DevOps Integration using Jenkins Pipeline

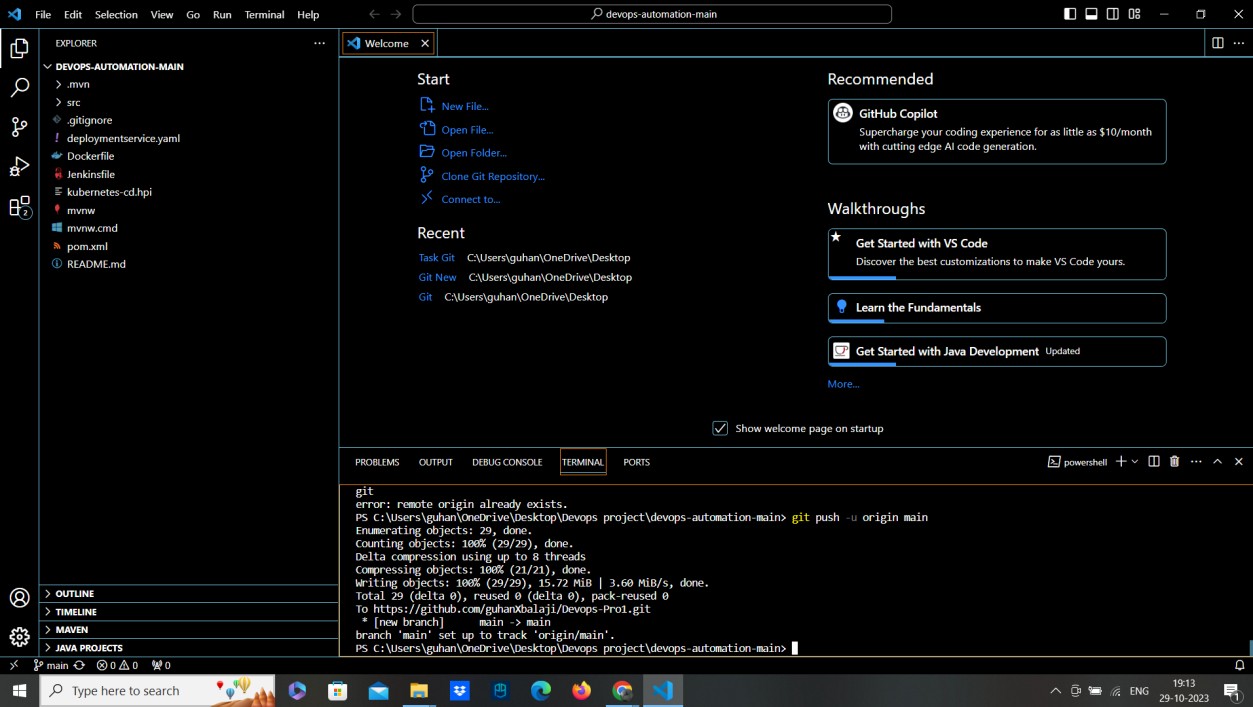
Name: S. Jeevanantham Balakrishnan

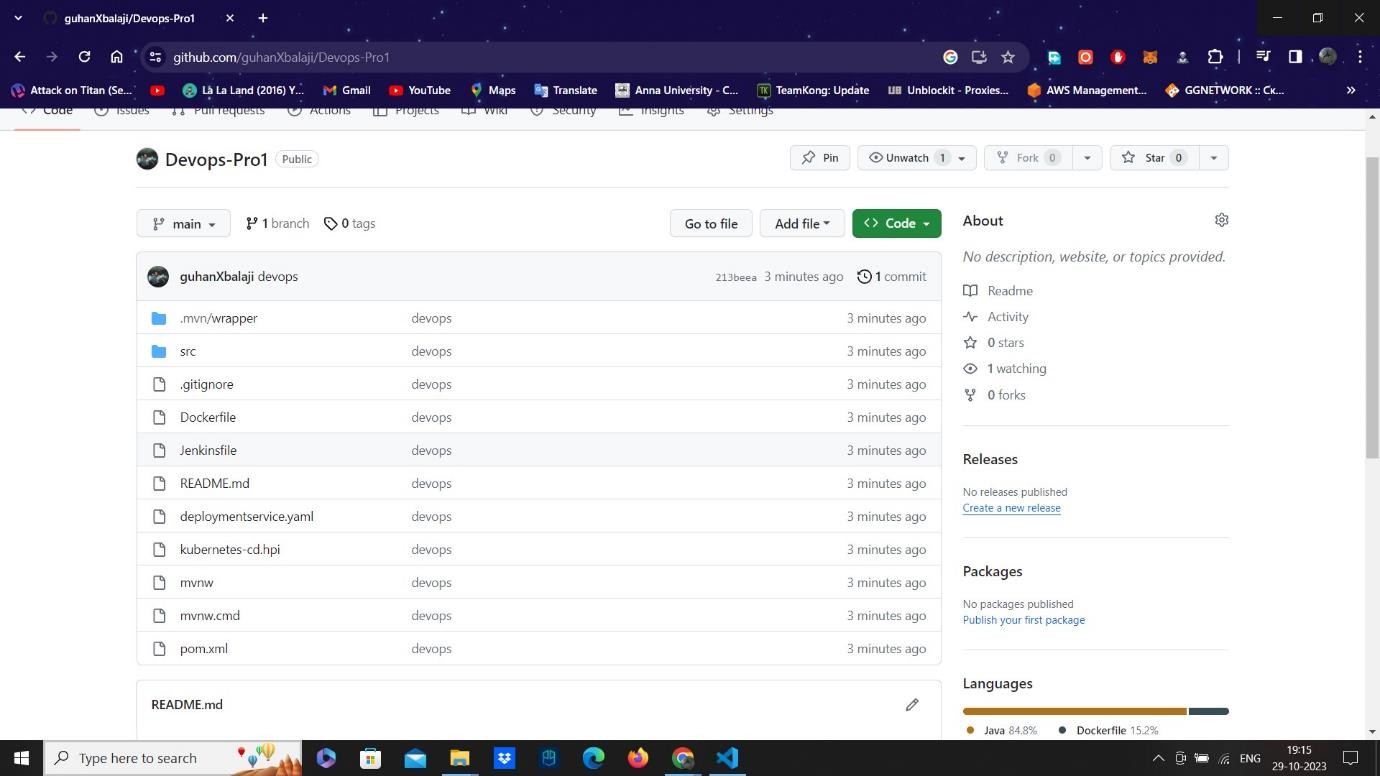
In this project we are going to build and push docker image using Jenkins pipeline. Building and pushing a Docker image using a Jenkins pipeline is a common and useful task for automating the deployment process.

Steps Involved:

Step 1:

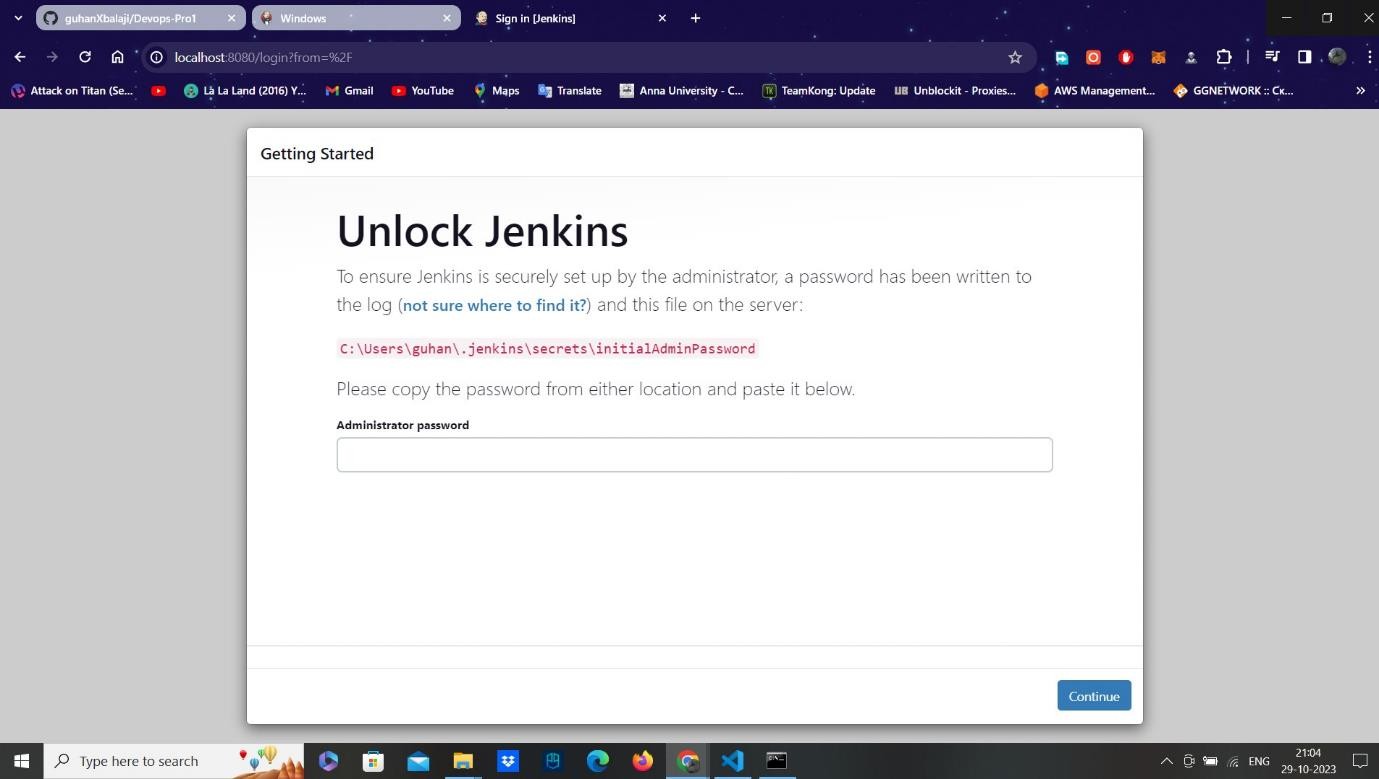
* Create a git hub repository for your application.
* Open the VS code and the open the application folder. Use git init to initialize the git repository.
* Add and commit the files. And add remote origin with url of git repository.
* Then Push to the master using push -u origin master.
* We finally committed the file of our application to our git hub.



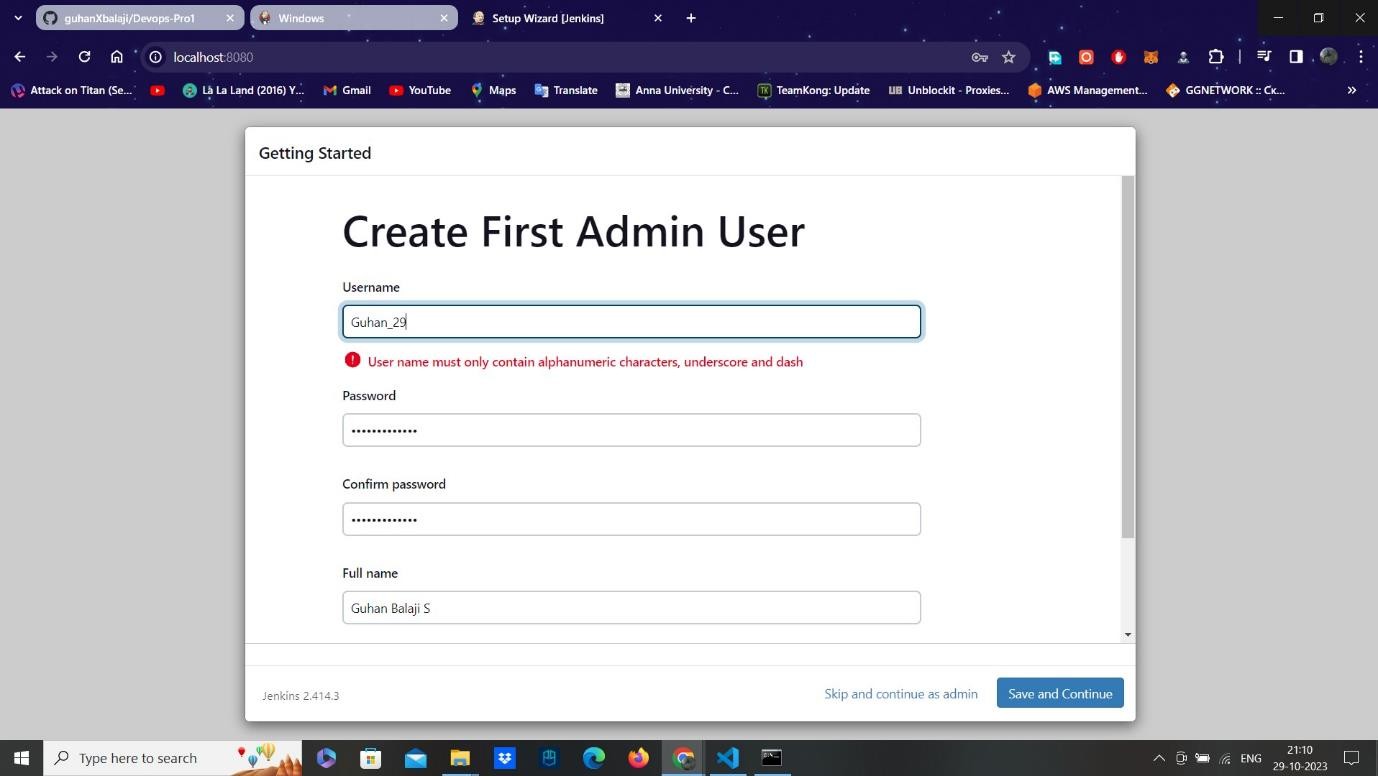


Step 2:

* Install openjdk 8 or higher in your pc and download Jenkins war file from the jenkins official website.
* Using command prompt as administrator. Run the jenkins war file.
* Now start Jenkins by opening a web browser and navigating to http://localhost:8080.
* Now login the Jenkins using the secret password, which will be found in

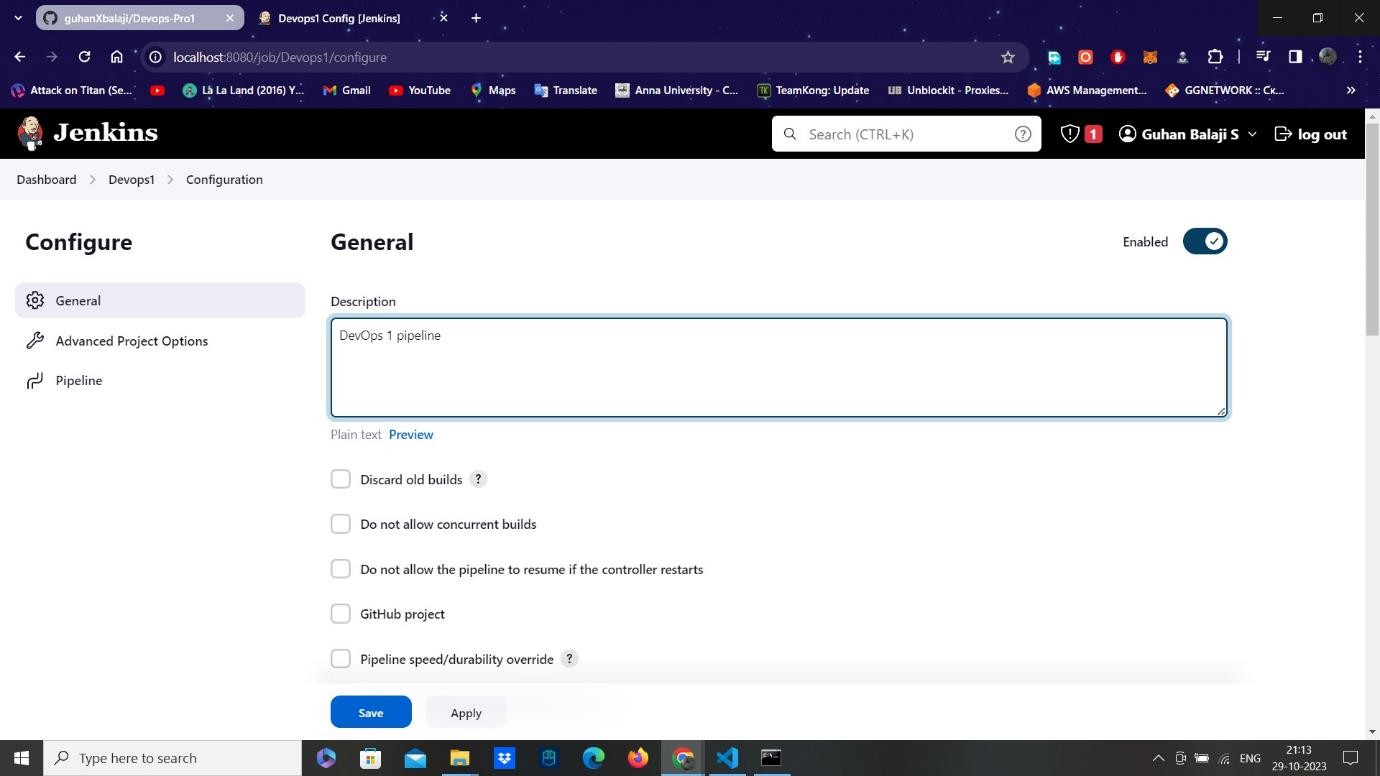


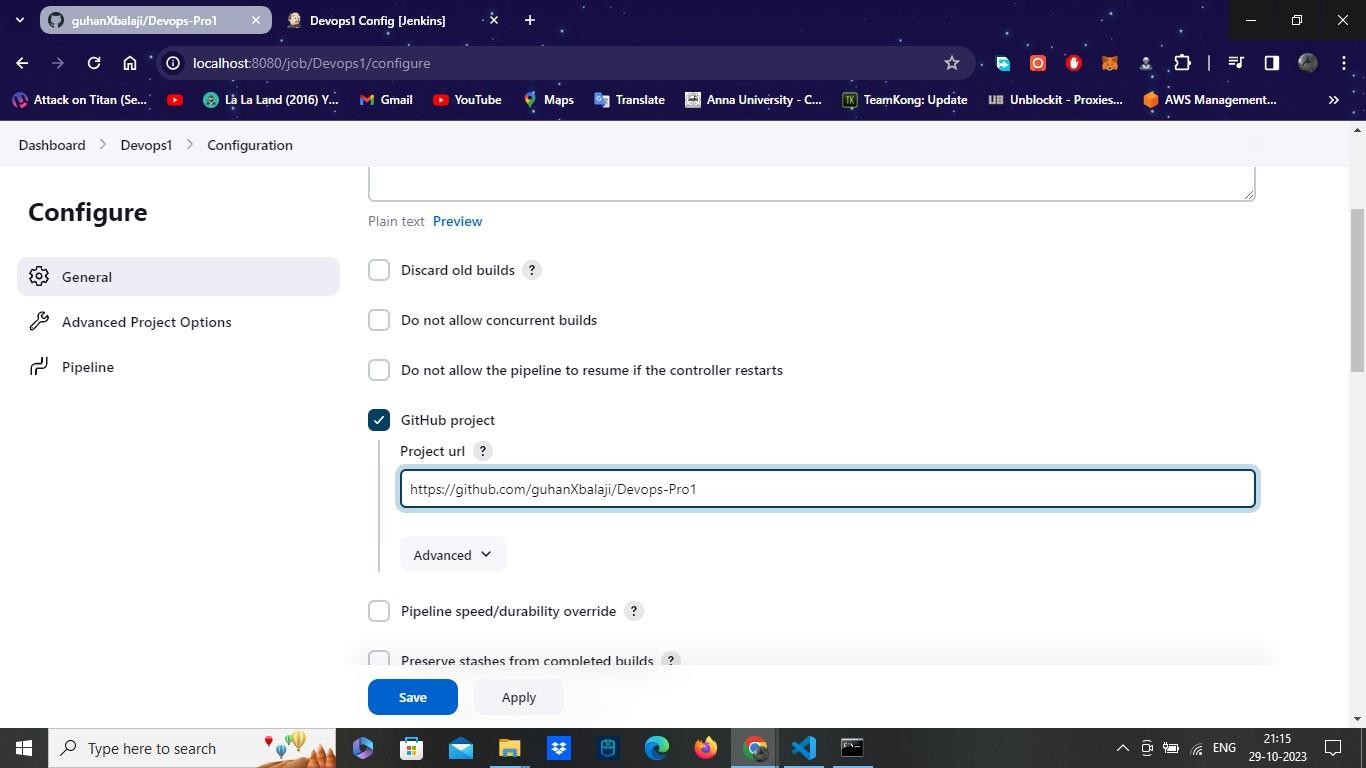
jenkins/secrets/initialadminpassword in your PC.

* Now login in the jenkins. Add username ad create new password. Install the plugins.

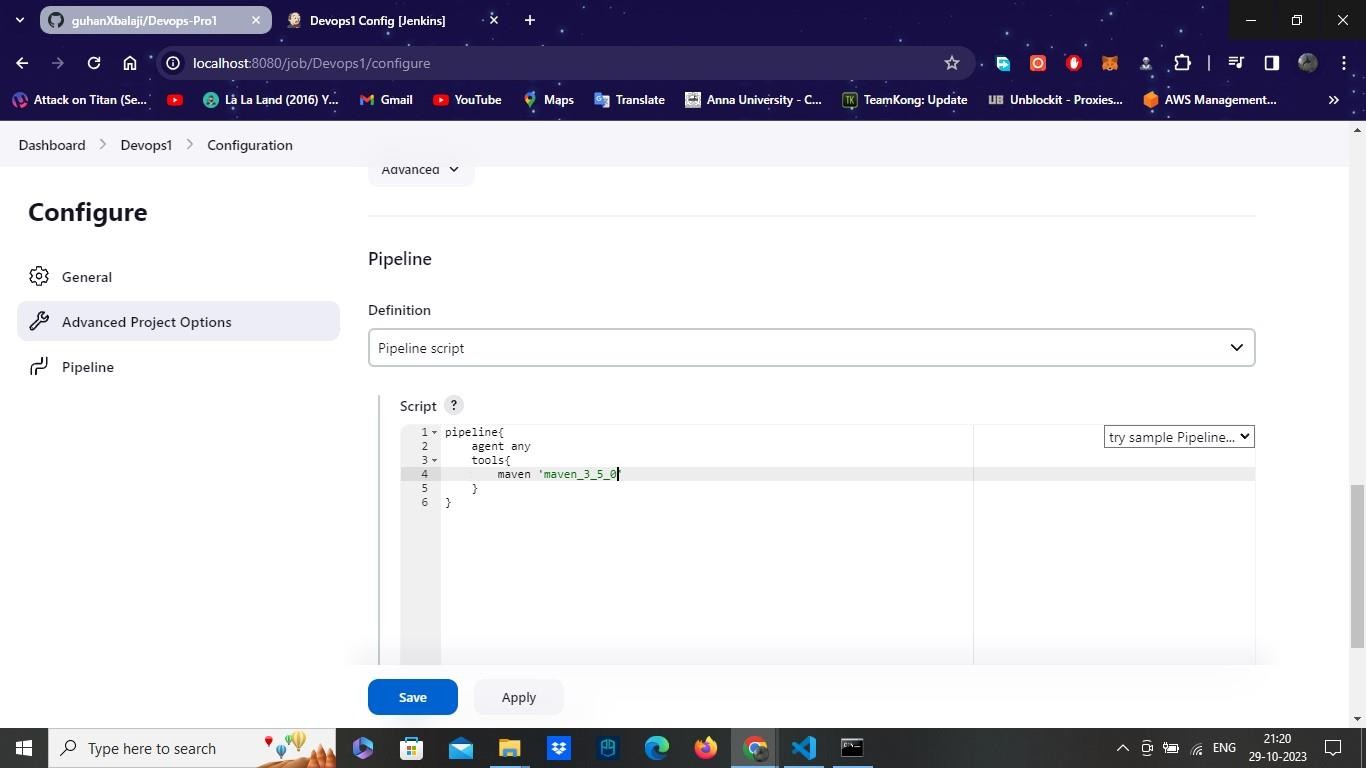
Step 3:

* Create a pipe line. Give the name and description.
* Select GitHub project and the URL of our repository. And also select build trigger snap.
* Now we should write the declarative pipeline. With any agent and tools with Maven

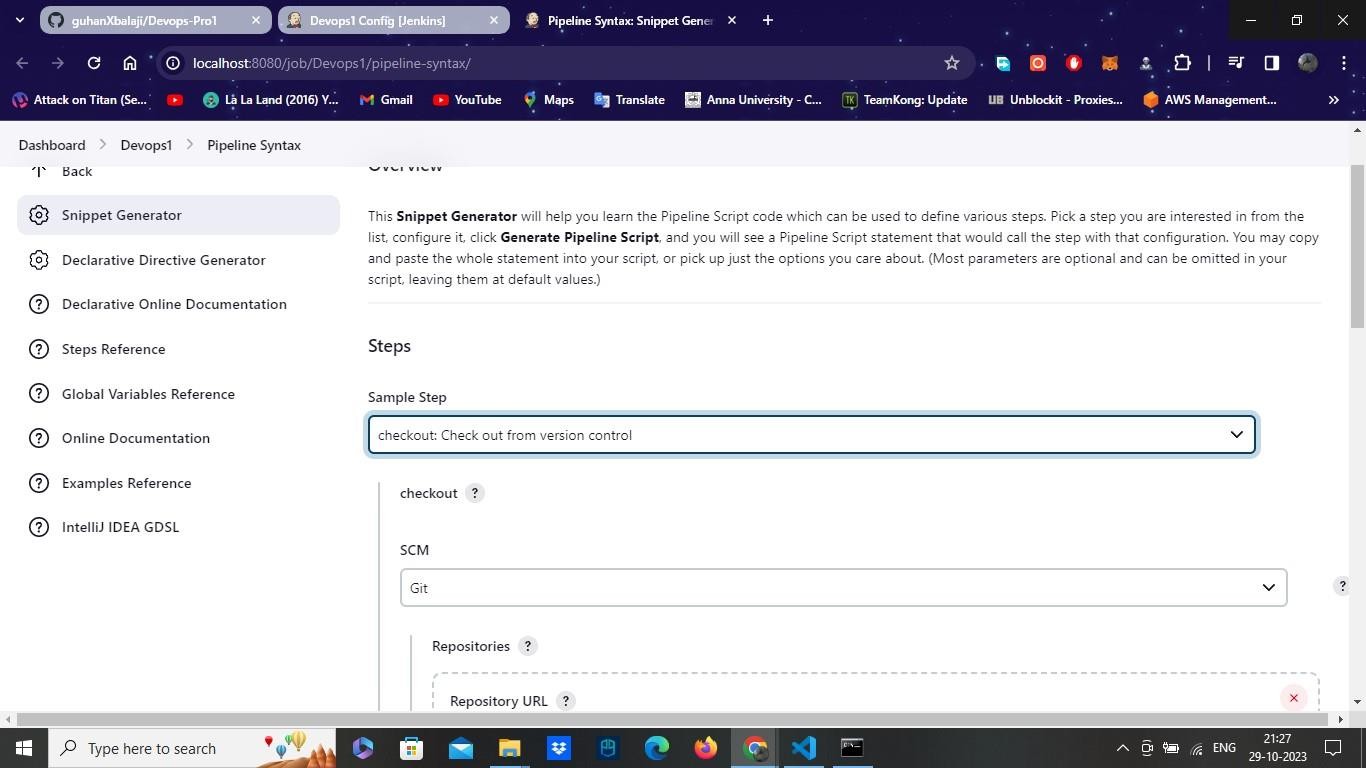


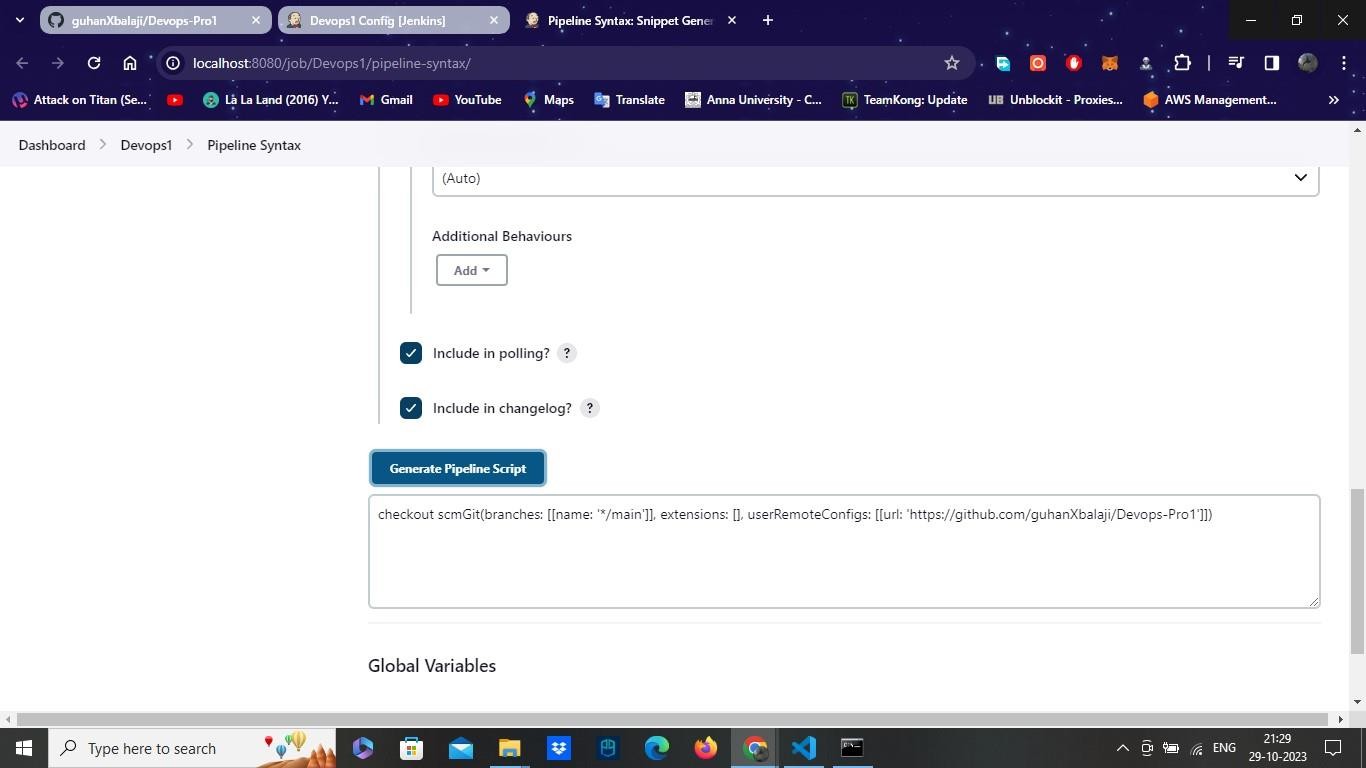


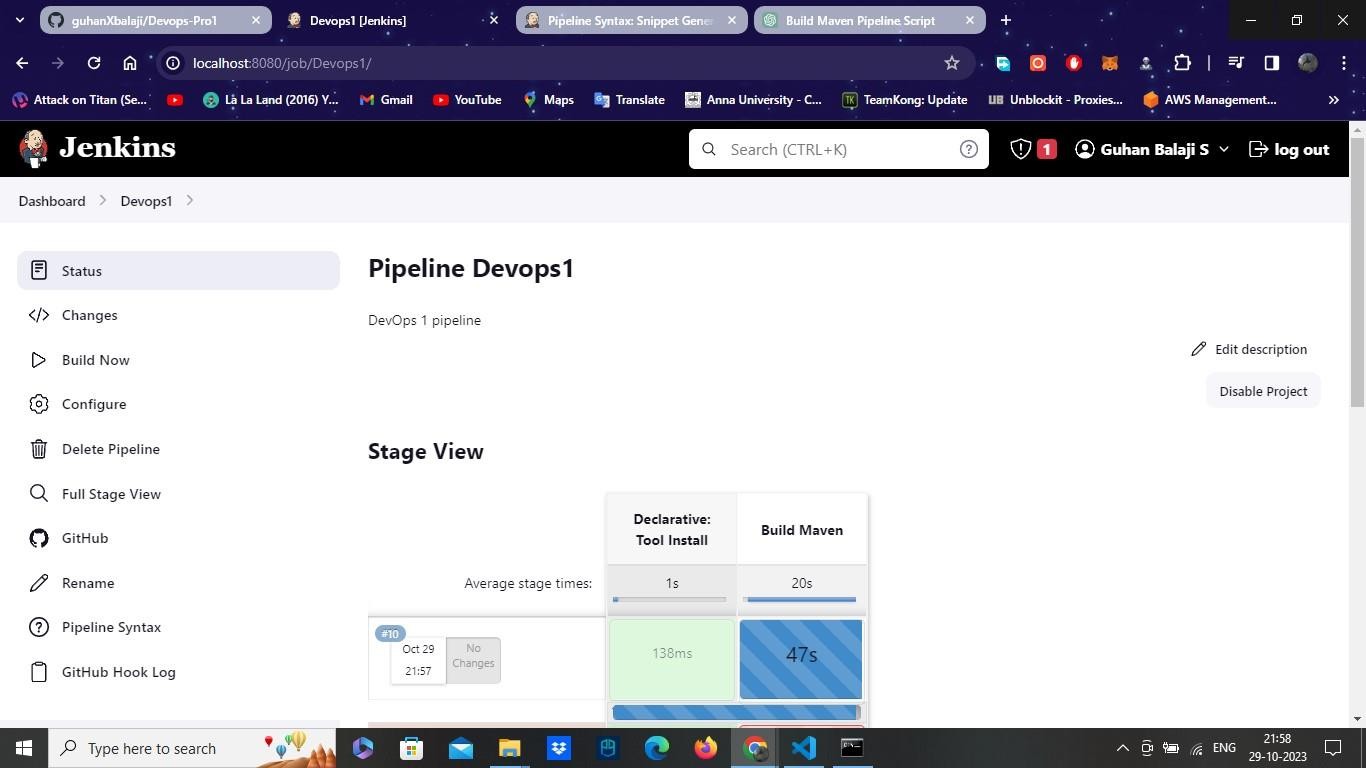
..

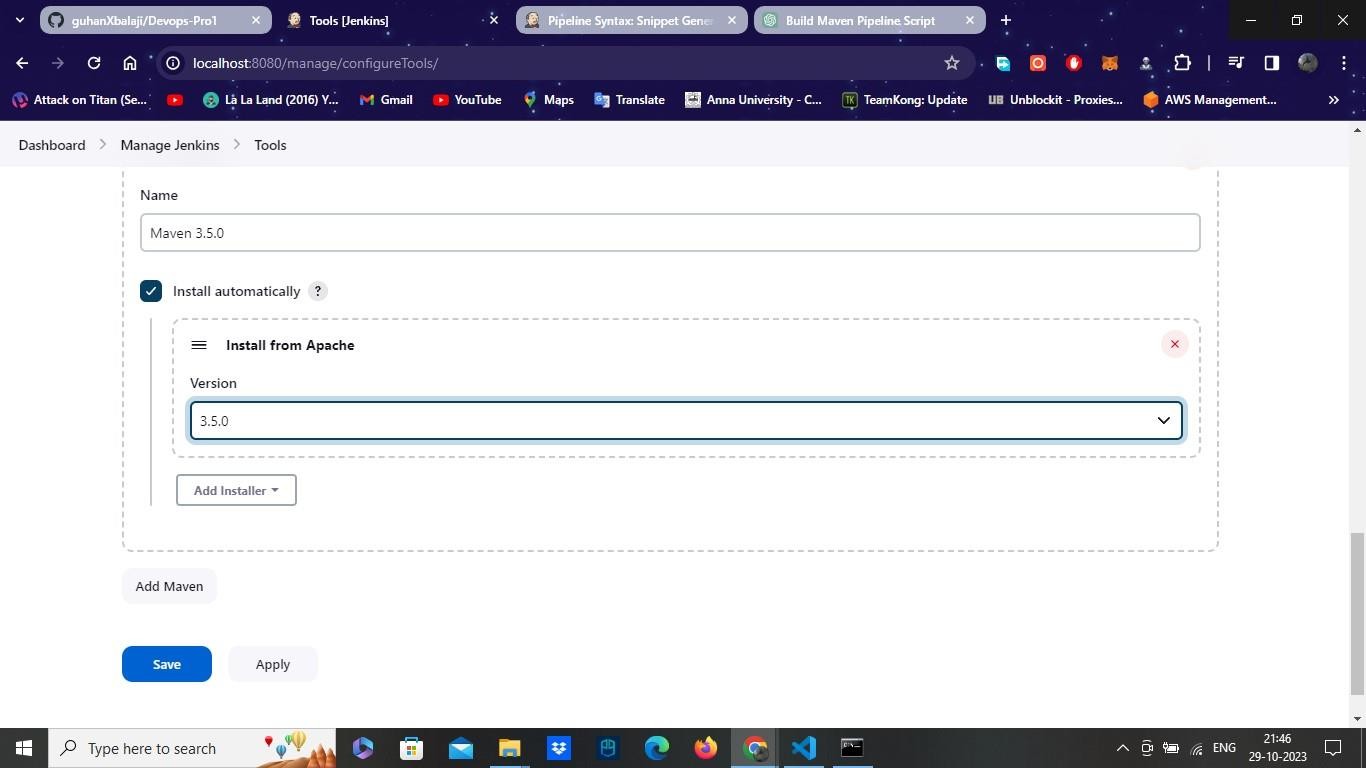


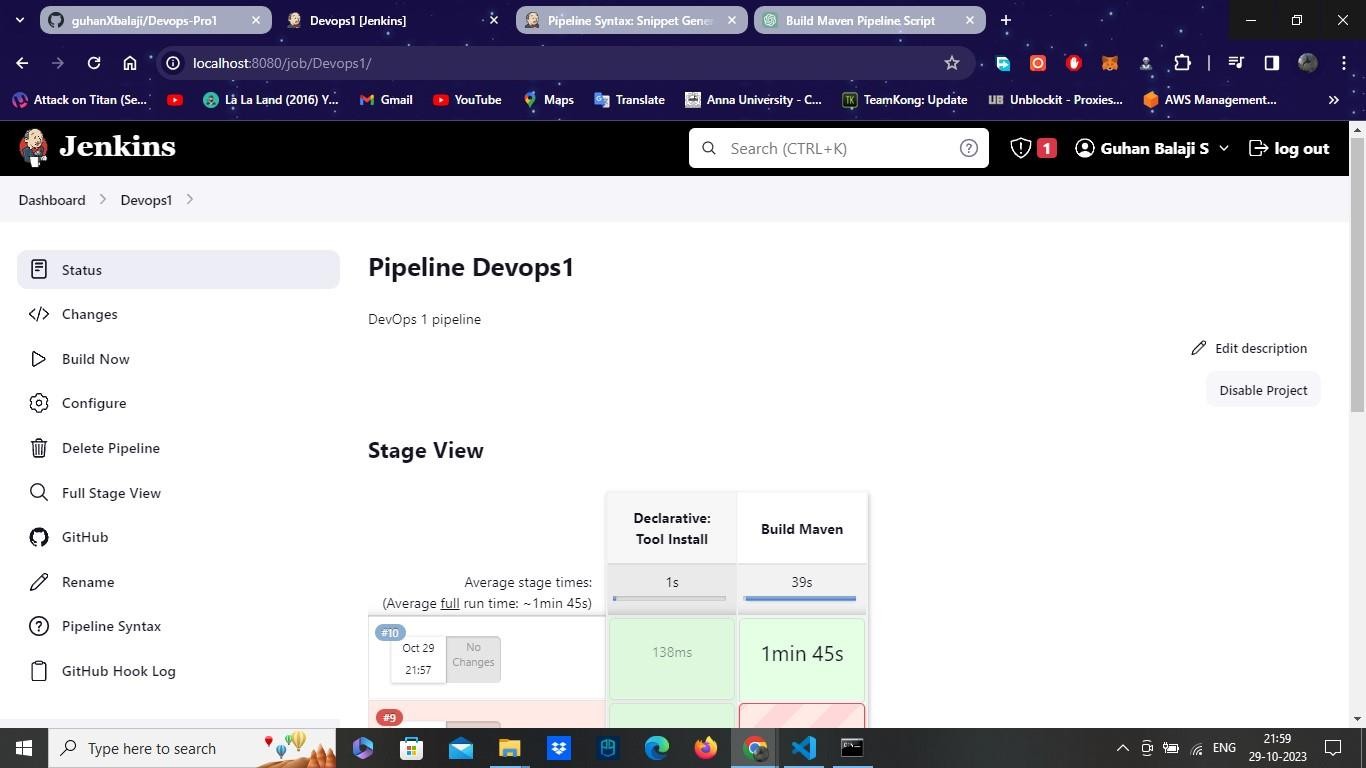
* Now add the stages in your script . Where add the stage as build maven.
* Use the pipeline syntax below and generate the script for git checkout .
* Select Checkout in sample step, git from scm and add your repository URL.
* Change the branch to master or main. And finally generate the script, paste in to the main pipeline script.
* And add the code bat ‘mvn clean install’ .
* And select build now and build it.







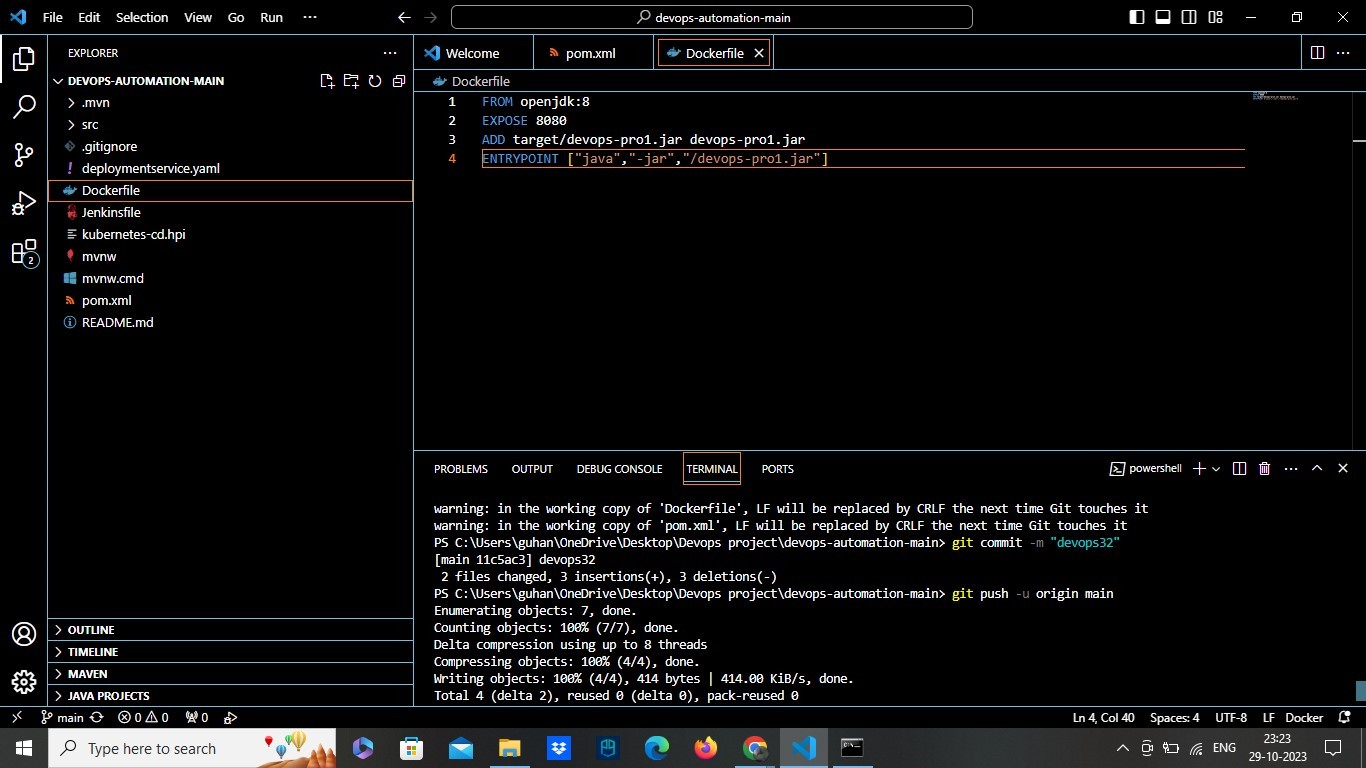




Step 4:

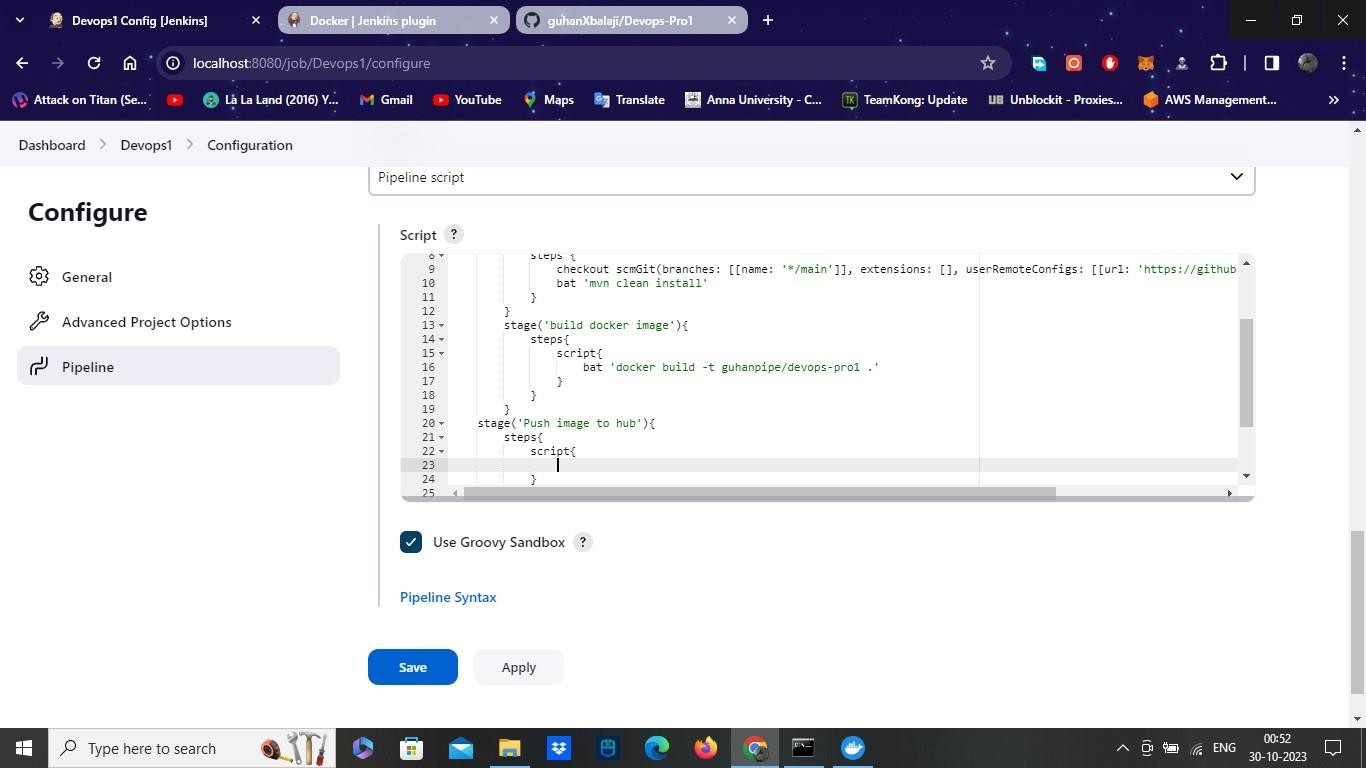
* Add the jar name in your both pom.xml and docker file from your application folder, which is presented in your GitHub Repository.
* And commit the message and push to your master repository again.

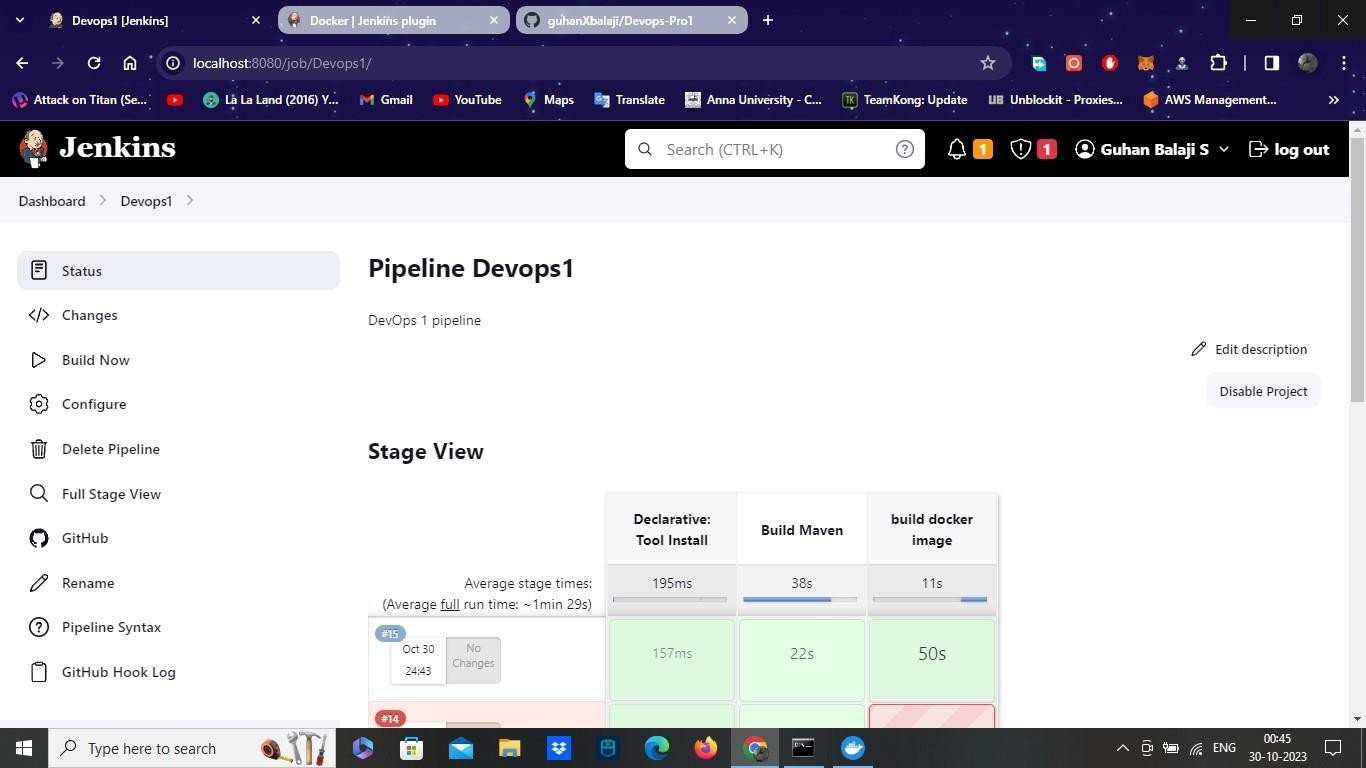
.

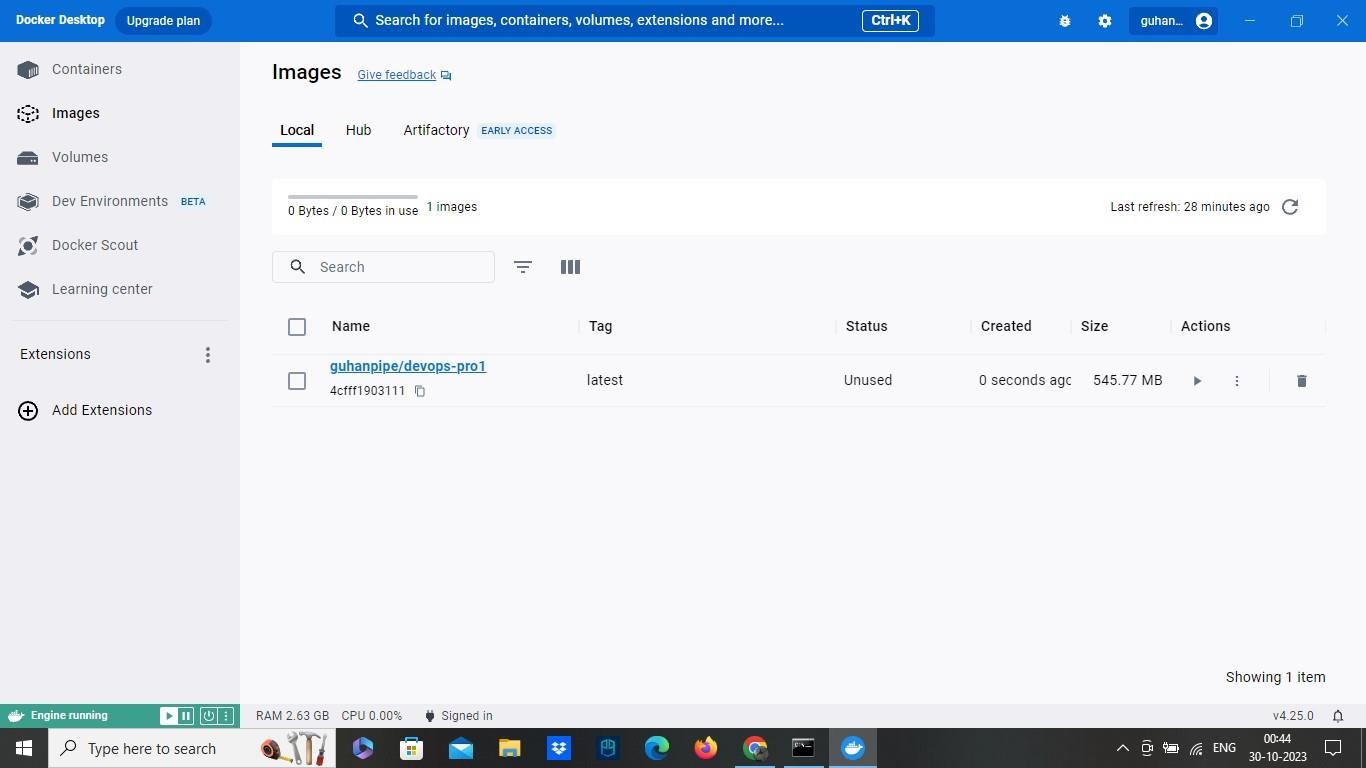


Step 5:

* Open the Docker desktop app and start the docker engine in your Pc.
* Now go back to configure in jenkins.
* Add a stage Build docker image. Add steps and script inside the stage.
* Add bat ‘ docker build -t guhanxdocker/imagename .’ which will build the docker image.
* Apply it and save it. And select build now to build again with this stages now.
* Note: Check the docker got added in your jenkins. if not install docker plugin from manage plugin and add docker in system configuration with the version you want.

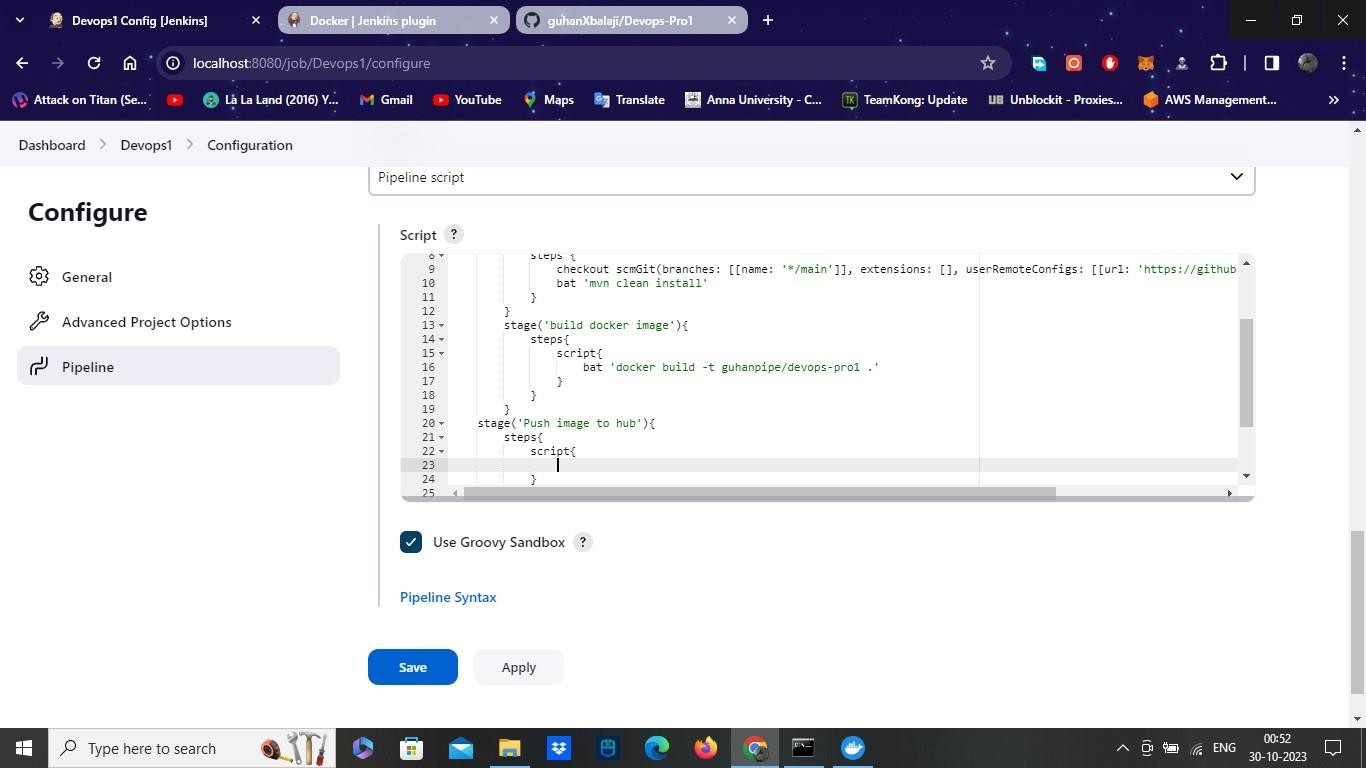


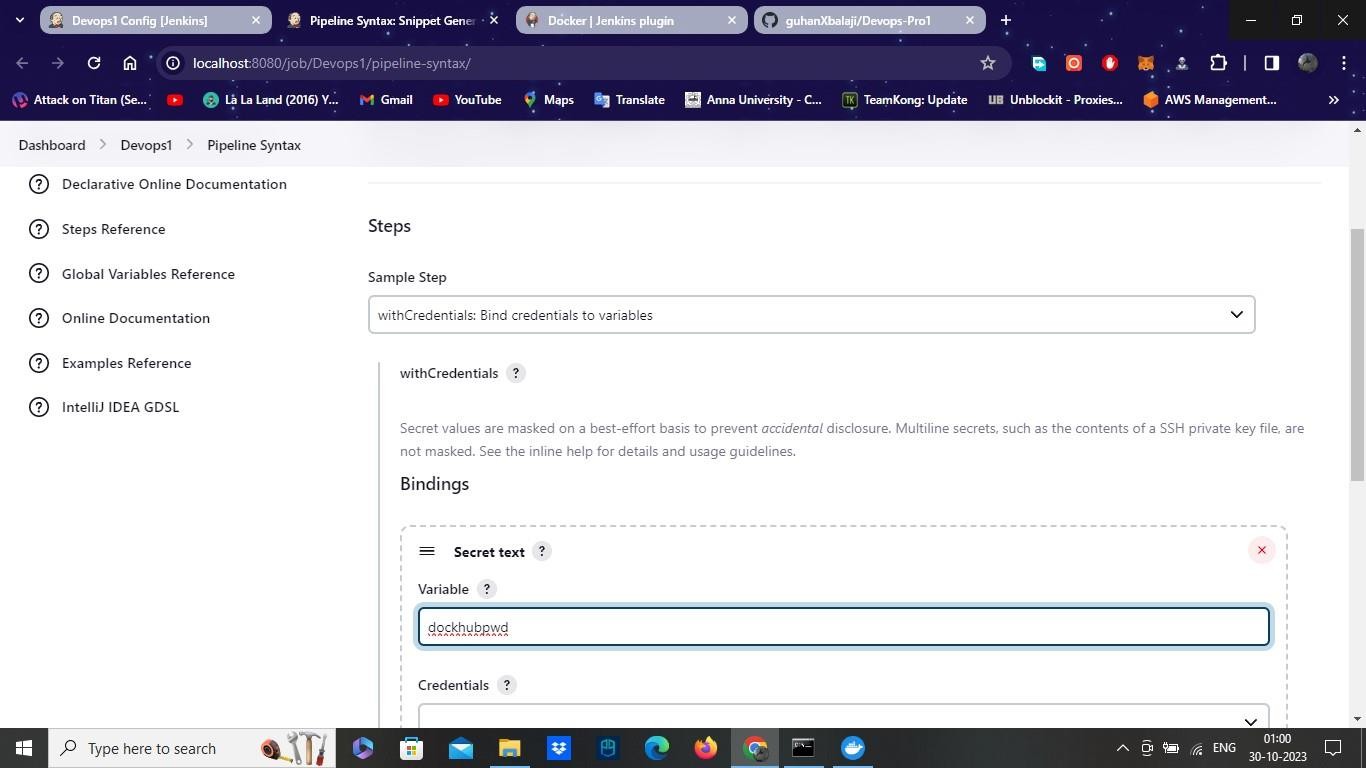


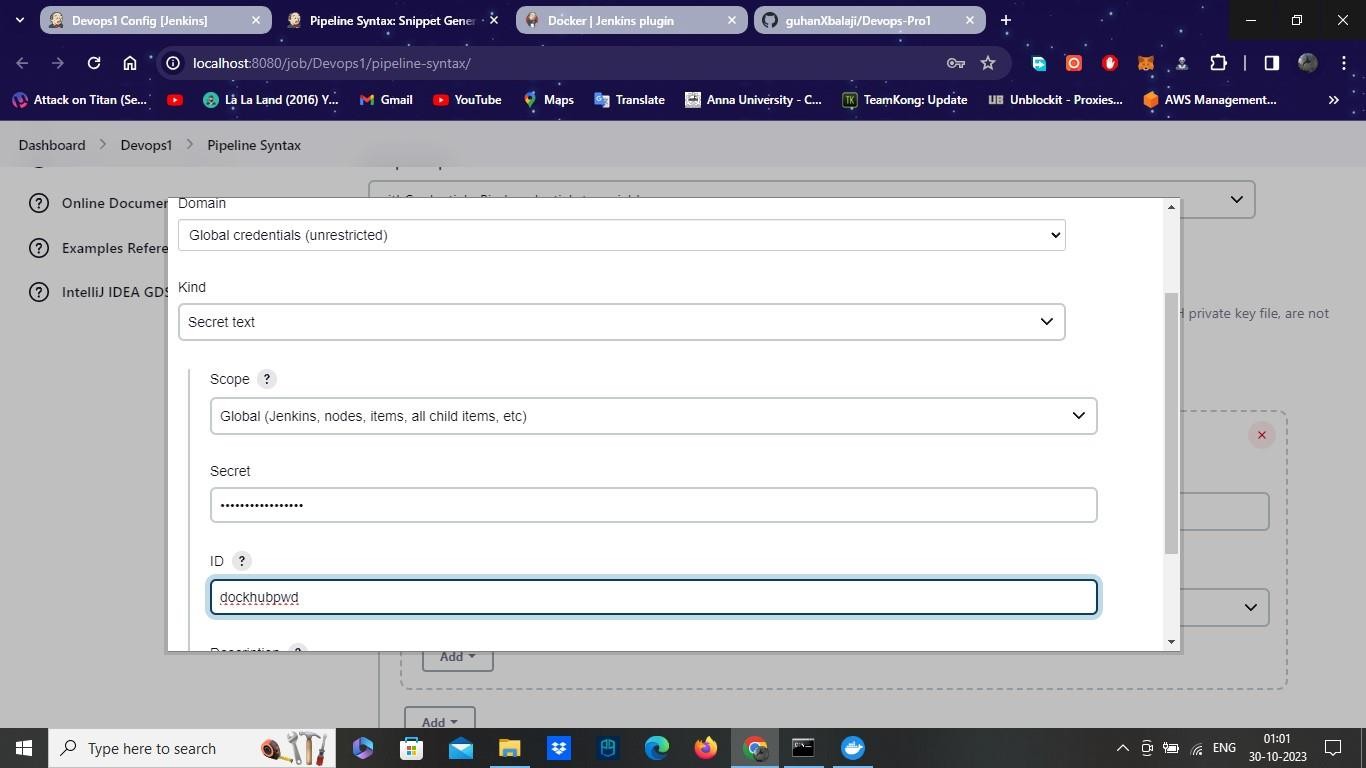


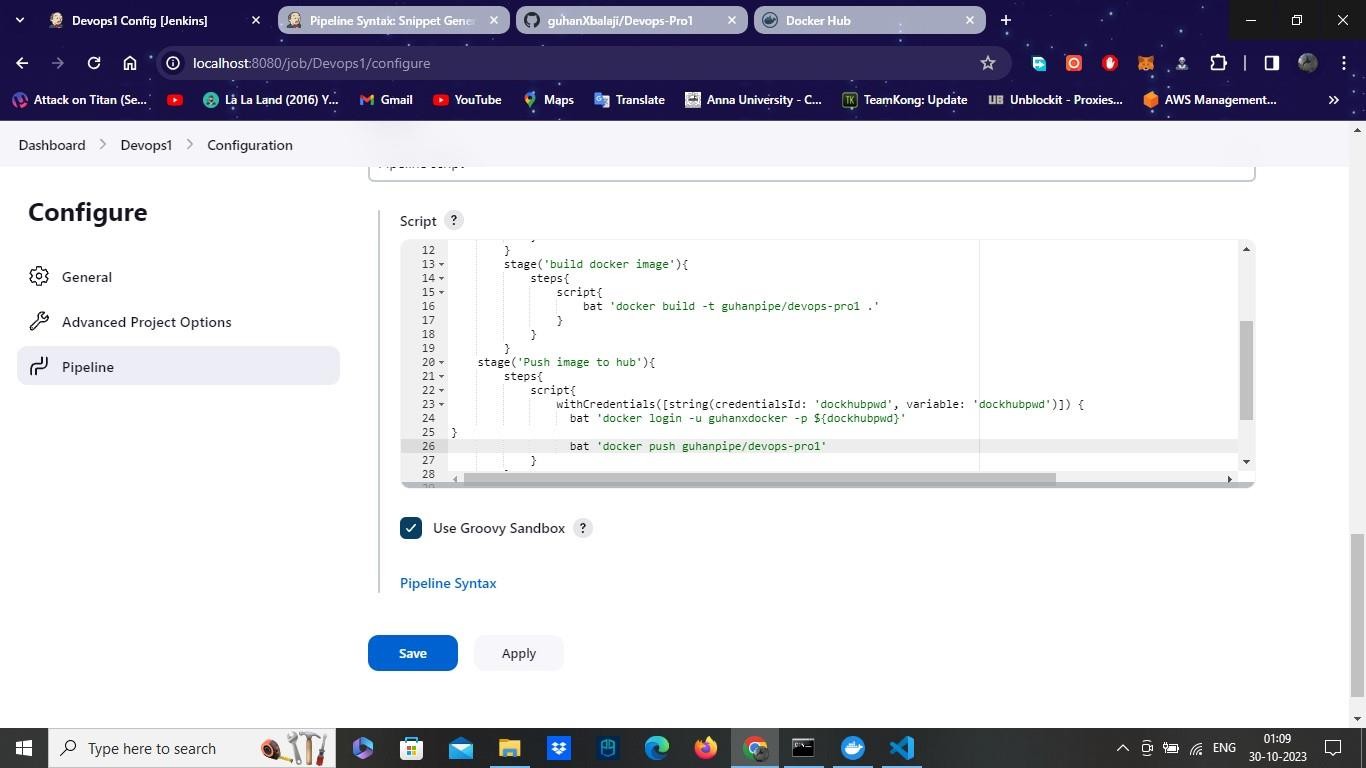
Step 6:

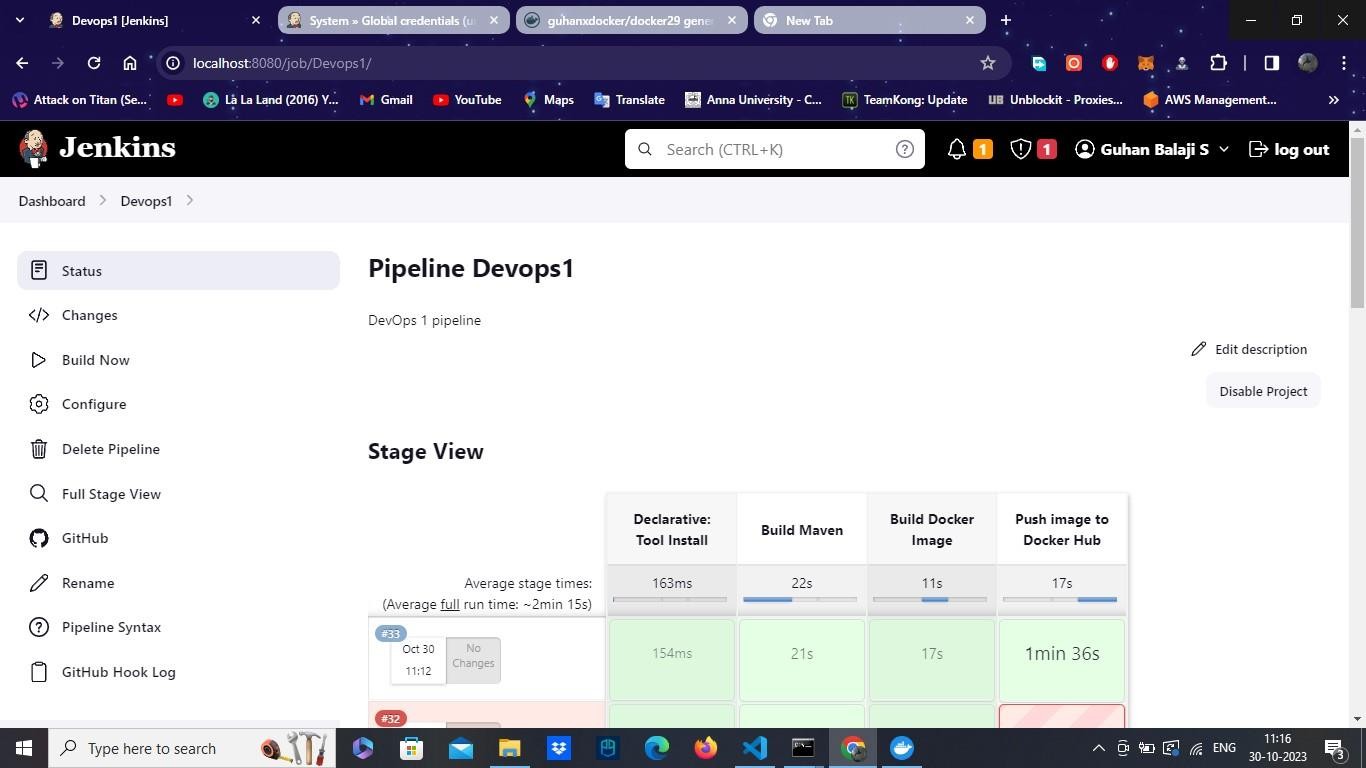
* Now we have to push the docker image we built to our docker hub.
* Add the stage push image to hub. Add a step with script. To add a script, use pipeline syntax.
* Select with credentials in sample step, add secret text. Add variable as dockerhubpwd and add credentials in secret text.
* Add Credentials, choose kind as secret text. Enter your docker hub password in secret. And enter any ID.
* Save it and choose the credentials we created now.
* Generate the script, copy it and paste in our declarative pipeline.
* And add a command below the generated script with bat ‘ docker login -u guhanxdocker -p ${}’.
* So, Now we can able to login in our docker hub with script we given above.
* Now to push to our docker hub.
* Add a command bat ‘ docker push our image name’.
* Apply and save it.
* And Finally select build now to start the build.





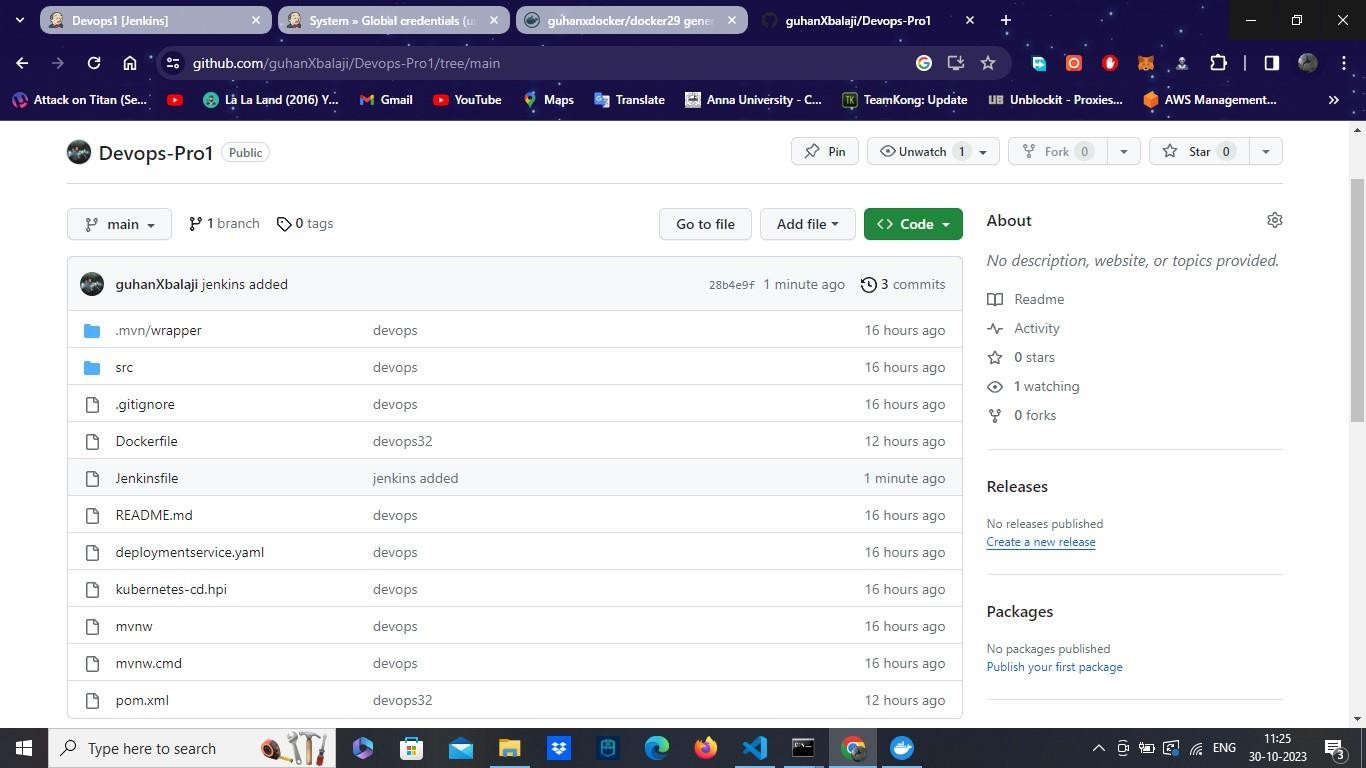


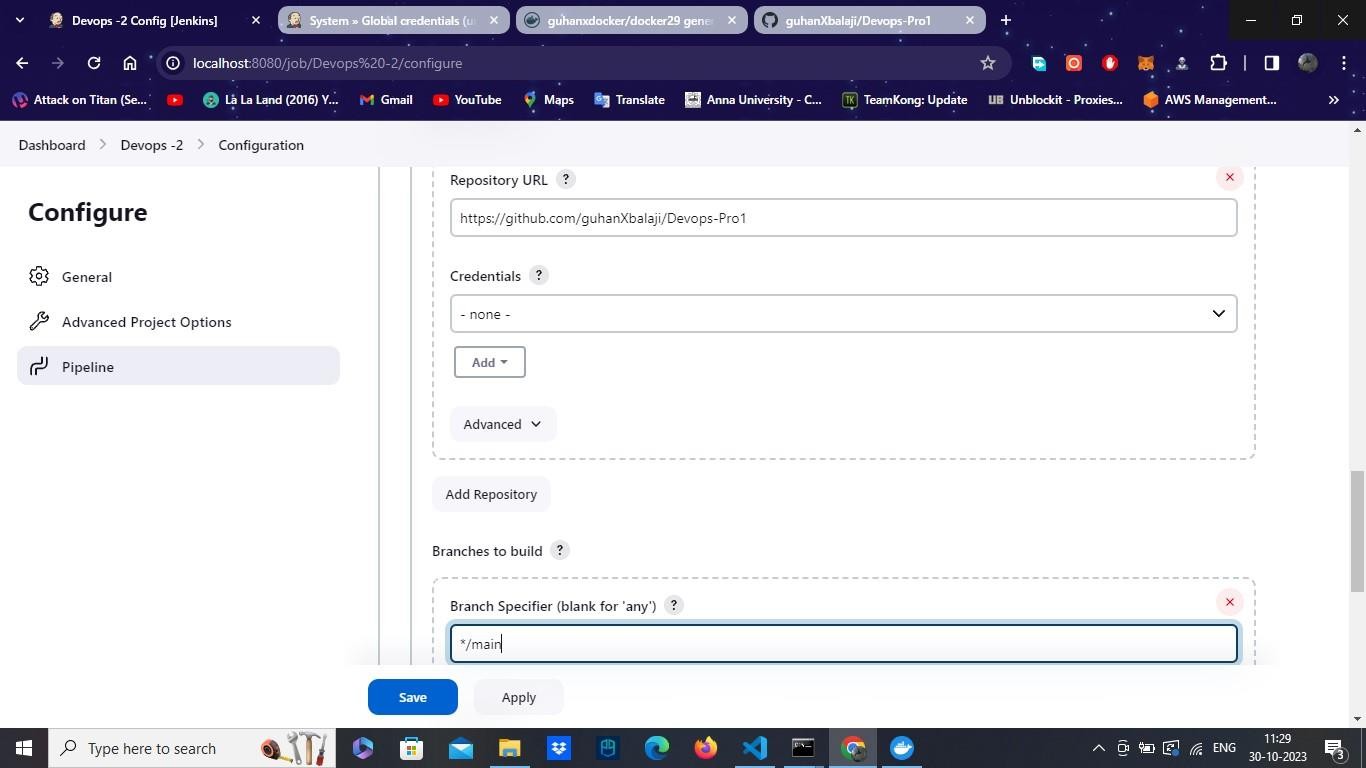
--

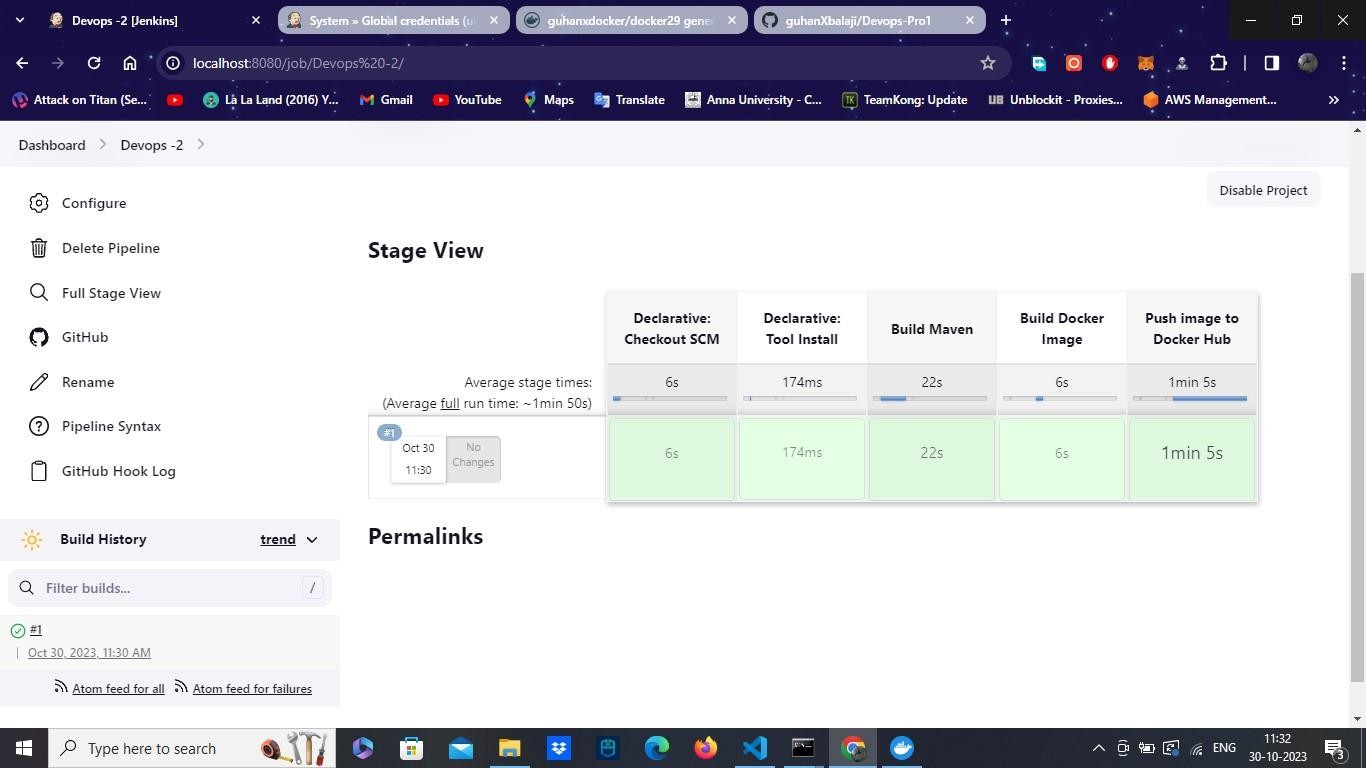


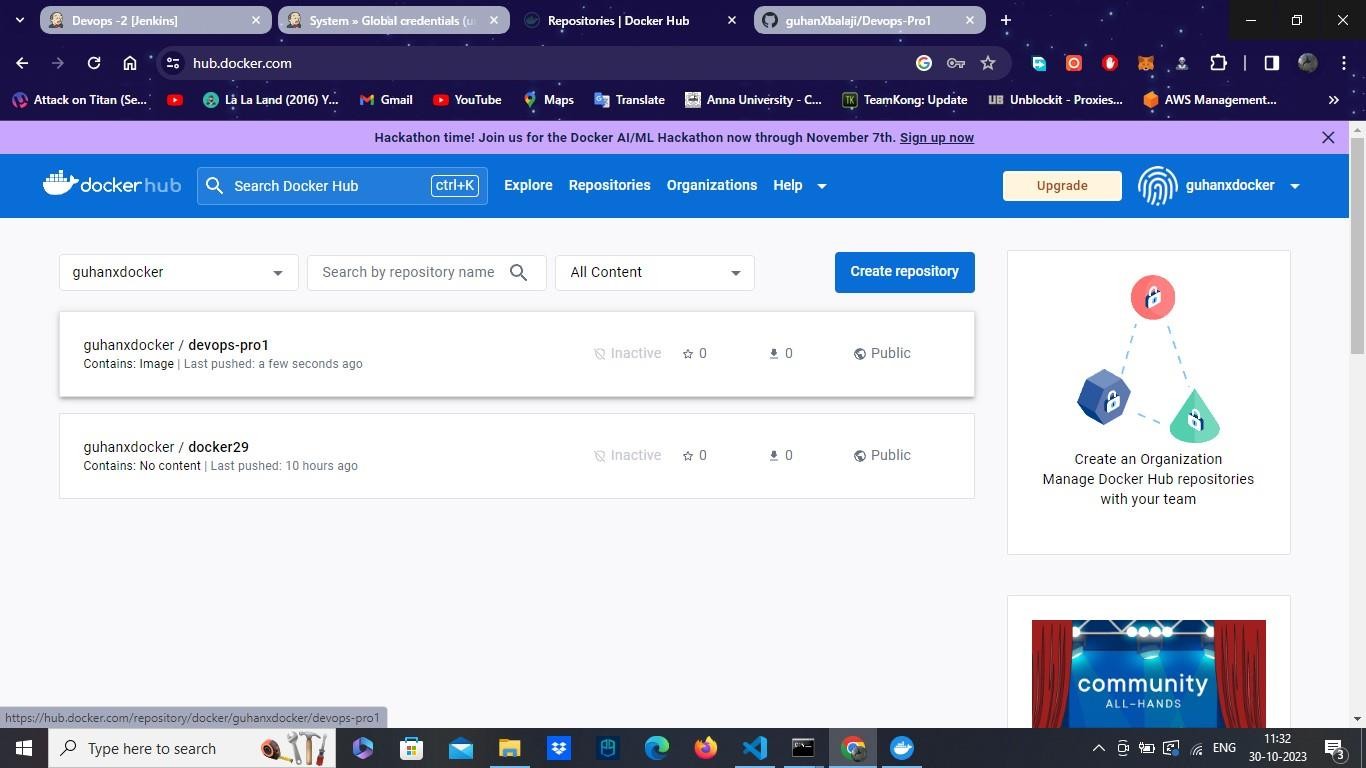
Step 7:

* We successfully pushed our image to our docker hub
* And the declarative pipeline script also runs without error.
* Now we can try it with using jenkins file.
* Create a jenkins file in our GitHub repository. Copy the pipeline script of our project we created.
* Now paste in to the jenkins file we created and save it.
* Add a New pipeline, give a name.
* Enter the project repo URL from git hub in git hub project.
* Now we select the pipeline script from scm instead of pipeline script.
* Select scm as git.
* Give the repository URL. Select branch as master or main and select path of the jenkins file.
* Save it and Select Build now, where it will read the Jenkinsfile and start to build.









Now finally the Docker image pushed to the hub with the help of Jenkins file also, This how we will build and push docker image using Jenkins pipeline. And this how DevOps end to end integration works.