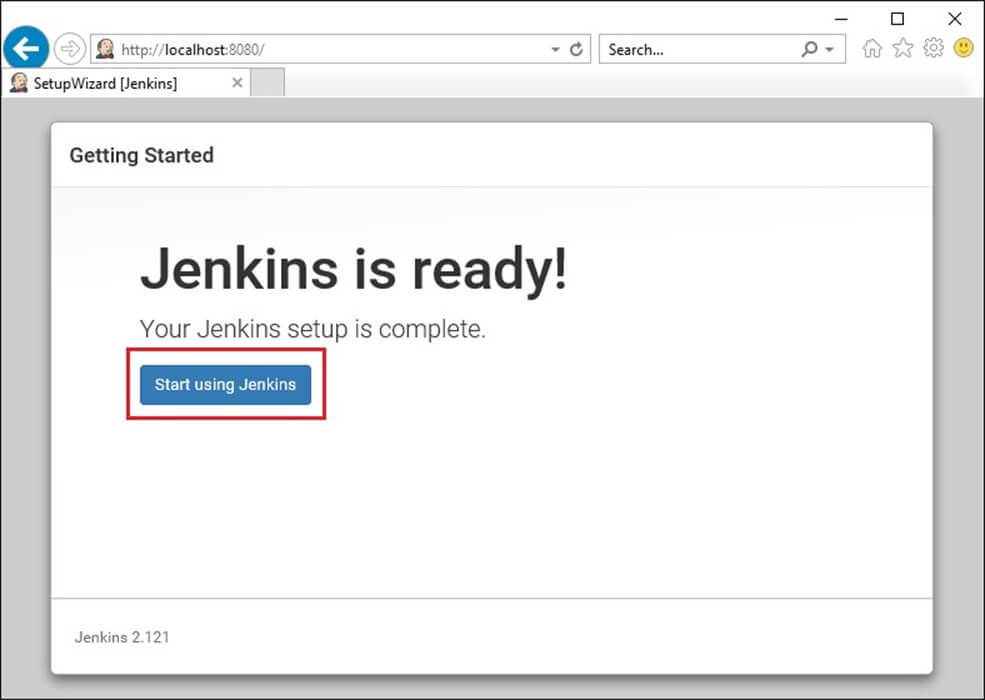
**Step 2)** After all suggested plugins were installed, the "Create First Admin User" panel will show up. Fill all the fields with desired account details and hit the "**Save and Finish**" button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo15.png)

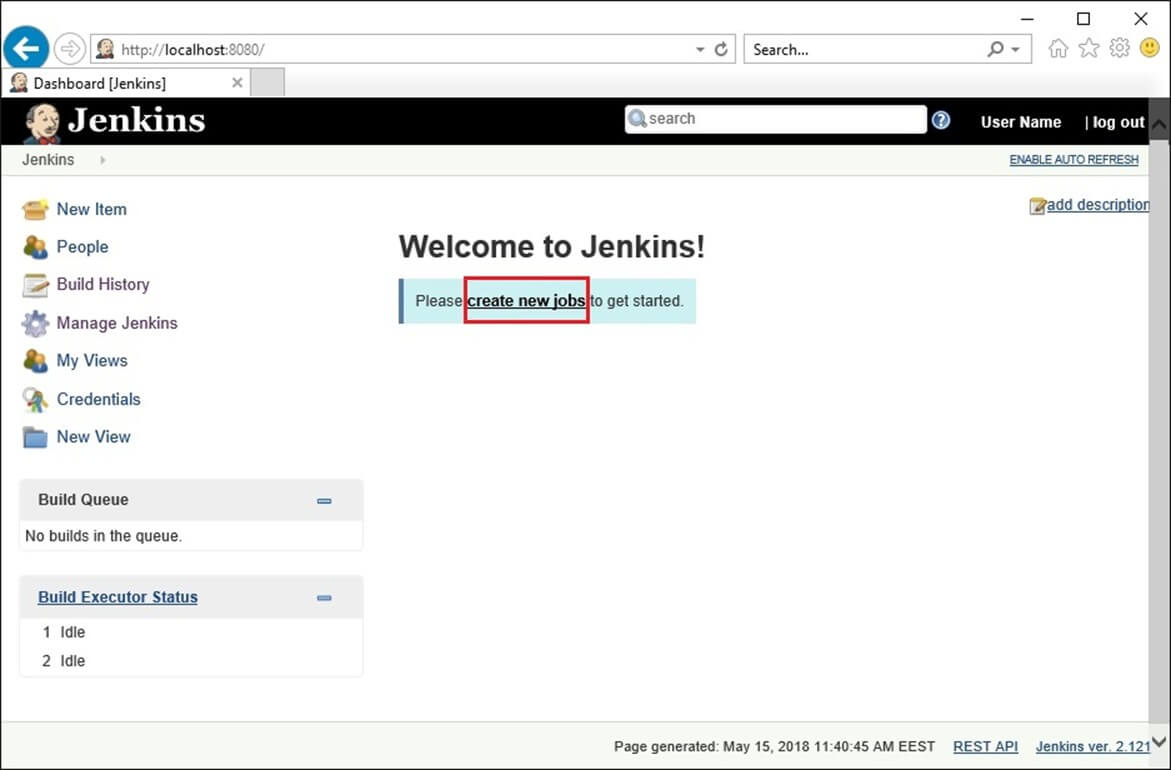
**Step 3)**Once you have filled the above data, finally it will ask for URL information where you can configure the default instance path for Jenkins. Leave it as it is to avoid any confusions later. However, if another application is already using 8080 port, you can use another port for Jenkins and finally save the settings, and you are done with installation of Jenkins. Hit the "**Save and Continue**" button:

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo16.jpg)

Congratulations! We have successfully installed a new Jenkins Server. Hit the "Start using Jenkins" button.

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo17.jpg)

Below you can find the Jenkins instance up and run, ready to create first Jenkins jobs:

[](https://www.guru99.com/images/1/063018_1023_HowtoDownlo18.jpg)

#### [Using the .war file to start Jenkins](https://www.vogella.com/tutorials/Jenkins/article.html#using-the-war-file-to-start-jenkins)

Download the jenkins.war file from [Jenkins Homepage](https://jenkins.io/). From this file you can start Jenkins directly via the command line with java -jar jenkins\*.war.

If you start it locally, you find it running under the following URL: <http://localhost:8080/>

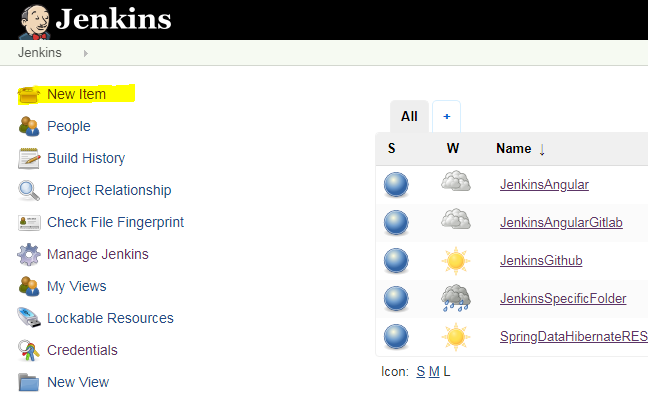
To run it in your Tomcat server, put the .war file into the webapps directory. If you start Tomcat, your Jenkins installation will be available under

http://localhost:8080/jenkins

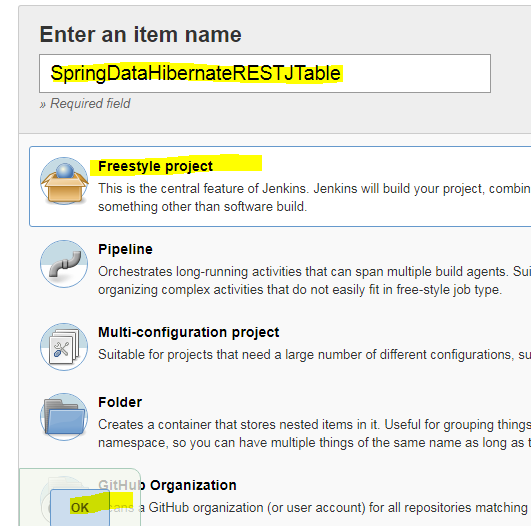
|  |  |
| --- | --- |
|  | If the jenkins.war is deployed in your webapps directory, but cannot be started and the tomcat manager says ﻿FAIL - Application at context path /jenkins could not be started, you may need to grant the permissons for ﻿JENKINS\_HOME. |

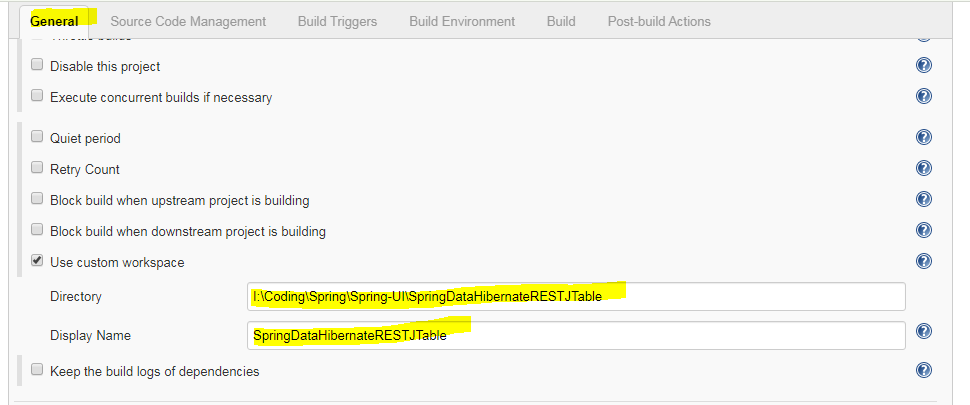
* **Local Java Job Using Jenkins:**

Step 1) Login to Jenkins and click on New Item

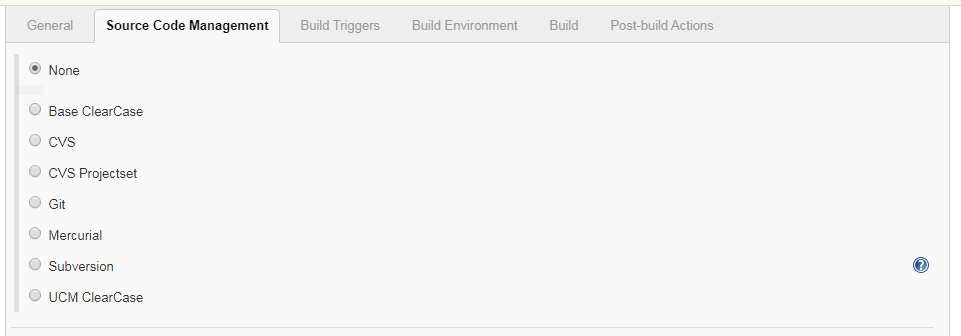


Step 2) Select Freestyle Project and provide project name. Click ok.

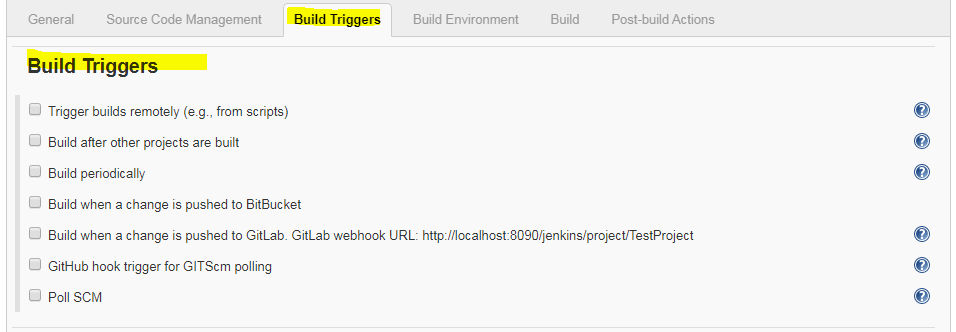


Step 3) General Tab : Provide Custom Workspace 

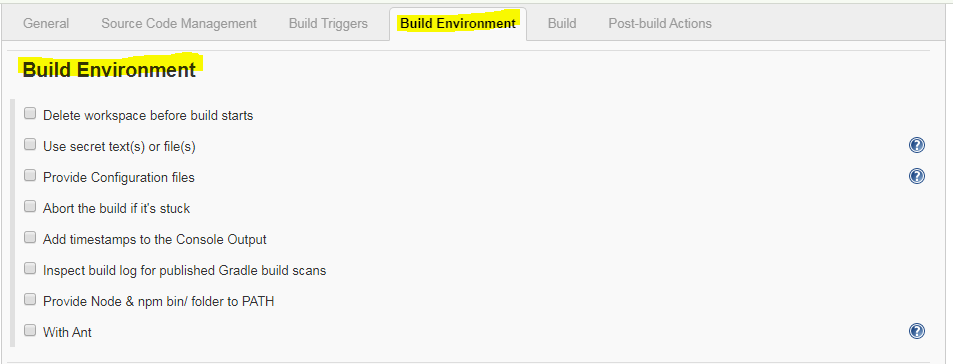
Step 4) Source Code Management: Select None



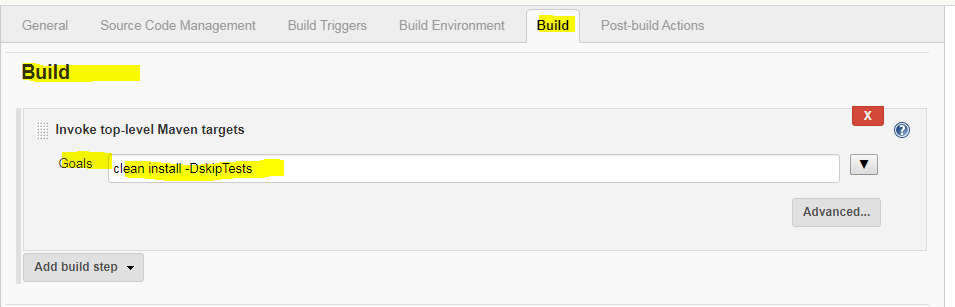
Step 5) Build Triggers



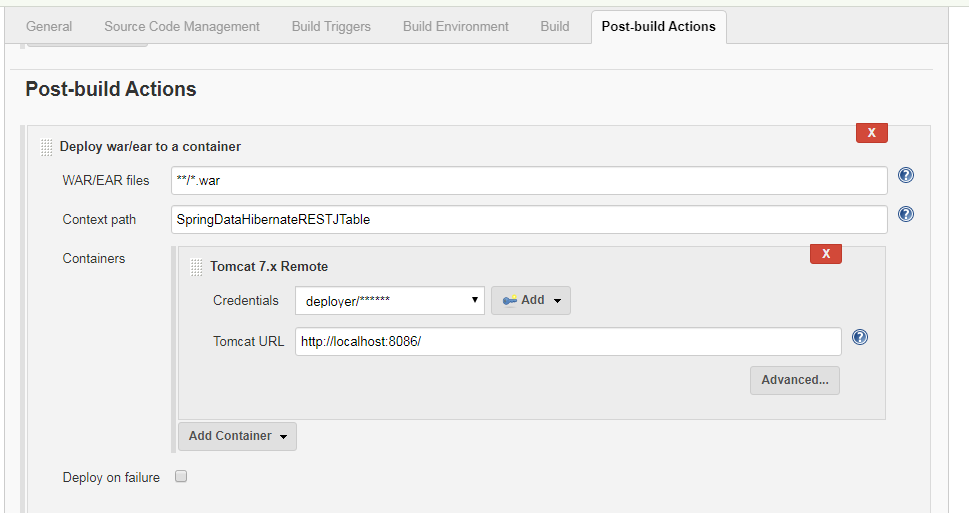
Step 6) Build Environment



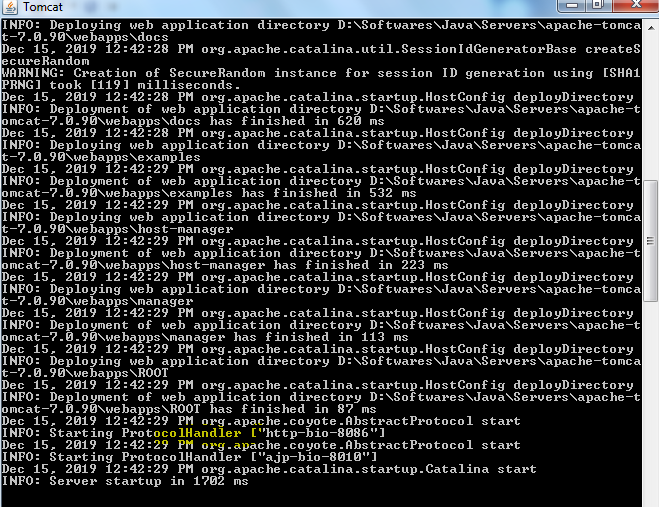
Step 7) Build



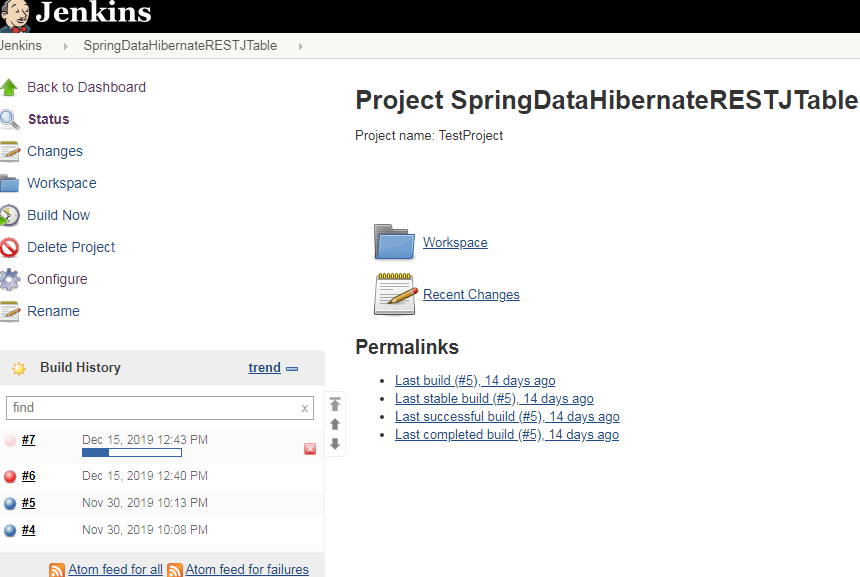
Step 8) Post Build Actions: After successful Build, deploy the war to Tomcat server running on port 8086



Step 9) Save and Start the Tomcat on Port no 8086



Step 10) Build Now

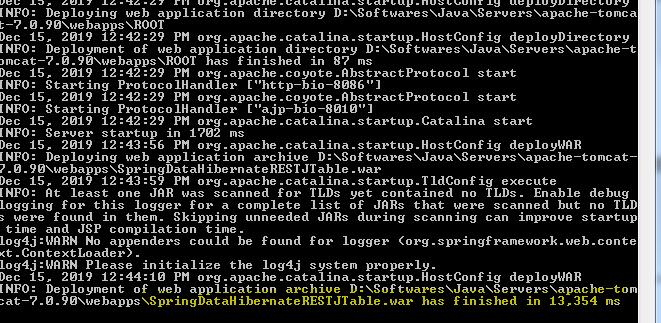


Build #7 will get started. Click on the Build and then Console output

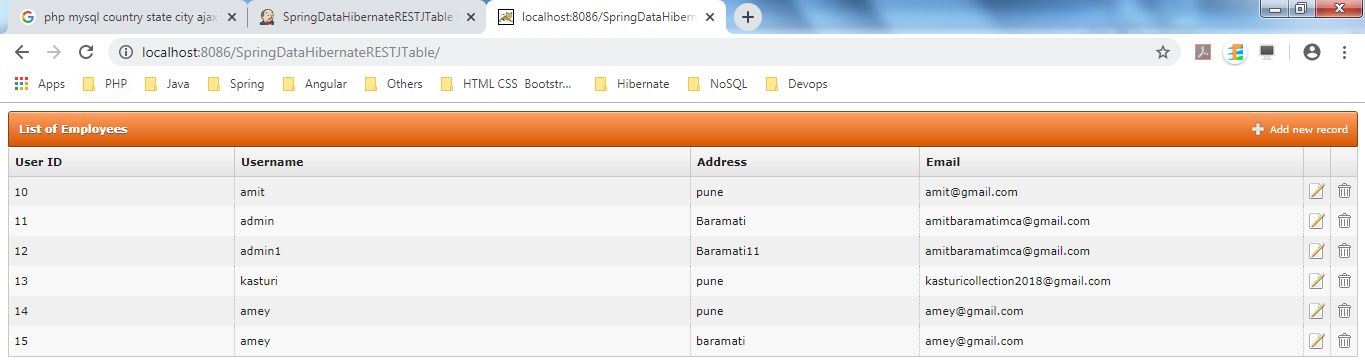




Build Successfully Completed and deployed on Tomcat running on Port 8086



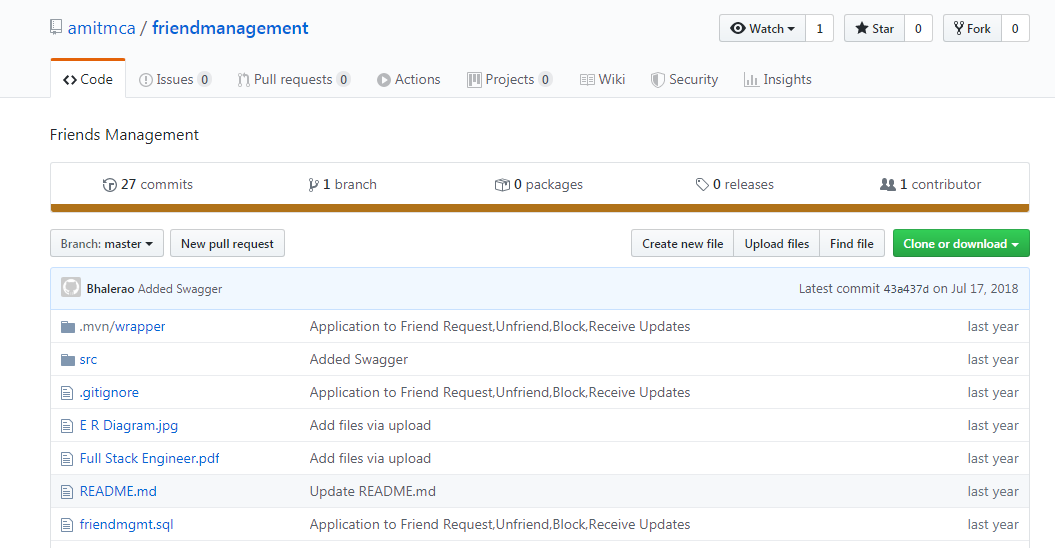
Try to access the Application



* **Jenkins-Github Integration:**

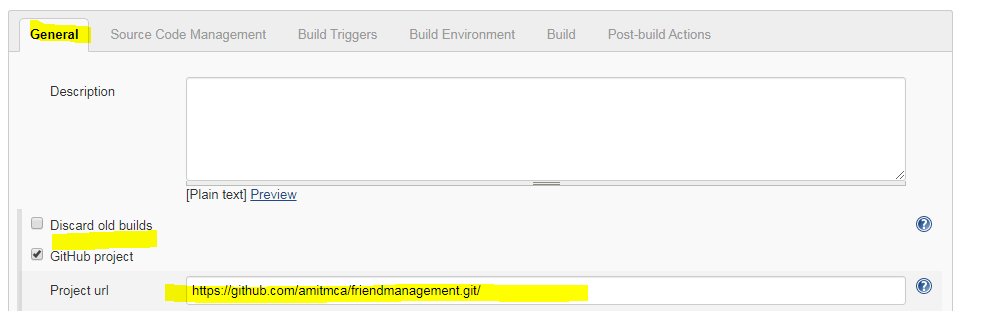
**Pre-requisite:** Make sure you have **Git** and **Github** plugins installed in Jenkins. Use Manage Jenkins.

**Github Repo: https://github.com/amitmca/friendmanagement.git/**

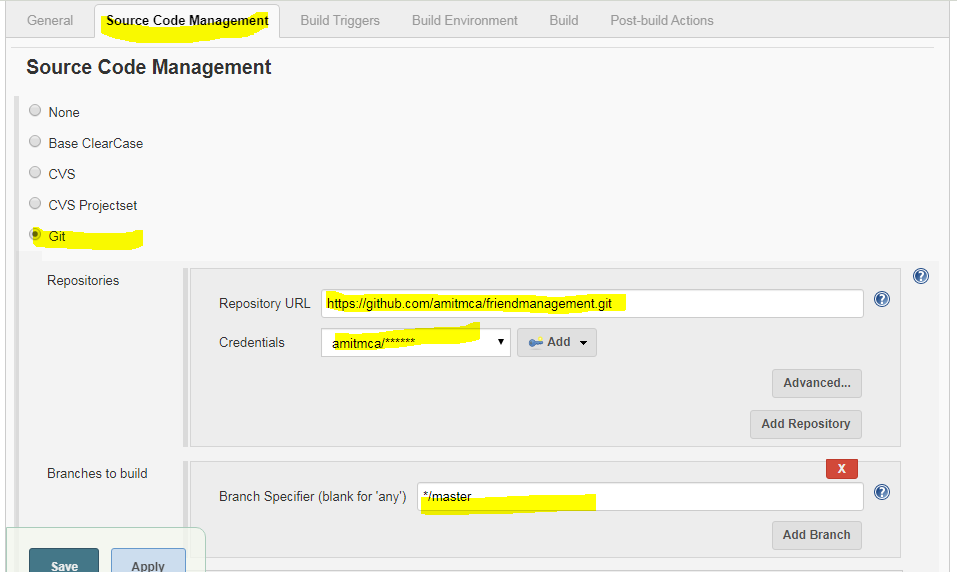


**Step 1)** Create FreeStyle project with name “JenkinsGithub”

**Step 2) General Tab:**

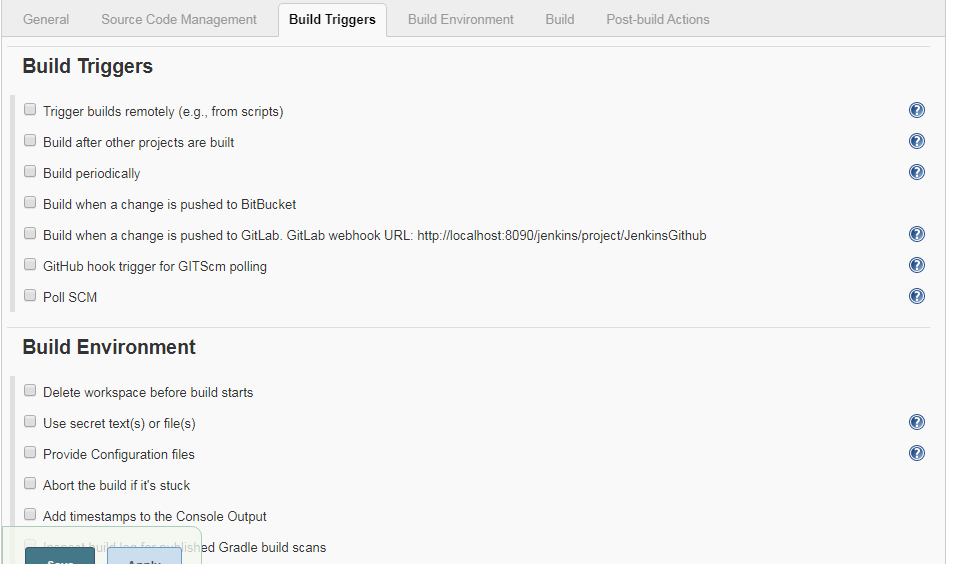


**Step 3) Source Code Management**

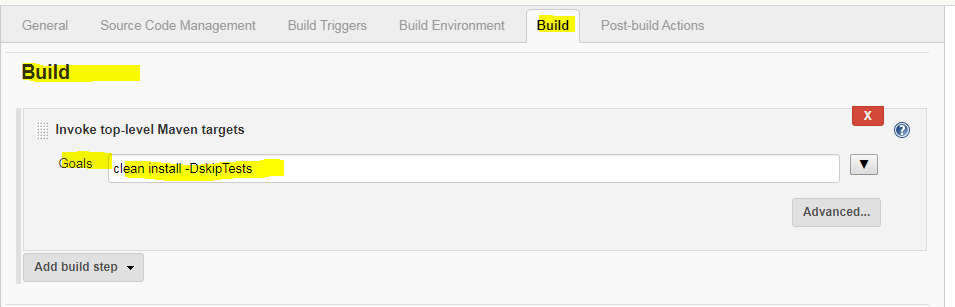


* Select Git as Source Code management
* Provide Git Repository
* Provide Credentials by clicking Add
* Select Branches to Build(Default it is \*/master)

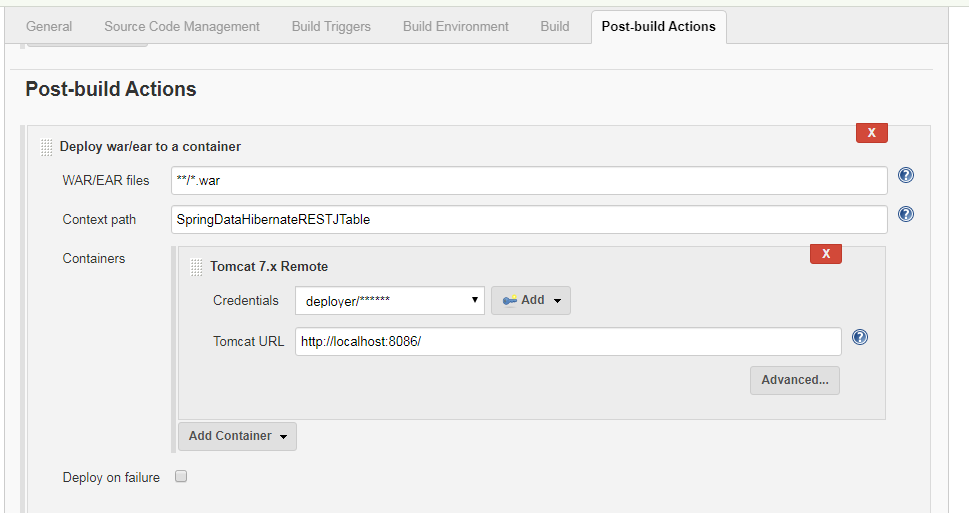
**Step 4) Build Triggers and Build Environment**



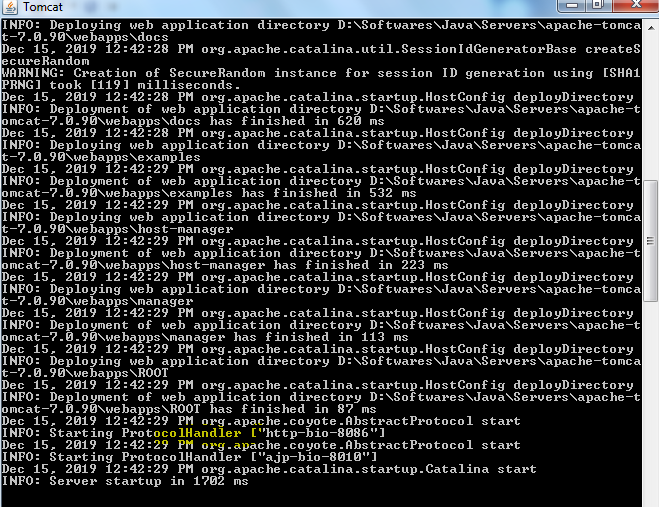
**Step 5) Build**



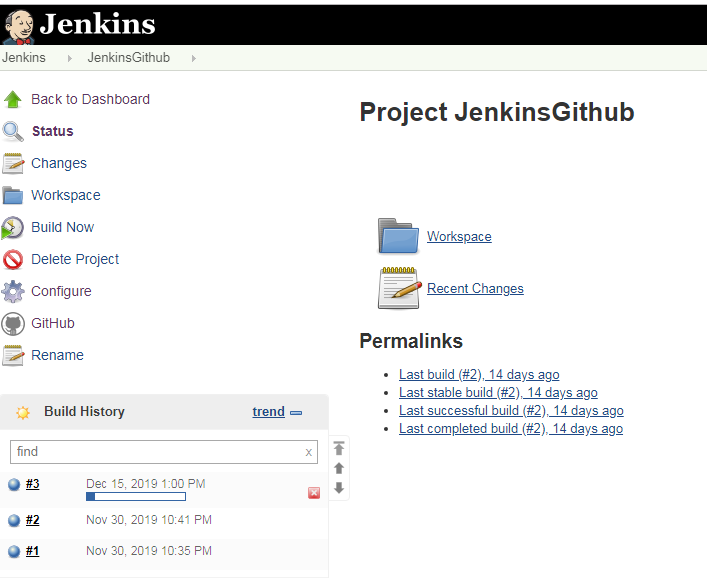
**Step 6)** Post Build Actions: After successful Build, deploy the war to Tomcat server running on port 8086



**Step 7)** Save and Start the Tomcat on Port no 8086



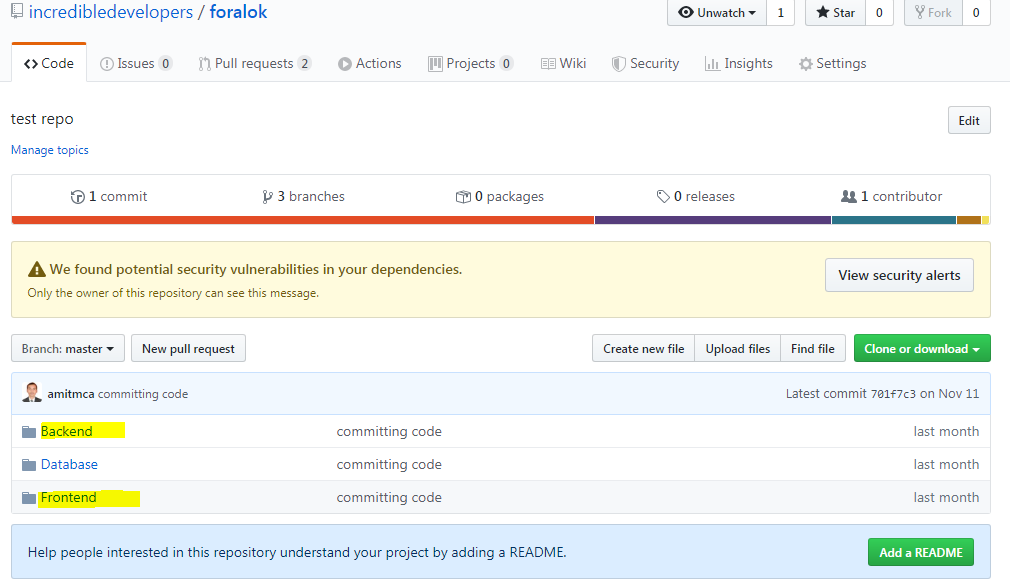
**Step 8) Build Now**



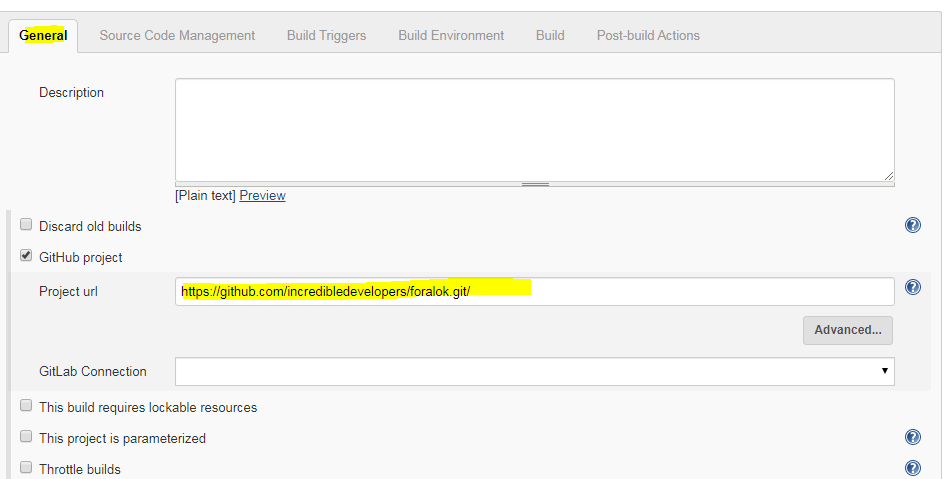
Build #3 will get started. Click on the Build and then Console output

* **Jenkins-Github Integration from Specific Folder:**

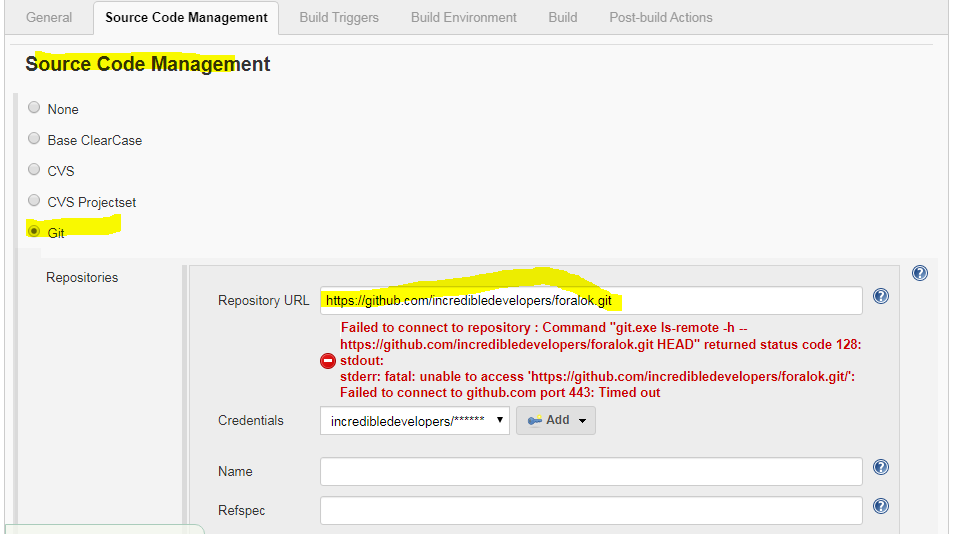
When we want to build a specific folder from Github

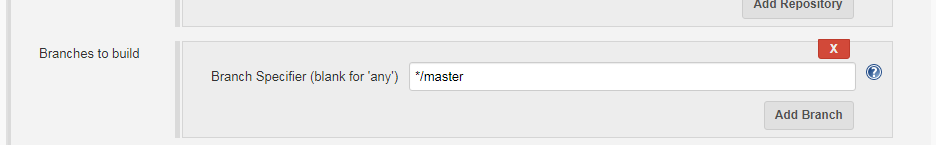


**Step 1) General Tab:** Specify Project url by selecting Github Project

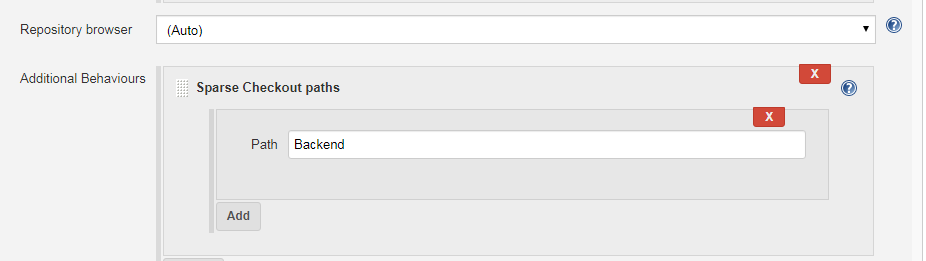


**Step 2) Source Code Management:**

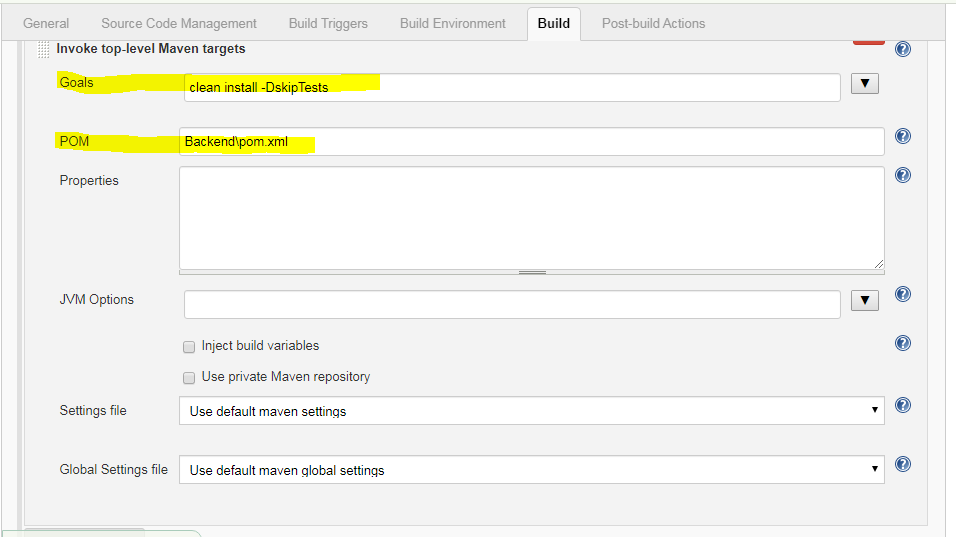




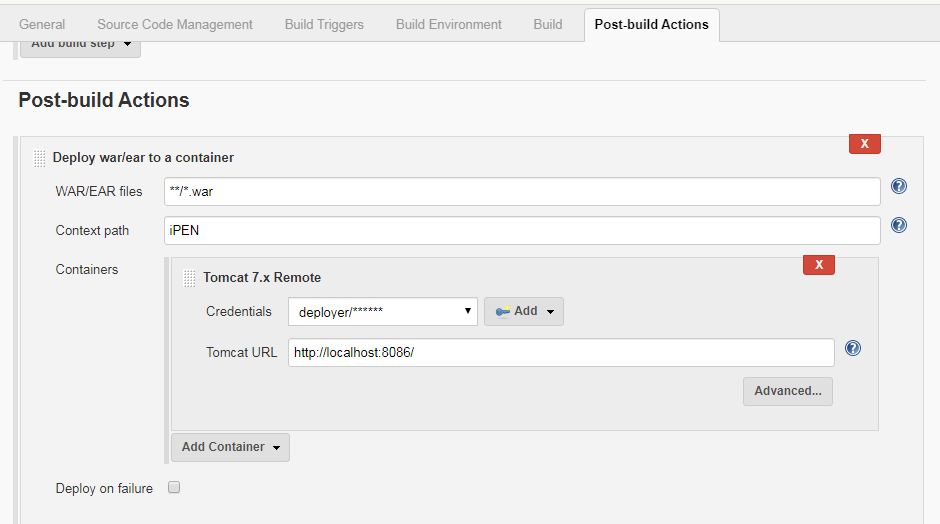
**Step 3) Click on Additional Behaviours and select Sparse Checkout Paths**



**Step 4) Build**

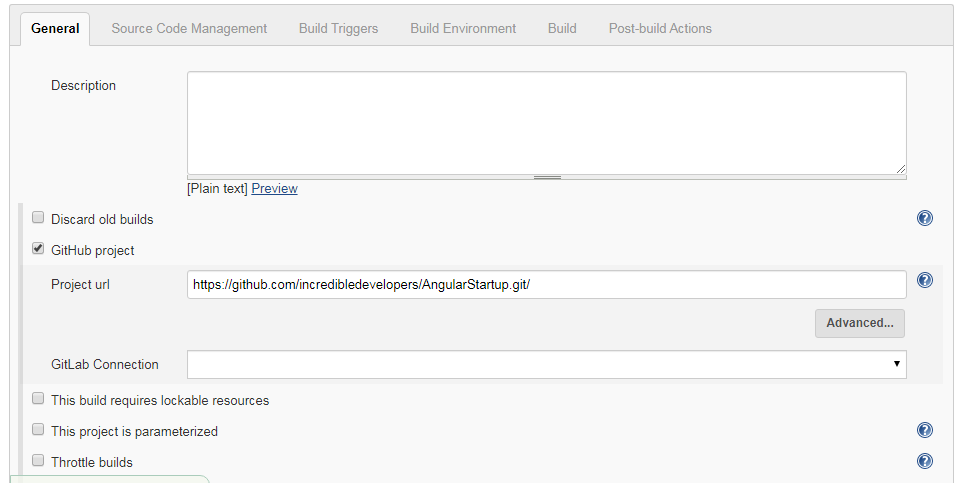


**Step 5) Post-build Actions**

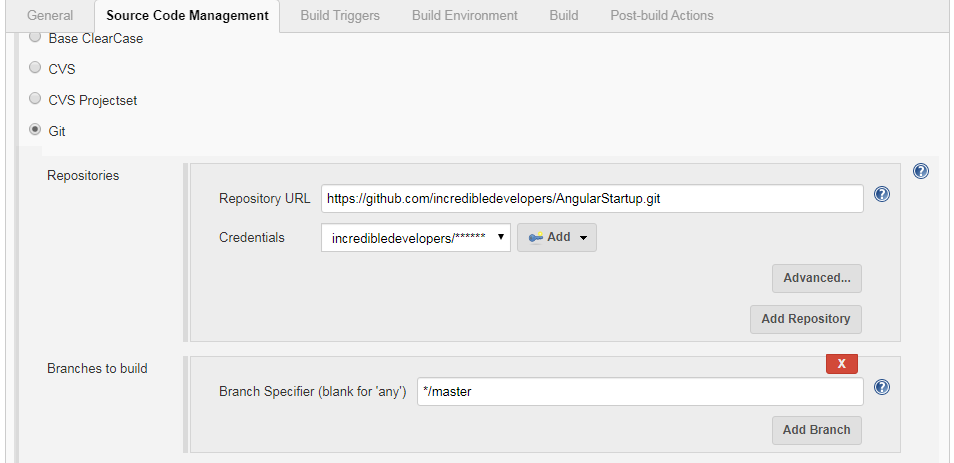


* **Jenkins Github Angular Integration :**

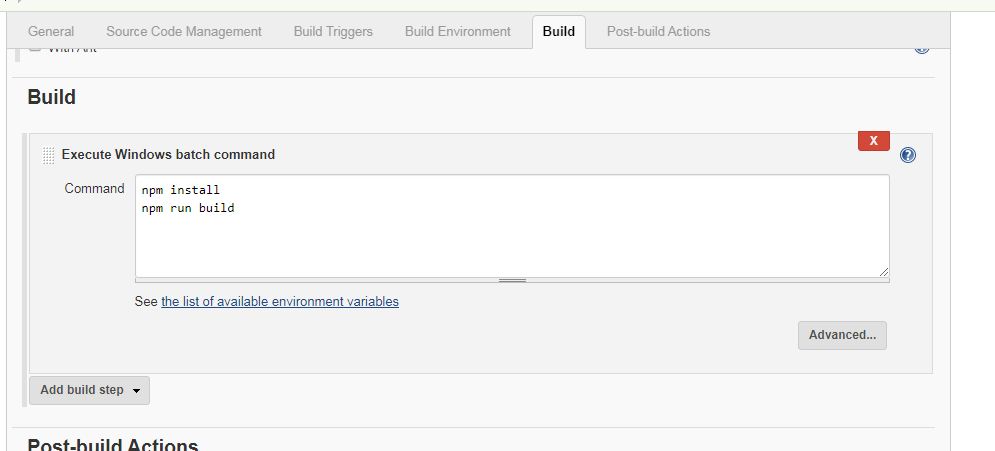
**Step 1) General**



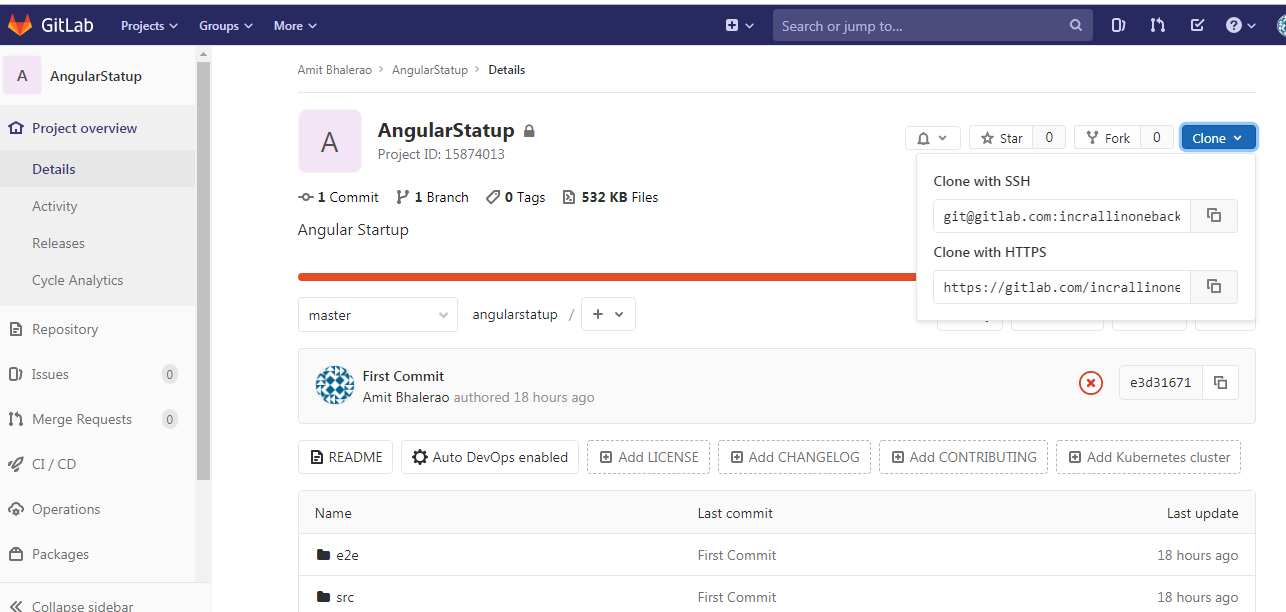
**Step 2) Source Code Management**



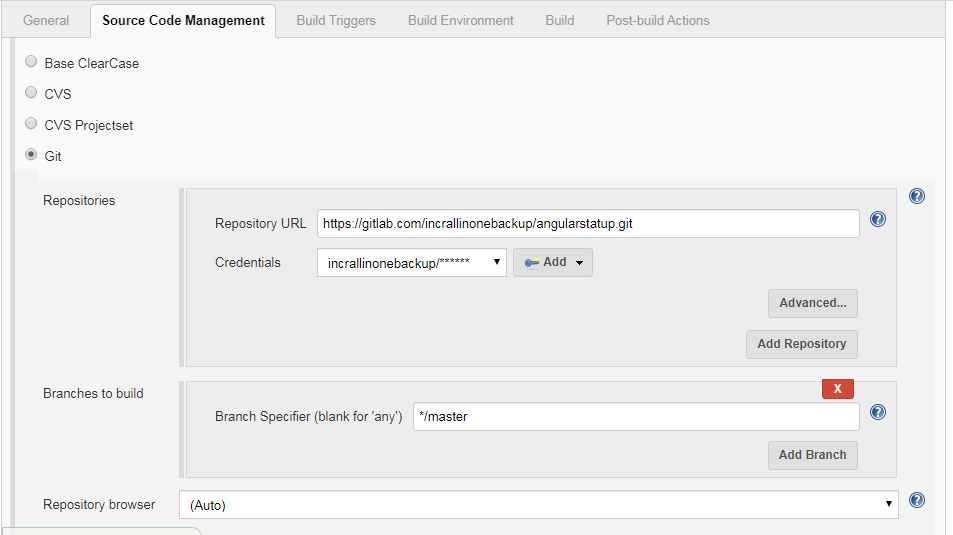
**Step 3) Build**



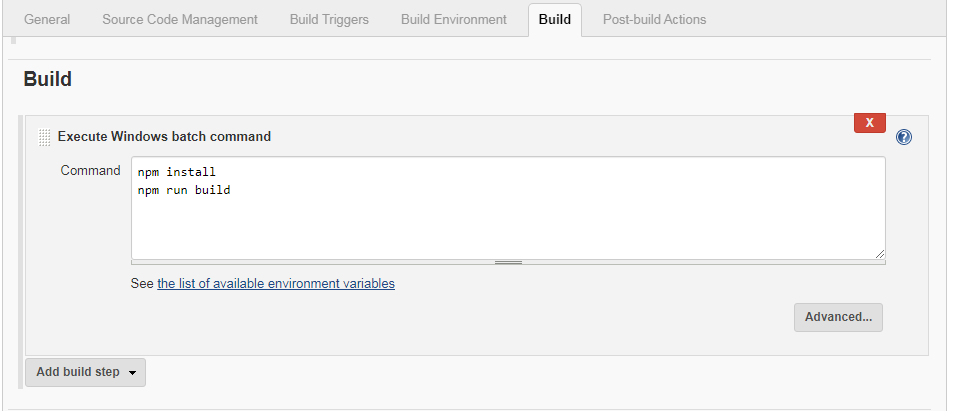
* **Setting Up Jenkins Github Webhook**
* **Jenkins GitLab Angular Integration :**



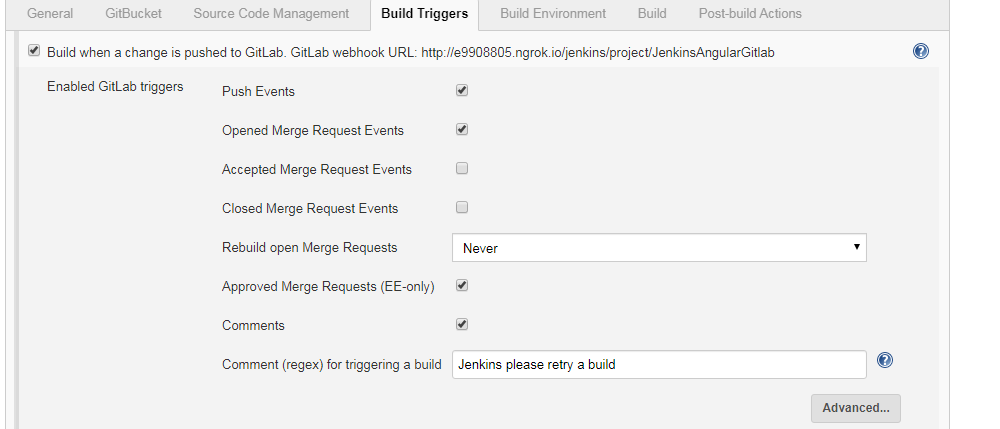
**Step 1) Source Code Management**



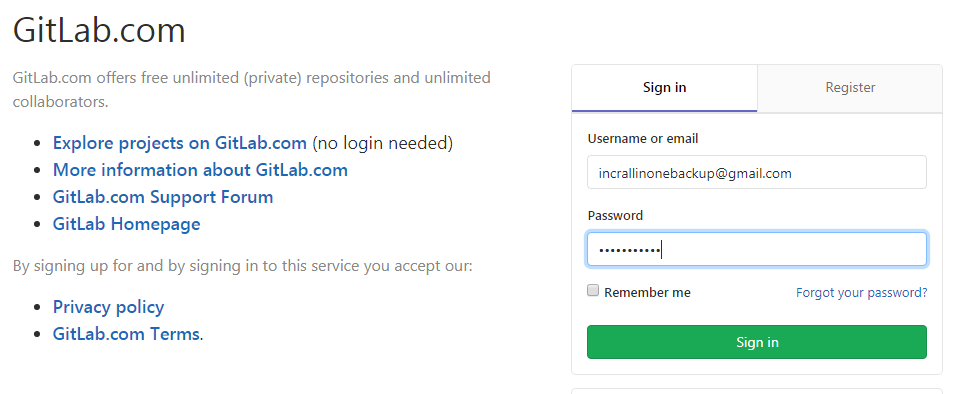
**Step 2) Build**



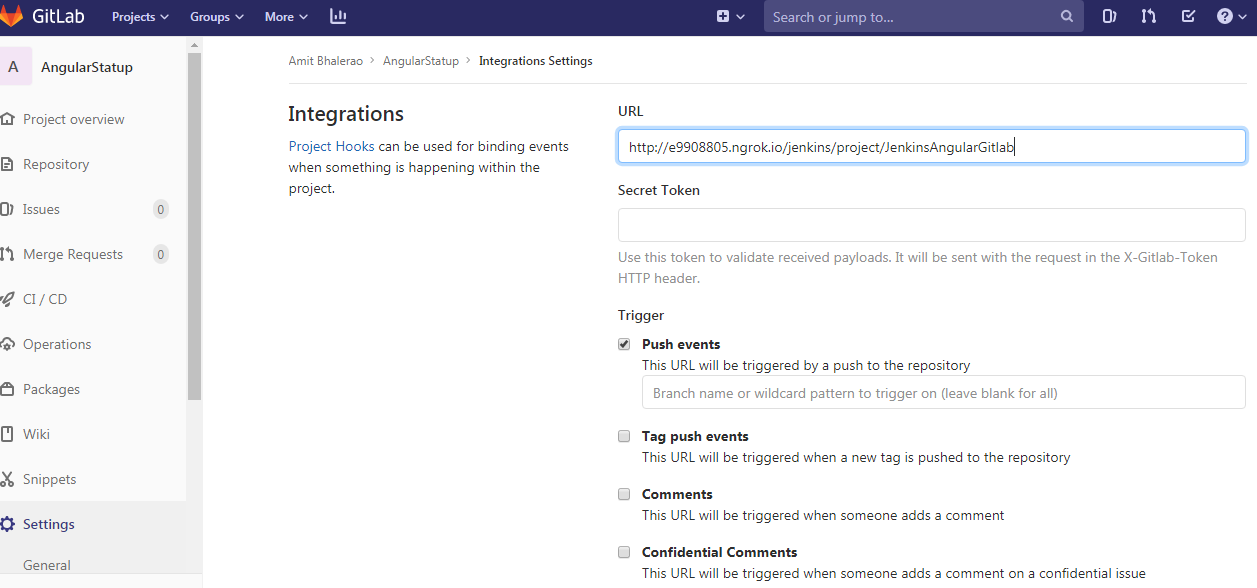
* **Jenkins Gitlab Webhook :**
  1. Create Build Configuration using above steps
  2. Open your Jenkins job configuration and configure it like below in Build Triggers section

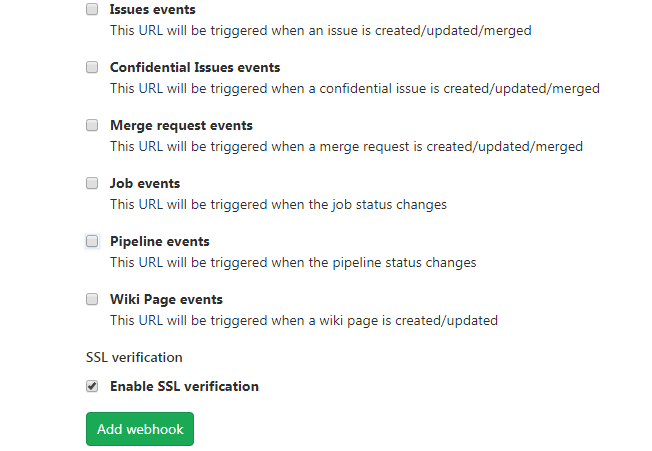


* 1. Login to GitLab

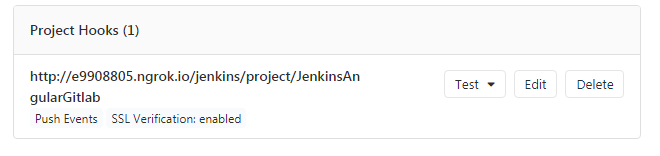


* 1. Select your project and navigate to Settings > Integrations





Click on Add webhook.It will create a webhook inside Project Hooks section as below



Click on Test > Push Events



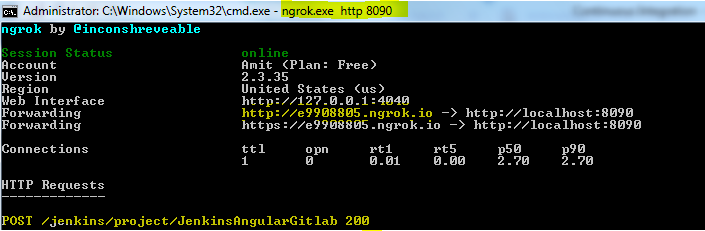
This indicates that your Jenkins Gitlab web hook is communicating properly.

Here the URL: <http://e9908805.ngrok.io> is an id generated using **ngrok** (ngrok.com)

Ngrok is a cross-platform application that enables developers to expose a local development server to the Internet with minimal effort

**Steps to expose your local development server (Teamcity server running on port 80)**

* Download and extract ngRok in folder ngrok-stable-windows-amd64
* Open the Command Prompt
* Use command : ngrok.exe http 8089



Now your localhost running on Port 8089 is exposed to internet and is accessible from <http://e9908805.ngrok.io>

* 1. Commit and Push some code in Gitlab Repo and see whether Jenkins build triggered or not

