

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
```

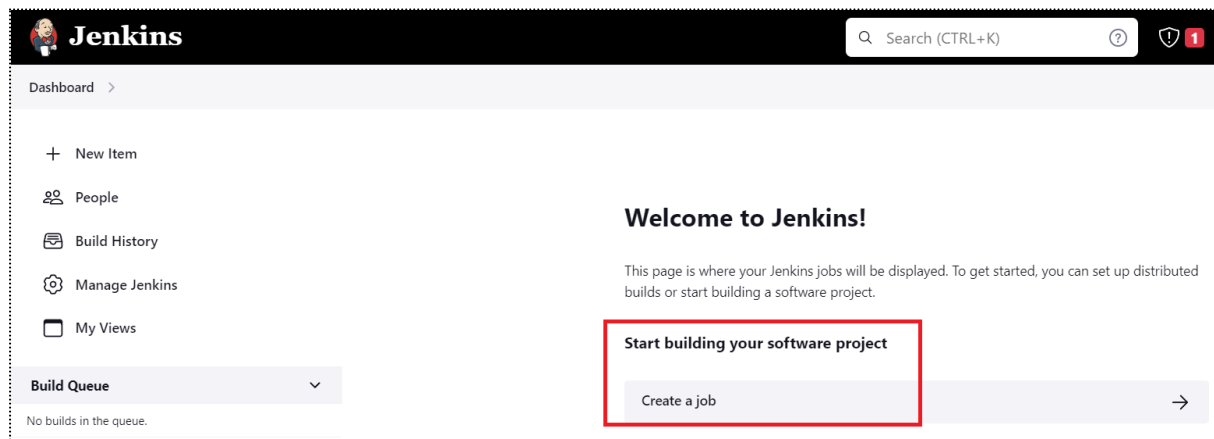
http://ip_address_or_domain:8080


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.




 **Jenkins** Q S

Dashboard >


Enter an item name

Job1

» Required field




Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM used for something other than software build.





Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building and/or organizing complex activities that do not easily fit in free-style job type.


Select Build Steps > Exdcute Shell:


Configure


 General

 Source Code Management

 Build Triggers

 Build Environment

 **Build Steps**

 Post-build Actions

☐ Delete workspace before build starts

☐ Use secret text(s) or file(s) ?

☐ Add timestamps to the Console Output

☐ Inspect build log for published build scans

☐ Terminate a build if it's stuck

☐ With Ant ?

Build Steps

Add build step ^

Filter

Execute Windows batch command

Execute shell

Invoke Ant

Invoke Gradle script

Invoke top-level Maven targets

Run with timeout

Set build status to "pending" on GitHub commit

Save Apply

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

Do as shown and click Save:

Build Steps

≡ Execute shell ?

Command

See [the list of available environment variables](#)

`java -version`

Advanced ▾

Add build step ▾

Save

Apply

Click Build Now:

Dashboard > Job1 >

☰ Status

</> Changes

📁 Workspace

▶ Build Now

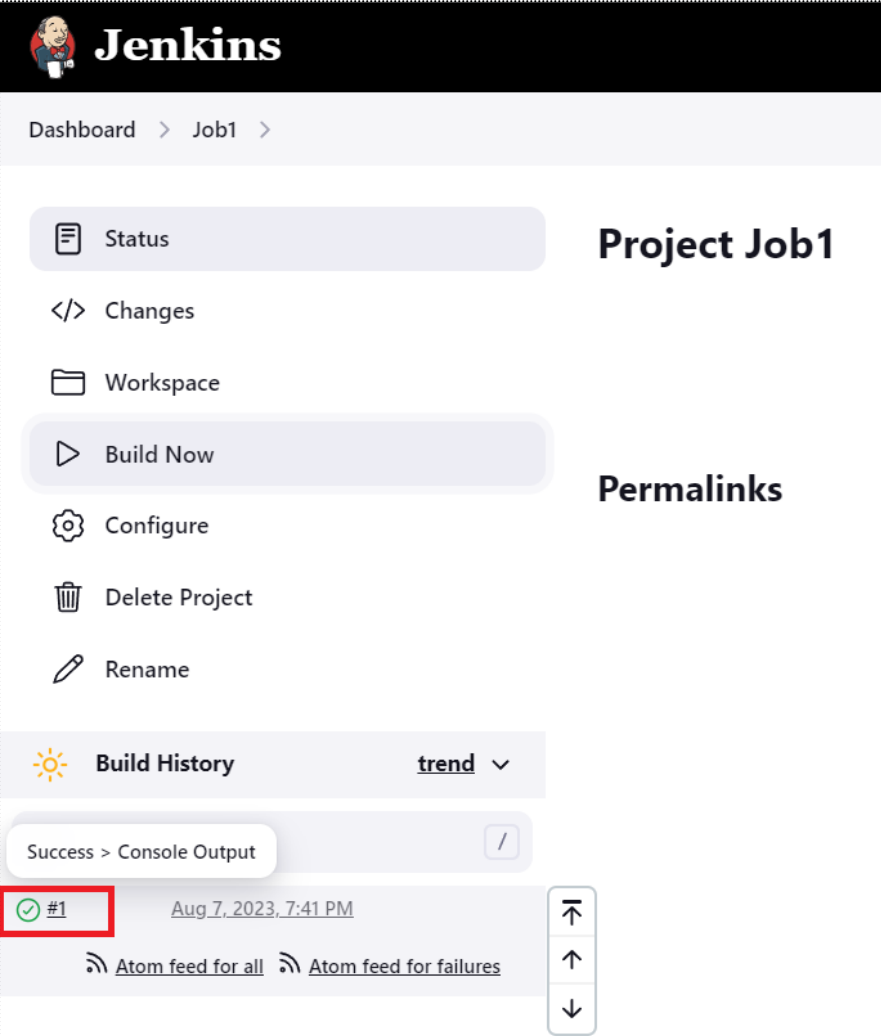
⚙️ Configure

🗑️ Delete Project

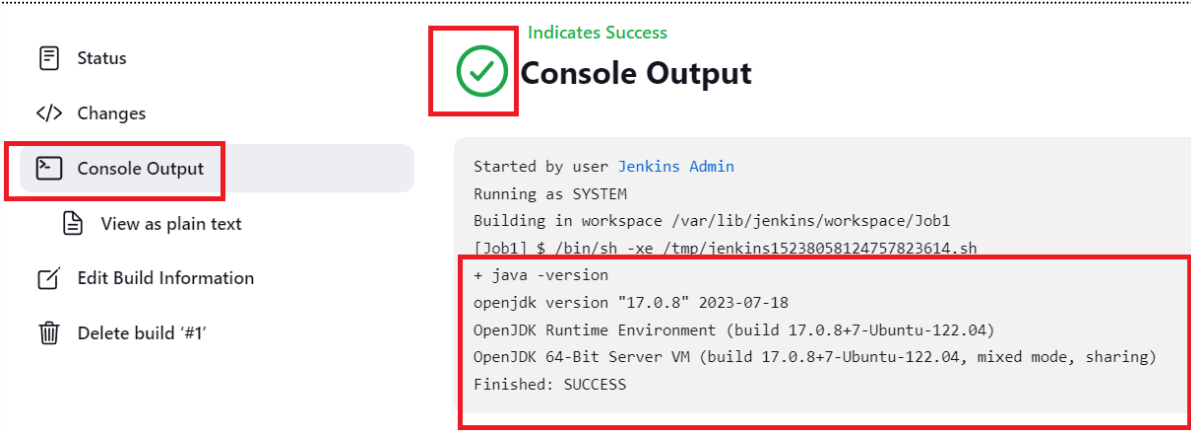
Project Job1

Permalinks

Click this:



The Jenkins dashboard for 'Project Job1' shows a sidebar with navigation links: Status, Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main area displays 'Permalinks' and a 'Build History' section. The 'Build History' section shows a single build, '#1', which is highlighted with a red box. The build status is 'Success' and the console output is visible. The build was started on 'Aug 7, 2023, 7:41 PM'.



The 'Console Output' view for build '#1' is shown. The 'Console Output' link in the sidebar is highlighted with a red box. The main area displays the console output, which is also highlighted with a red box. The output shows the build was started by 'Jenkins Admin', running as 'SYSTEM', and successfully completed. The console output text is as follows:

```
Started by user Jenkins Admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Job1
[Job1] $ /bin/sh -xe /tmp/jenkins15238058124757823614.sh
+ java -version
openjdk version "17.0.8" 2023-07-18
OpenJDK Runtime Environment (build 17.0.8+7-Ubuntu-122.04)
OpenJDK 64-Bit Server VM (build 17.0.8+7-Ubuntu-122.04, mixed mode, sharing)
Finished: SUCCESS
```

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.

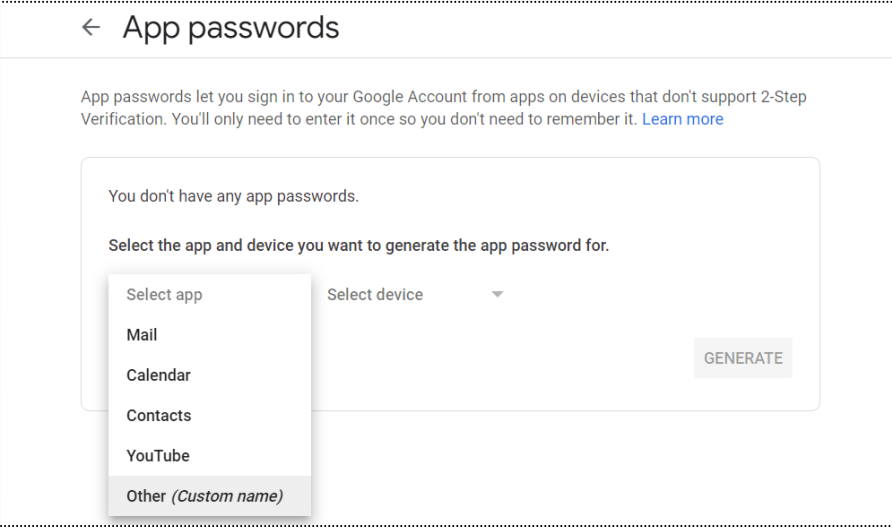
Using GMAIL as SMTP Server:

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 Appsswords: <https://myaccount.google.com/apppasswords>

In the drop down select Other(custom name)



← App passwords

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)

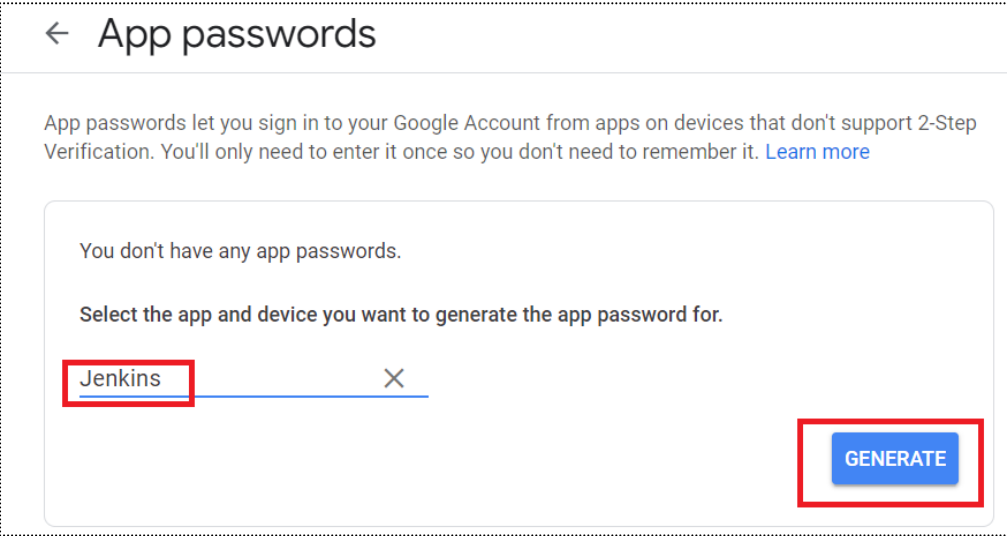
You don't have any app passwords.

Select the app and device you want to generate the app password for.

Select app: Mail, Calendar, Contacts, YouTube, **Other (Custom name)**

Select device: ▼

GENERATE



← App passwords

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)

You don't have any app passwords.

Select the app and device you want to generate the app password for.

Jenkins X

GENERATE

Note the Password Generated:

Generated app password

Your app password for your device

16-character app password

How to use it

Go to the settings for your Google Account in the application or device you are trying to set up. Replace your password with the 16-character password shown above. Just like your normal password, this app password grants complete access to your Google Account. You won't need to remember it, so don't write it down or share it with anyone.

[DONE](#)

Email

securesally@gmail.com

Password

••••••••••

Setup SMTP configuration in jenkins for using the gmail

+ New Item

People

Build History

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Manage Jenkins

Building on the built-in node can be a security issue. You should consider using a separate node for building.

System Configuration

System

Configure global settings and paths.

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

E-mail Notification

SMTP server

smtp.gmail.com

Default user e-mail suffix ?

@gmail.com

Advanced ^

Edited

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

username : (Your gmail id)

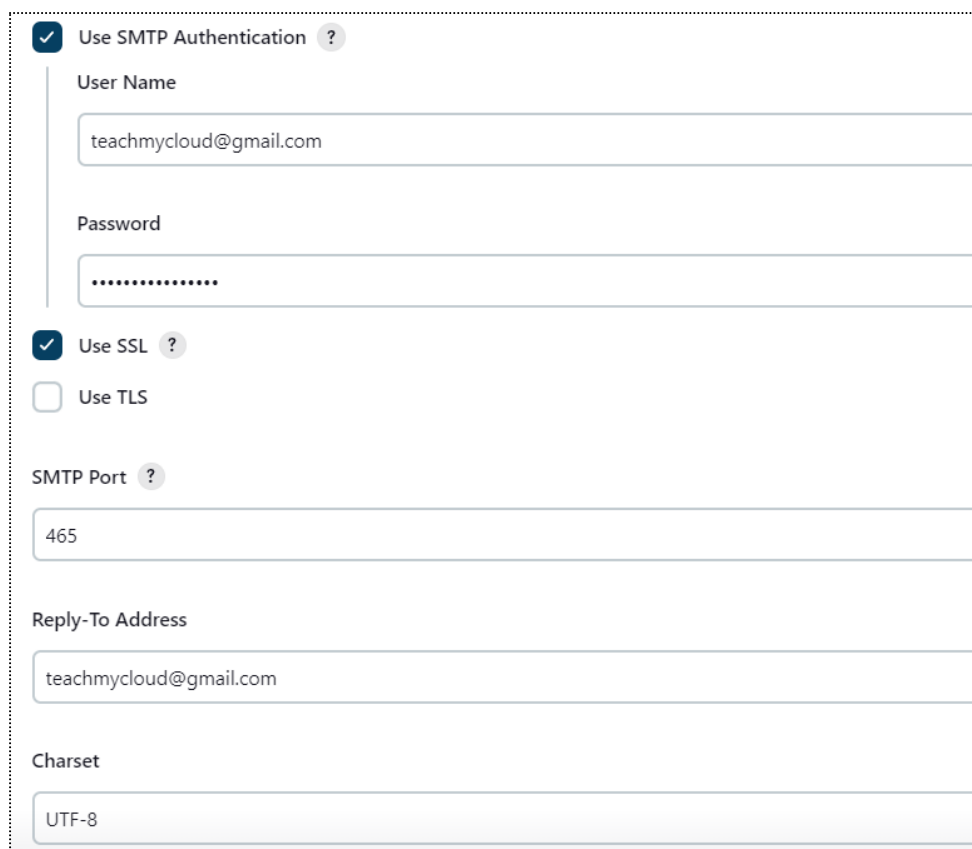
password : (application specific password generated from previous step)

check **use SSL**

SMTP port : **465**

Reply to address : (Your gmail id) *(optional)*

Charset : *UTF-8 (by default it is UTF-8)*



The screenshot shows a configuration form for SMTP settings. It includes a section for 'Use SMTP Authentication' which is checked, with fields for 'User Name' (teachmycloud@gmail.com) and 'Password' (masked with dots). Below this is a section for 'Use SSL' (checked) and 'Use TLS' (unchecked). The 'SMTP Port' is set to 465. The 'Reply-To Address' is teachmycloud@gmail.com. The 'Charset' is set to UTF-8.

☒ Use SMTP Authentication ?

User Name

teachmycloud@gmail.com

Password

.....

☒ Use SSL ?

☐ Use TLS

SMTP Port ?

465

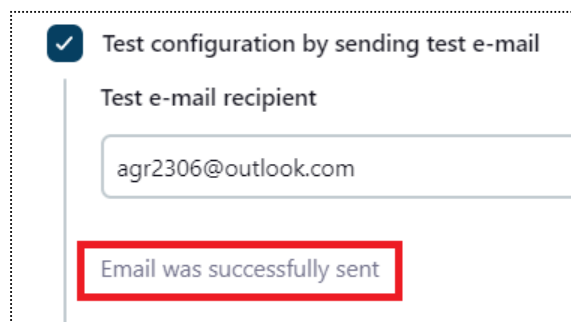
Reply-To Address

teachmycloud@gmail.com

Charset

UTF-8

select **Test configuration mail**



The screenshot shows the 'Test configuration mail' section. It is checked, and the 'Test e-mail recipient' is agr2306@outlook.com. A red box highlights the message 'Email was successfully sent'.

☒ Test configuration by sending test e-mail

Test e-mail recipient



agr2306@outlook.com

Email was successfully sent

Verify Email Notifications

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


Post-build Actions

 **E-mail Notification** 

Recipients

Whitespace-separated list of recipient addresses. May reference build parameters like \$PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.


☒ Send e-mail for every unstable build





☒ Send separate e-mails to individuals who broke the build


In the Build Steps change the code to: `java -version`


You see that the Build Job fails!


 Workspace



 **Build Now**



 Configure

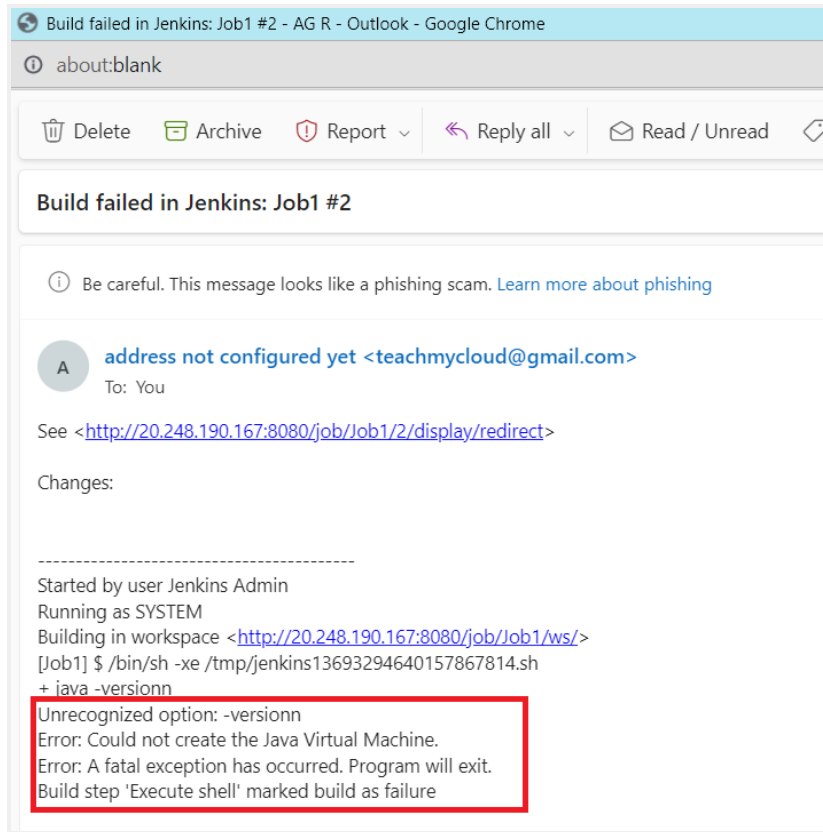
 Delete Project

 Rename

 **Build History** trend ▾

| | |
|--|--------------------------------------|
|  #2 | Aug 7, 2023, 9:40 PM |
|  #1 | Aug 7, 2023, 7:41 PM |

 [Atom feed for all](#)  [Atom feed for failures](#)



Observer that the email 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...

A

address not configured yet

Build failed in Jenkins: Job... 3:11 AM

See <http://20.248.190.167:8080/job...

A

address not configured yet

Test email #7 3:04 AM

This is test email #7 sent from Jenkins

Yesterday

Z

ZeroSSL

Certificate Issued: tea... Mon 10:57 AM

ZeroSSL Account agr2306@outlook....

M3

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](#)

T

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 -versionnn

Unrecognized option: -versionnn

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