Jenkins Documentation

Popular Jenkins plugins used in CI pipelines:

- **Git Plugin:** Integrates Jenkins with Git repositories for source code management.
- **Pipeline Plugin:** Enables Jenkins pipelines, defining complex build and deployment workflows as code.
- **Build Tools Plugins:** Provides support for specific build tools like Maven, Gradle, Ant, etc.
- **Artifact Repository Plugins:** Integrates with artifact repositories like Nexus or Artifactory to store and manage build artifacts.
- **JUnit Plugin:** Allows publishing and analyzing JUnit test reports.
- **Docker Plugin:** Integrates Jenkins with Docker for building and managing containerized applications.
- **AWS SDK Plugin:** Enables Jenkins to interact with Amazon Web Services (AWS) services.
- **Slack Plugin:** Sends notifications to Slack channels for build results and status updates.
- **Email Extension Plugin:** Sends email notifications with build results to team members.
- **SonarQube Scanner Plugin:** Integrates with SonarQube for code quality analysis.
- **Deploy to Container Plugin:** Allows deployment of applications to containers like Tomcat, Kubernetes, etc.
- **SSH Plugin:** Enables Jenkins to execute commands on remote machines over SSH.

Popular plugins used by Jenkins for test automation:

- **JUnit Plugin:** This plugin enables Jenkins to parse and display JUnit test results, making it easier to analyze and visualize test outcomes.
- **TestNG Plugin:** Integrates Jenkins with the TestNG testing framework, allowing the reporting and visualization of TestNG test results.
- **Cucumber Reports Plugin:** Enables Jenkins to generate and display reports for Cucumber BDD (Behavior-Driven Development) test results.
- **Selenium Plugin:** Integrates Jenkins with the Selenium automation testing framework, making it easier to execute web application tests.
- **Robot Framework Plugin:** Enables Jenkins to integrate with the Robot Framework, making it simpler to execute and report on acceptance test-driven development (ATDD) tests.
- **Behave Plugin:** Integrates Jenkins with Behave, a BDD test framework for Python, facilitating the execution and reporting of Behave tests.

- **JUnit Attachments Plugin:** Extends the JUnit Plugin to allow attachments to test results, useful for adding screenshots or log files for failed tests.
- **xUnit Plugin:** Allows Jenkins to work with various test result formats, such as JUnit, NUnit, MSTest, and others.
- **Test Result Trend Plugin:** Helps track and visualize historical test result trends over time.
- **Zephyr for Jira Test Management Plugin:** Integrates Jenkins with Jira and Zephyr for managing and executing test cases from Jenkins.

Static Application Security Testing (SAST) tools:

- **SonarQube:** SonarQube is a widely-used SAST tool that provides comprehensive code analysis for various programming languages, detecting security vulnerabilities, bugs, and code smells.
- **Checkmarx:** Checkmarx is a powerful SAST tool that scans source code for security vulnerabilities in different programming languages, helping to identify and fix potential threats.
- **Fortify:** Fortify is another popular SAST tool that performs static analysis on source code to detect and prioritize security vulnerabilities.
- **Veracode:** Veracode offers a SAST solution that identifies security vulnerabilities and provides insights into code quality.
- **PMD:** PMD is a source code analyzer that focuses on finding common programming flaws, potential bugs, and performance issues in Java and other languages.
- **FindBugs:** FindBugs is a static code analyzer that detects possible bugs in Java programs.
- **ESLint:** ESLint is a popular SAST tool for JavaScript code, helping to enforce coding standards and identify potential security issues.
- **Bandit:** Bandit is a security SAST tool specifically designed for Python applications, searching for common security issues in Python code.
- **Brakeman:** Brakeman is a security scanner for Ruby on Rails applications, focusing on identifying security vulnerabilities in Ruby code.
- **SpotBugs:** SpotBugs (formerly known as FindBugs) is a static analysis tool for Java that detects bugs in Java code.

DAST (Dynamic Application Security Testing) tools:

- **OWASP ZAP (Zed Attack Proxy):** OWASP ZAP is an open-source DAST tool used for finding vulnerabilities in web applications. It provides an API that can be integrated into CI pipelines to automate security scanning.

- **Burp Suite:** Although Burp Suite is mainly known as a manual security testing tool, its Professional edition offers automated scanning capabilities that can be integrated into CI pipelines.
- **Acunetix:** Acunetix is a web vulnerability scanner that can automatically detect and report security vulnerabilities in web applications. It provides integrations with CI/CD tools like Jenkins, Azure DevOps, and GitLab CI.
- **Netsparker:** Netsparker is an automated web application security scanner that can be integrated into CI pipelines to detect various vulnerabilities.
- **AppSpider:** AppSpider, by Rapid7, is a DAST tool designed to scan web applications for security vulnerabilities. It offers automation options for CI/CD integration.
- **Qualys Web Application Scanning (WAS):** Qualys WAS is a cloud-based DAST tool that can be used to automate web application security testing in CI pipelines.
- **Tenable.io Web Application Scanning:** Tenable.io provides DAST capabilities to scan web applications for vulnerabilities, which can be integrated into CI/CD pipelines.

Examples of Web Apps that DAST tools like AppSpider can test:

Some of the environments and technologies that AppSpider can test include:

- **Web Applications:** AppSpider can scan and test traditional web applications, including those built using PHP, Java, .NET, Ruby on Rails, and other common web development languages.
- **Single-page Applications (SPAs):** AppSpider can handle Single-page Applications built with modern JavaScript frameworks like Angular, React, and Vue.js.
- **RESTful APIs:** It supports testing RESTful APIs, which have become a crucial part of many web applications and services.
- **Web Services:** AppSpider can scan and test SOAP and other types of web services.
- **Web Servers:** It can scan web applications hosted on various web servers like Apache, Nginx, Microsoft IIS, etc.
- **Content Management Systems (CMS):** AppSpider can test web applications built on popular CMS platforms like WordPress, Joomla, Drupal, etc.
- **Authentication Mechanisms:** It can test web applications that use various authentication mechanisms, including basic authentication, form-based authentication, OAuth, and more.

In a CI pipeline, the typical types of automated tests that are executed to catch bugs or identify potential issues are:

- **Unit Tests:** These tests check individual units or components of the software in isolation to ensure they function correctly.

- **Integration Tests:** Integration tests verify that different units or components of the system work together as expected.
- **Functional Tests:** Functional tests validate the behavior of the software from the end-user's perspective, ensuring it meets the specified requirements.
- **Regression Tests:** As mentioned earlier, regression tests ensure that new code changes do not introduce unintended bugs or regressions in the existing functionality.
- **Performance Tests:** Performance tests assess the software's responsiveness and scalability under various conditions.
- **Security Tests:** Security tests examine the software for potential vulnerabilities and weaknesses.

Installing Jenkins

Setup Java

```
Unset
sudo apt install openjdk-17-jdk

#set the JAVA_HOME path
sudo nano /etc/profile
export JAVA_HOME=/usr/lib/jvm/java-1.17.0-openjdk-amd64
export PATH=$JAVA_HOME/bin:$PATH

source /etc/profile

echo $JAVA_HOME
/usr/lib/jvm/java-1.17.0-openjdk-amd64
```

Setup Jenkins

```
Unset
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo
tee \
    /usr/share/keyrings/jenkins-keyring.asc > /dev/null
```

```
Unset
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
    https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
    /etc/apt/sources.list.d/jenkins.list > /dev/null
```

```
Unset
sudo apt-get update
sudo apt-get install jenkins
sudo systemctl status jenkins
sudo systemctl enable --now jenkins
```

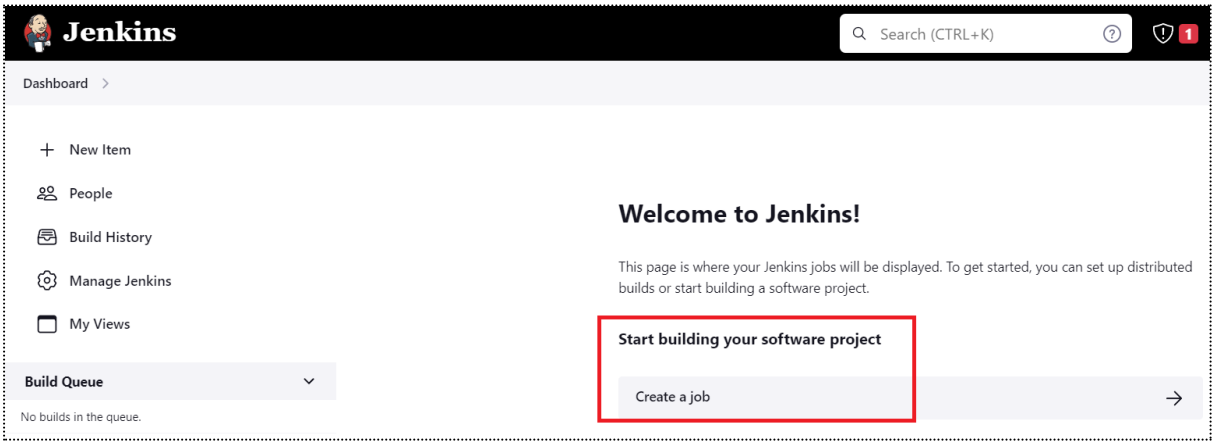
http://ip_address_or_domain:8080

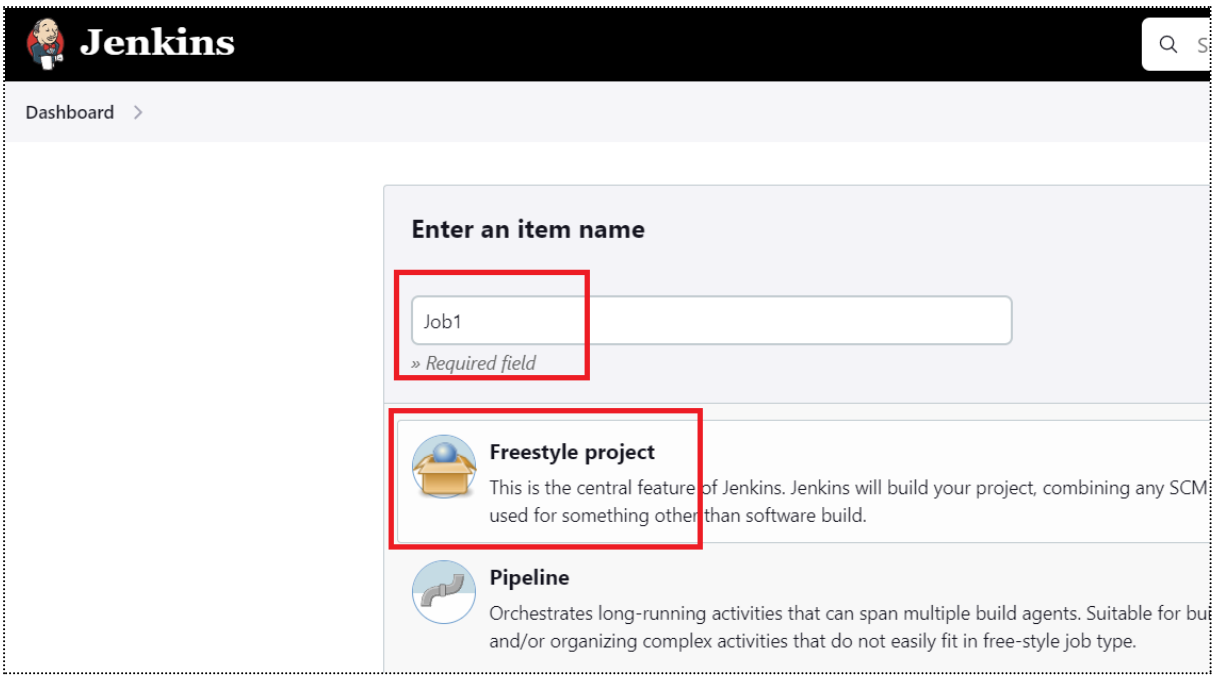
Follow the onscreen to complete the setup!

Create the first JOB in Jenkins

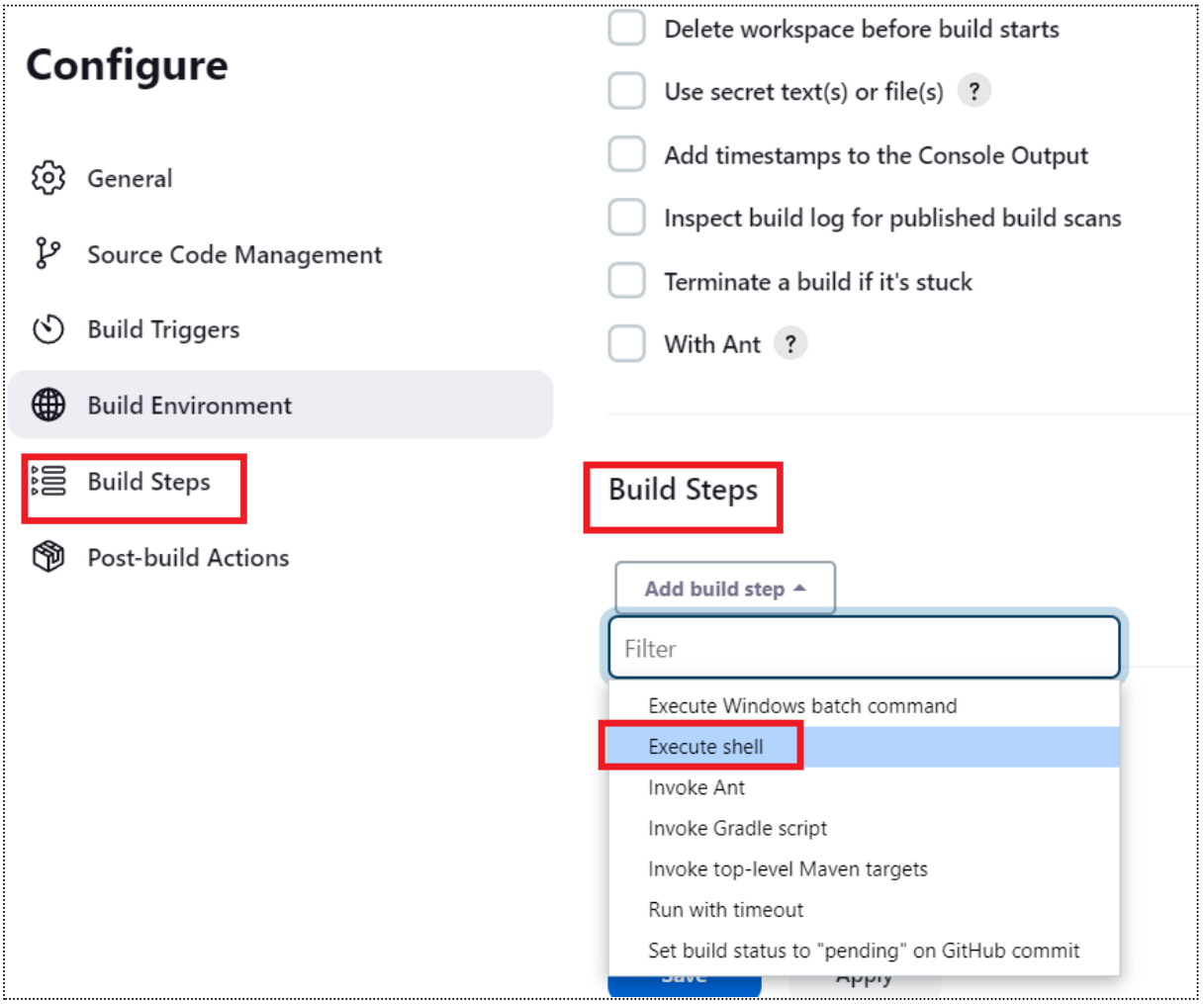
A JOB is an ACTION that Jenkin performs like Deploy job, Test job etc.,

Jenkins calls the JOBS as ITEMS or PROJECTS.



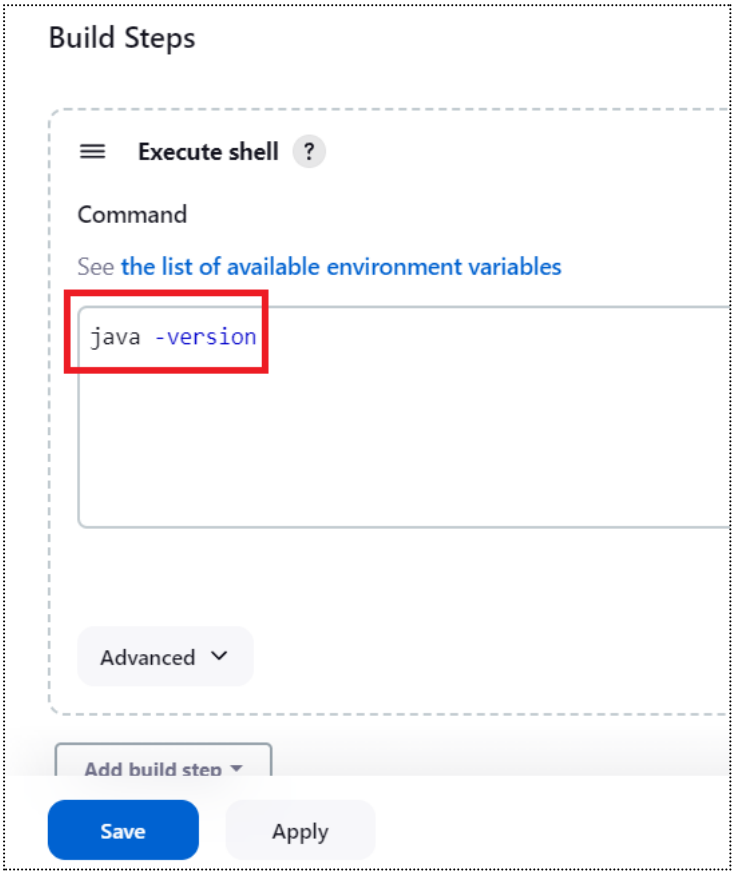


Select Build Steps > Exdcute Shell:

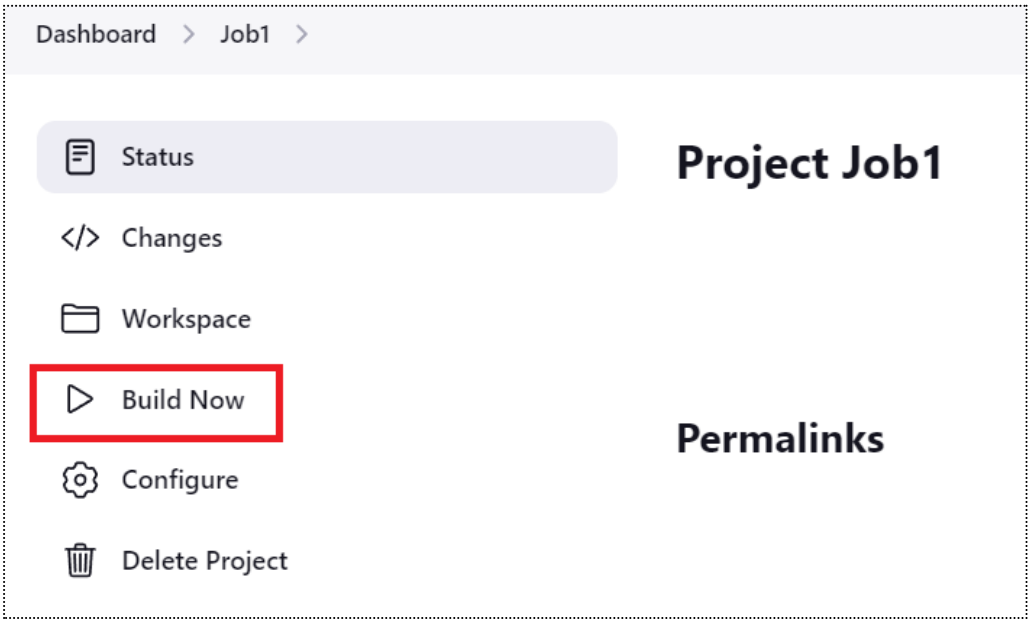


Execute Shell, can run SHELL Commands on the Linux Server where Jenkins is installed.

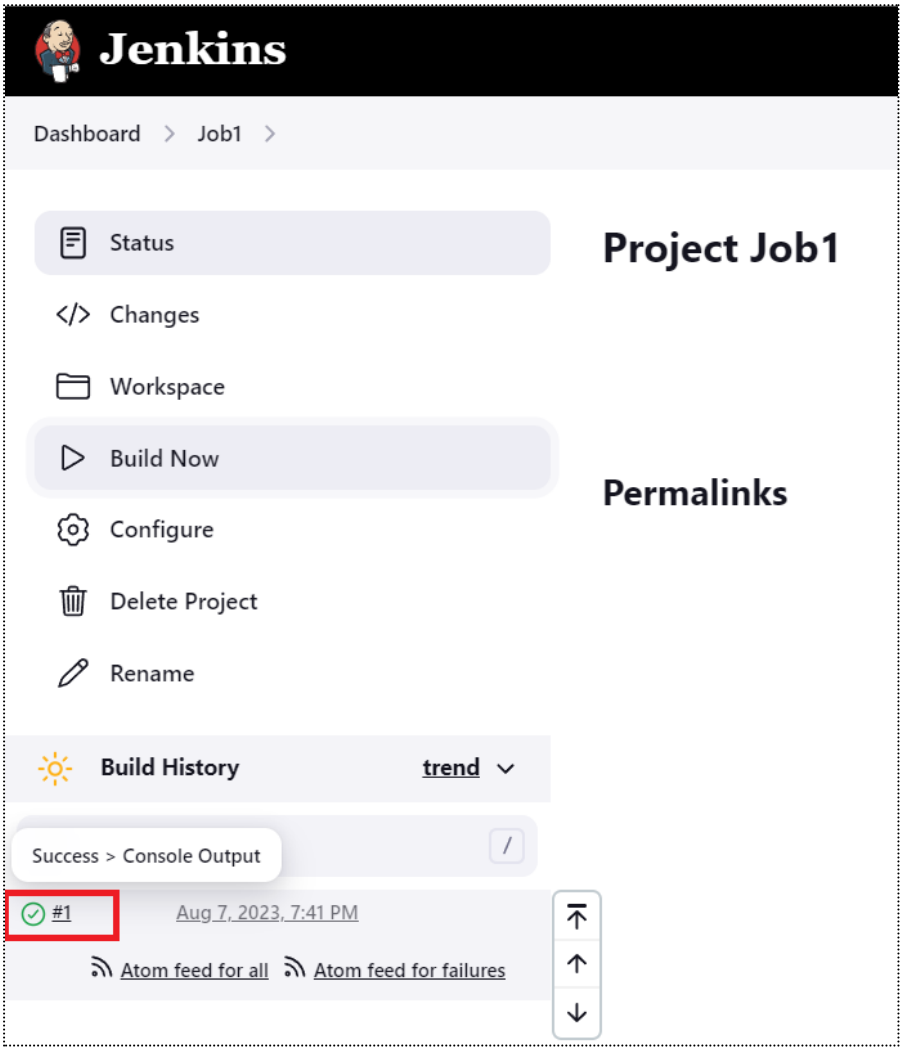
Do as shown and click Save:

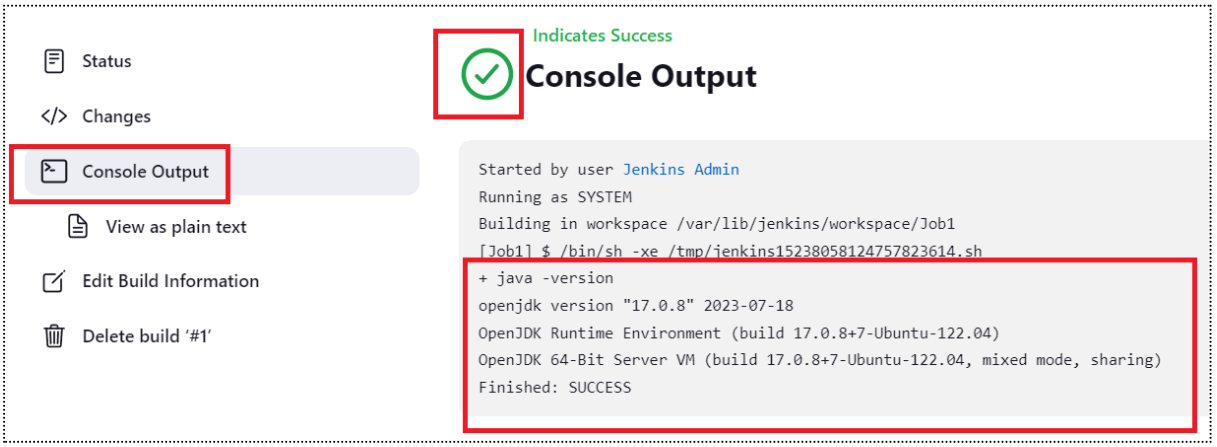


Click Build Now:



Click this:





The Build Succeeded!

Notifications in jenkins

- Failed Jobs have to be informed to respective teams.
- This is called FEEDBACK or REPORTING.
- Jenkins sends Notifications in the form of e-mails.
- Jenkins uses an SMTP Server to send the e-mails.

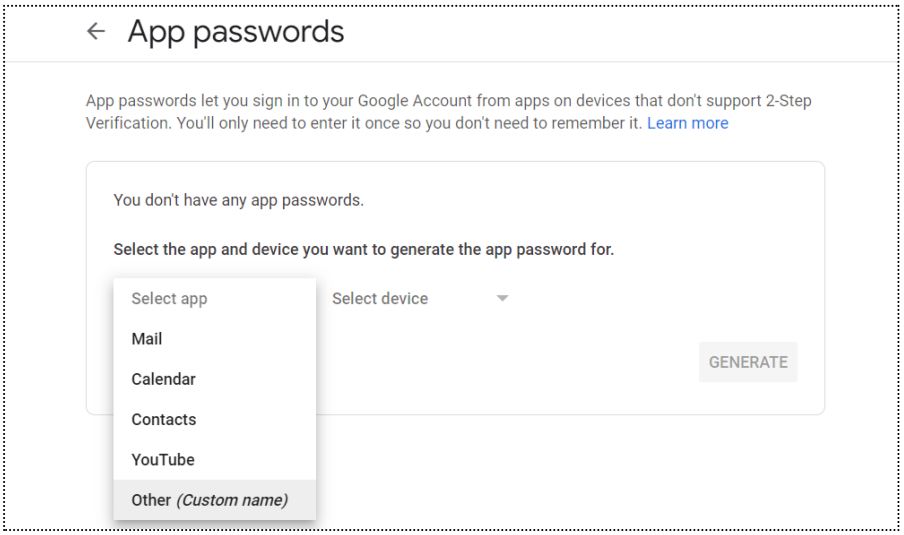
<u>Using GMAIL as SMTP Server:</u>

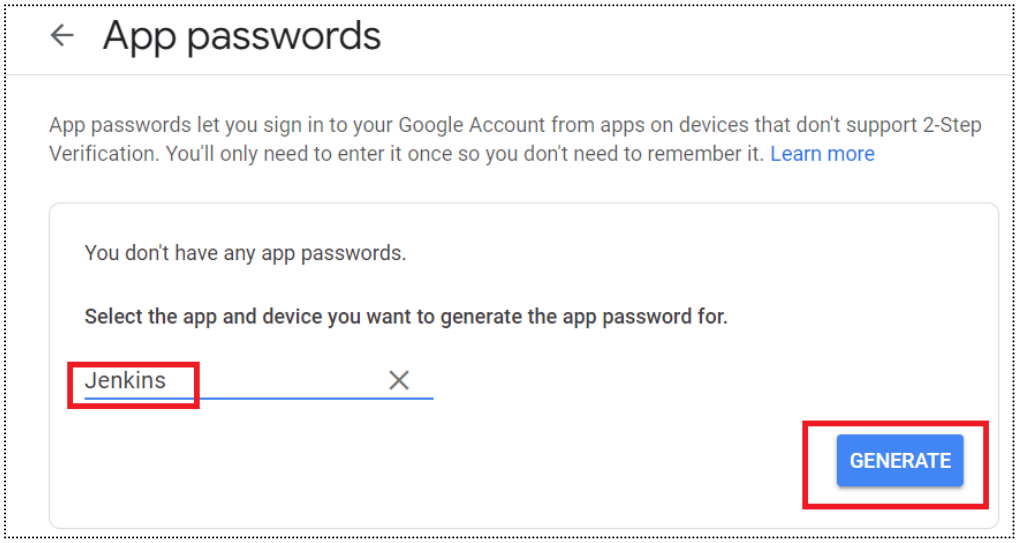
Turn on 2-Step Verification

- Open your Google Account.
- In the navigation panel, select Security.
- Under “Signing in to Google,” select 2-Step Verification.

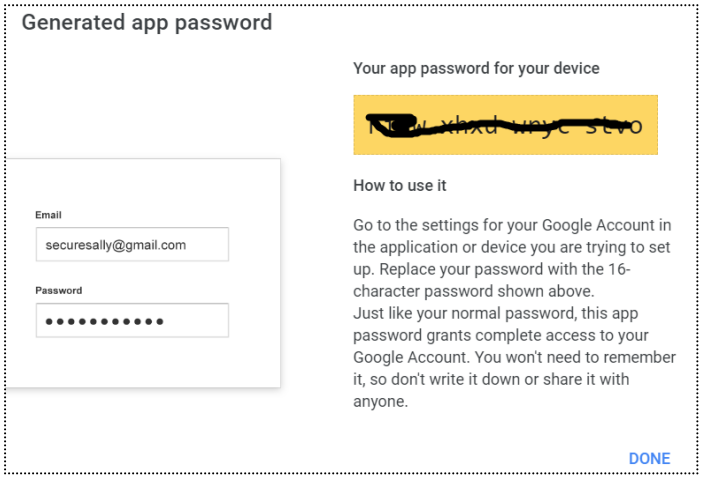
Go to App Apsswords: https://myaccount.google.com/apppasswords

In the drop down select Other(custom name)

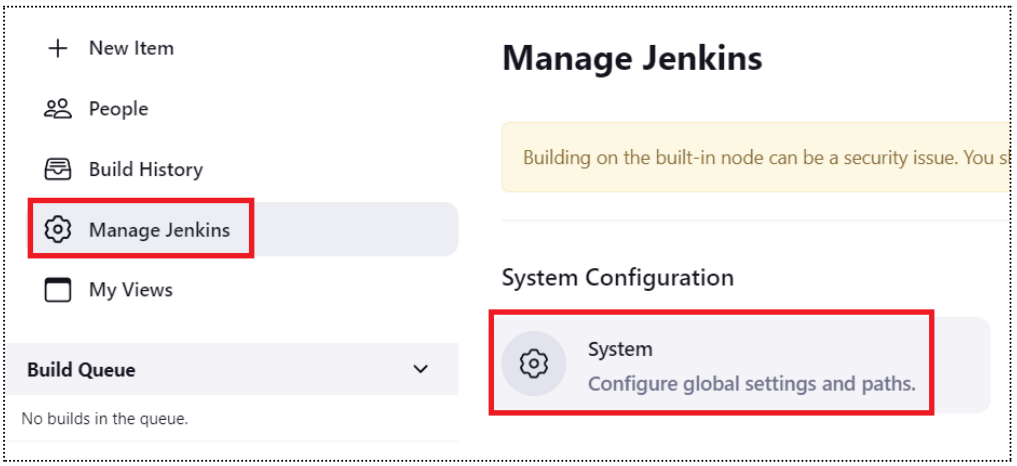




Note the Password Generated:

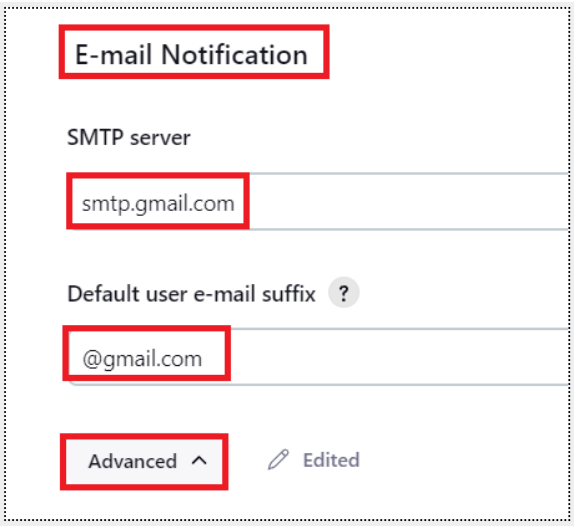


Setup SMTP configuration in jenkins for using the gmail



Go to Jenkins >> configure system >> scroll down to email notification section

Enter the following parameters:
smtp server : smtp.gmail.com
default user email suffix : @gmail.com

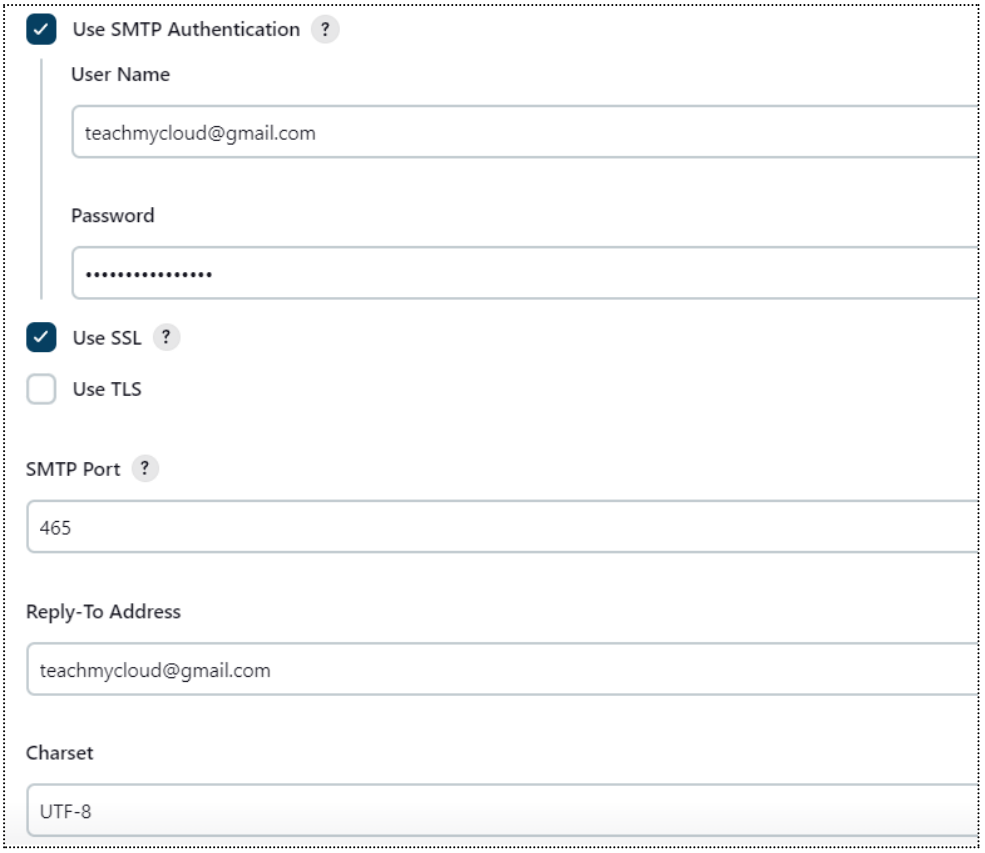


select **advanced**
check **smtp authentication**

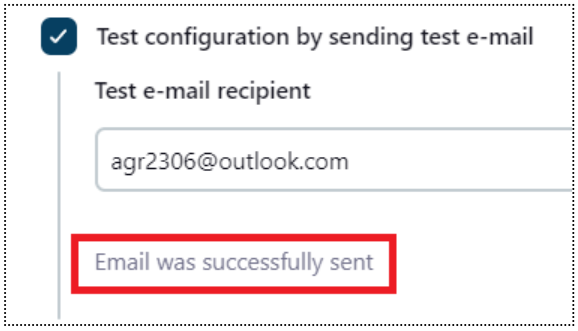
username : (Your gmail id)
password : (application specific password generated from previous step)

check **use SSL**

SMTP port : **465**
Reply to address : (Your gmail id) *(optional)*
Charset : *UTF-8 (by default it is UTF-8)*



select **Test configuration mail**

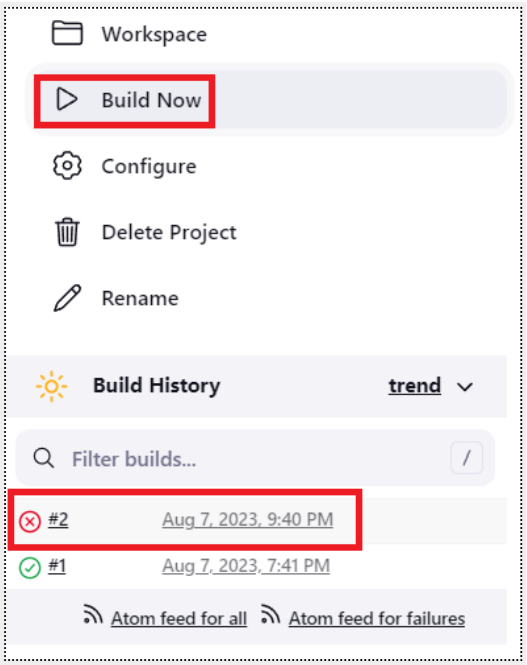


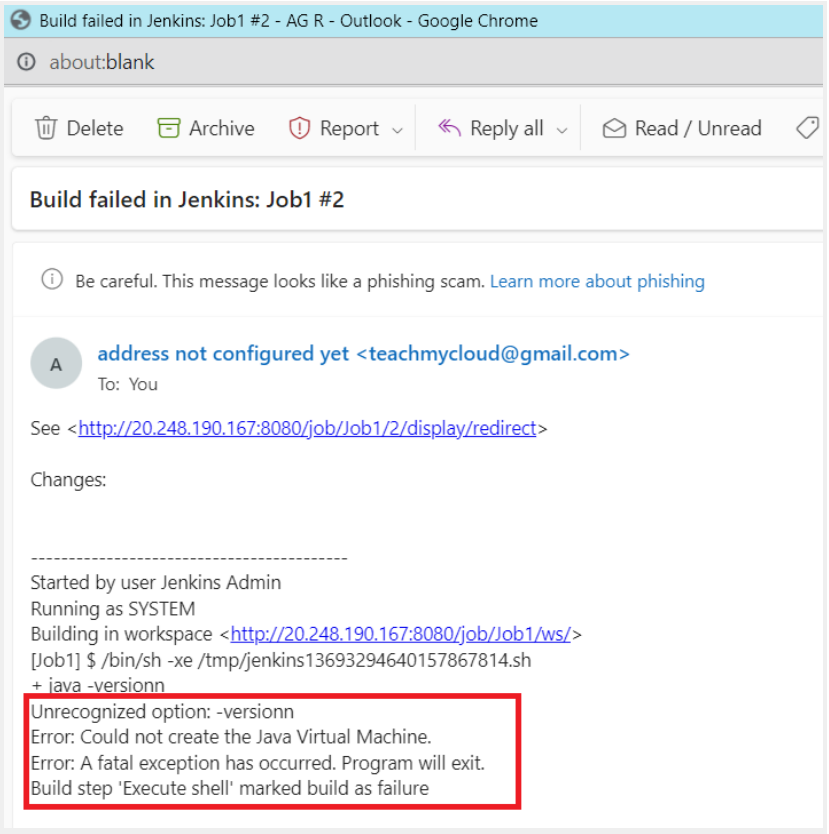
Verify Email Notifications

In jenkins configure the Job1 we created earlier as shown and re-run the job:

In the Build Steps change the code to: java -versionn

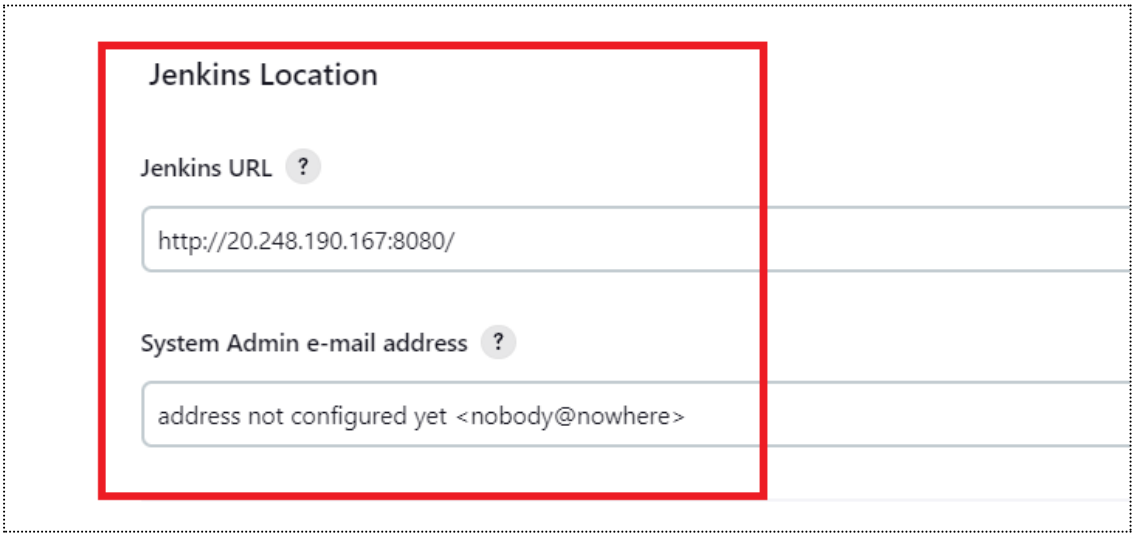
You see that the Build Job fails!





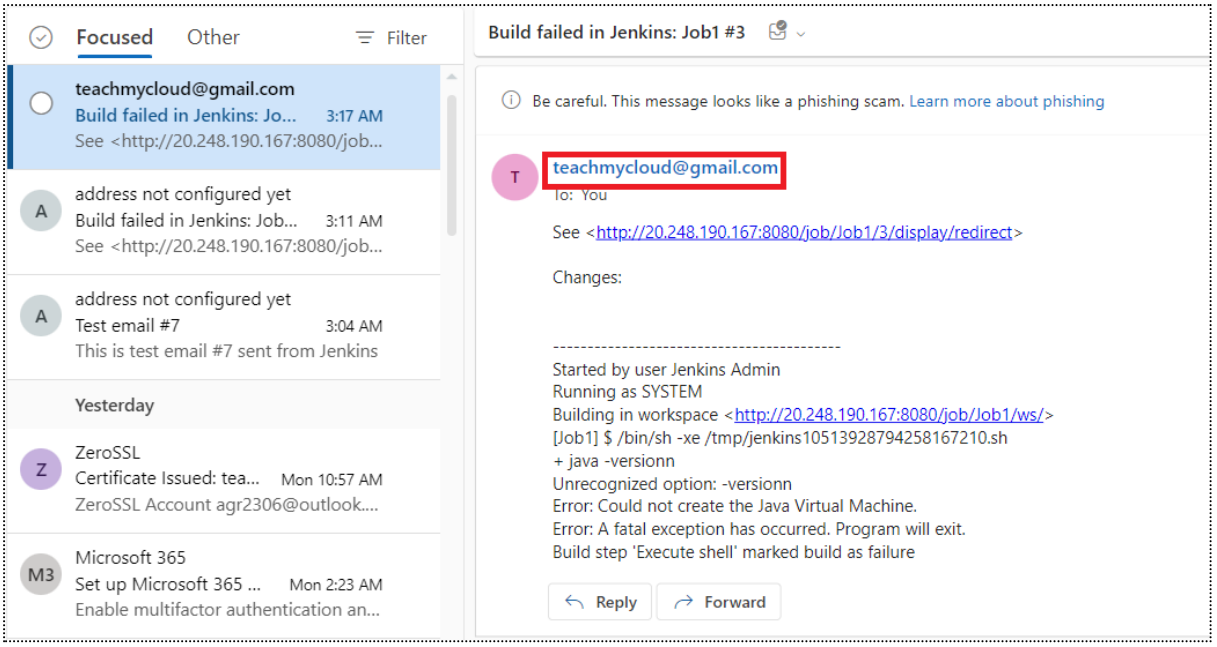
Observer that the emal address from Jenkins shows as:
address not configured yet <teachmycloud@gmail.com>

To clarify that use this setting: Manage Jenkins -> Configure System -> Jenkins Location -> System Admin e-mail address



Change it to your email, as in my case: teachmycloud@gmail.com

Rebuild and Verify!


Focused
Other
Filter
teachmycloud@gmail.com
Build failed in Jenkins: Jo...
3:17 AM
See <http://20.248.190.167:8080/job...
address not configured yet
Build failed in Jenkins: Job...
3:11 AM
See <http://20.248.190.167:8080/job...
address not configured yet
Test email #7
3:04 AM
This is test email #7 sent from Jenkins
Yesterday
ZeroSSL
Certificate Issued: tea...
Mon 10:57 AM
ZeroSSL Account agr2306@outlook....
Microsoft 365
Set up Microsoft 365 ...
Mon 2:23 AM
Enable multifactor authentication an...
Build failed in Jenkins: Job1 #3
Be careful. This message looks like a phishing scam. Learn more about phishing
teachmycloud@gmail.com
To: You
See <http://20.248.190.167:8080/job/Job1/3/display/redirect>
Changes:

Started by user Jenkins Admin
Running as SYSTEM
Building in workspace <http://20.248.190.167:8080/job/Job1/ws/>
[Job1] $ /bin/sh -xe /tmp/jenkins10513928794258167210.sh
+ java -versionn
Unrecognized option: -versionn
Error: Could not create the Java Virtual Machine.
Error: A fatal exception has occurred. Program will exit.
Build step 'Execute shell' marked build as failure
Reply
Forward