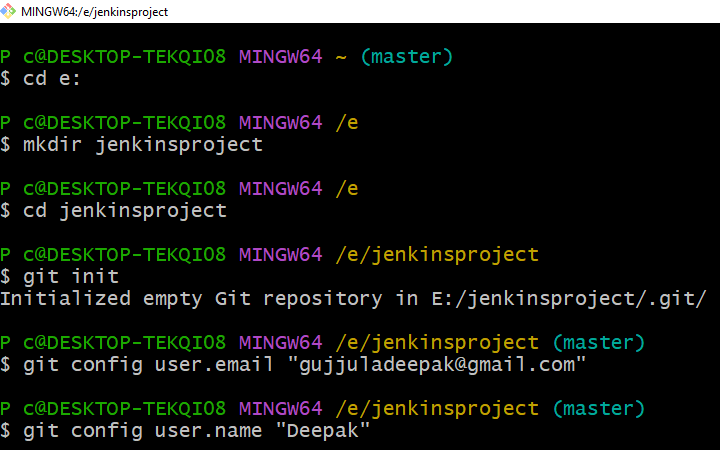
**Lab 5:**

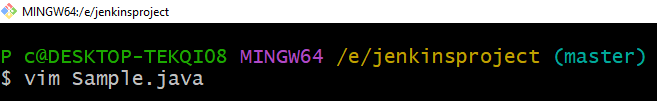
**Demonstrate continuous integration and development using Jenkins.**

Open Git Bash

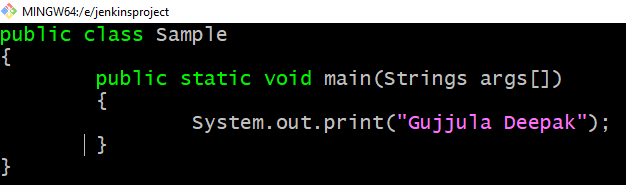


Write simple java program

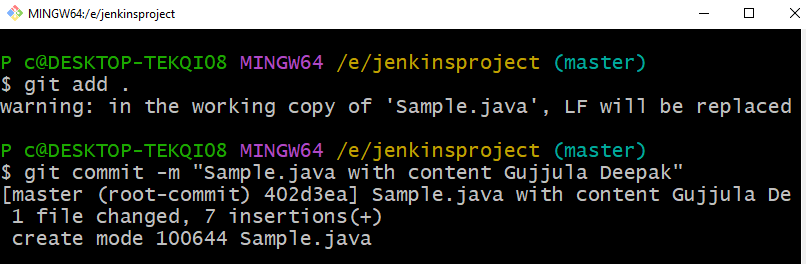
Use the command ***vim Sample.java***

  
Text editor will be opened

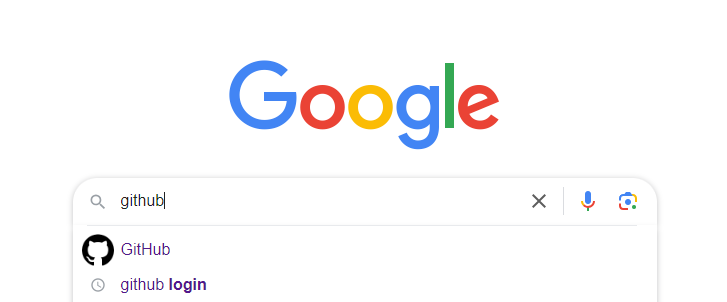
Press ***i*** to write the code

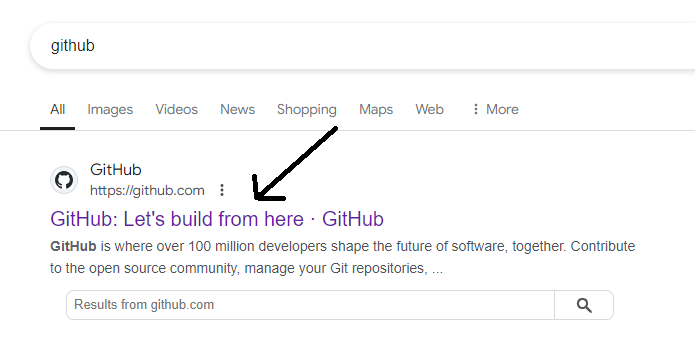
****

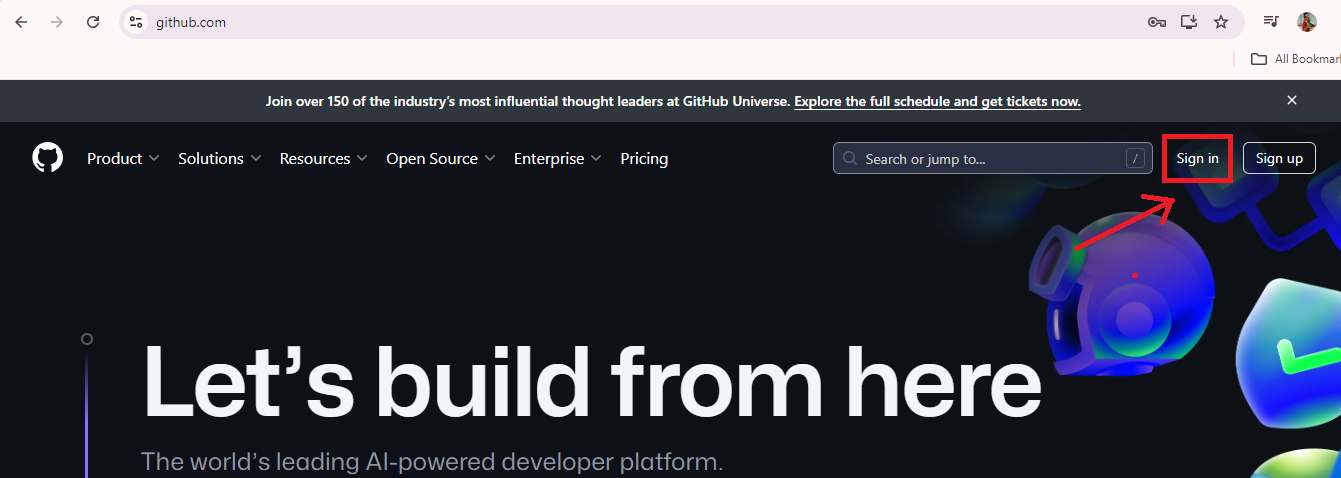
After writing the code press ***esc*** and type ***:wq*** and press enter

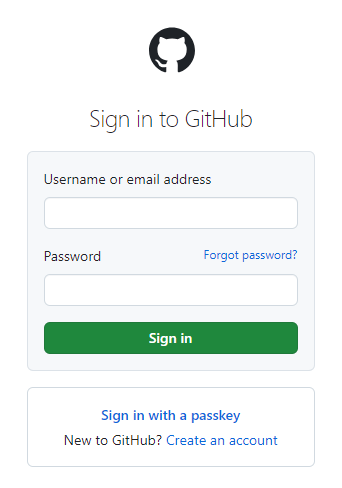


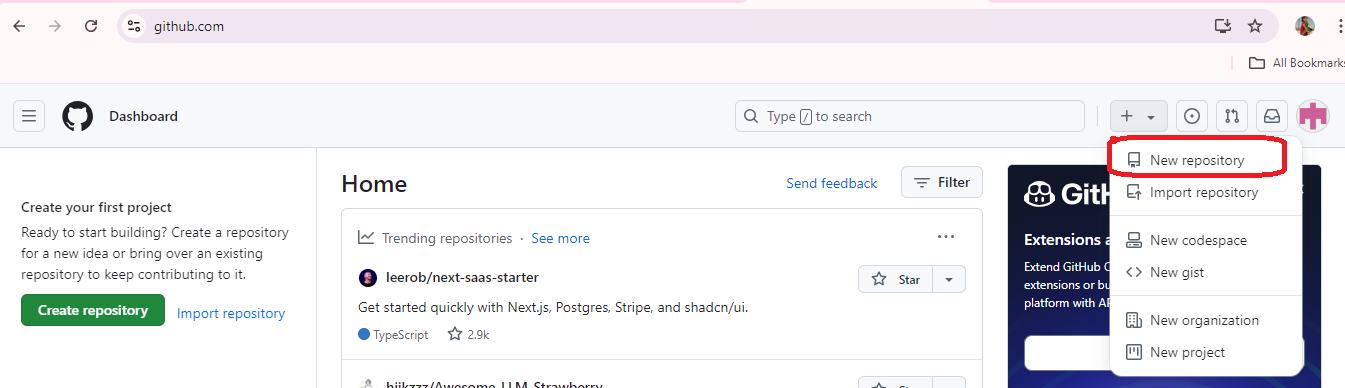
**Open any browser and Open github website**

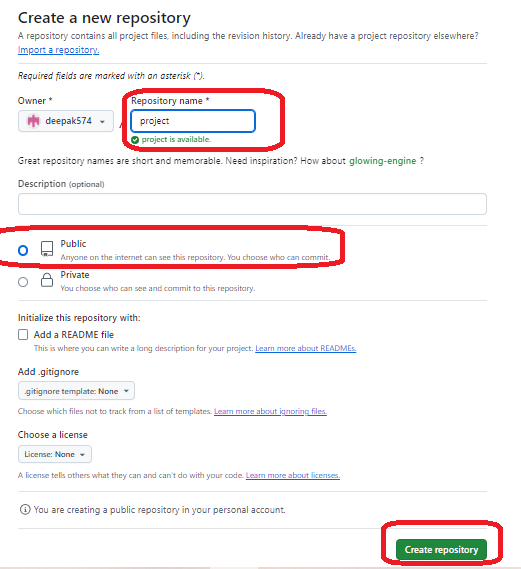
****

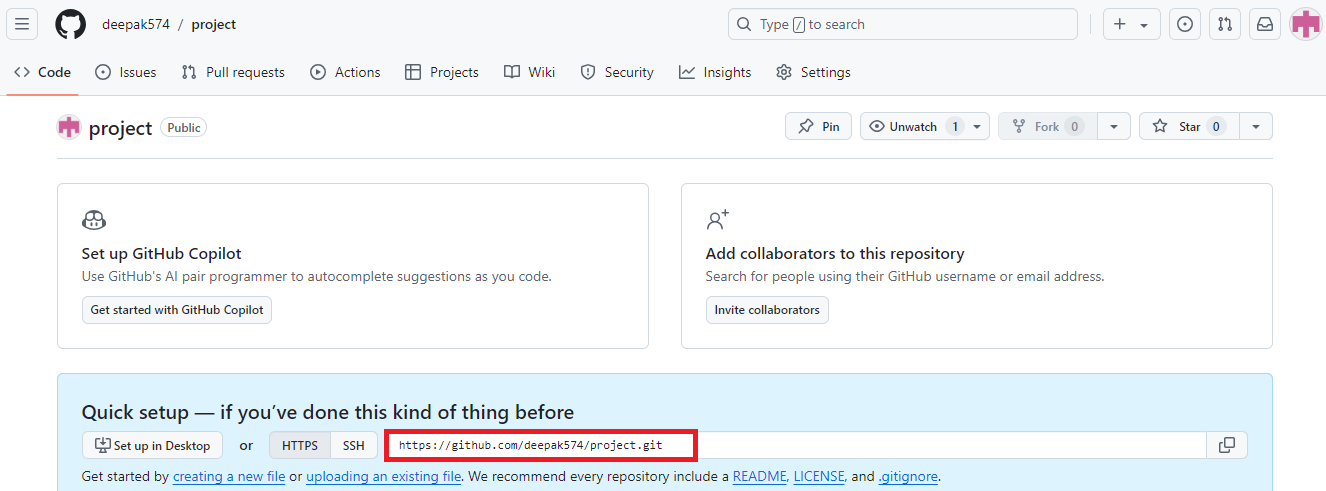
****

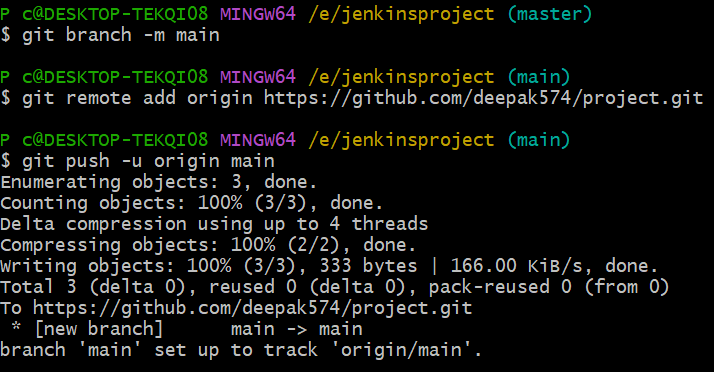
****

****

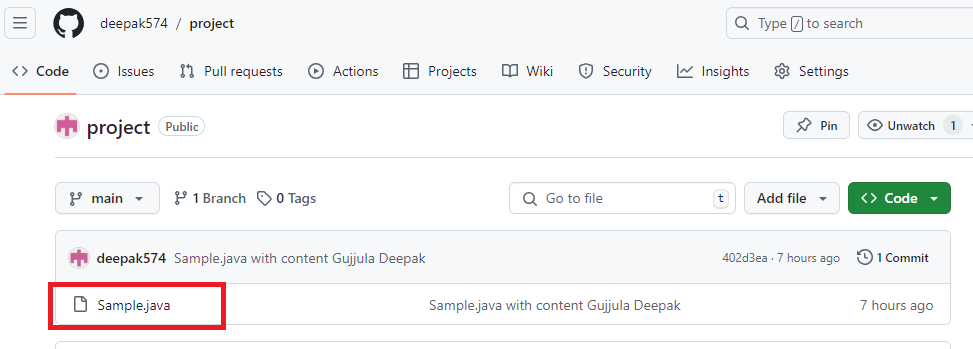
****

****

****

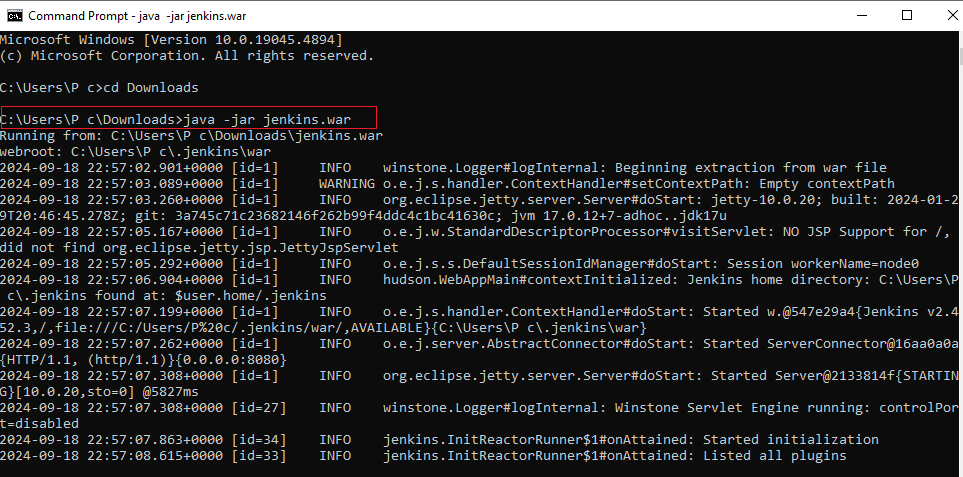
****

**Refresh the github website**

****

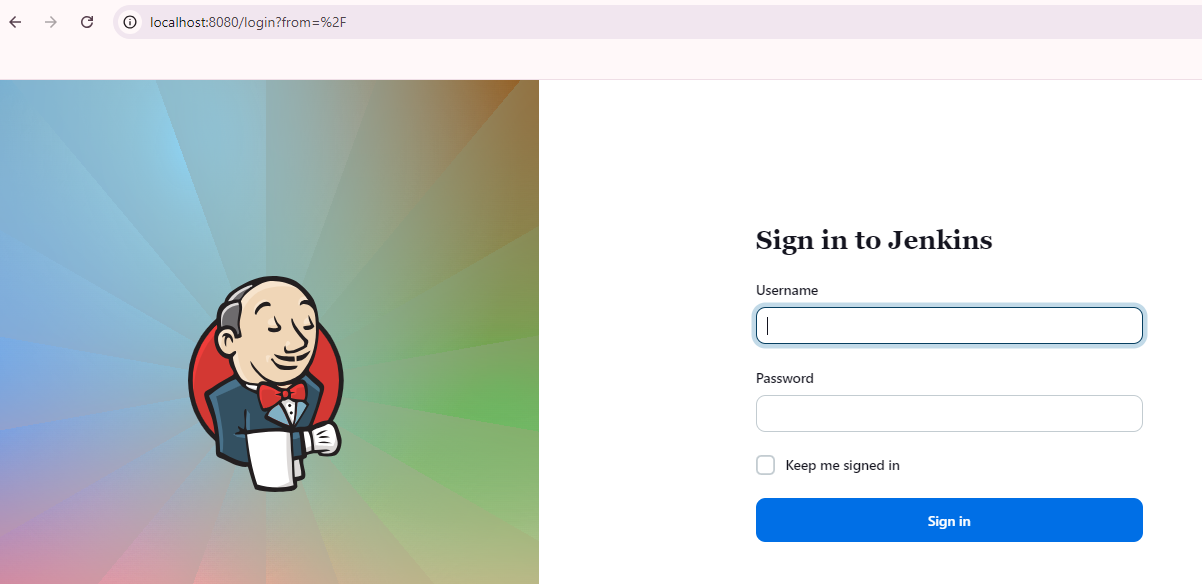
**Now we have to open jenkins**

**Open command prompt, go to the path where jenkins installed**

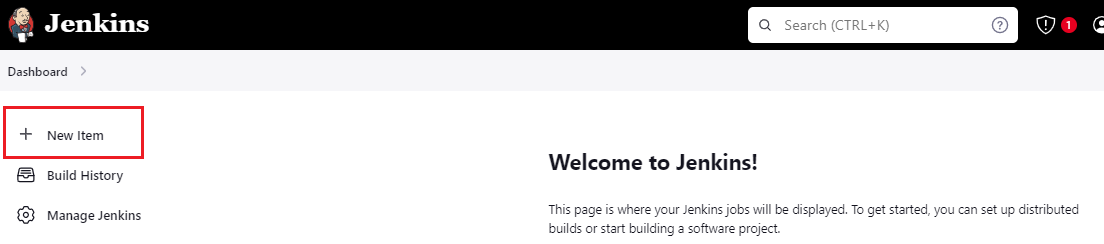
****

**Minimize this cmd prompt and open any browser**

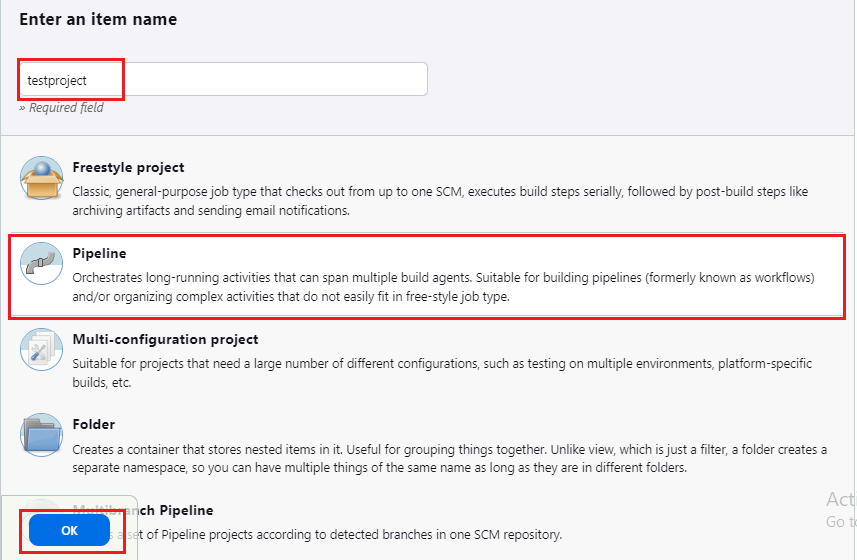
**In the search bar type localhost:8080**

****

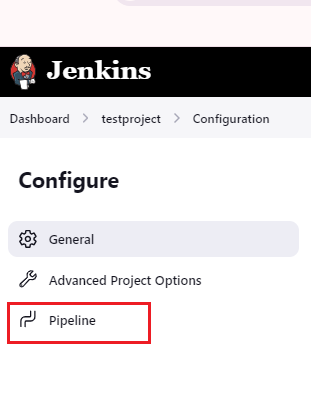
**Give the username , password and click sign in button**

****

**Click on New Item**

****

**Give item name as *testproject,* select *Pipeline* and click on *ok* button**

****

**Click on Pipeline and in the script box type the following code**

pipeline {

agent any

stages {

stage('Checkout Code') {

steps {

git branch: 'main', url: '<https://github.com/deepak574/project.git>'

}

}

stage('Build') {

steps {

bat 'javac Sample.java'

}

}

stage('Test') {

steps {

bat 'java Sample'

}

}

stage('Deploy') {

steps {

echo 'Deploying application...'

// Add your deployment commands here (like copying files to a server)

}

}

post

{

success

{

mail to:’[gujjuladeepak@gmail.com](mailto:gujjuladeepak@gmail.com)’,

subject: “Build success”,

Body: ‘Your build was successfull’

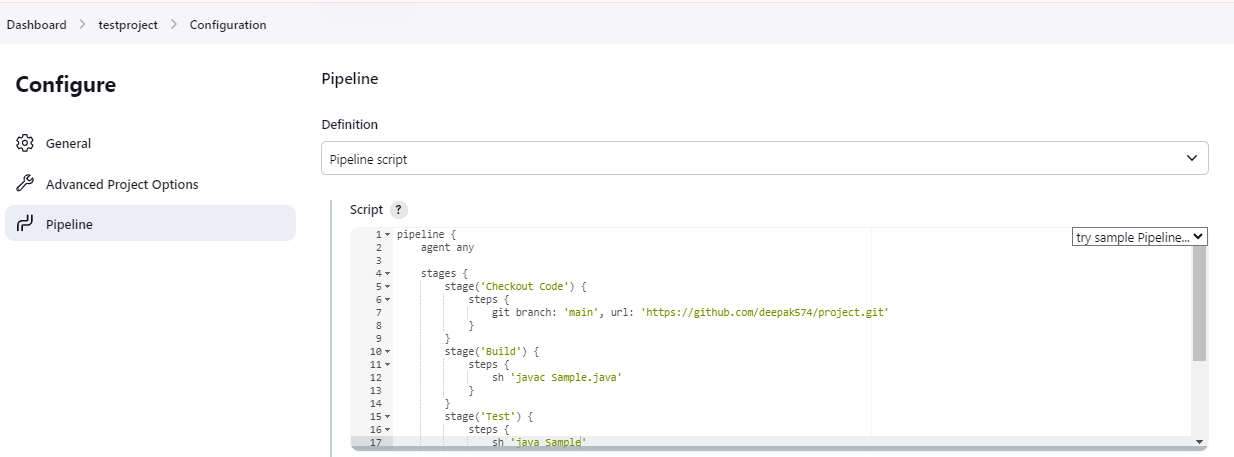
}

}

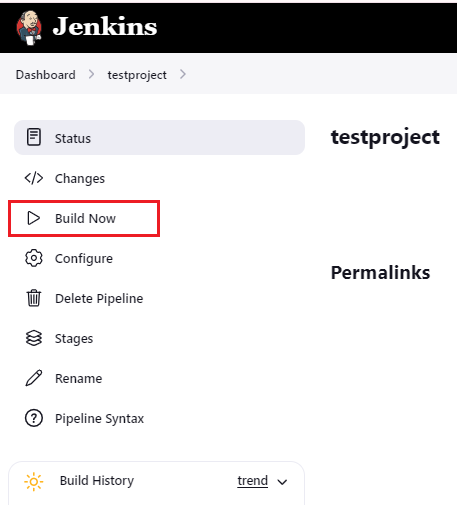
}

}

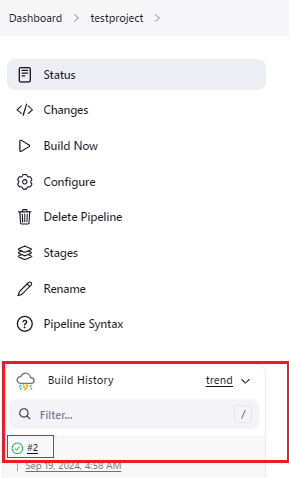
**Note : here url should be ours github url.**

****

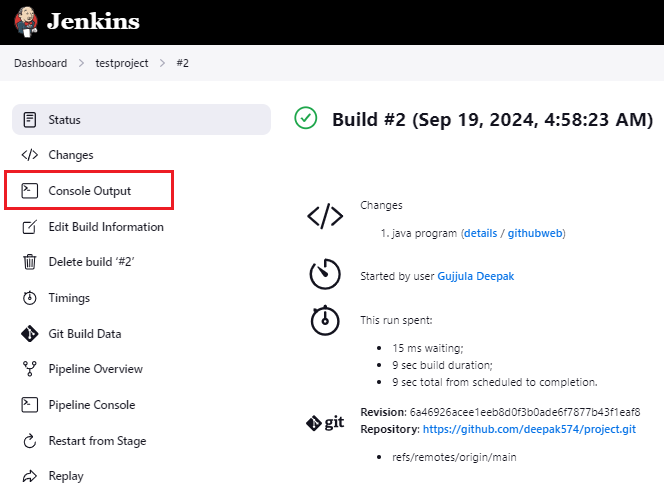
**After typing the code ,click on save button**

****

**Click on Build now button**

****

**After clicking on Build Now , the script will be executed and in the Build History we can see it.  
Now after successful execution click on #2**

****

**Now click on Console Output**

**Output:**

**Started by user****[Gujjula Deepak](http://localhost:8080/user/gujjuladeepak)**

**[Pipeline] Start of Pipeline**

**[Pipeline] node**

**Running on****[Jenkins](http://localhost:8080/computer/(built-in)/)**

**in C:\Users\P c\.jenkins\workspace\testproject**

**[Pipeline] {**

**[Pipeline] stage**

**[Pipeline] { (Checkout Code)**

**[Pipeline] git**

**The recommended git tool is: NONE**

**No credentials specified**

**> git.exe rev-parse --resolve-git-dir C:\Users\P c\.jenkins\workspace\testproject\.git # timeout=10**

**Fetching changes from the remote Git repository**

**> git.exe config remote.origin.url**[**https://github.com/deepak574/project.git**](https://github.com/deepak574/project.git) **# timeout=10**

**Fetching upstream changes from** [**https://github.com/deepak574/project.git**](https://github.com/deepak574/project.git)

**> git.exe --version # timeout=10**

**> git --version # 'git version 2.45.2.windows.1'**

**> git.exe fetch --tags --force --progress --** [**https://github.com/deepak574/project.git**](https://github.com/deepak574/project.git) **+refs/heads/\*:refs/remotes/origin/\* # timeout=10**

**> git.exe rev-parse "refs/remotes/origin/main^{commit}" # timeout=10**

**Checking out Revision 6a46926acee1eeb8d0f3b0ade6f7877b43f1eaf8 (refs/remotes/origin/main)**

**> git.exe config core.sparsecheckout # timeout=10**

**> git.exe checkout -f 6a46926acee1eeb8d0f3b0ade6f7877b43f1eaf8 # timeout=10**

**> git.exe branch -a -v --no-abbrev # timeout=10**

**> git.exe branch -D main # timeout=10**

**> git.exe checkout -b main 6a46926acee1eeb8d0f3b0ade6f7877b43f1eaf8 # timeout=10**

**Commit message: "java program"**

**> git.exe rev-list --no-walk 402d3ea25873de9dad6dd7f4705bcc23ec4cab6c # timeout=10**

**[Pipeline] }**

**[Pipeline] // stage**

**[Pipeline] stage**

**[Pipeline] { (Build)**

**[Pipeline] sh**

**+ javac Sample.java**

**[Pipeline] }**

**[Pipeline] // stage**

**[Pipeline] stage**

**[Pipeline] { (Test)**

**[Pipeline] sh**

**+ java Sample**

**Gujjula Deepak**

**[Pipeline] }**

**[Pipeline] // stage**

**[Pipeline] stage**

**[Pipeline] { (Deploy)**

**[Pipeline] echo**

**Deploying application...**

**[Pipeline] }**

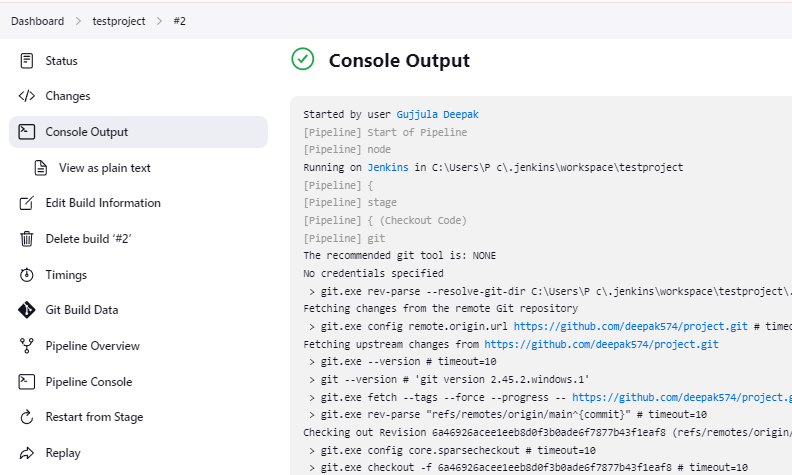
**[Pipeline] // stage**

**[Pipeline] }**

**[Pipeline] // node**

**[Pipeline] End of Pipeline**

**Finished: SUCCESS**

****

To configure Jenkins to poll your GitHub repository at regular intervals (every 5 minutes, for example), you can use the Poll SCM option.

### Jenkins will automatically check GitHub for changes at regular intervals and trigger a build if there are any.

### 

### 

### 

### 

### 

### 

### 

### 

### 

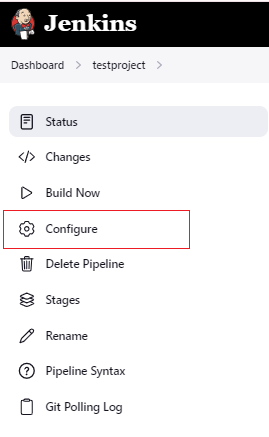
### 

### 

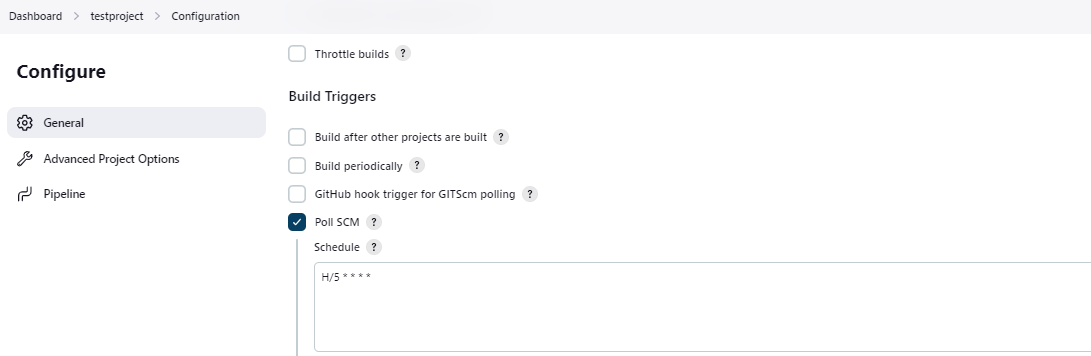
### 

### **Steps to Set Poll SCM in Jenkins:**

1. Open your Jenkins job.

****

1. Click on Configure.
2. Scroll down to the Build Triggers section.
3. Check the box labeled Poll SCM.
4. In the Schedule field, enter H/5 \* \* \* \*

****

6. Click apply & Save.

H/5 means Jenkins will poll every 5 minutes, but the H ensures that the polling happens at different minutes for different jobs, reducing the load on the server.

The other \* symbols mean "every hour", "every day", "every month", and "every day of the week", respectively.

**How to check?**

Modify the java program,add to the staging area,commit and push to github.  
Wait for 5 min and check jenkins the job will be executed automatically