DevOps

> **Product Name:** Integration wth jenkins and Docker.
> **Platform** : Nodejs.
> **Phase:** Deployement
> **Date:** Nov 10, 2022
> **Creator** : **Fayaz Shaik**

Requirements :

1. Jenkins Login credentials.
2. Docker hub Account

Setting up the Jenkins Job first

Add These Files to your VS-Code

- Create a file called `jenkinsfile` in your project and add the below code to the file.

```
/* Requires the Docker Pipeline plugin */
pipeline {
    agent { docker { image 'node:16.17.1-alpine' } }
    stages {
        stage('build') {
            steps {
                sh 'node --version'
            }
        }
    }
}
```

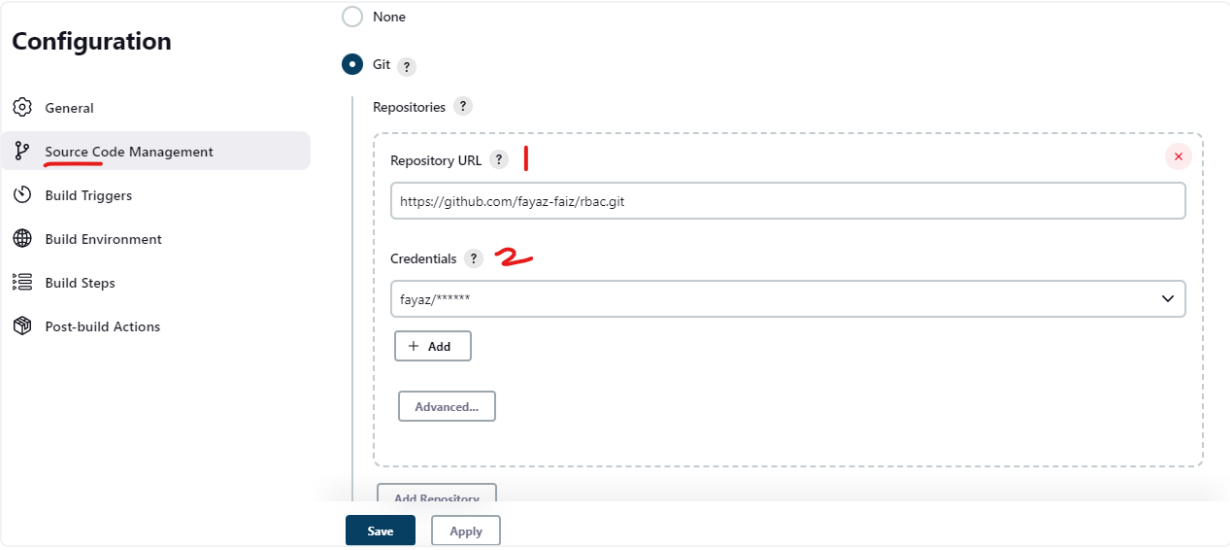
- Create another file called `Dockerfile` in your project and add the below code to the file.

```
FROM node

WORKDIR /rbac

COPY package.json /rbac/
```

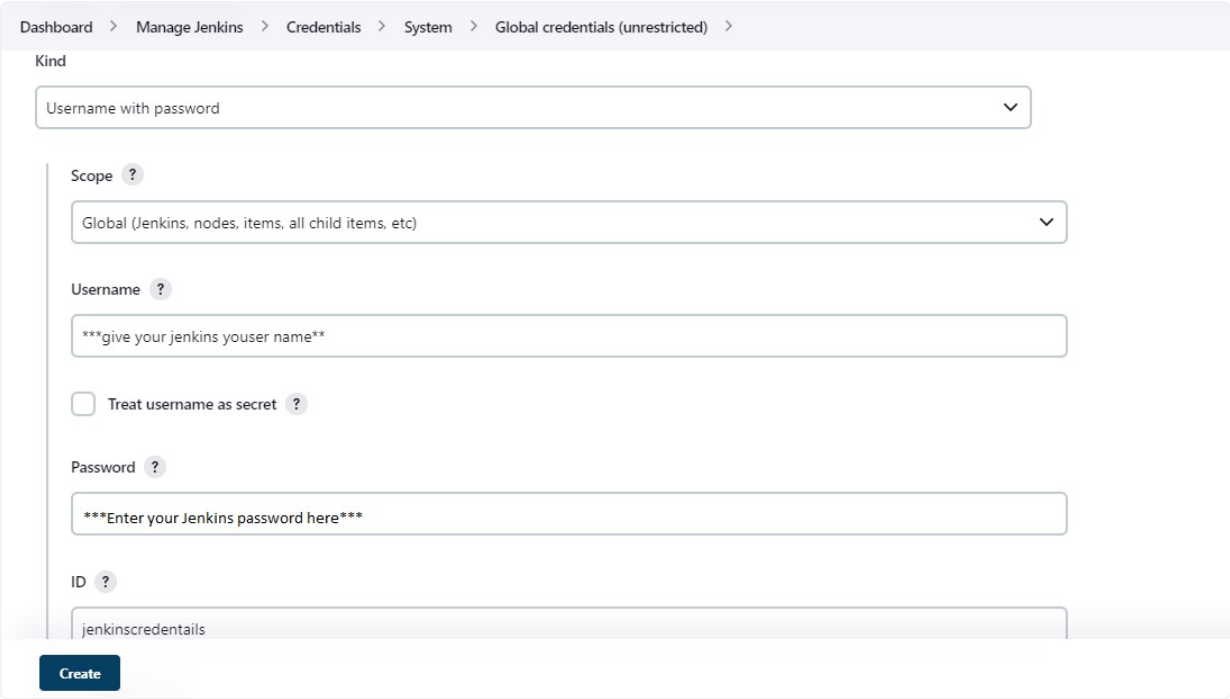
```
RUN npm cache clean --force

RUN npm install

COPY . /rbac/

EXPOSE 8081

CMD ["npm","start"]

```

Jenkins setup: CI/CD Pipeline for a NodeJS Application with Jenkins :

- First Login into the Jenkins application click on NEW ITEM and select a FREESTYLE project give the item name you want. (ex: your project name) now your job is created.
- Go to the configuration and select the source code manager as shown below.



1. In the Repository URL paste the GitHub link where your project is available
2. credentials you need to add the Jenkins credentials to get this you need to set up The credentials here only admins can add these credentials. To add These follow the below steps.
    - Click on Dashboard and select manage Jenkins.
    - Click on Manage Credentials and click on the system then click on global.
    - Click on add credentials as shown below and click on create button.
    -



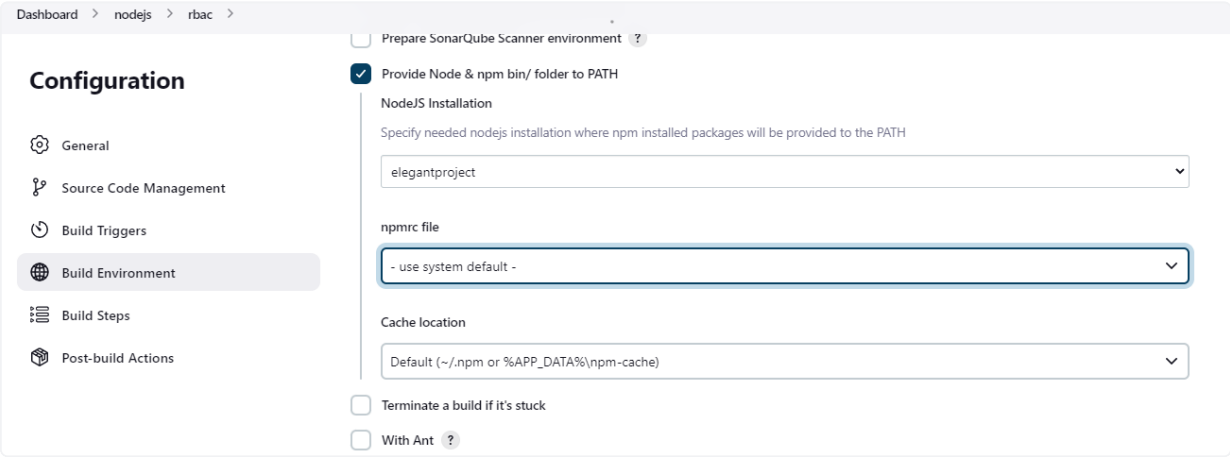
- After setting up Jenkins we need to configure NodeJS on Jenkins.

Install NodeJS plugin:

  - Open Jenkins: **Manage Jenkins > Plugin Manager > Install NodeJS plugin.**
- Global Tool Configuration :
  - Open Jenkins: **Manage Jenkins > Global Tool Configuration > NodeJS**
  - Set your compatible node version. We can set multiple NodeJS versions for multiple applications. (here In this we are using 16.16.0 V)

Now go to the job configuration again and select :

Build Environment > Provide Node & npm bin/ folder to PATH
Select the Nodejs version that is compatible with your application.



Build > Execute shell:

```
#!/bin/bash
echo "------> Install node modules <------"
npm install
```

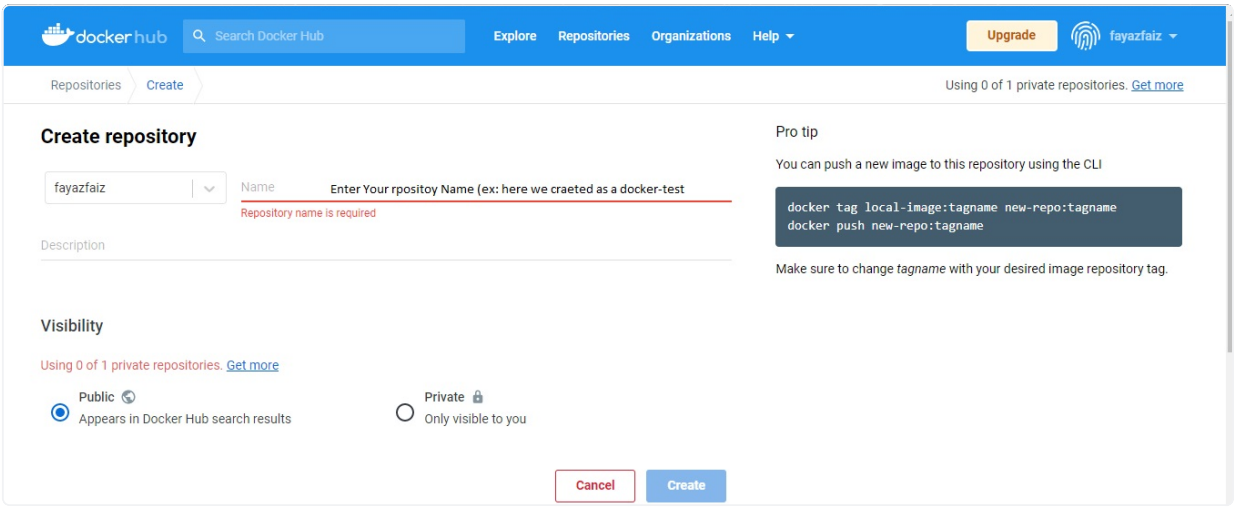
Apply and Save

*** Build Application with Jenkins Pipeline ***

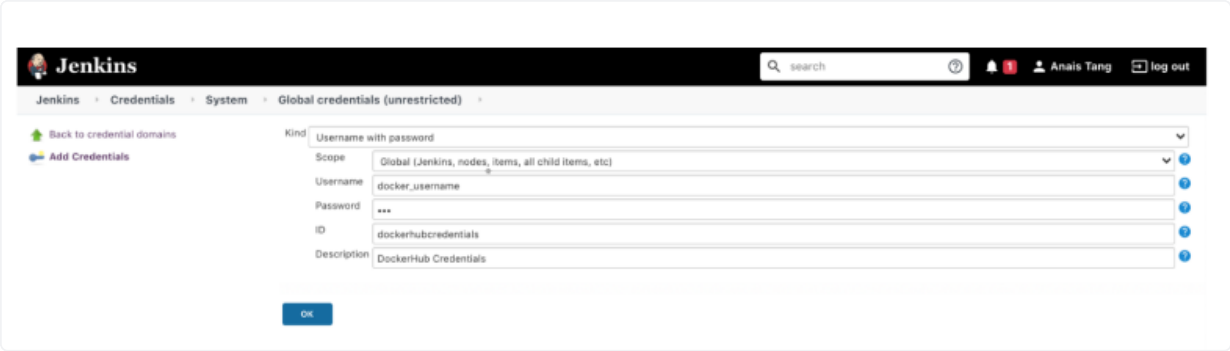
Integrate your application with Docker To create a Docker Image.

Before we create a new build, we require **two** elements:

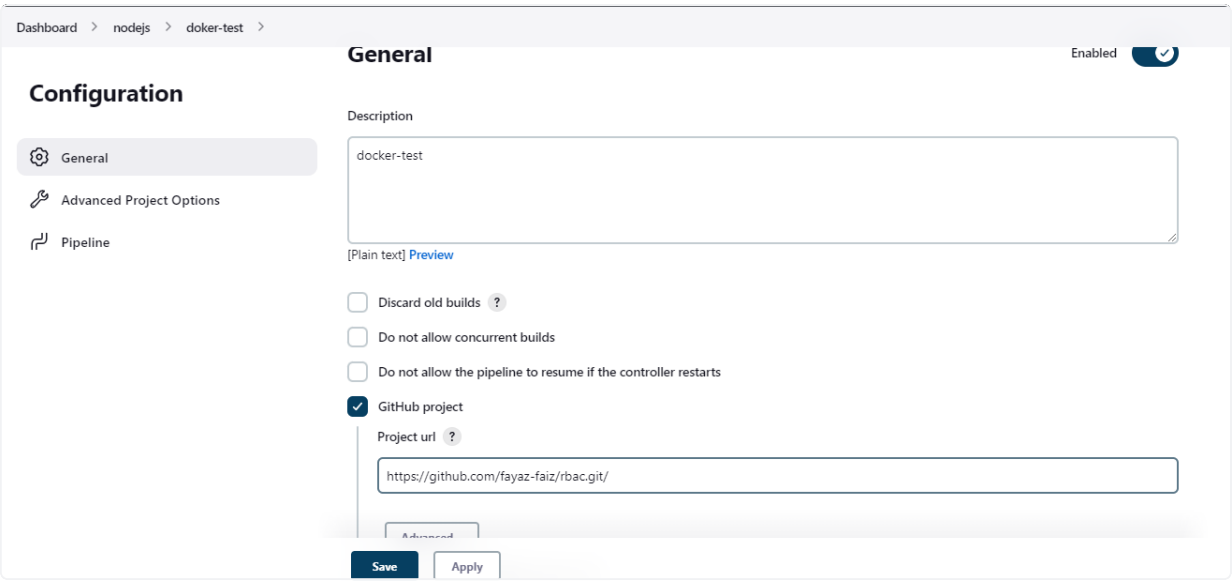
- Name of the Docker **Repository**
- Dockerhub **credentials** on Jenkins
- Create your Account by using this Link https://hub.docker.com/
- Open up a new tab and Login to your **Dockerhub** account. Click on **Create Repository,** enter a name and click **Create**. This will indicate to Jenkins **where** the Docker Image will be pushed.
-



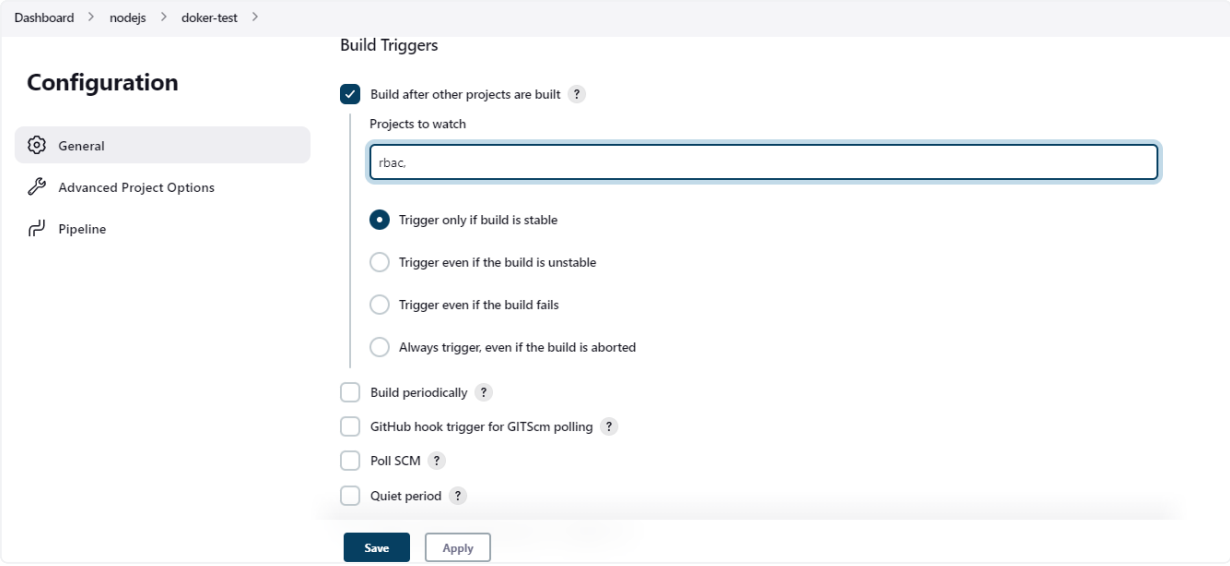
- In order for Jenkins to communicate with Dockerhub, we need to enter our credentials. Make sure the **Credentials** plugin is installed and navigate to **Manage Jenkins > Manage Credentials > System > Global Credentials > Add Credentials**.
- Enter your **Dockerhub credentials** including **username** and **password.** Assign it with an **ID** name called `dockerhub` or `dockerhubcredentials` and give it a memorable description.



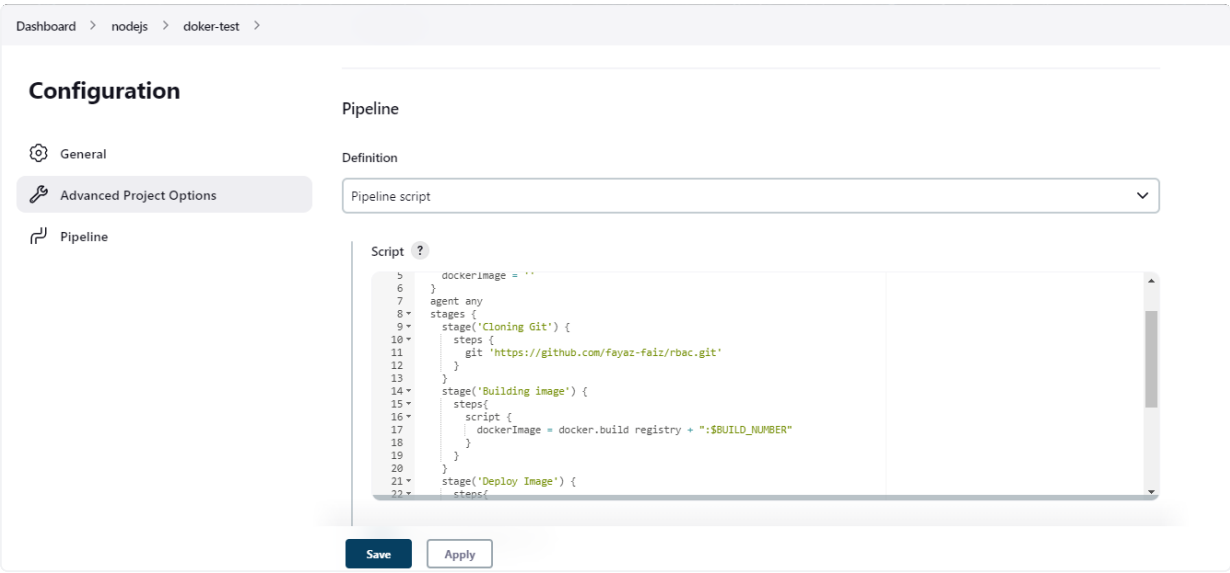
- Once saved, navigate back to the home page and click on **New Item.**
- Enter your **project name** and select **Pipeline** and click **OK**:
- In the **General Section,** enter the same project URL as the previous Jenkins build we configured earlier.
-



- In the **Build Triggers** section, select **Build after other projects is built** and select the **name** of the Continuous Integration build we configured earlier. In my case, the build is called `rbac`. This build will be automatically triggered once the freestyle project has passed the tests successfully. Next, select **Trigger only if the build is stable**.



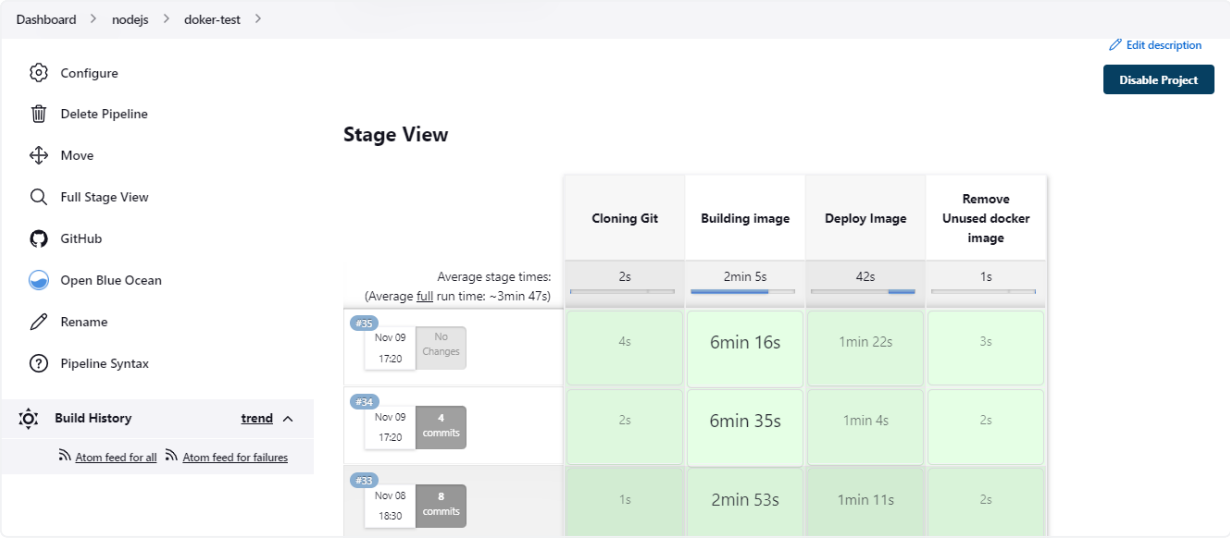
- In the **Pipeline** section, we will be creating a script to tell Jenkins to **create** a Docker Image and **deploy** it onto Dockerhub. The pipeline script is based on the **Apache Groovy** programming language.
-




```
//Pipeline Script
pipeline {
  environment {
    registry = "fayazfaiz/docker-test"   // This is docker repository name
    registryCredential = 'dockerhubcredentials'
    dockerImage = ''
  }
  agent any
```

```
    stages {
      stage('Cloning Git') {
        steps {
          git 'https://github.com/fayaz-faiz/rbac.git'
        }
      }
      stage('Building image') {
        steps{
          script {
            dockerImage = docker.build registry + ":$BUILD_NUMBER"
          }
        }
      }
      stage('Deploy Image') {
        steps{
          script {
            docker.withRegistry( '', registryCredential ) {
              dockerImage.push()
            }
          }
        }
      }
      stage('Remove Unused docker image') {
        steps{
          sh "docker rmi $registry:$BUILD_NUMBER"
        }
      }
    }
  }
```

- Apply and save and Click on **BUILD NOW**



Build now if success

- If You got any errors go to the output see the error and fix that then come back to Jenkins and start Bui now again.