# Title: Setting up Jenkins in Local Mac and Execute XCUITEST

Setting up Jenkins in Mac

Steps to follow

1. Navigate to [https://jenkins.io/download/](https://www.google.com/url?q=https://jenkins.io/download/&sa=D&ust=1575460936809000)

There will be a separate section for Download Jenkins 2.206 for:

Under that tap on the macOS or [**https://jenkins.io/download/weekly/macos**](https://www.google.com/url?q=https://jenkins.io/download/weekly/macos&sa=D&ust=1575460936809000)

Or Download from the below link which lists all the versions

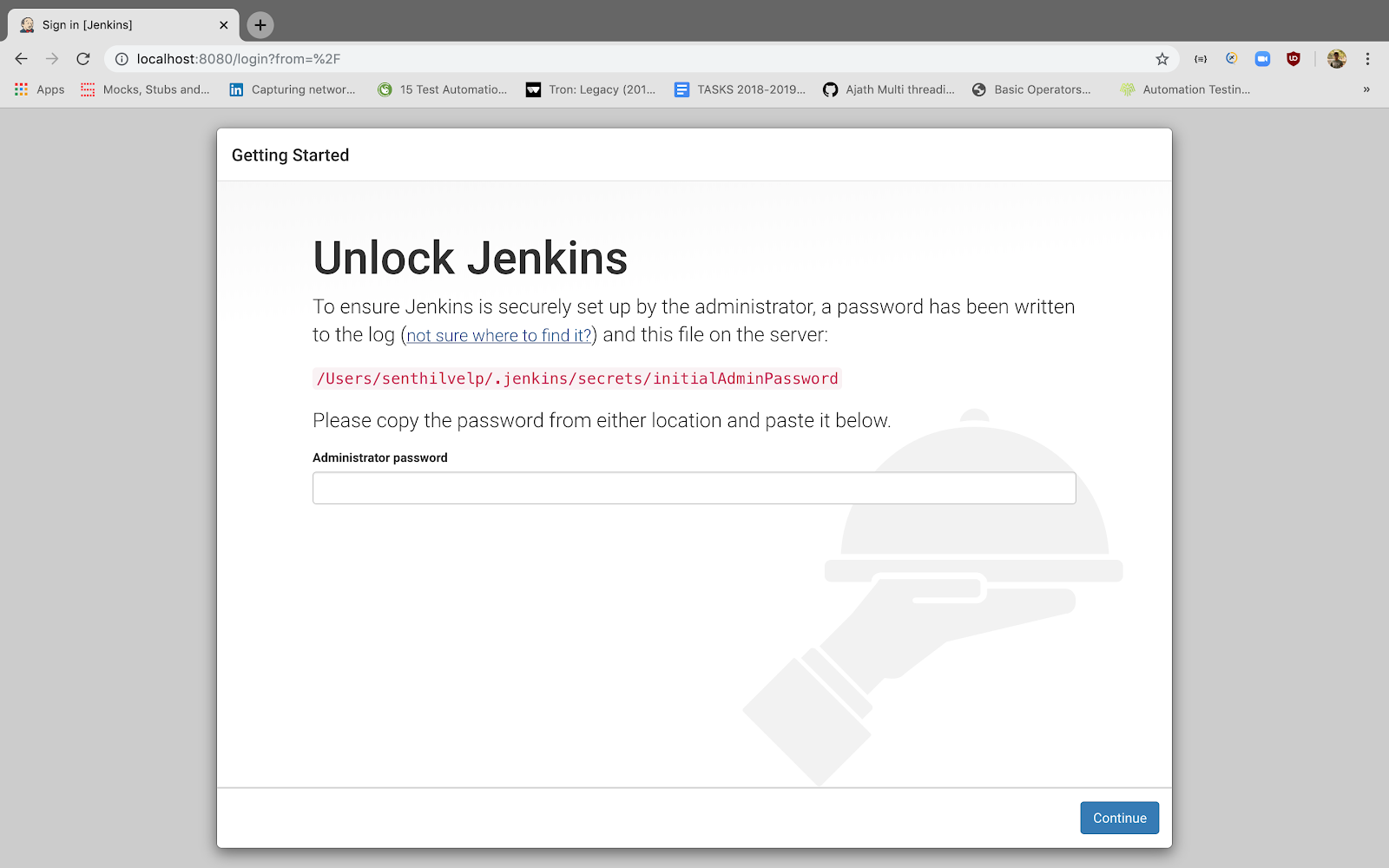
[http://mirrors.jenkins-ci.org/osx/](https://www.google.com/url?q=http://mirrors.jenkins-ci.org/osx/&sa=D&ust=1575460936810000)

1. If you didn't download from the mirrors URL, Jenkins can be now installed using brew

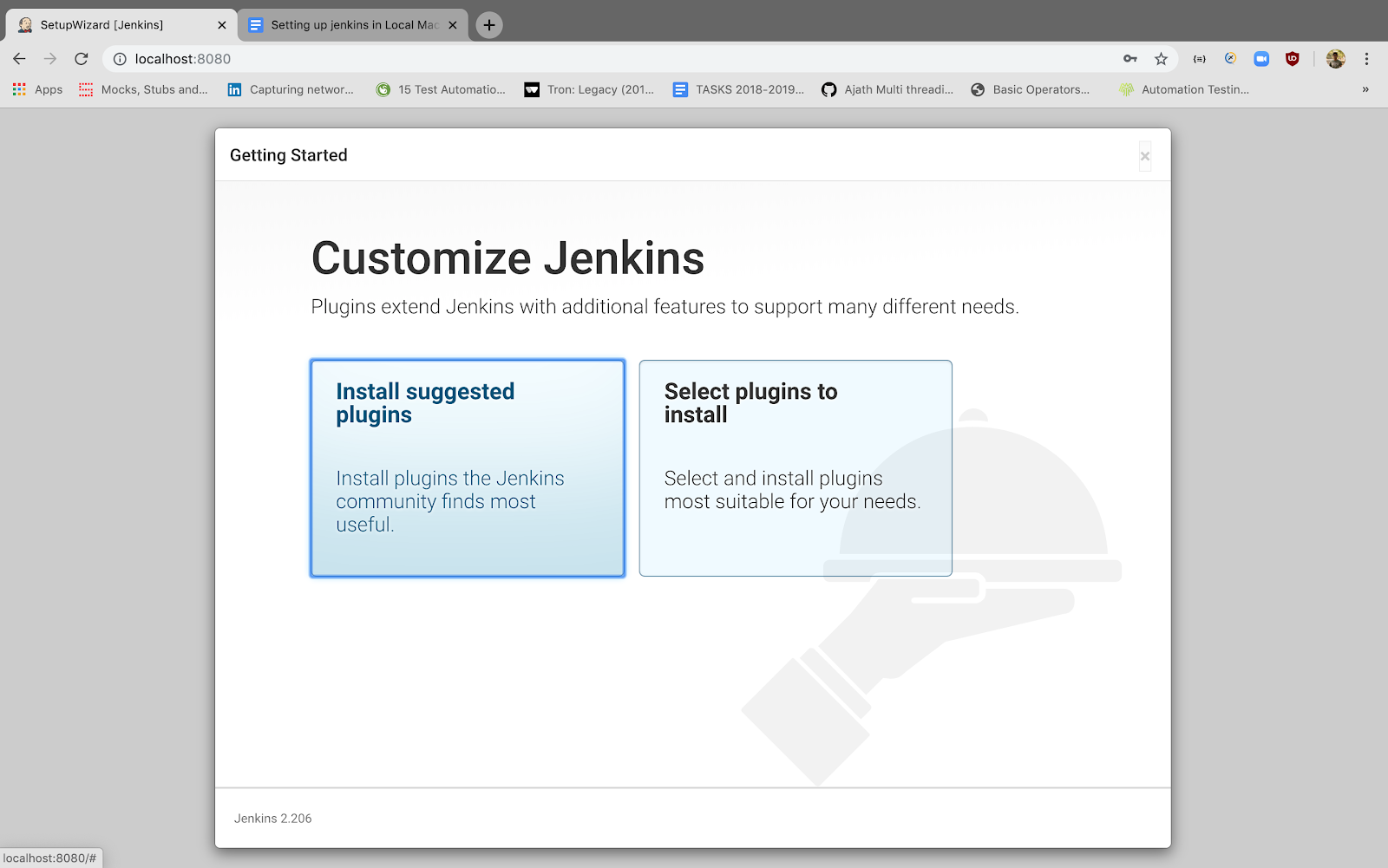
**brew install jenkins**

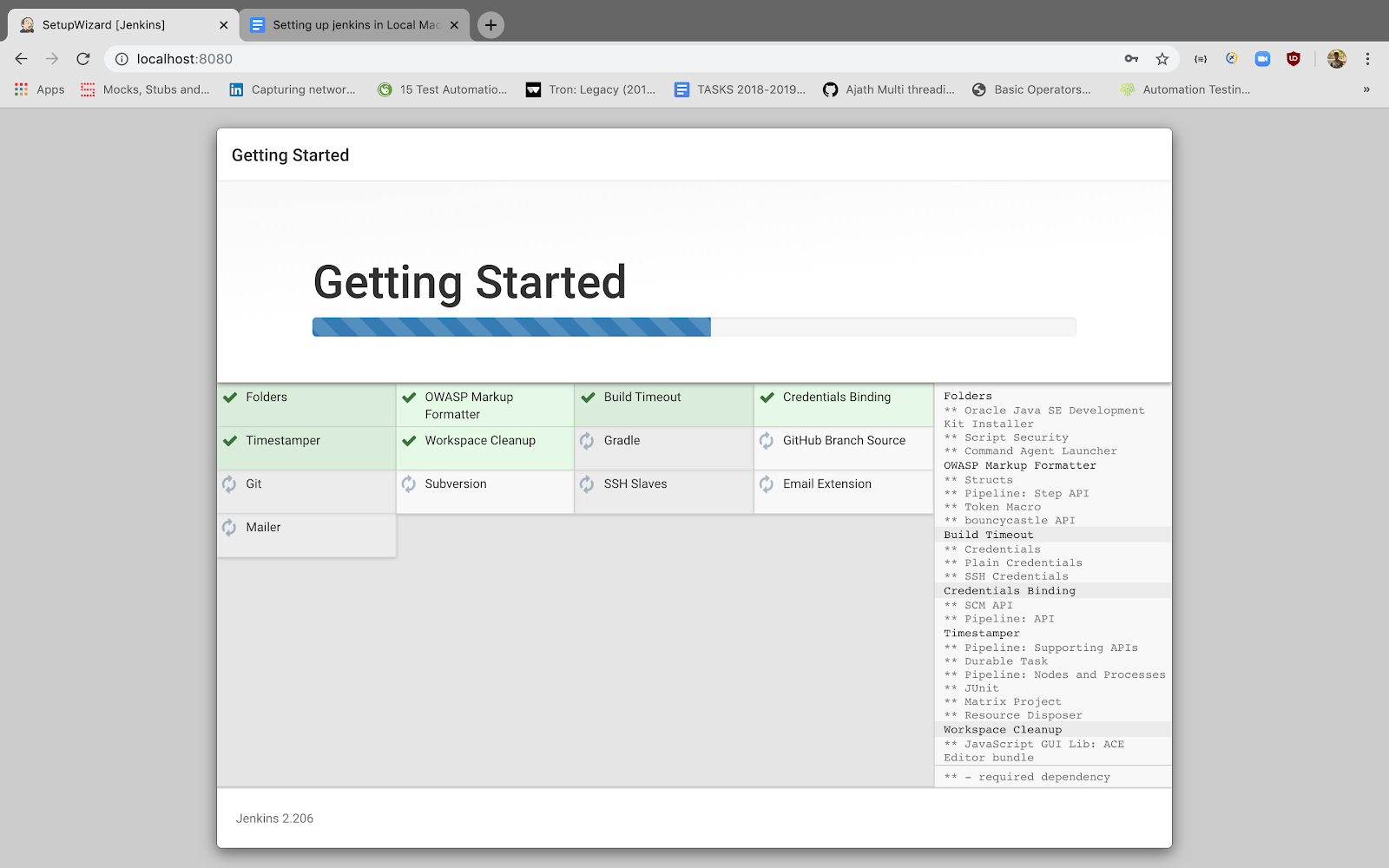
                After installation completed, use  
              **brew services start jenkins**

                Jenkins now will be started in the [http://localhost:8080/](https://www.google.com/url?q=http://localhost:8080/&sa=D&ust=1575460936811000)

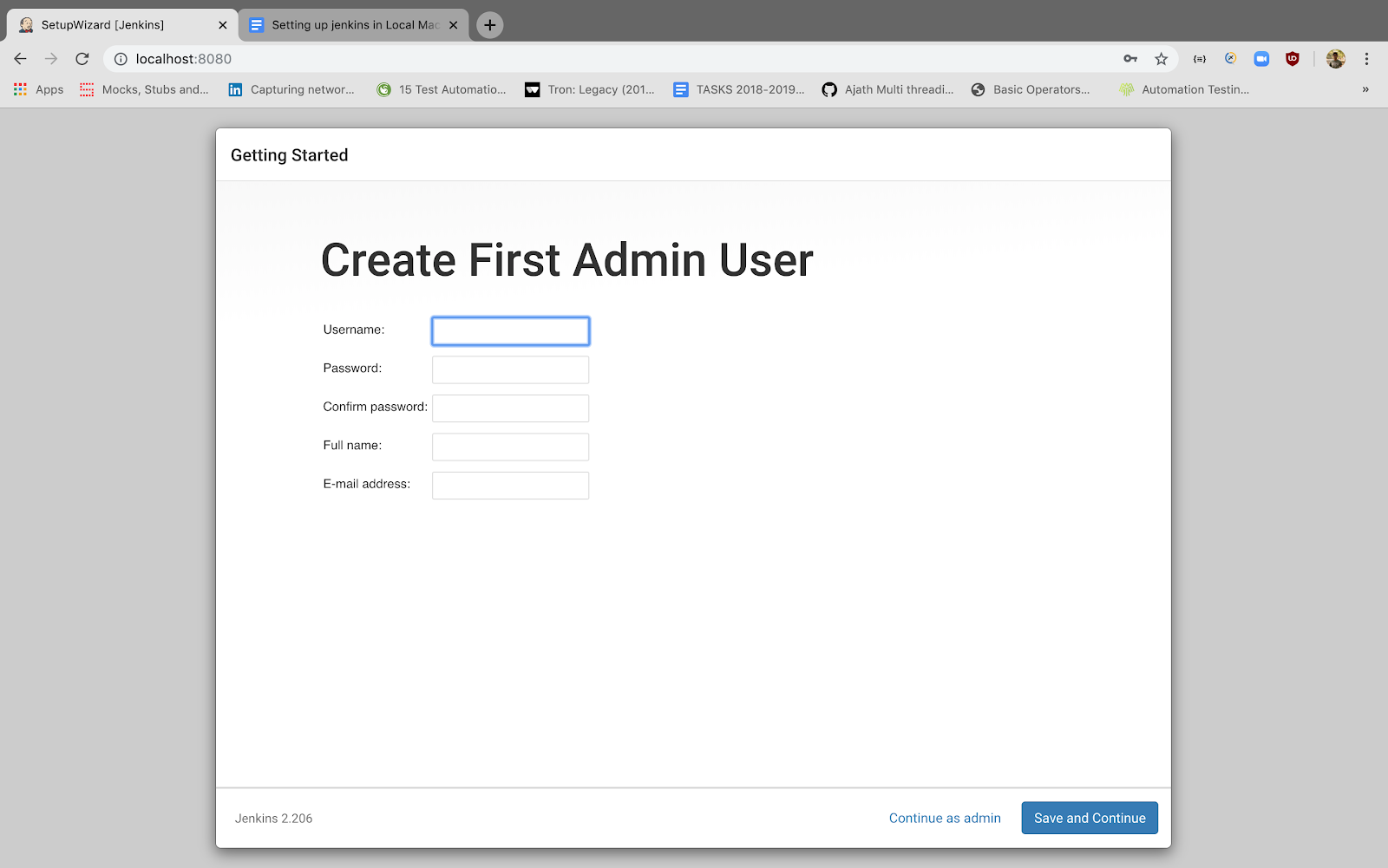


1. The initial password will be saved in the shown highlighted Path
2. Copy and paste it in the terminal (**cat /Users/senthilvelp/.jenkins/secrets/initialAdminPassword**)
3. Copy and paste the password in the text box and tap continue
4. In the next screen either tap the Install suggested plugins or Select plugins to install and wait for the plugin to get downloaded

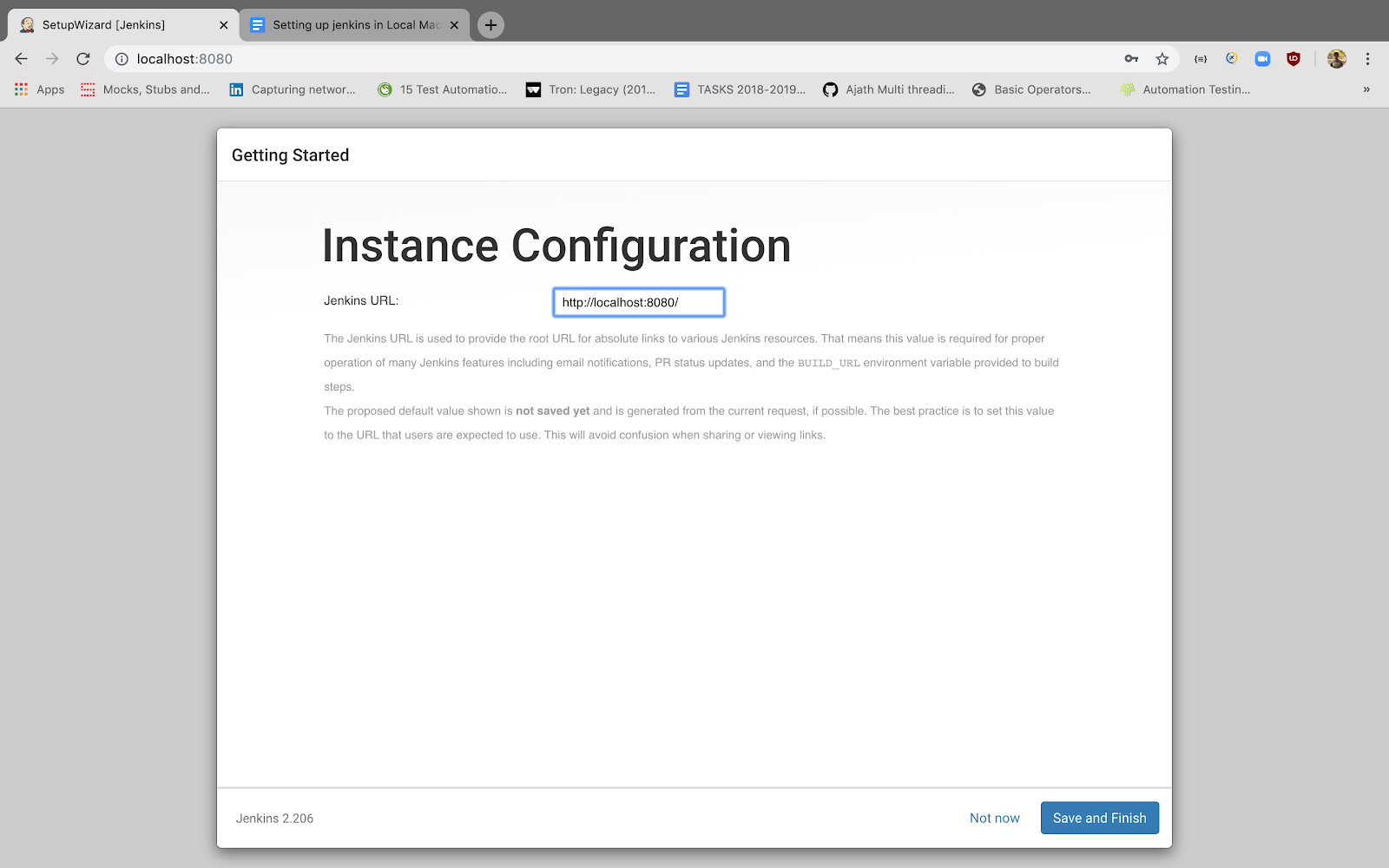




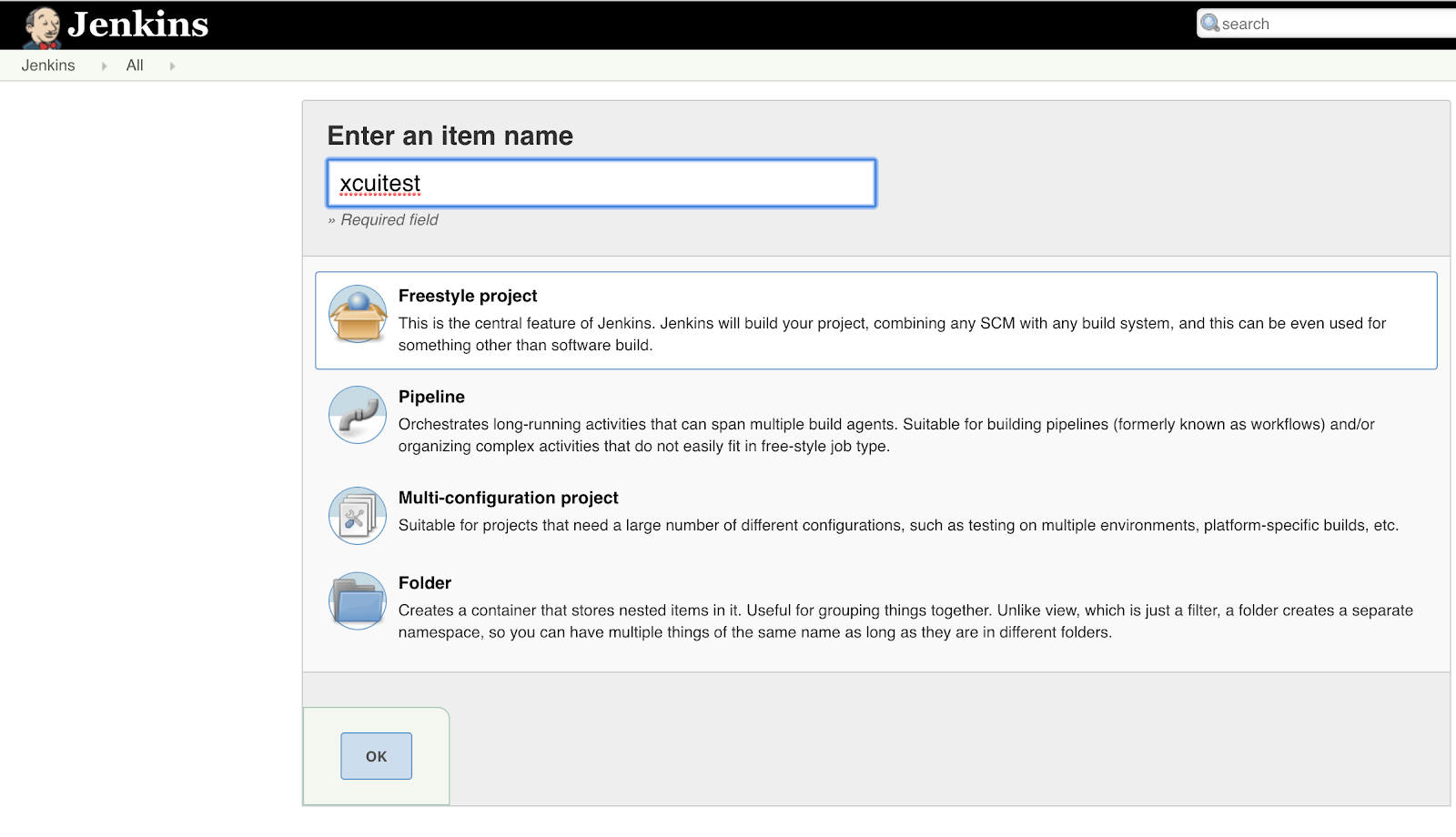
1. After installation completed, Please create a user and tap on save and continue in the below screen



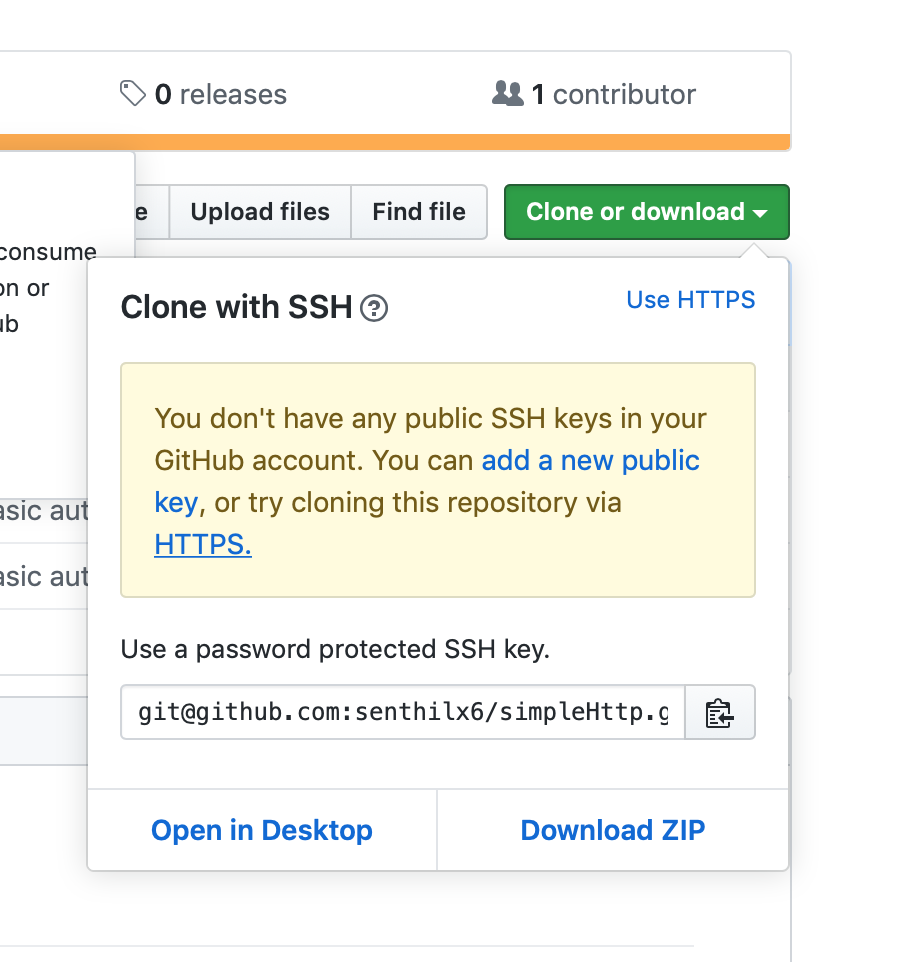
1. Enter the default URL to be used for Jenkins and tap Save and Finish

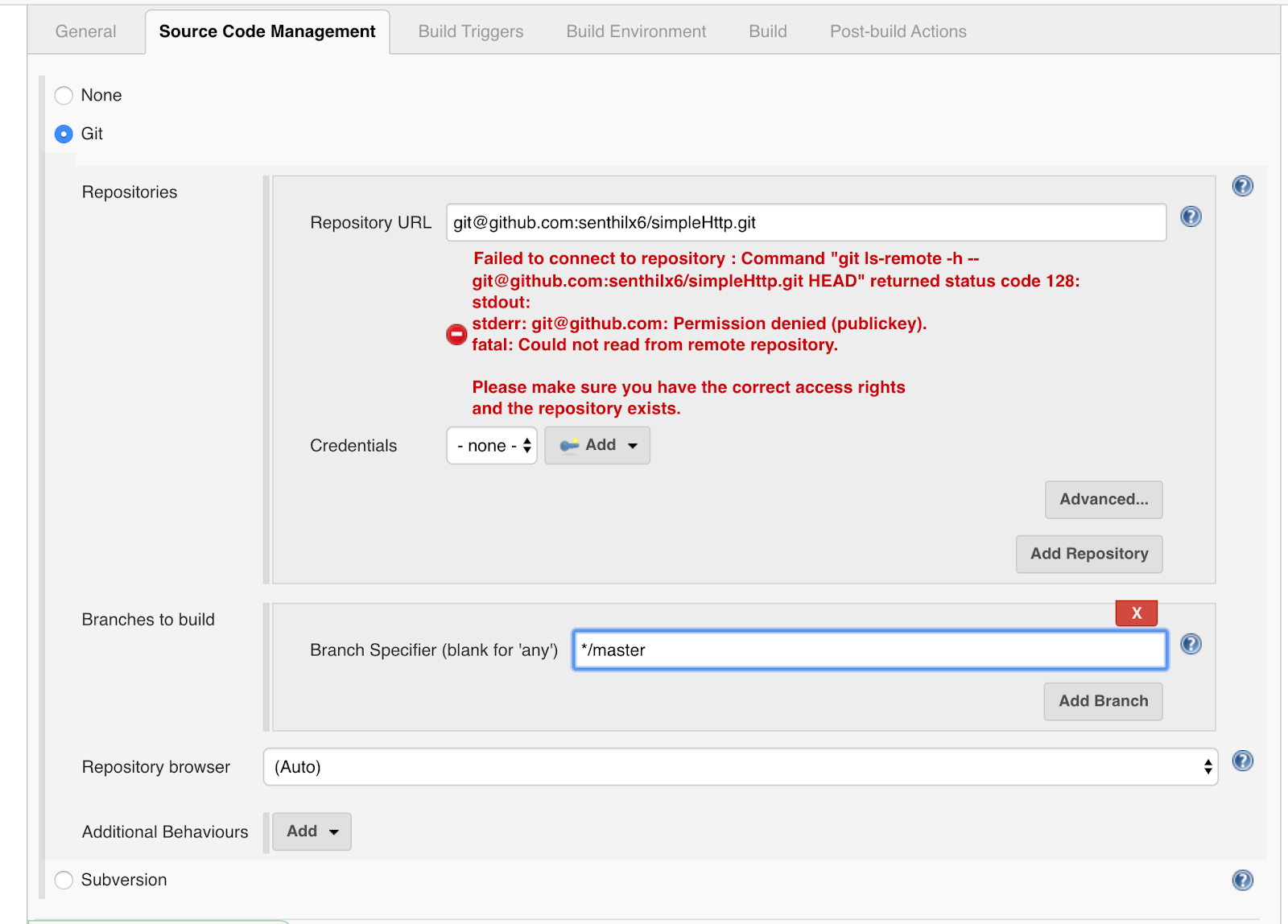


1. Tap on New item in the home screen and select freeline project in below screen and Tap OK

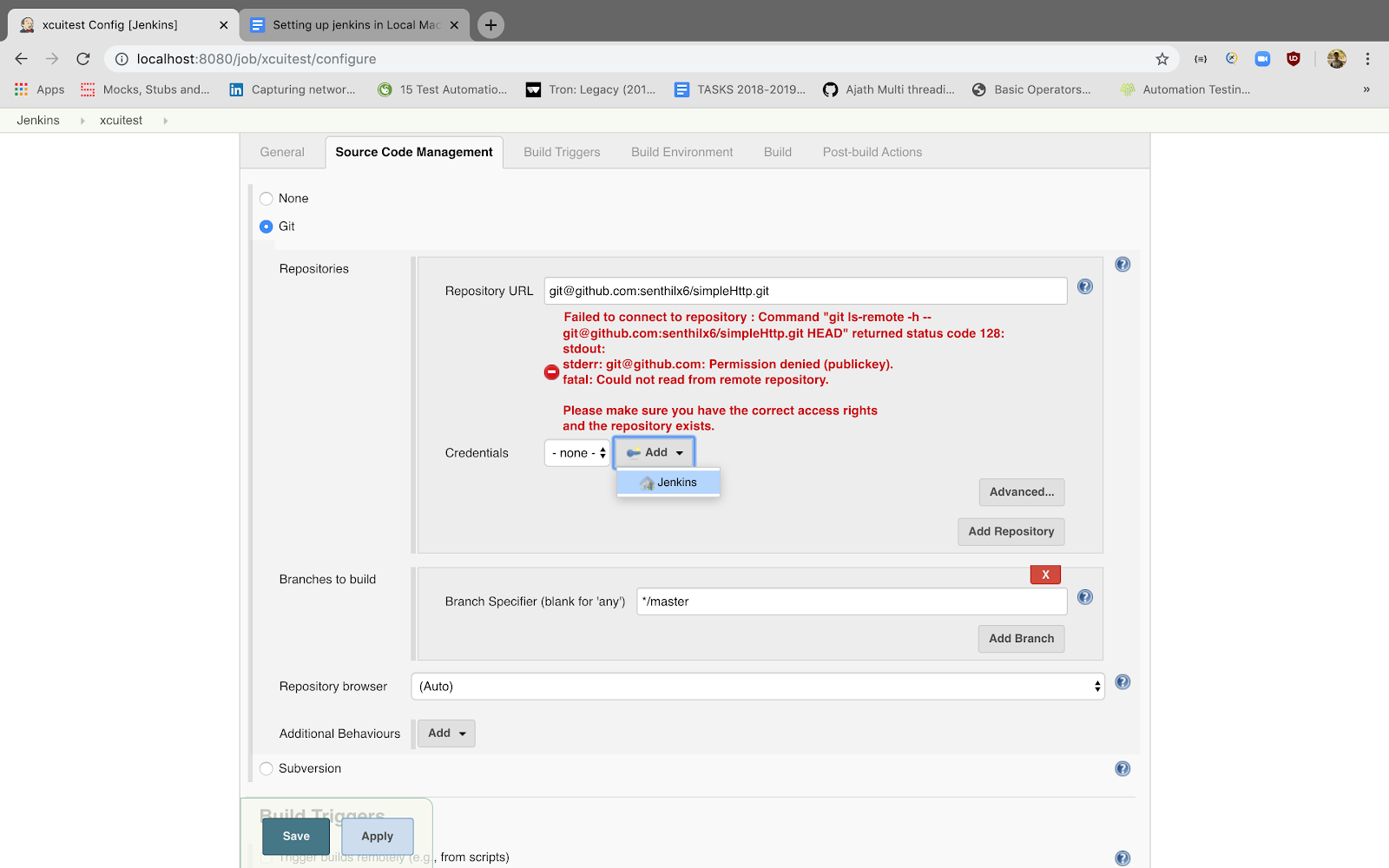


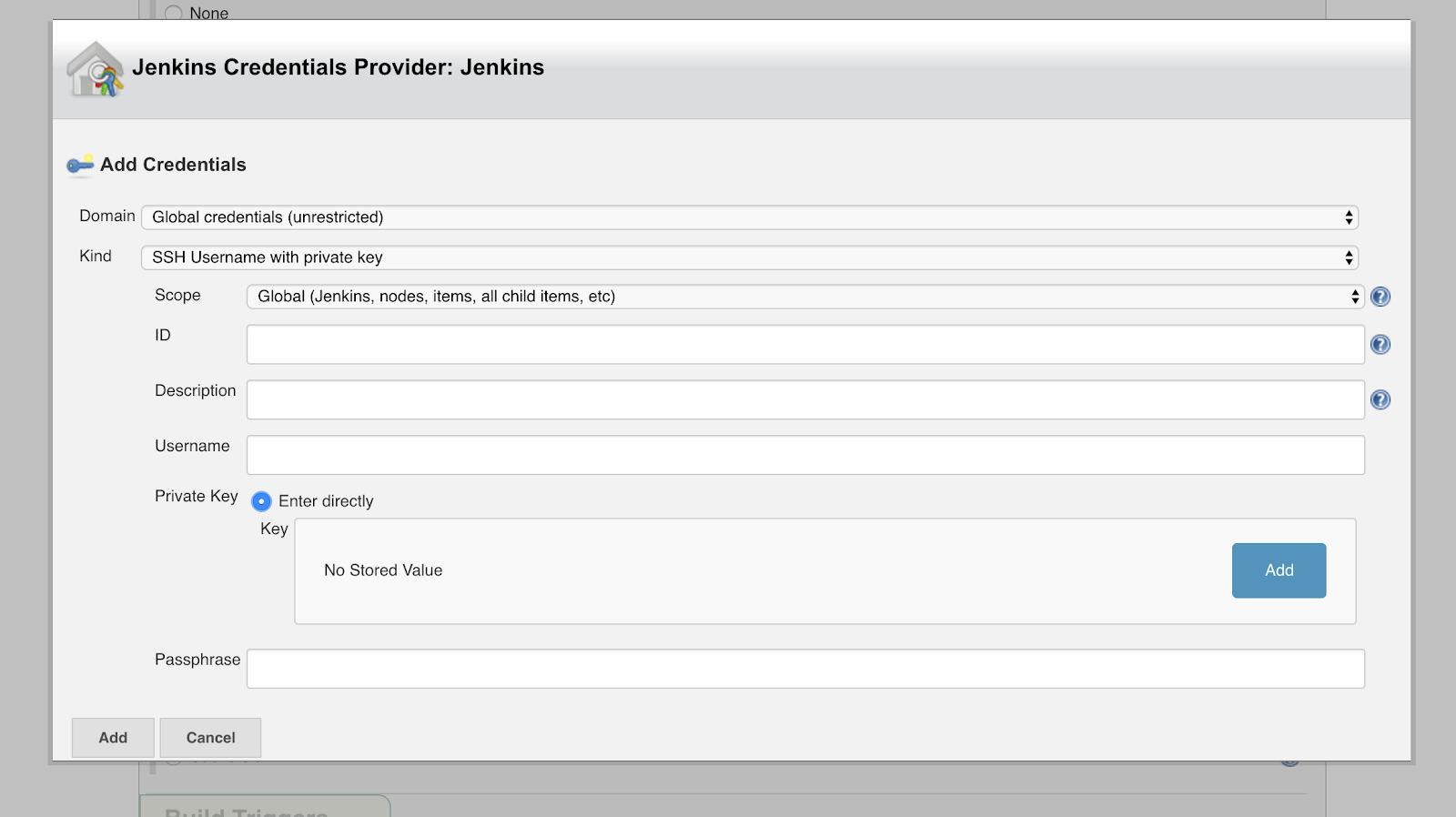
1. Enter git ssh repo url and branch name in the SCM Code management





1. Since our repos are private, we need to add Credentials, Tap on the Add button and Add Jenkins button

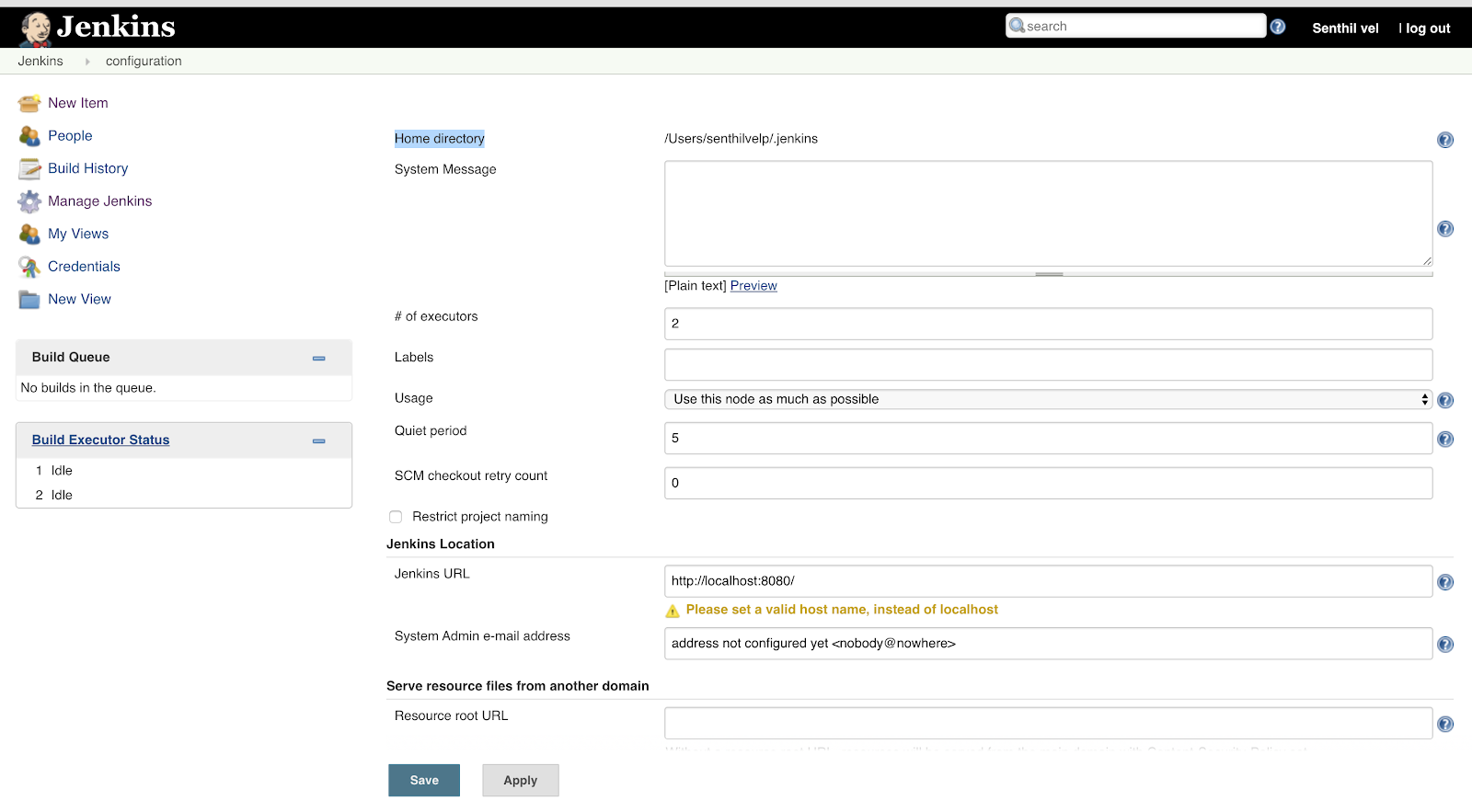




1. Enter the git username and private key, Tap Add button it will show a text box, copy the  private key from id\_rsa in .ssh folder and tap Add  and save the Job,

1. Before Running the job , if necessary , make ssh-add in the terminal , once identity is added , trigger the Job , The default time out for git clone is 10 minutes, (For heavy Repos Cloning first time will hit a timeout, once the First job is failed once again Trigger the job, this will fetch the repo successfully)

1. To know the repo cloned location, go [http://localhost:8080/configure](https://www.google.com/url?q=http://localhost:8080/configure&sa=D&ust=1575460936817000)  
   There Home Directory Location will be shown



|  |
| --- |
| Here my Home Directory is shown as /**Users/senthilvelp/.jenkins**              Open the terminal and navigate to your Jenkins Home Directory and type ls in the terminal, it will list a              A directory called workspace, so inside this directory all the jobs cloned repo will be there        For example, if my job name is xcuitest , then the path where my repos cloned will be /Users/senthilvelp/.jenkins/workspace/xcuitest     EXECUTING XCUITEST **Steps to Follow**     1. Setup the ruby version using rvm and install Pod gem required for the pod install and Set up golang in the local mac machine 2. Xcode 11 provides test plans as a feature , Segregate your test cases to test plans which will be helpful in running the test cases via xcodebuild , for creating test plans in Xcode refer the below link [https://shashikantjagtap.net/wwdc19-getting-started-with-test-plan-for-xctest/](https://www.google.com/url?q=https://shashikantjagtap.net/wwdc19-getting-started-with-test-plan-for-xctest/&sa=D&ust=1575460936823000) 3. If you don't have Xcode 11 , test can be run using **-only-testing Packagename/ClassName/Testmethod** 4. After creating the xctest plans to list out the test plans in your scheme, each scheme can have any number of test plans **xcodebuild -showTestPlans -scheme YOUR\_SCEME** 5. The far most important thing needed we need to remove is Always include new test cases checkbox which will prevent the ram shoot up error, Refer the below screenshot to check     Tap on the Edit scheme option and Tap on the Test option in the below screen      Double click on any test plans and then uncheck the automatically add test cases checkbox as shown below       1. Make sure all your test plans are disabled with the automatically add test cases checkbox 2. Once all set up has done we have build Go for project using the below command , since your ios repo will be different location and Golanf repo will be different location , Every Golang project a make file which will create executables , for that go to the Jenkins Tap on the Job name tap Configure, Under build Section Tap Add build step and add execute shell as shown below screenshot and the make commands        1. The next is we need to build our app in order to run the test over the app, For that xcodebuild command can be used, So add one more execute shell     In that execute shell enter **#!/bin/bash** In the first line of the text box, which will tell Jenkins the following script are executed as bash commands This is known as shebang, it tells the interpreter that the following lines are written for bash    For building Xcode for testing, we need CODE\_SIGN\_IDENTITY and**CODE\_SIGNING\_RELEASE, CODE\_SIGN\_IDENTITY** is used to build apps with different certificates CODE\_SIGNING\_RELEASE is used for taking debug and release build so give the below command in the Execute shell    **xcodebuild  clean build-for-testing -workspace Freshsales.xcworkspace -scheme SanityTest -sdk iphonesimulator -derivedDataPath /Users/testing/Documents/Firebase -destination 'platform=iOS Simulator,name=iPhone Xs 13.2,OS=13.2.2' CODE\_SIGN\_IDENTITY="" CODE\_SIGNING\_RELEASE=NO | xcpretty -s**              In the above command, we have mentioned the derivedDataPath path where the app will be built and once can see the list of simulators available in the machine by **xcrun simctl list** , Select a device from the below list and set it in the destination    In the above command you can see a third party plugin will be used for formatting the outputs created that is pretty ([https://github.com/xcpretty/xcpretty](https://www.google.com/url?q=https://github.com/xcpretty/xcpretty&sa=D&ust=1575460936831000)),**before using the above command please install xcpretty using gem install xcpretty**     1. So we have done with Building up both our App and GoLang, now we start with automating the test cases using the below command, Add a new Execute shell   **xcodebuild -workspace Freshsales.xcworkspace -scheme SanityTest -destination 'platform=iOS Simulator,name=iPhone Xs 13.2,OS=13.2.2' -derivedDataPath /Users/testing/Documents/Firebase test-without-building -testPlan SalesActivitySanityTest | xcpretty -r junit -o /Users/Shared/Jenkins/Home/workspace/SanityTests/junit-report/salesactivity.xml || true**  In the above command you can see xcpretty have the options to Integrate with junit reports or Html reports, For my convenience, I have given JUnit report and its location where it should be saved, Always make sure the location given should be inside the Jenkins home directory        **For Xcode version Less than 11 please use the below command**    **xcodebuild test-without-building -xctestrun /Users/user/Documents/FIREBASE/Freshsales-bqgxugtoohekbjdhtlsfimlggamz/Build/Products/SanityTest\_iphonesimulator12.4-x86\_64.xctestrun -destination 'platform=iOS Simulator,name=iPhone X,OS=12.4' -only-testing:FreshsalesUITests/ContactDetailViewTest/testEditContact\_C9275 -only-testing:FreshsalesUITests/ContactDetailViewTest/testNewAccountCreate -only-testing:FreshsalesUITests/ContactDetailViewTest/testDealCreateWithContactAndAccount\_C60326757 -only-testing:FreshsalesUITests/ContactDetailViewTest/testDealCreateWithContact\_C9253 -only-testing:FreshsalesUITests/DealDetailViewTest/testchangeStageinLanding\_C7578093 -only-testing:FreshsalesUITests/DealDetailViewTest/testEditDeal\_C8346782 -only-testing:FreshsalesUITests/DealDetailViewTest/testcreateDealwithMultipleContact\_C9493\_C9472\_C9470**      In the above command, you can see **-xctestrun** is provided, which will be generated once App build for testing is completed and it will be in the **derived path** as mentioned and **-only-testing** which includes test methods to be ran     1. Now we need to show the output result, for that we need to integrate Junit result plugin, So go to Post-build action and Add JUnit reporter there as below        1. Now save the configuration and tap trigger build , for the test to get started and the result will show as below         **Note :**  **The above screenshot uses multi-job plugin , for parallel running to test plans, since freshsales having test plans based on modules like Lead, Contact, Account and Deals, Have created a separate job for each test plans and integrated them to a single Main job**    **For multiple jobs to run, we don't need to build the project each and every time , so just use xcodebuild test command to execute test and all the child job should refer to the same workplace, which can be done as shown in the below screenshot, Just tap the Advanced options in the general tab and it will get expanded and provide the custom location** |