ATTACK DEFENSE

by PentesterAcademy

Name	Fix the App: Java WebApp
URL	https://attackdefense.com/challengedetails?cid=353
Type	Fix the Code : Web Applications

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Challenge Description

DevOps practices are to combine software development (Dev) and IT operations (Ops) in order to improve the delivery process. DevOps pipelines are chained tasks and components that run in a sequence to cover different phases of software compilation, packaging, automated testing, and test deployment.

In this lab, we have a DevSecOps pipeline for a Maven web application. The pipeline consists of the following components (and tasks):

- VS Code Server (For pulling, modifying, and pushing the code)
- GitLab server (For hosting code)
- Jenkins server (For integrating). Different phases and components used:
    - Build: Maven
    - Code testing: Maven
    - Test Deployment: Ansible
    - Dynamic Testing: Selenium
- Test server (For test deployment)
- Archery Sec server (For Vulnerability management)

It is suggested to play the DevOps focused lab before playing this lab.

DevSecOps refer to introducing security in different stages of the DevOps process. This is done to catch the vulnerabilities/insecurities as soon as possible in the pipeline. In this lab, the pipeline consists of the following components (and tasks):

● Automated Code Review:	DevSkim
● Sensitive Information Scan phase:	Detect Secrets
● Software Component Analysis:	OWASP Dependency-Check
● Static Code Analysis:	FindSecBugs
● Dynamic Application Security Testing:	OWASP ZAP
● Compliance as Code:	Inspec

Objective: Fix the Issues in the stages of the pipeline and Find the flags!

Instructions:

- The GitLab server is reachable with the name 'gitlab'
- Gitlab credentials:

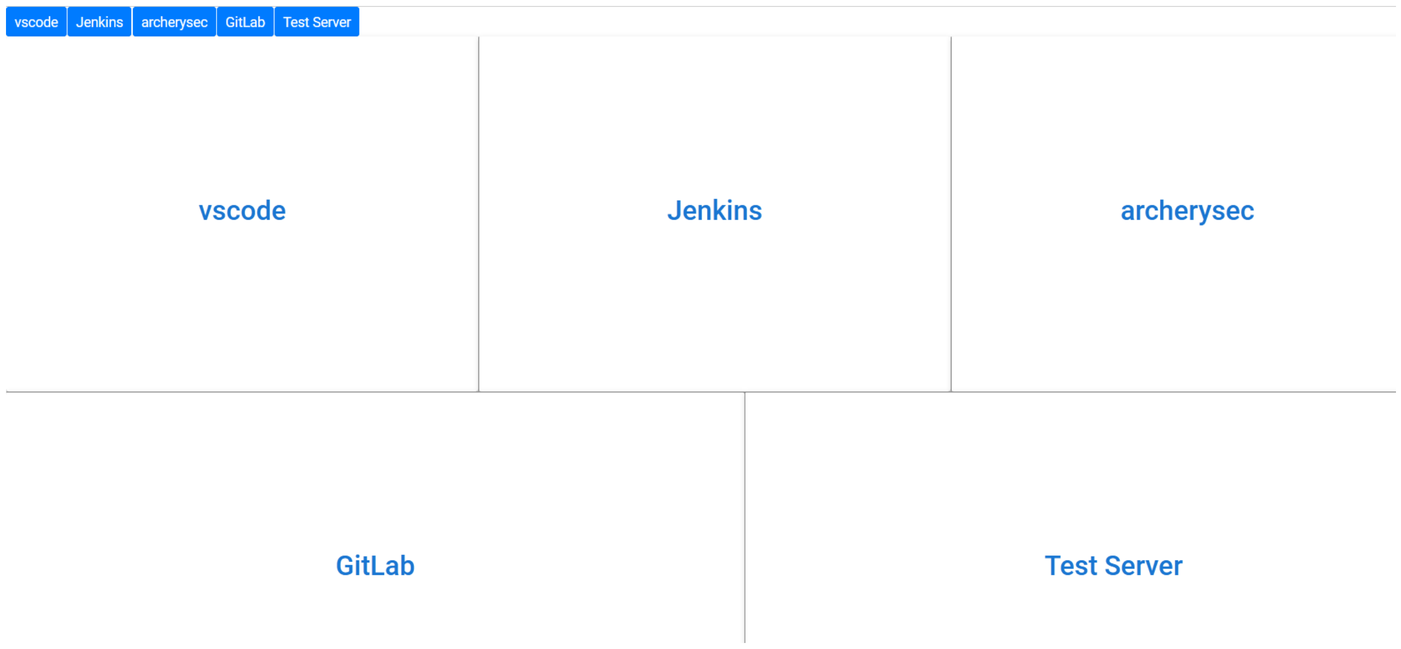
Username	**Password**
root	welcome123

- The Archery server is reachable by the name "archerysec"
- ArcherySec credentials:

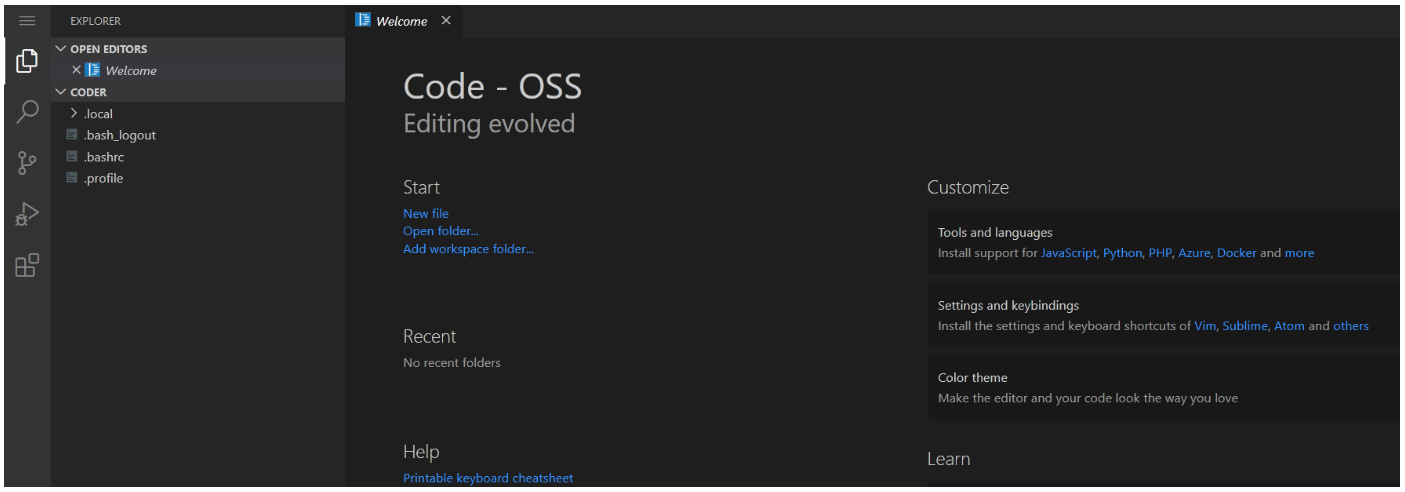
Username	**Password**
admin	admin

Lab Setup

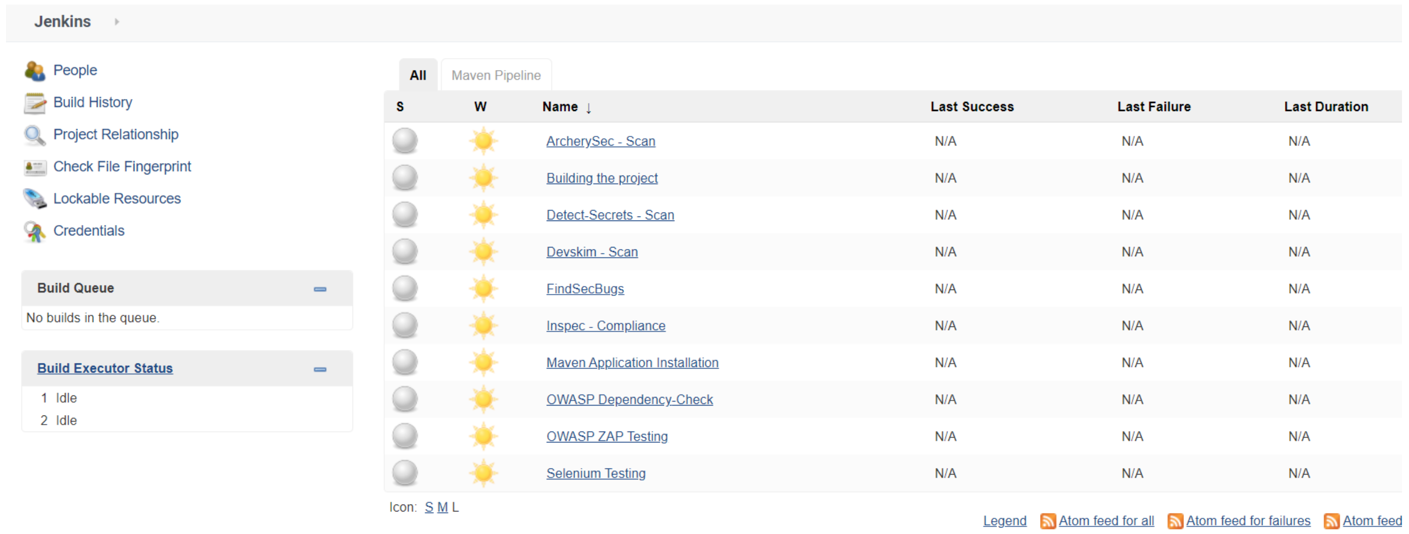
On starting the lab, the following interface will be accessible to the user.



On choosing (clicking the text in the center) top left panel, **vscode** will open in a new tab



Similarly on selecting the top middle panel, a web UI of **Jenkins** will open in a new tab.



Jenkins

- People
- Build History
- Project Relationship
- Check File Fingerprint
- Lockable Resources
- Credentials

Build Queue

No builds in the queue.

Build Executor Status

1 Idle
2 Idle

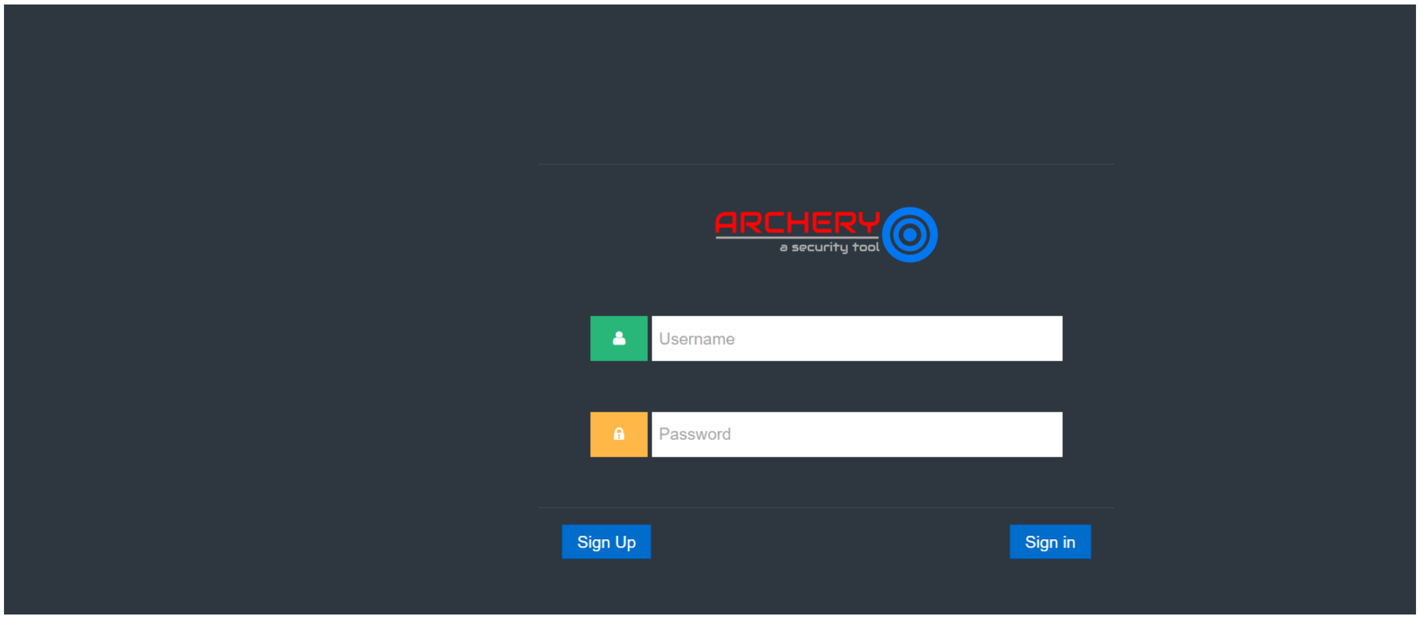
All | Maven Pipeline

S	W	Name ↓	Last Success	Last Failure	Last Duration
		ArcherySec - Scan	N/A	N/A	N/A
		Building the project	N/A	N/A	N/A
		Detect-Secrets - Scan	N/A	N/A	N/A
		Devskim - Scan	N/A	N/A	N/A
		FindSecBugs	N/A	N/A	N/A
		Inspec - Compliance	N/A	N/A	N/A
		Maven Application Installation	N/A	N/A	N/A
		OWASP Dependency-Check	N/A	N/A	N/A
		OWASP ZAP Testing	N/A	N/A	N/A
		Selenium Testing	N/A	N/A	N/A

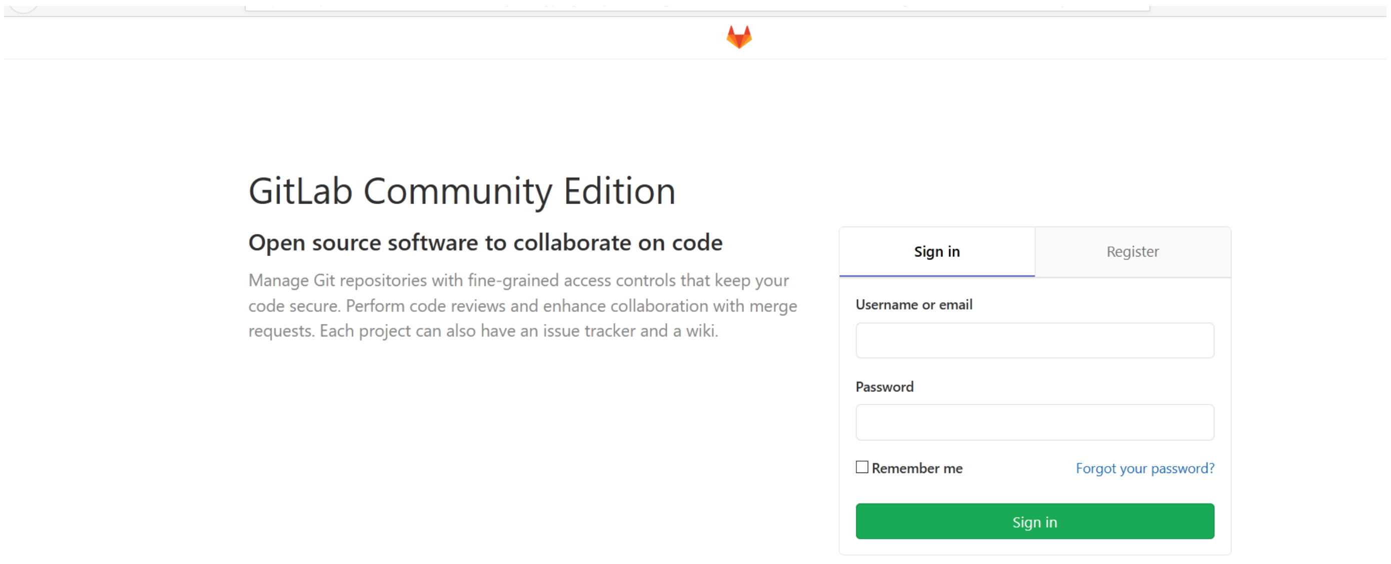
Icon: S M L

Legend | Atom feed for all | Atom feed for failures | Atom feed

On selecting the top right panel, a web UI of **ArcherySec**will open in a new tab.



On selecting the bottom left panel, a web UI of **Gitlab** will open in a new tab.


GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

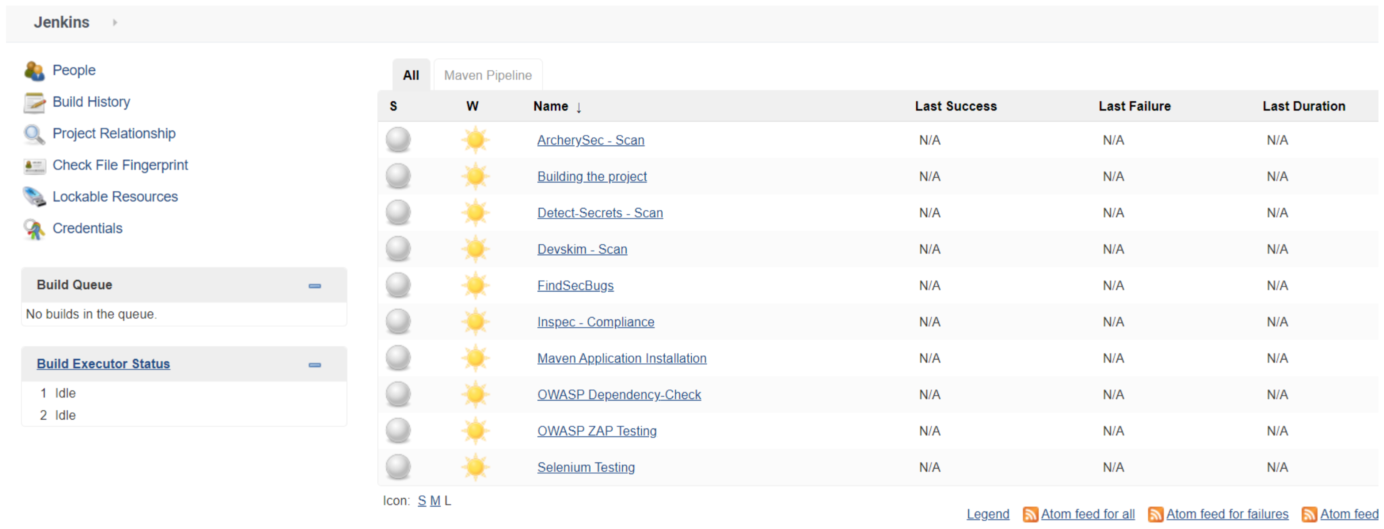
And on selecting the bottom right panel, a web UI of **Test Server** will open in a new tab.

```
Bad Gateway
```

The page will reload until the test-server has started running the web service at port 8080

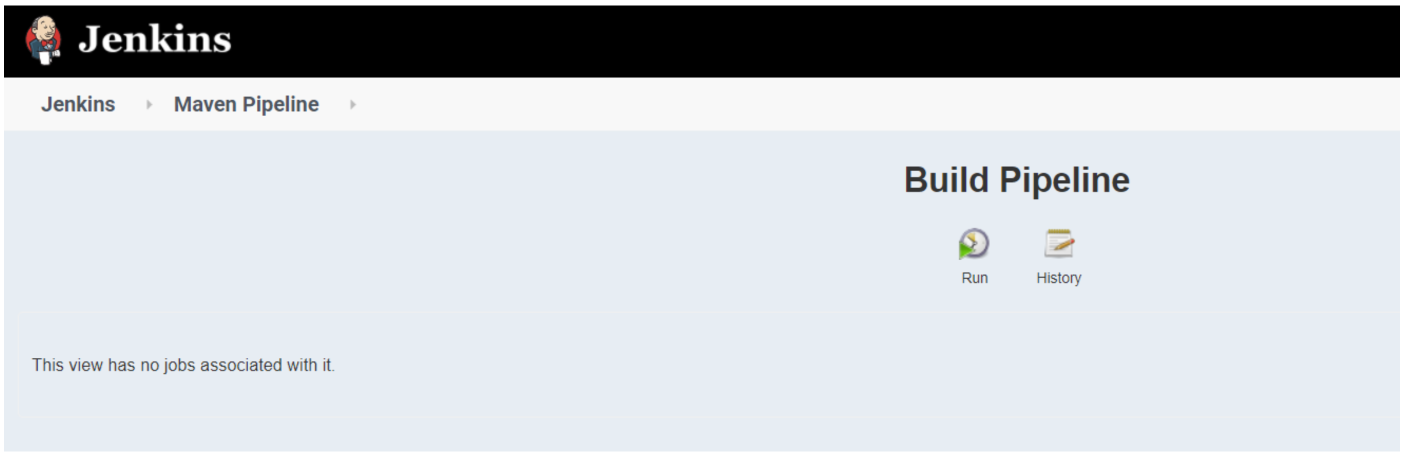
Solution

Step 1: Open the Jenkins page



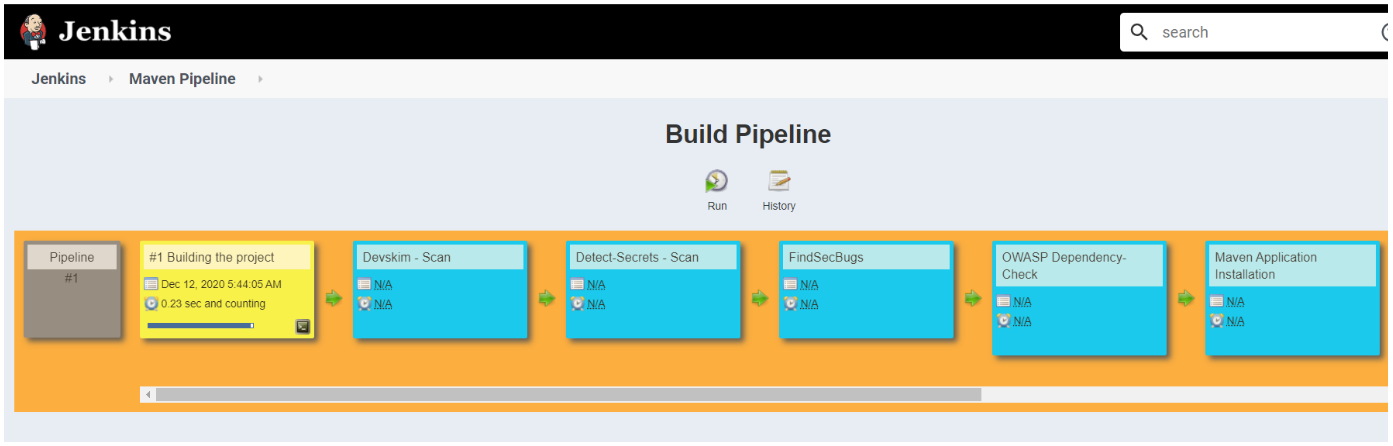
S	W	Name ↓	Last Success	Last Failure	Last Duration
		ArcherySec - Scan	N/A	N/A	N/A
		Building the project	N/A	N/A	N/A
		Detect-Secrets - Scan	N/A	N/A	N/A
		Devskim - Scan	N/A	N/A	N/A
		FindSecBugs	N/A	N/A	N/A
		Inspec - Compliance	N/A	N/A	N/A
		Maven Application Installation	N/A	N/A	N/A
		OWASP Dependency-Check	N/A	N/A	N/A
		OWASP ZAP Testing	N/A	N/A	N/A
		Selenium Testing	N/A	N/A	N/A

There are 10 jobs present in the Jenkins interface. Navigate to the Maven Pipeline view section.

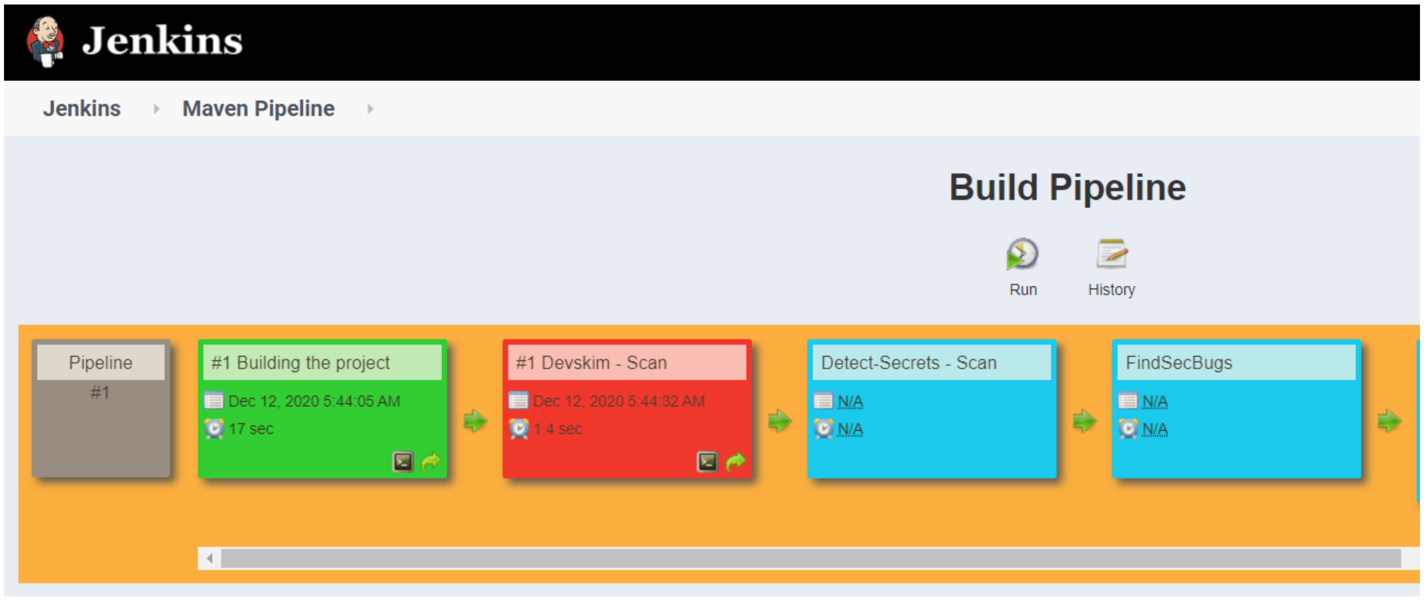


This view has no jobs associated with it.

Click on the Run button to start building the pipeline.



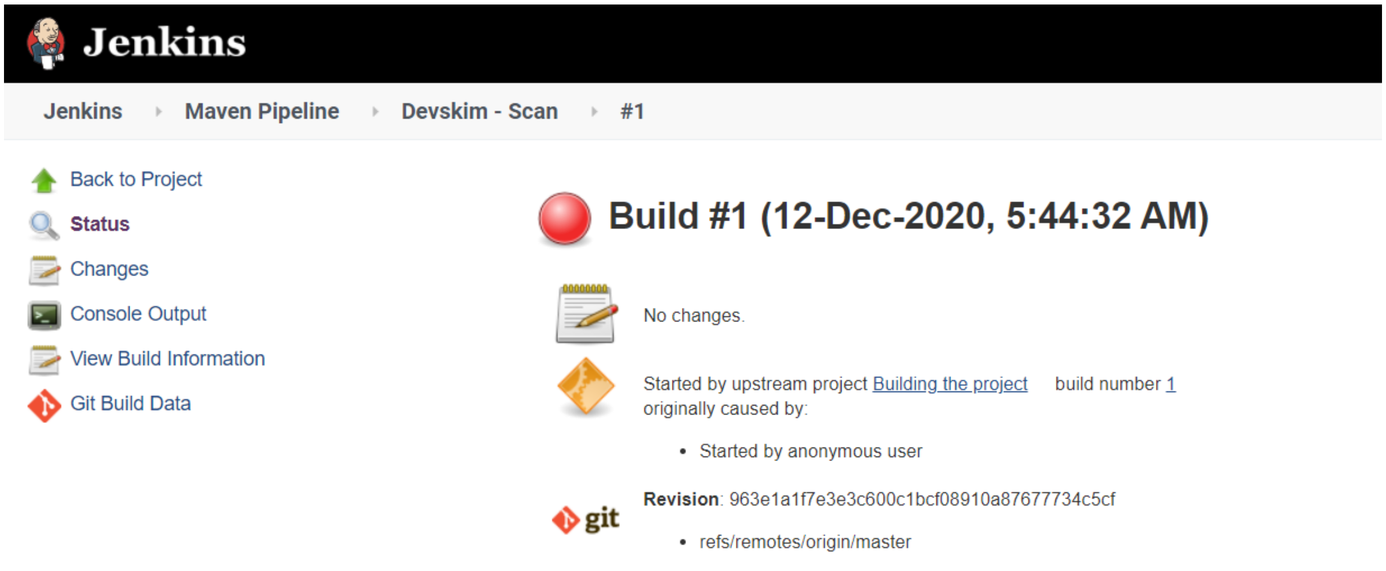
The projects will start building one by one. Keep reloading the page in intervals to check the changes on the page.



The build failed.

Devskim Issue

Step 1: Click on the 'Devskim - Scan' to check the job build page.


Jenkins

Jenkins › Maven Pipeline › Devskim - Scan › #1

Back to Project
Status
Changes
Console Output
View Build Information
Git Build Data

Build #1 (12-Dec-2020, 5:44:32 AM)

No changes.

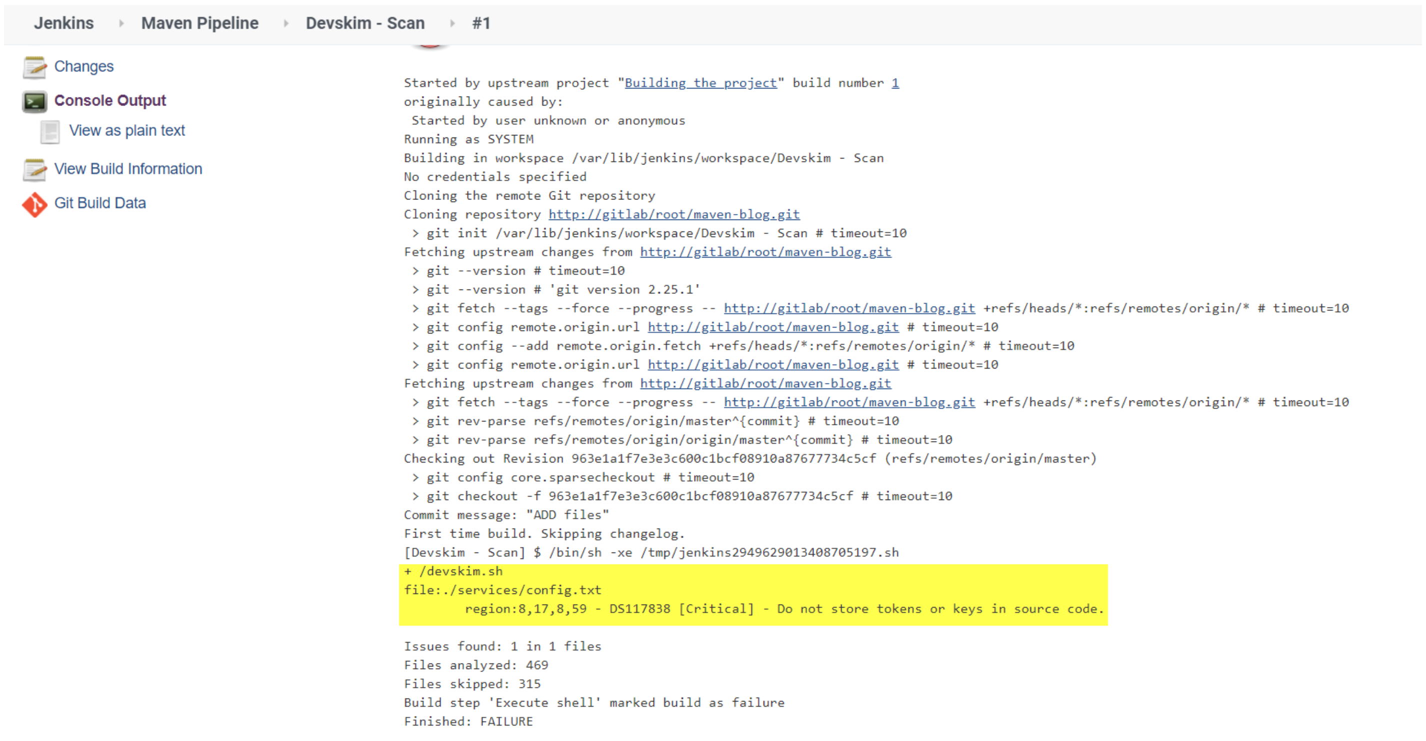
Started by upstream project Building the project build number 1
originally caused by:

- Started by anonymous user

Revision: 963e1a1f7e3e3c600c1bcf08910a87677734c5cf

- refs/remotes/origin/master

Step 2: Click on the “Console Output” to check the issues found by devskim tool.

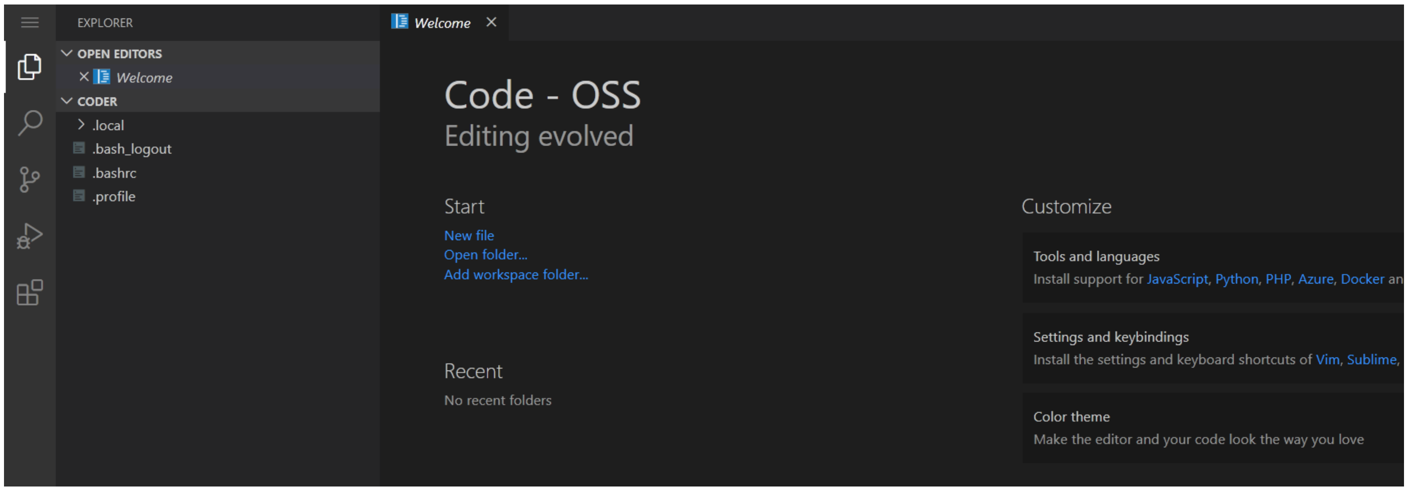

Jenkins › Maven Pipeline › Devskim - Scan › #1

Changes
Console Output
View as plain text
View Build Information
Git Build Data

```
Started by upstream project "Building the project" build number 1
originally caused by:
 Started by user unknown or anonymous
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Devskim - Scan
No credentials specified
Cloning the remote Git repository
Cloning repository http://gitlab/root/maven-blog.git
 > git init /var/lib/jenkins/workspace/Devskim - Scan # timeout=10
Fetching upstream changes from http://gitlab/root/maven-blog.git
 > git --version # timeout=10
 > git --version # 'git version 2.25.1'
 > git fetch --tags --force --progress -- http://gitlab/root/maven-blog.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url http://gitlab/root/maven-blog.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url http://gitlab/root/maven-blog.git # timeout=10
Fetching upstream changes from http://gitlab/root/maven-blog.git
 > git fetch --tags --force --progress -- http://gitlab/root/maven-blog.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 963e1a1f7e3e3c600c1bcf08910a87677734c5cf (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 963e1a1f7e3e3c600c1bcf08910a87677734c5cf # timeout=10
Commit message: "ADD files"
First time build. Skipping changelog.
[Devskim - Scan] $ /bin/sh -xe /tmp/jenkins2949629013408705197.sh
+ /devskim.sh
file:./services/config.txt
        region:8,17,8,59 - DS117838 [Critical] - Do not store tokens or keys in source code.

Issues found: 1 in 1 files
Files analyzed: 469
Files skipped: 315
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```

The devskim identified hardcoded keys in the services directory.

Step 3: Open the vscode server page.



EXPLORER

OPEN EDITORS

× Welcome

CODER

.local

.bash_logout

.bashrc

.profile

Welcome ×

Code - OSS

Editing evolved

Start

New file

Open folder...

Add workspace folder...

Recent

No recent folders

Customize

Tools and languages

Install support for JavaScript, Python, PHP, Azure, Docker an

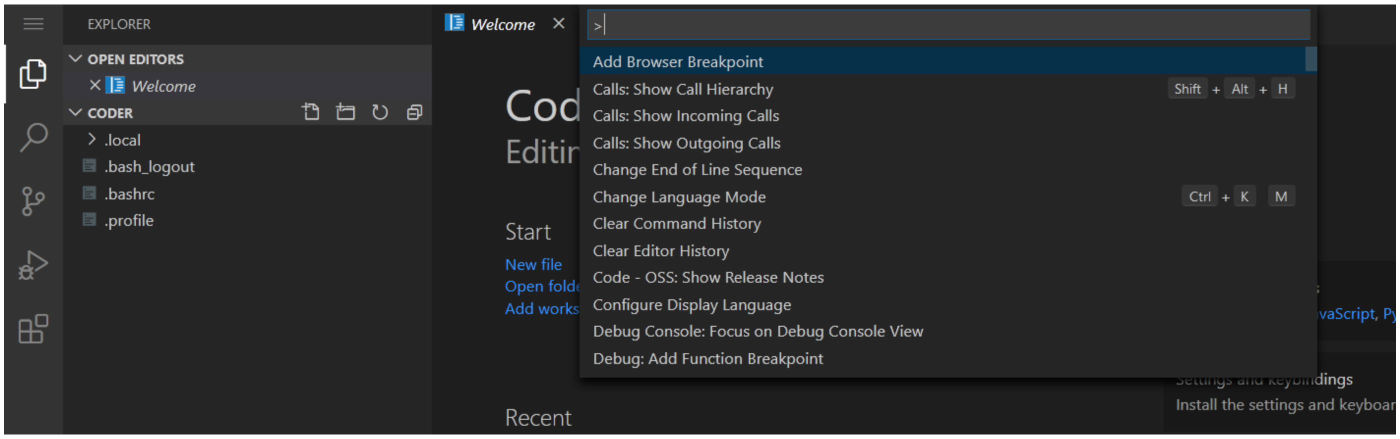
Settings and keybindings

Install the settings and keyboard shortcuts of Vim, Sublime,

Color theme

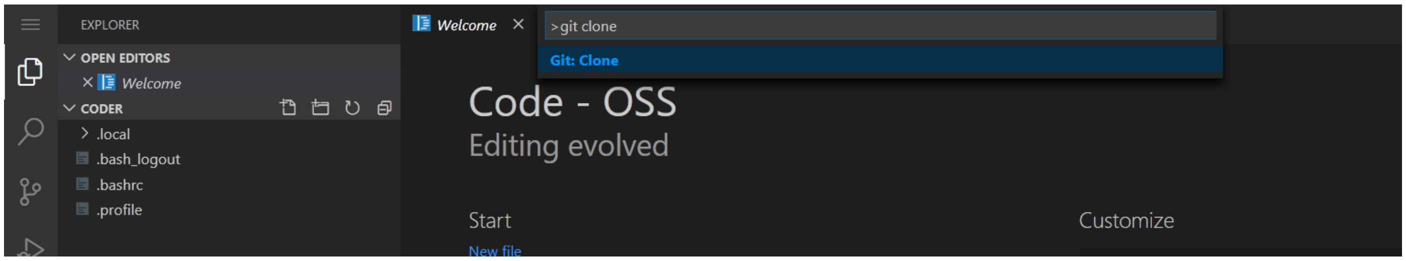
Make the editor and your code look the way you love

Press CTRL + SHIFT + P to open the command palette or click on the settings bar available in the bottom left and choose the “Command Palette” option.

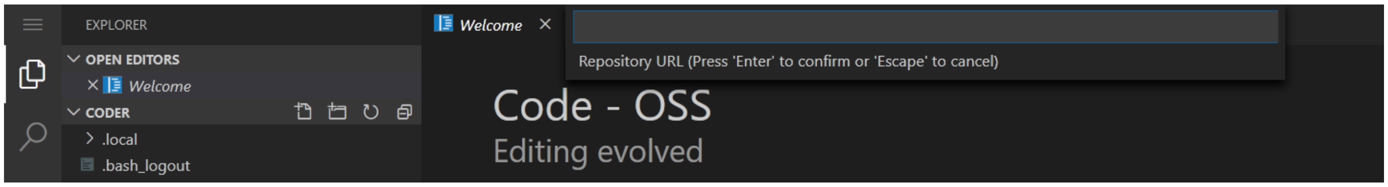


```
>|
Add Browser Breakpoint
Calls: Show Call Hierarchy                    Shift + Alt + H
Calls: Show Incoming Calls
Calls: Show Outgoing Calls
Change End of Line Sequence
Change Language Mode                          Ctrl + K  M
Clear Command History
Clear Editor History
Code - OSS: Show Release Notes
Configure Display Language
Debug Console: Focus on Debug Console View
Debug: Add Function Breakpoint
```

Step 4: Enter the command “git clone” in the command palette in order to clone the repository and make changes.



Press enter to choose the Git Clone option.

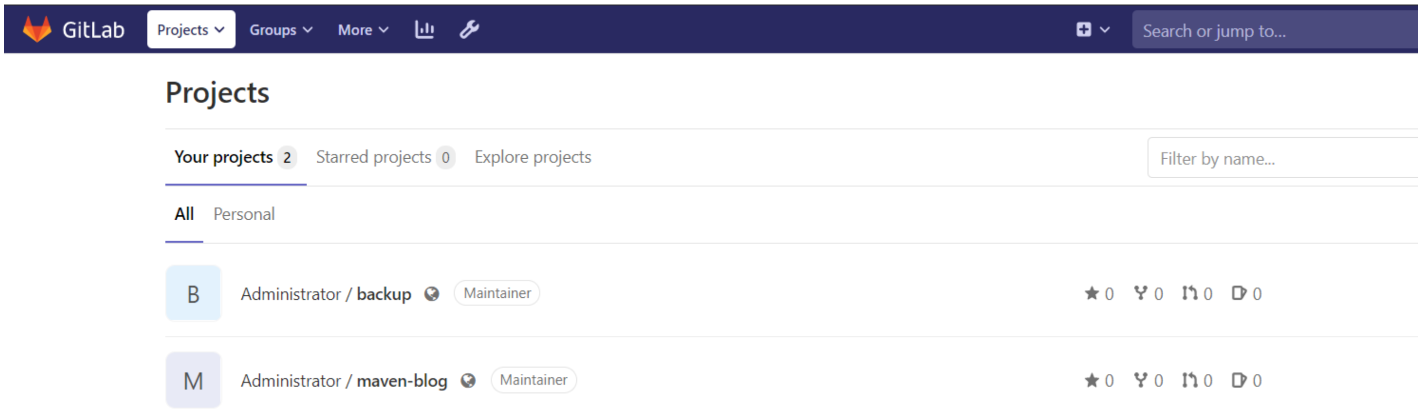


Repository URL (Press 'Enter' to confirm or 'Escape' to cancel)

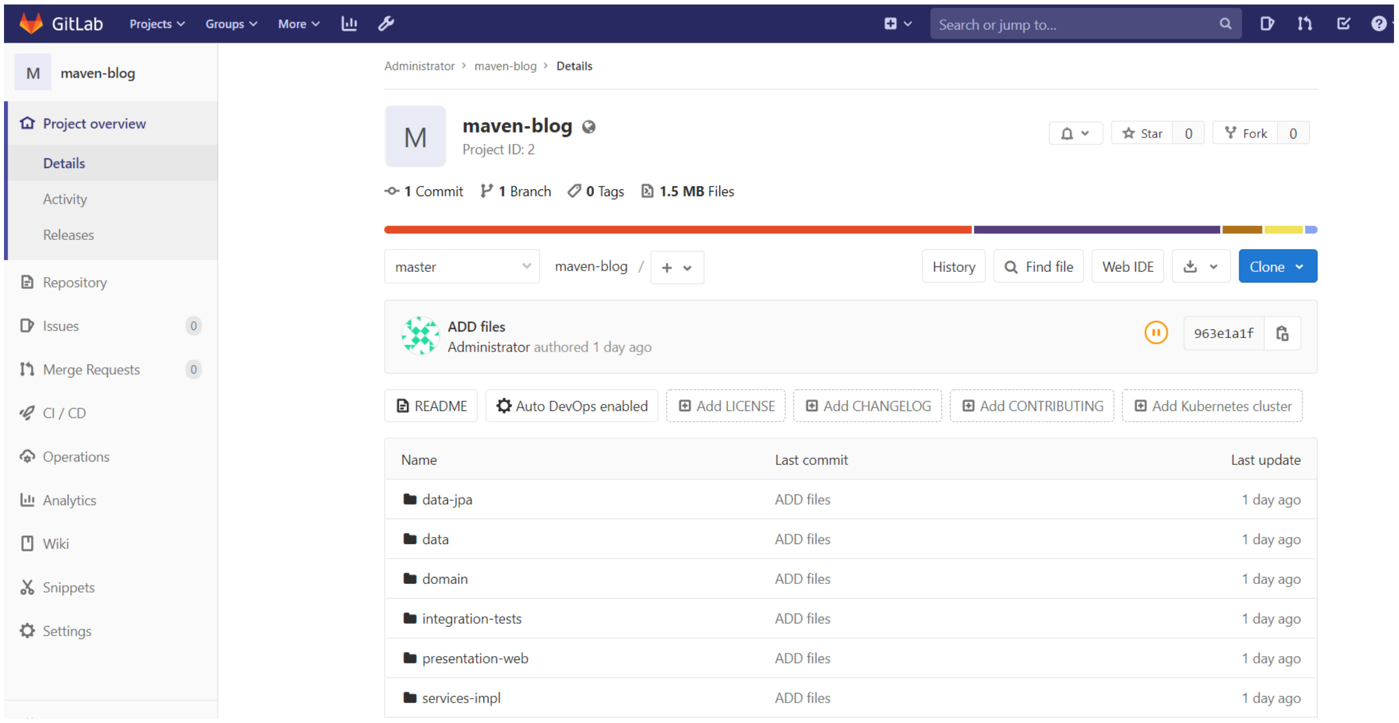
Step 5: Open the gitlab page and log in using the credentials provided in the challenge description.

Credentials:

- **Username:** root
- **Password:** welcome123



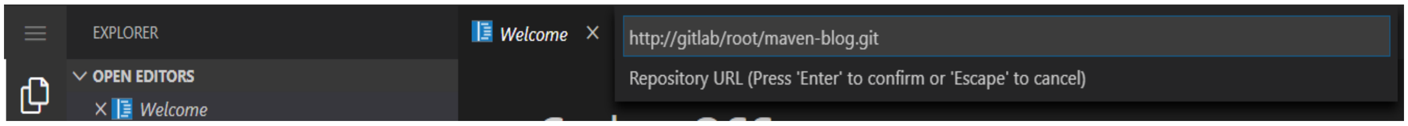
Click on the maven-blog link to open the repository page.



Name	Last commit	Last update
data-jpa	ADD files	1 day ago
data	ADD files	1 day ago
domain	ADD files	1 day ago
integration-tests	ADD files	1 day ago
presentation-web	ADD files	1 day ago
services-impl	ADD files	1 day ago

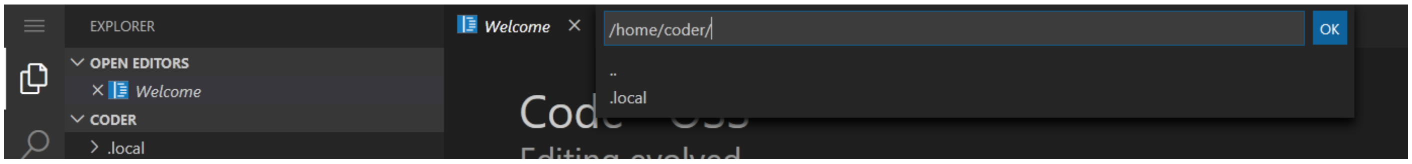
Copy the git link to the GitLab repository.

Step 6: Enter the link into the clone function at vscode server.



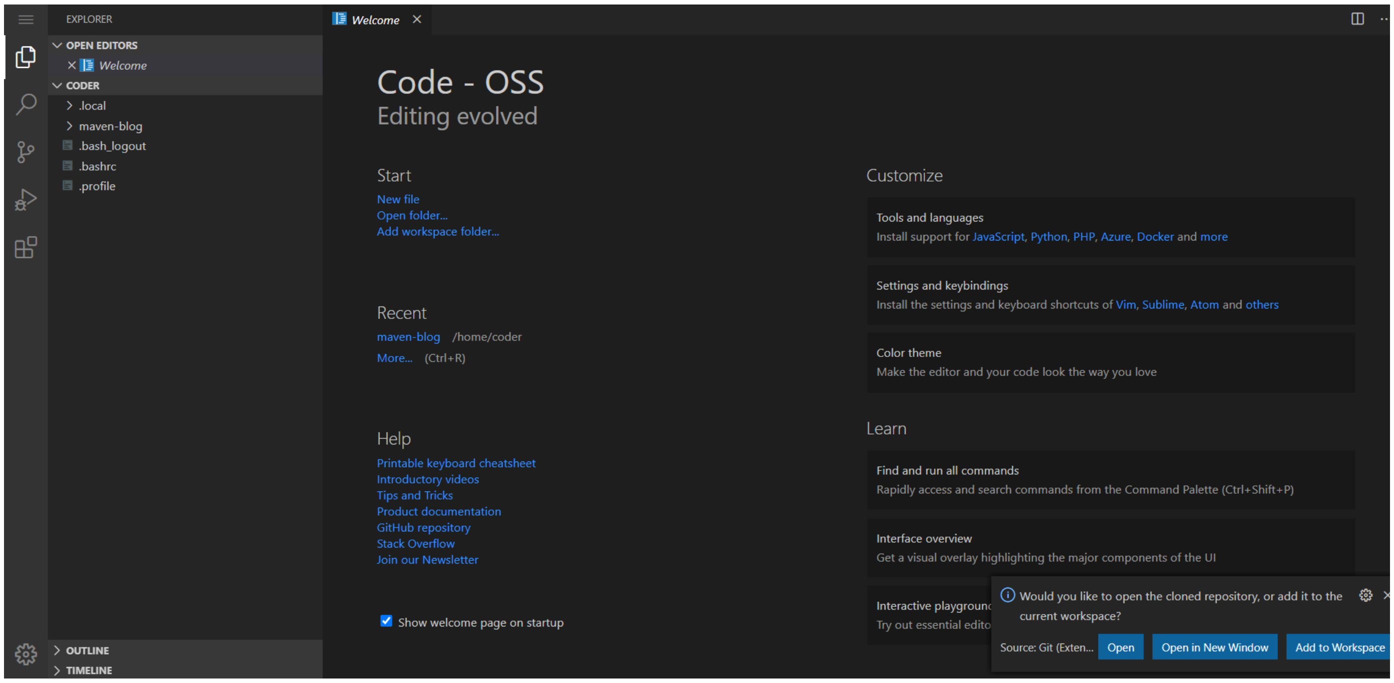
```
http://gitlab/root/maven-blog.git
Repository URL (Press 'Enter' to confirm or 'Escape' to cancel)
```

Press Enter.



```
/home/coder/
```

Choose the path to clone to the repository.



Code - OSS
Editing evolved

Start
New file
Open folder...
Add workspace folder...

Recent
maven-blog /home/coder
More... (Ctrl+R)

Help
Printable keyboard cheatsheet
Introductory videos
Tips and Tricks
Product documentation
GitHub repository
Stack Overflow
Join our Newsletter

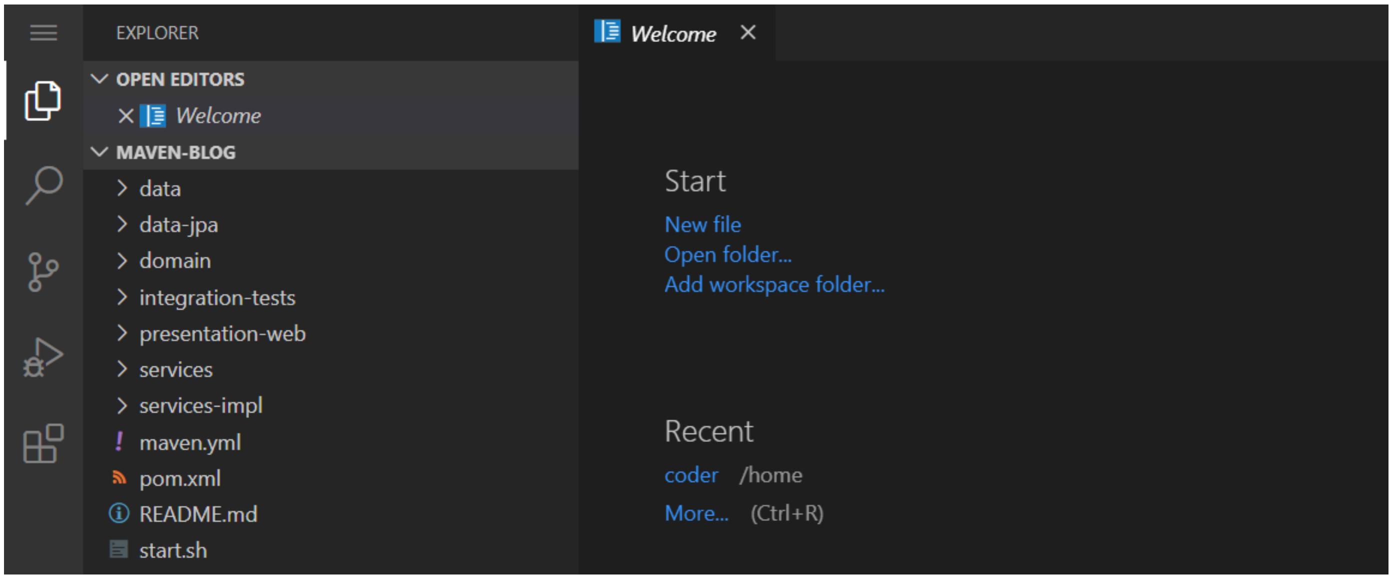
Show welcome page on startup

Customize
Tools and languages
Install support for JavaScript, Python, PHP, Azure, Docker and more
Settings and keybindings
Install the settings and keyboard shortcuts of Vim, Sublime, Atom and others
Color theme
Make the editor and your code look the way you love

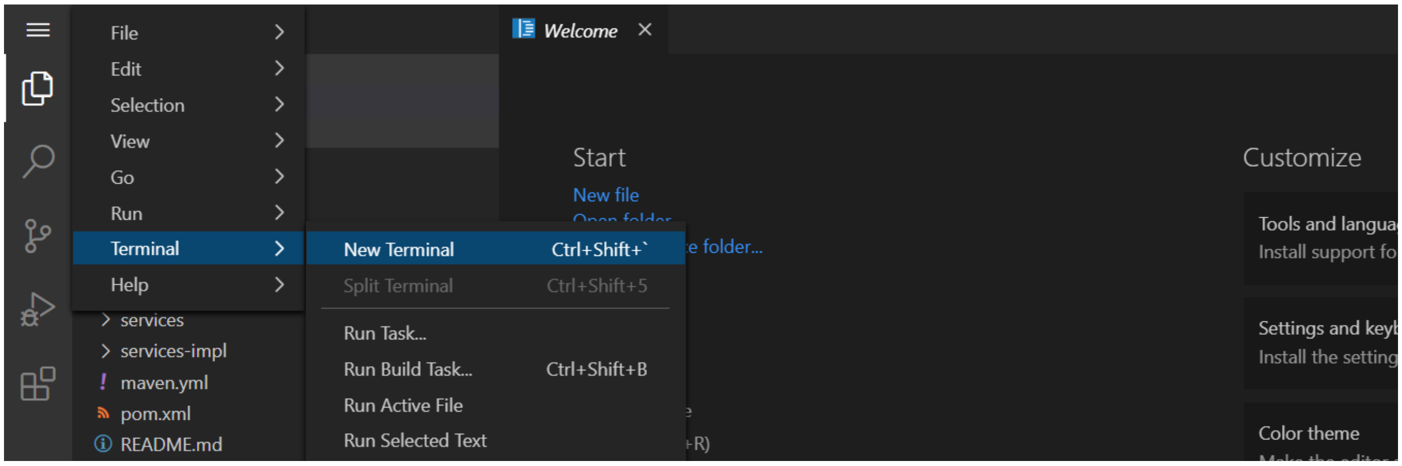
Learn
Find and run all commands
Rapidly access and search commands from the Command Palette (Ctrl+Shift+P)
Interface overview
Get a visual overlay highlighting the major components of the UI
Interactive playgroun
Try out essential edito

Would you like to open the cloned repository, or add it to the current workspace?
Source: Git (Exten... Open | Open in New Window | Add to Workspace

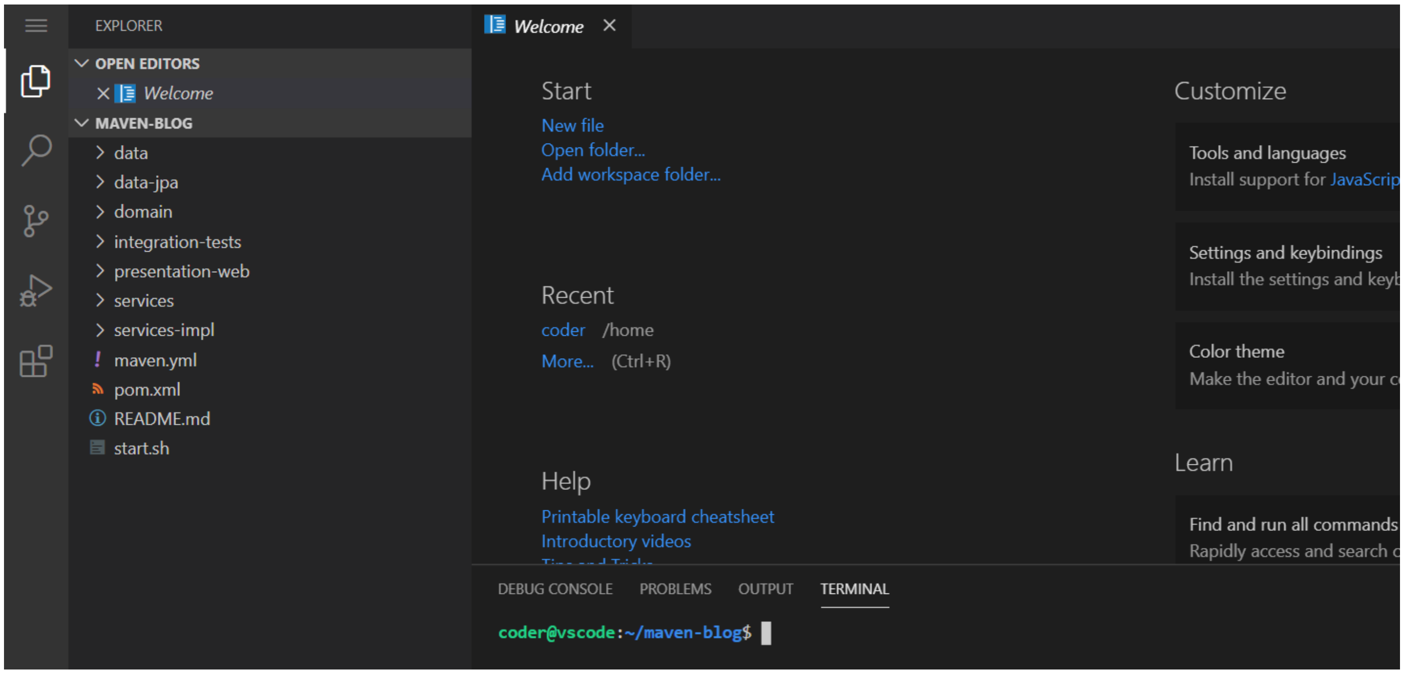
Click on the “Open” button to open the source code directory of the maven-blog.



Step 7: Open the Terminal in the directory.



Select the “New Terminal” option.

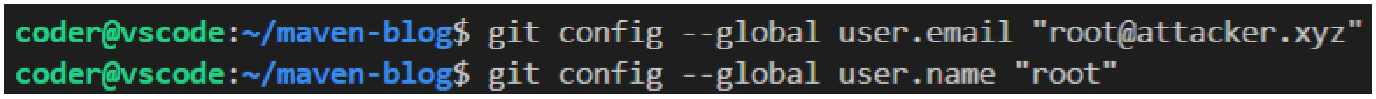


```
coder@vscode:~/maven-blog$
```

Step 8: Enter the git credentials into the terminal to save the identity of the user.

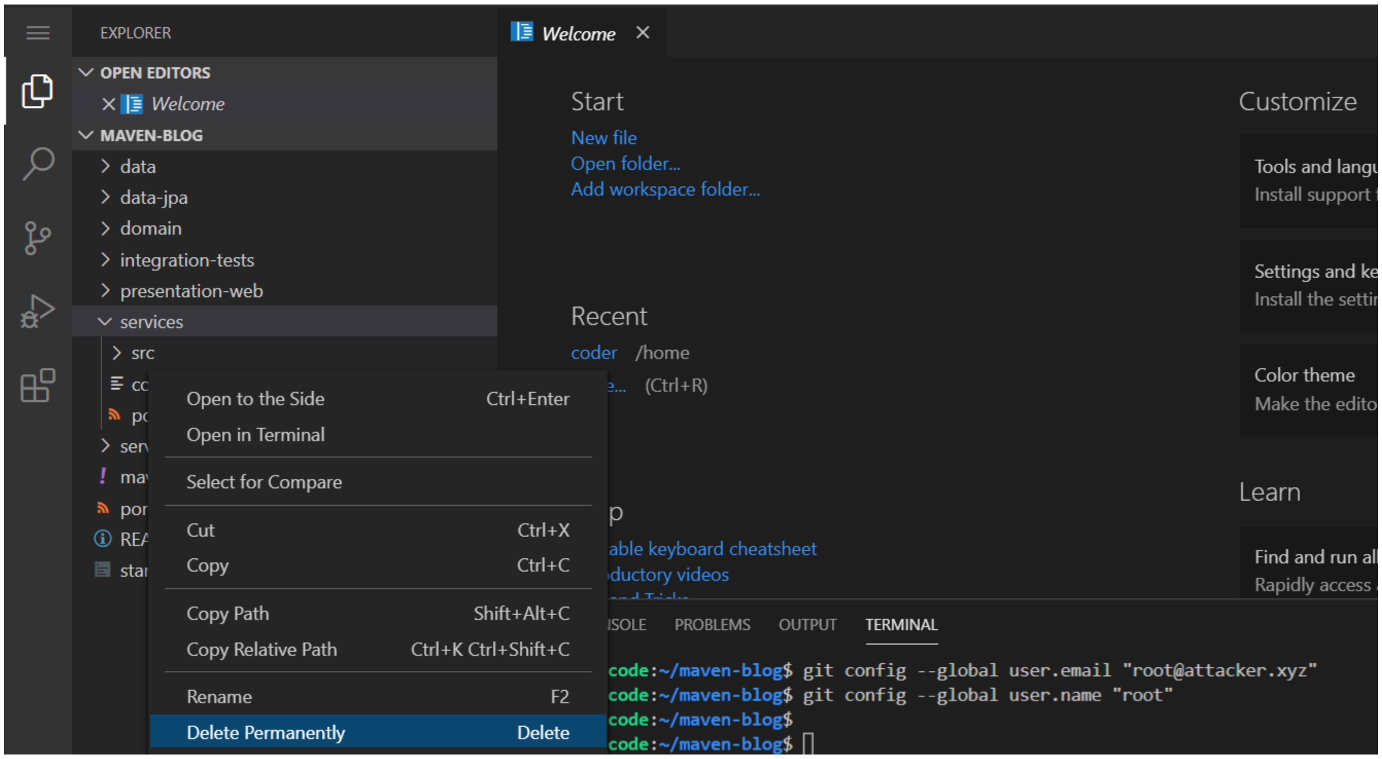
Commands:

```
git config --global user.email "root@attacker.xyz"
git config --global user.name "root"
```



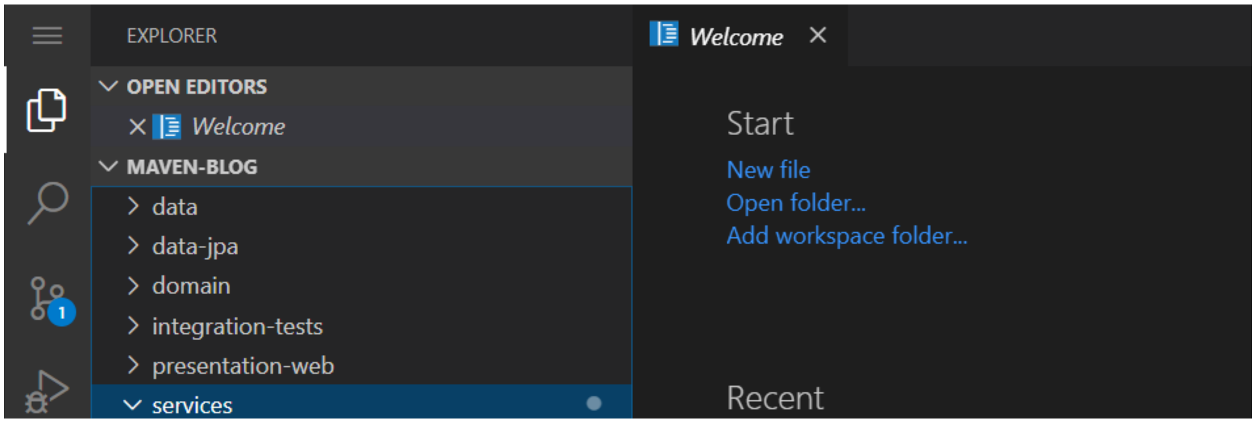
```
coder@vscode:~/maven-blog$ git config --global user.email "root@attacker.xyz"
coder@vscode:~/maven-blog$ git config --global user.name "root"
```

Step 9: Navigate inside the services directory and right-click on the config.txt file.

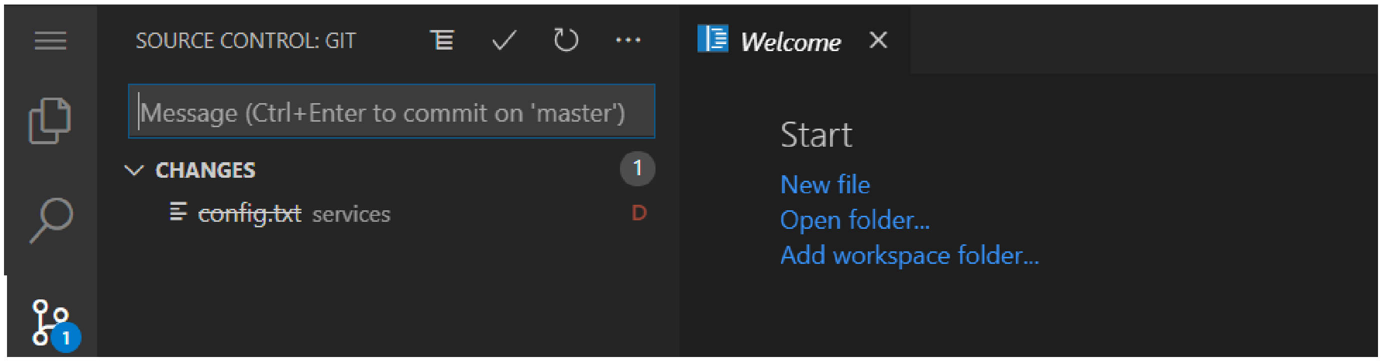


```
code:~/maven-blog$ git config --global user.email "root@attacker.xyz"
code:~/maven-blog$ git config --global user.name "root"
code:~/maven-blog$
code:~/maven-blog$
```

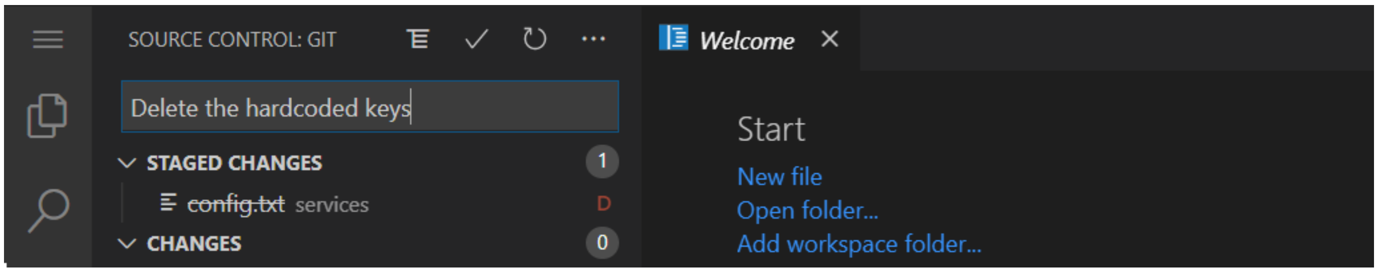
Select “Delete Permanently” to delete the backup.sql file from the project.



Step 10: Navigate to the Source Control section.

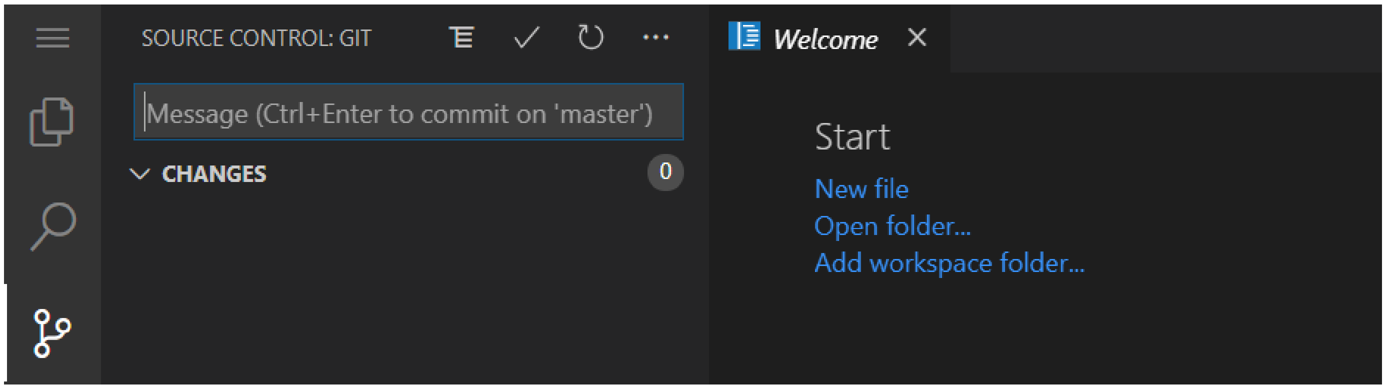


Enter a message in the message field to set a comment on the commit.

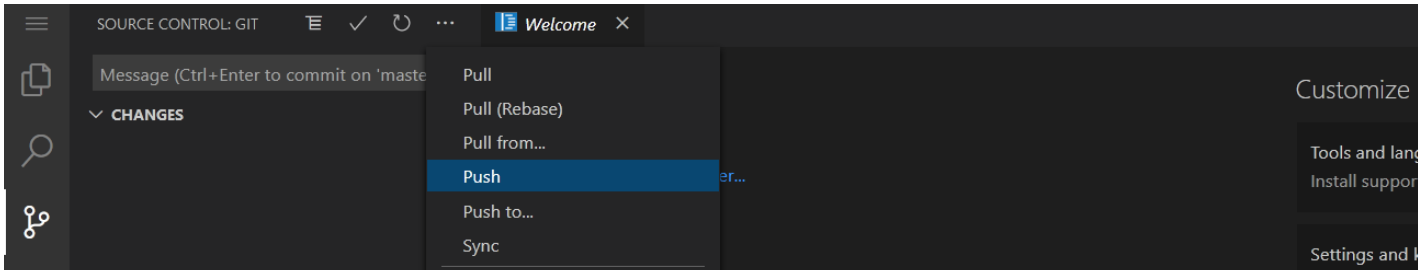


Delete the hardcoded keys

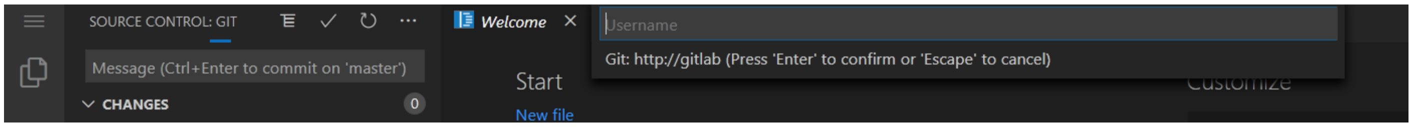
Click on the Commit button (Tick icon) and Choose the option “Always” to stage the commits always after the changes are made to the files.



Step 11: Click on the "More Actions" button (3 dots)



And select the Push button to push the changes to the remote repository.

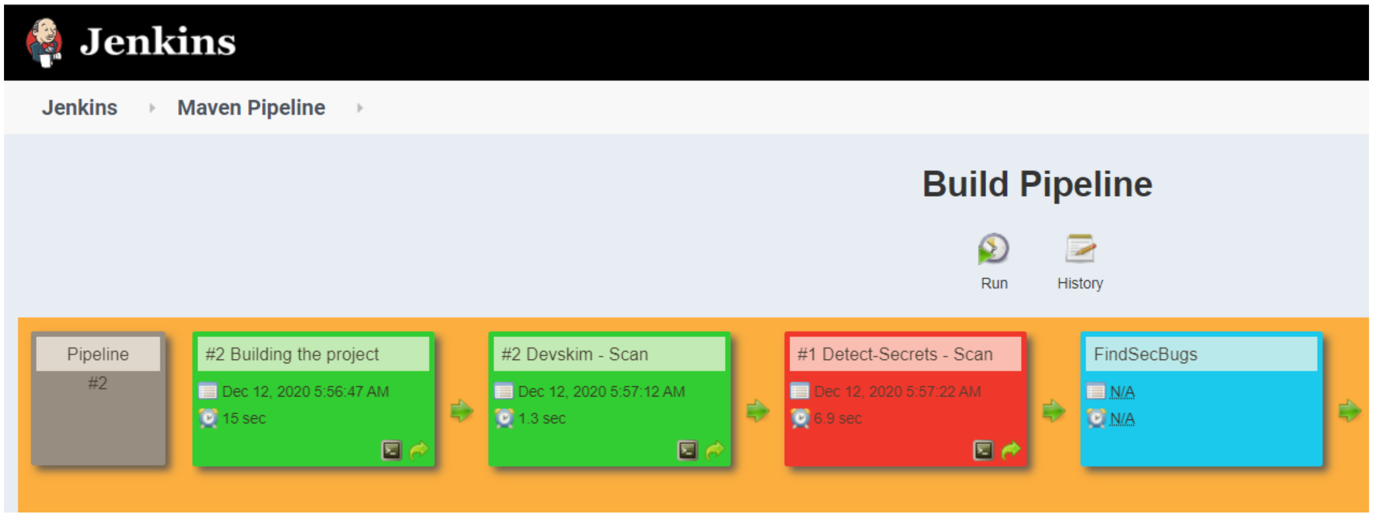


Git: http://gitlab (Press 'Enter' to confirm or 'Escape' to cancel)

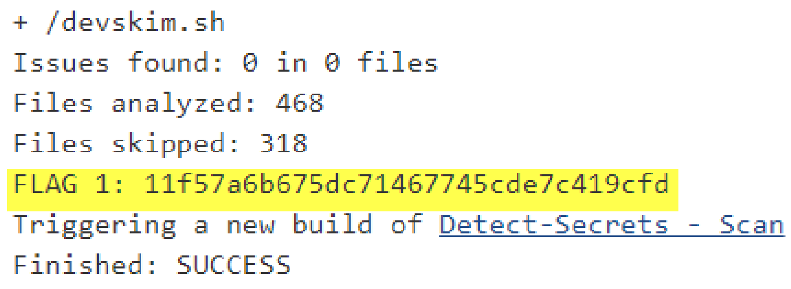
Enter the git username and password when asked in the fields. The credentials are the same from gitlab.

Note: As soon as the code pushed to the remote repository, the pipeline will start building automatically.

Step 12: Navigate back to the Pipeline view to check if Devskim stage passes or not.



The Devskim stage has passed successfully. The logs can be checked from clicking on the "Devskim - Scan" and opening the console output.

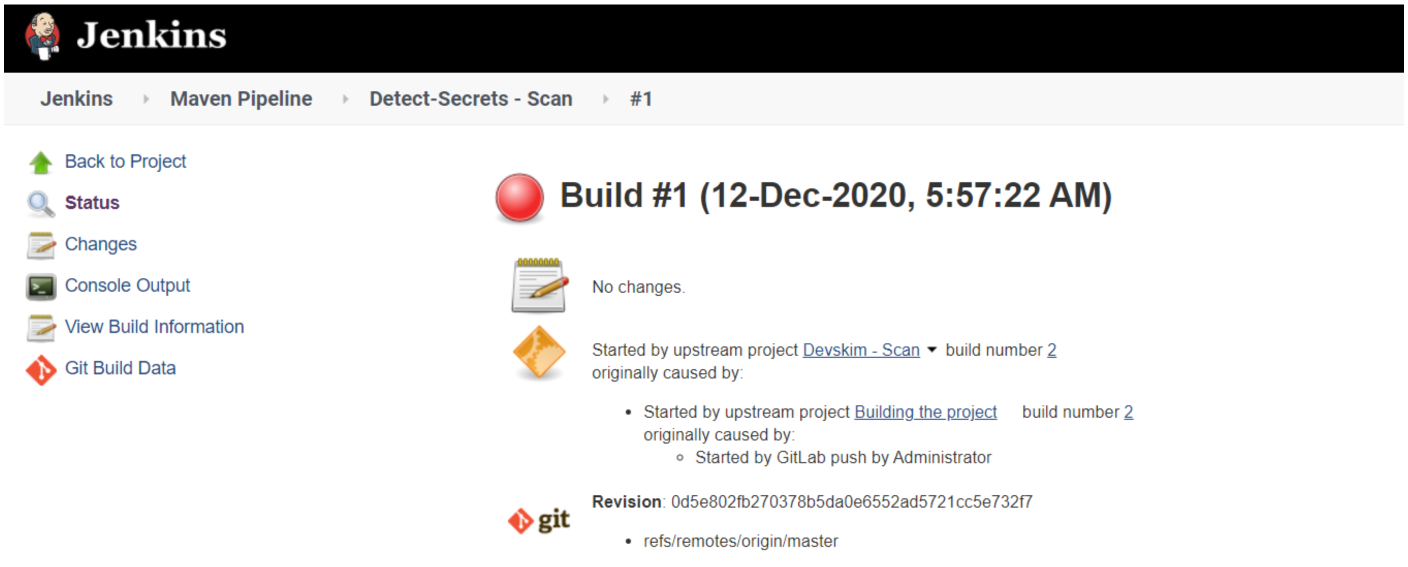


```
+ /devskim.sh
Issues found: 0 in 0 files
Files analyzed: 468
Files skipped: 318
FLAG 1: 11f57a6b675dc71467745cde7c419cfd
Triggering a new build of Detect-Secrets - Scan
Finished: SUCCESS
```

FLAG 1: 11f57a6b675dc71467745cde7c419cfd

Detect-Secrets issue

Step 1: Click on the Detect-Secrets Scan' to check the job build page.



Build #1 (12-Dec-2020, 5:57:22 AM)

No changes.

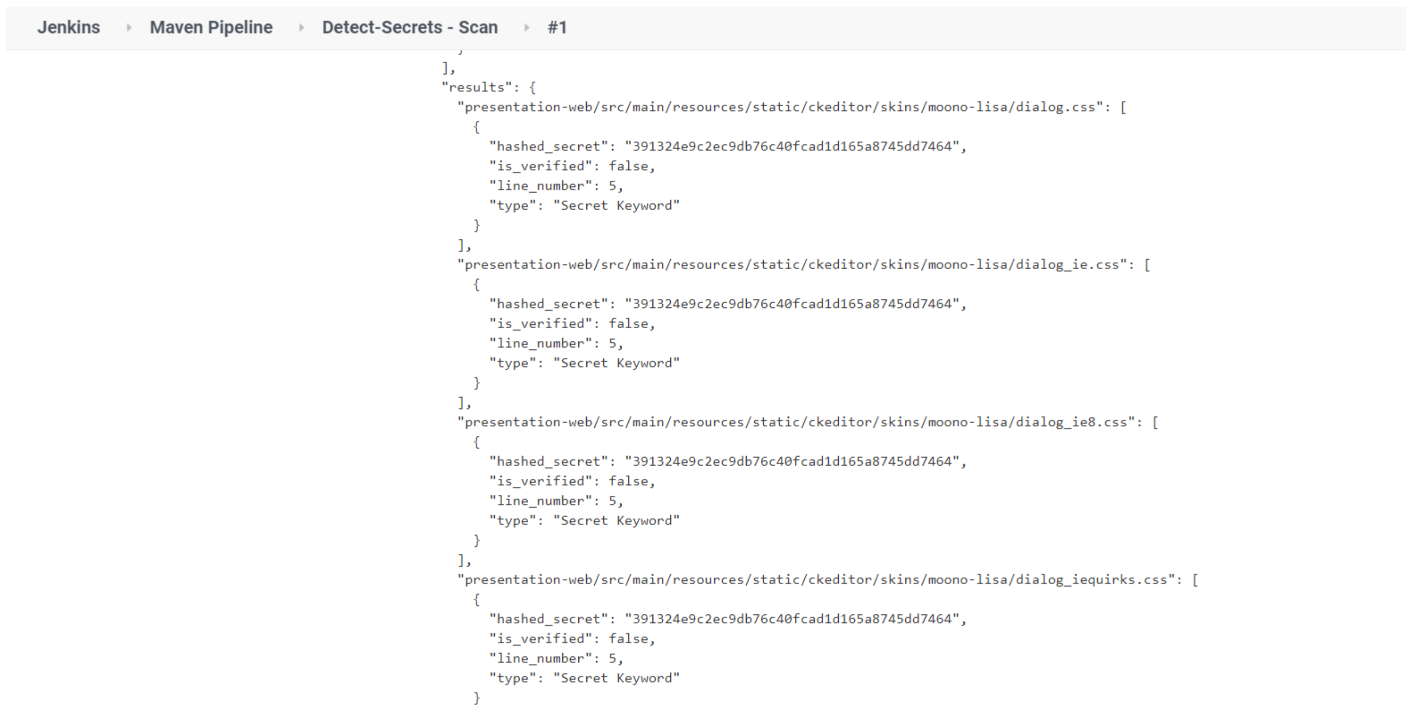
Started by upstream project Devskim - Scan build number 2
originally caused by:

- Started by upstream project Building the project build number 2
  originally caused by:
  - Started by GitLab push by Administrator

Revision: 0d5e802fb270378b5da0e6552ad5721cc5e732f7

- refs/remotes/origin/master

Step 2: Click on the "Console Output" to check the issues found by Detect-Secrets tool.


Jenkins › Maven Pipeline › Detect-Secrets - Scan › #1

```
],
"results": {
  "presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/dialog.css": [
    {
      "hashed_secret": "391324e9c2ec9db76c40fcad1d165a8745dd7464",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/dialog_ie.css": [
    {
      "hashed_secret": "391324e9c2ec9db76c40fcad1d165a8745dd7464",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/dialog_ie8.css": [
    {
      "hashed_secret": "391324e9c2ec9db76c40fcad1d165a8745dd7464",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/dialog_iequirks.css": [
    {
      "hashed_secret": "391324e9c2ec9db76c40fcad1d165a8745dd7464",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
```

The Detect-Secrets identified several strings in the files which are flagged as sensitive information.

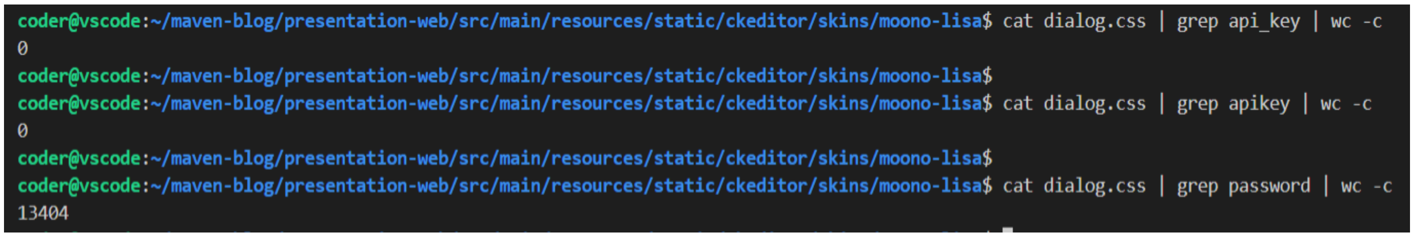
Files:

- dialog.css
- dialog_ie.css
- dialog_ie8.css
- dialog_iequirks.css

Step 3: Navigate to the directory of the files and check for sensitive keywords in the files such as api_key, apikey, password, and passwd, since these are the keywords which get flagged by the Detect-Secrets tool. (On VS Code Instance)

Commands:

```
cd presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/
cat dialog.css | grep api_key | wc -c
cat dialog.css | grep apikey | wc -c
cat dialog.css | grep password | wc -c
```

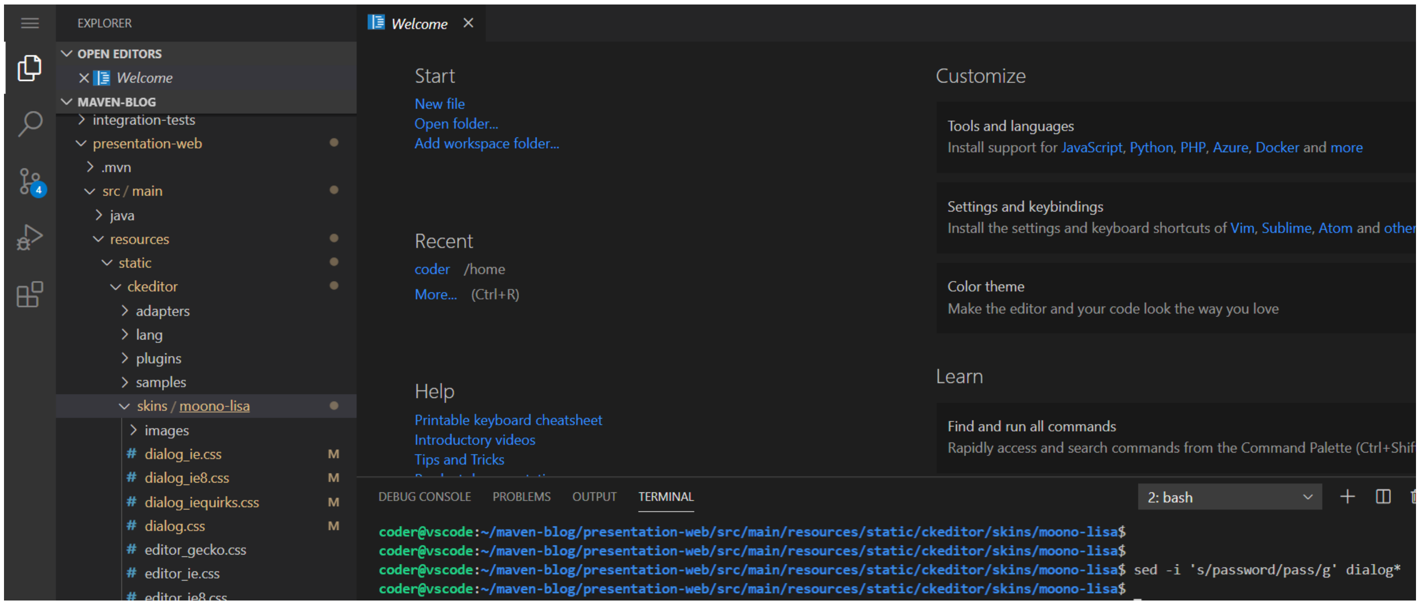


```
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$ cat dialog.css | grep api_key | wc -c
0
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$ cat dialog.css | grep apikey | wc -c
0
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$ cat dialog.css | grep password | wc -c
13404
```

The command will check for the keyword and return the number of bytes found. But only ‘password’ string returned output greater than 0. Hence, there is only ‘password’ keyword which is getting detected by Detect-Secrets.

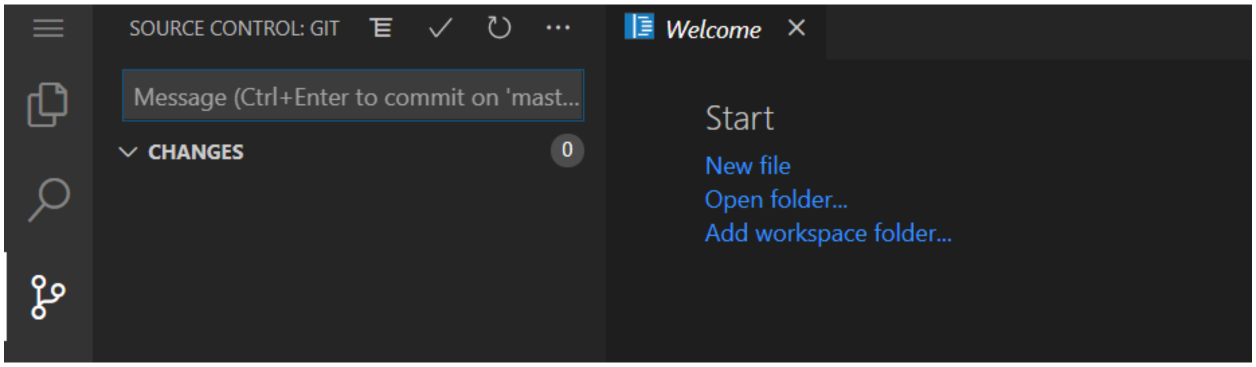
Step 4: Change all the occurrences of ‘password’ with pass in order to fix the issue.

Command: sed -i 's/password/pass/g' dialog*

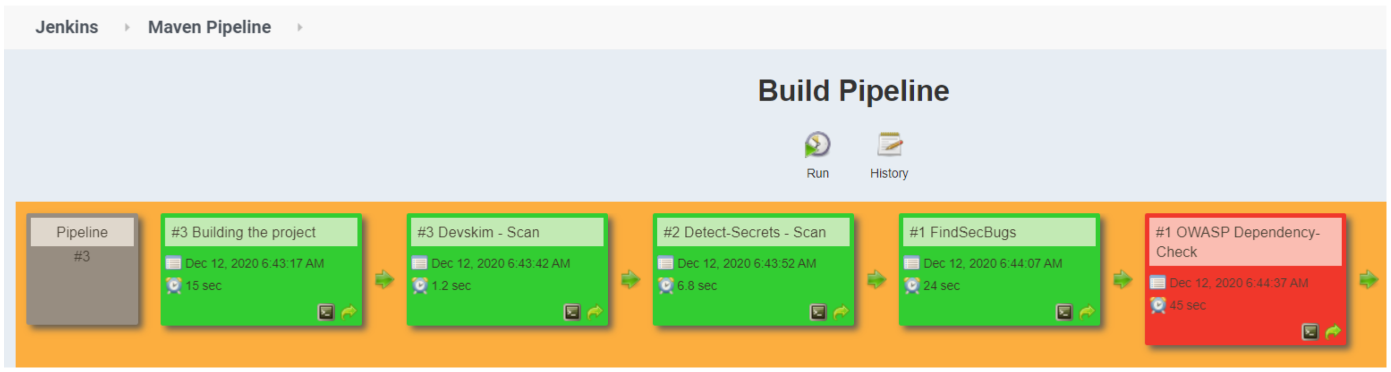


```
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$ sed -i 's/password/pass/g' dialog*
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$
```

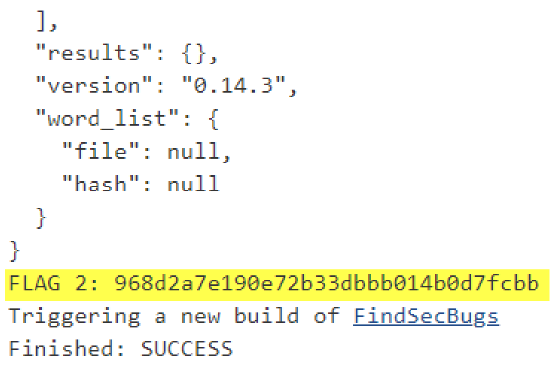
Step 5: Navigate to the SCM section and make commits to the changes. Push the changes to the remote repository.
(Same as Step 10 onwards of the DevSkim issue section)



Step 6: Check the pipeline for the changes.



The DetectSecrets stage has passed successfully. The logs can be checked from clicking on the "Detect-Secrets - Scan" and opening the console output.

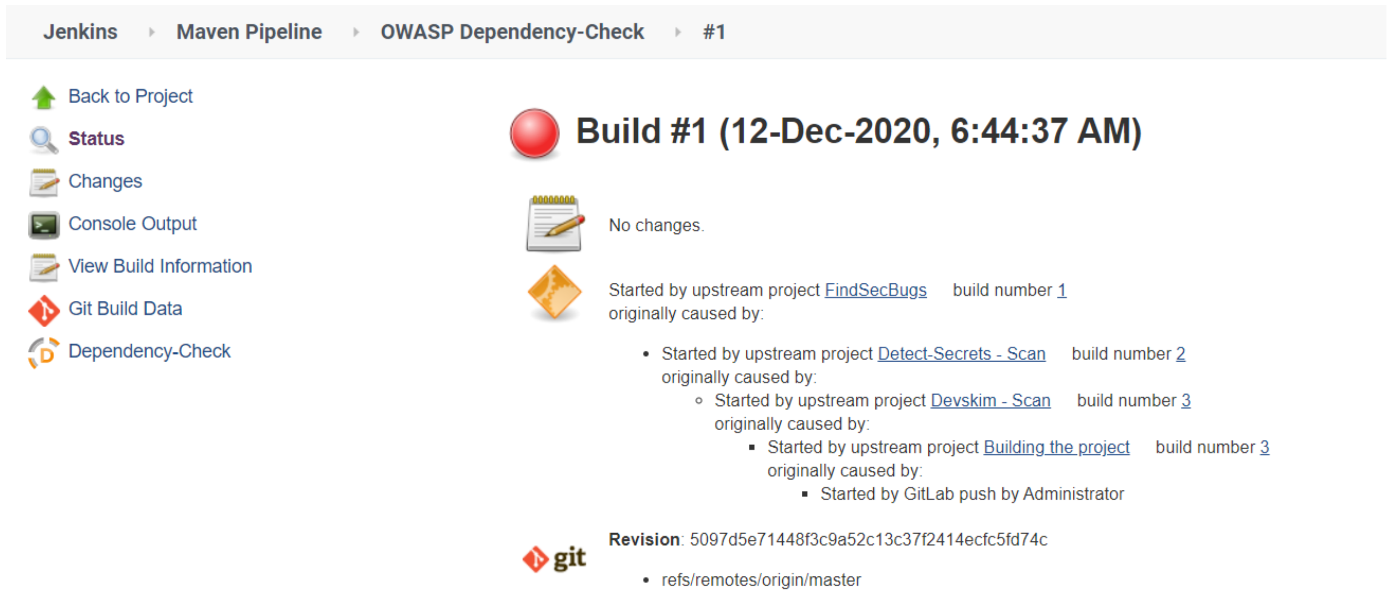


```
    ],
    "results": {},
    "version": "0.14.3",
    "word_list": {
      "file": null,
      "hash": null
    }
  }
FLAG 2: 968d2a7e190e72b33dbbb014b0d7fcbb
Triggering a new build of FindSecBugs
Finished: SUCCESS
```

FlAG 2: 968d2a7e190e72b33dbbb014b0d7fcbb

OWASP Dependency Check Issue

Step 1: Click on the ‘OWASP Dependency-Check’ to check the job build page.



Jenkins ‣ Maven Pipeline ‣ OWASP Dependency-Check ‣ #1

- Back to Project
- Status
- Changes
- Console Output
- View Build Information
- Git Build Data
- Dependency-Check

Build #1 (12-Dec-2020, 6:44:37 AM)

No changes.

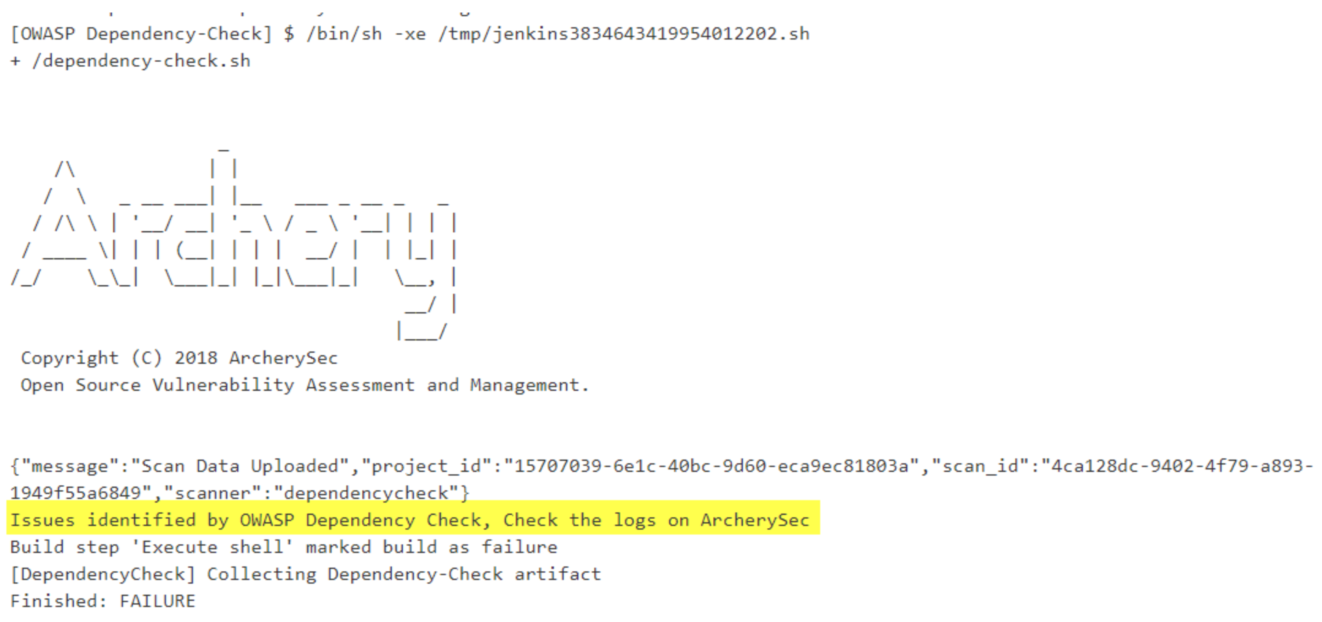
Started by upstream project FindSecBugs build number 1
originally caused by:

- Started by upstream project Detect-Secrets - Scan build number 2
  originally caused by:
  - Started by upstream project Devskim - Scan build number 3
    originally caused by:
    - Started by upstream project Building the project build number 3
      originally caused by:
      - Started by GitLab push by Administrator

git **Revision**: 5097d5e71448f3c9a52c13c37f2414ecfc5fd74c

- refs/remotes/origin/master

Step 2: Click on the “Console Output” to check the issues found by the Bandit tool.

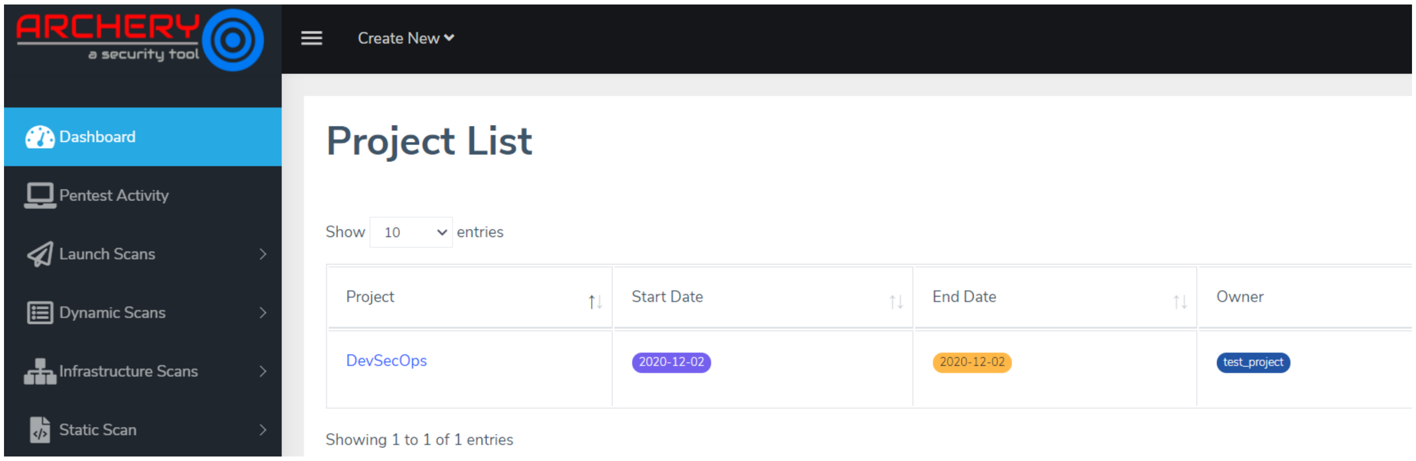


```
[OWASP Dependency-Check] $ /bin/sh -xe /tmp/jenkins3834643419954012202.sh
+ /dependency-check.sh


                   _
    /\            | |
   /  \   _ __ ___| |__   ___ _ __ _   _
  / /\ \ | '__/ __| '_ \ / _ \ '__| | | |
 / ____ \| | | (__| | | |  __/ |  | |_| |
/_/    \_\_|  \___|_| |_|\___|_|   \__, |
                                    __/ |
                                   |___/
 Copyright (C) 2018 ArcherySec
 Open Source Vulnerability Assessment and Management.


{"message":"Scan Data Uploaded","project_id":"15707039-6e1c-40bc-9d60-eca9ec81803a","scan_id":"4ca128dc-9402-4f79-a893-
1949f55a6849","scanner":"dependencycheck"}
Issues identified by OWASP Dependency Check, Check the logs on ArcherySec
Build step 'Execute shell' marked build as failure
[DependencyCheck] Collecting Dependency-Check artifact
Finished: FAILURE
```

Step 3: Open the ArcherySec page and log in using the credentials provided.

Credentials:

- **Username:** admin
- **Password:** admin



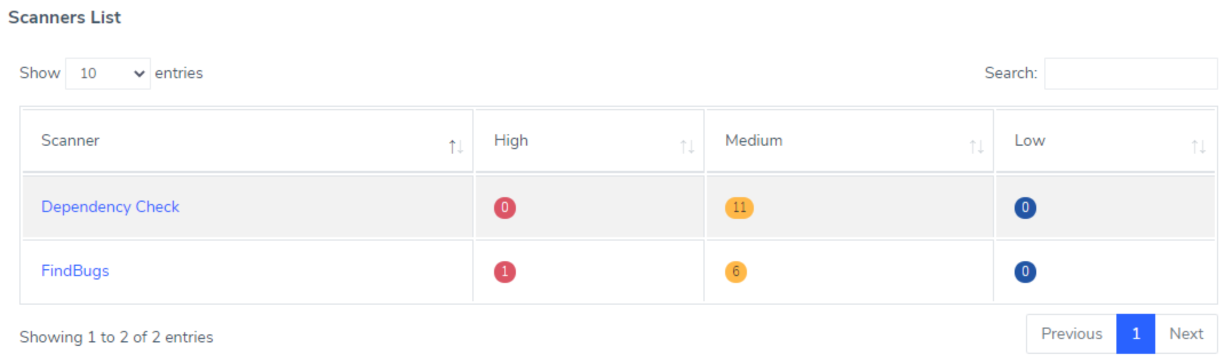
Project List

Show 10 entries

Project	Start Date	End Date	Owner
DevSecOps	2020-12-02	2020-12-02	test_project

Showing 1 to 1 of 1 entries

Click on the project name "DevSecOps" and see results from the bandit tool.



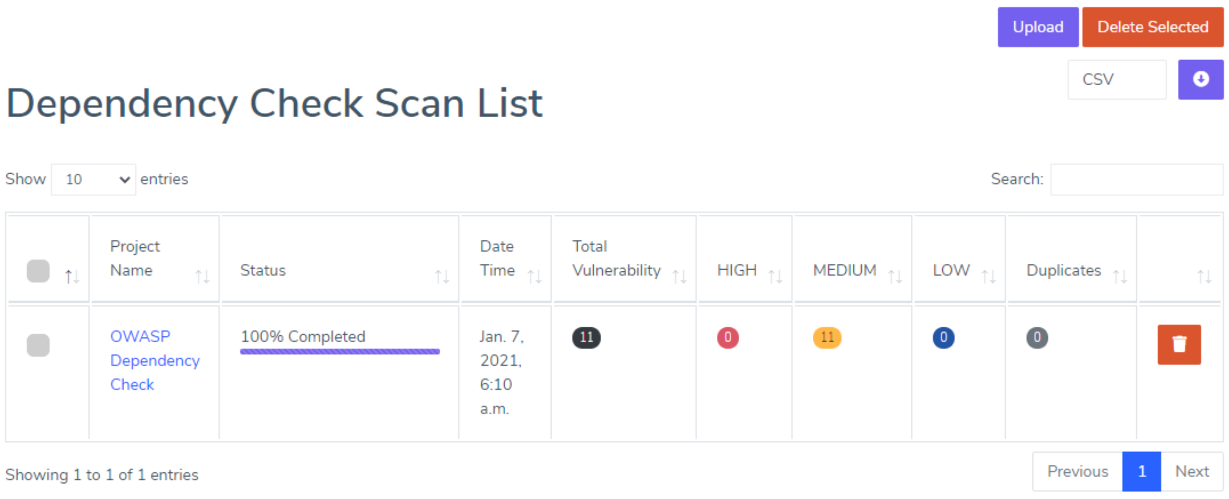
Scanners List

Show 10 entries Search:

Scanner	High	Medium	Low
Dependency Check	0	11	0
FindBugs	1	6	0

Showing 1 to 2 of 2 entries

Click on 'Dependency Check'.



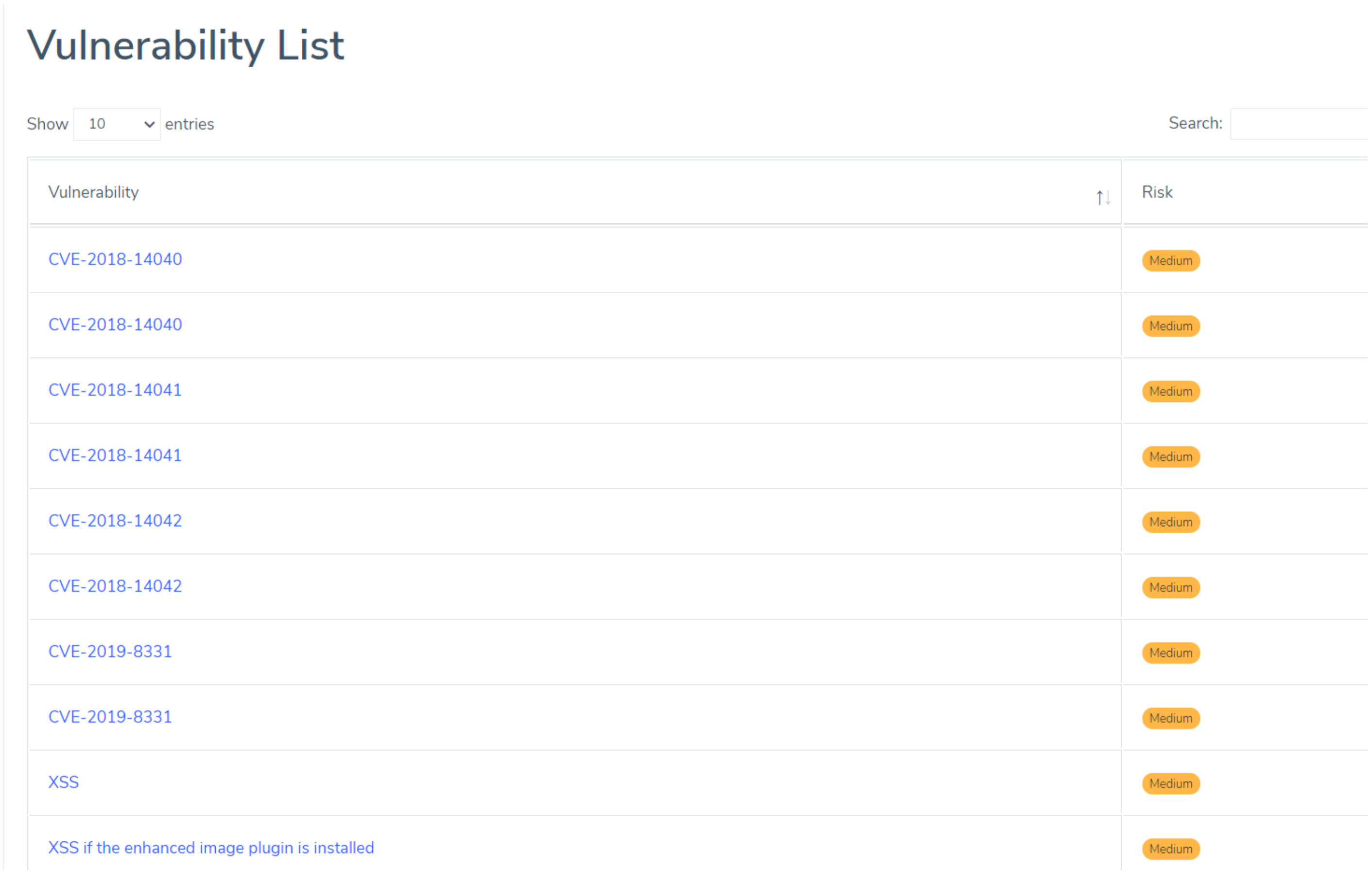
Dependency Check Scan List

Show 10 entries Search:

	Project Name	Status	Date Time	Total Vulnerability	HIGH	MEDIUM	LOW	Duplicates	
	OWASP Dependency Check	100% Completed	Jan. 7, 2021, 6:10 a.m.	11	0	11	0	0	

Showing 1 to 1 of 1 entries

Click on 'OWASP Dependency Check'.



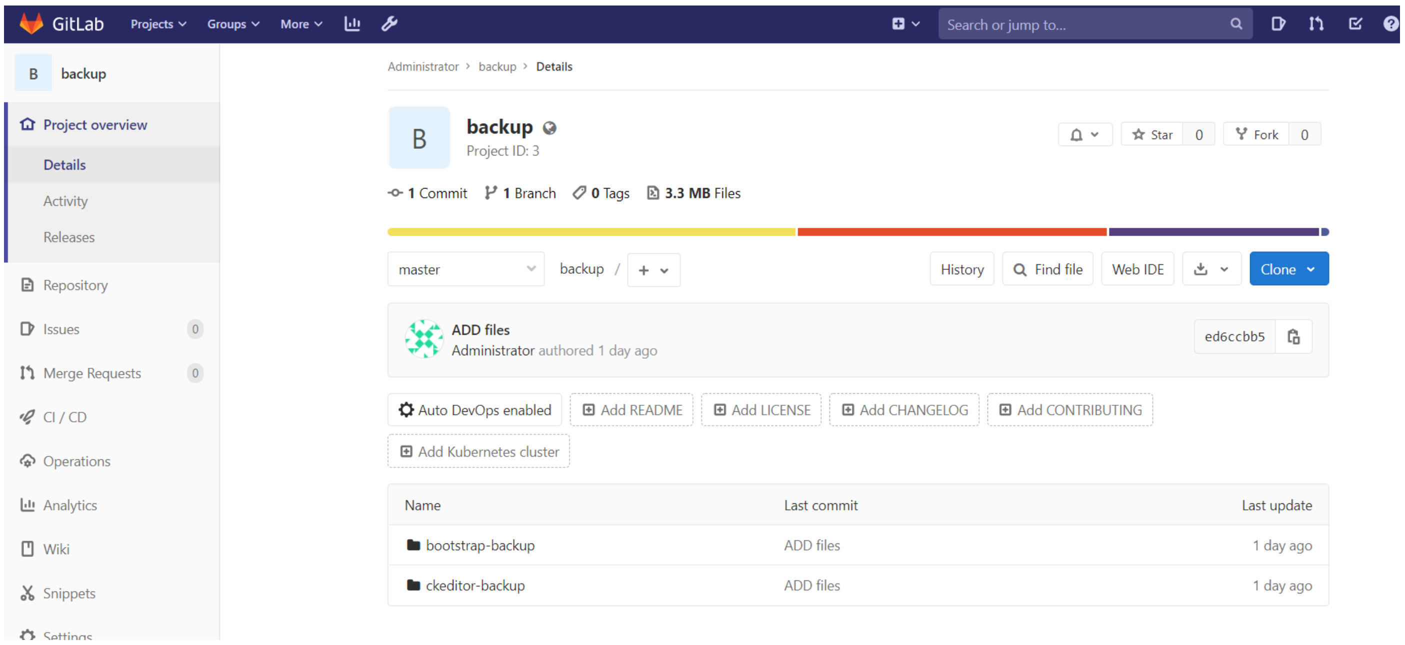
Vulnerability List

Show 10 entries Search:

Vulnerability	Risk
CVE-2018-14040	Medium
CVE-2018-14040	Medium
CVE-2018-14041	Medium
CVE-2018-14041	Medium
CVE-2018-14042	Medium
CVE-2018-14042	Medium
CVE-2019-8331	Medium
CVE-2019-8331	Medium
XSS	Medium
XSS if the enhanced image plugin is installed	Medium

Multiple CVE's have been found in the current version of Bootstrap library and XSS was found in ckeditor library.

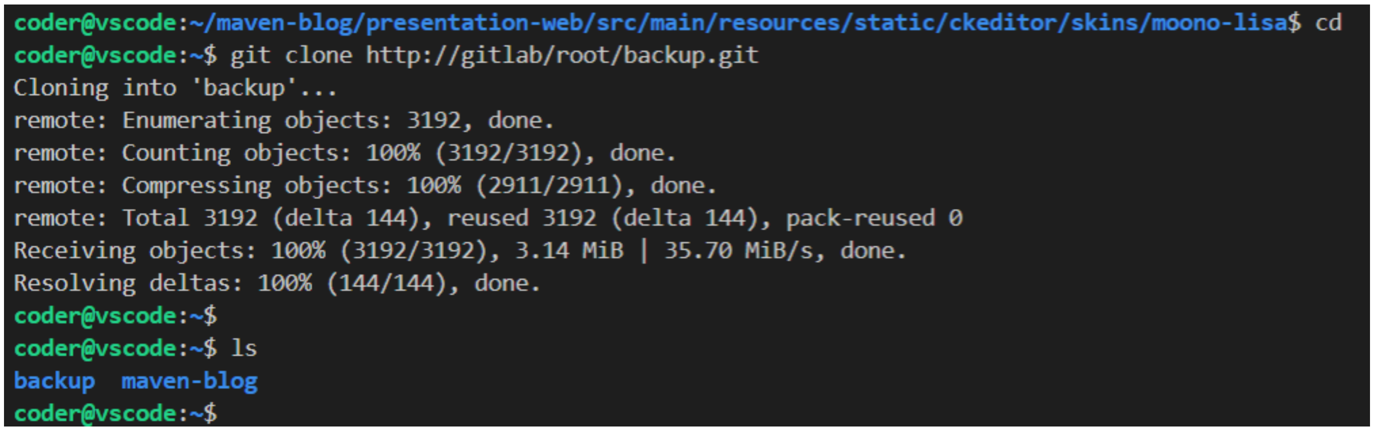
Step 4: Open the gitlab instance and navigate to the backup repository.



Name	Last commit	Last update
bootstrap-backup	ADD files	1 day ago
ckeditor-backup	ADD files	1 day ago

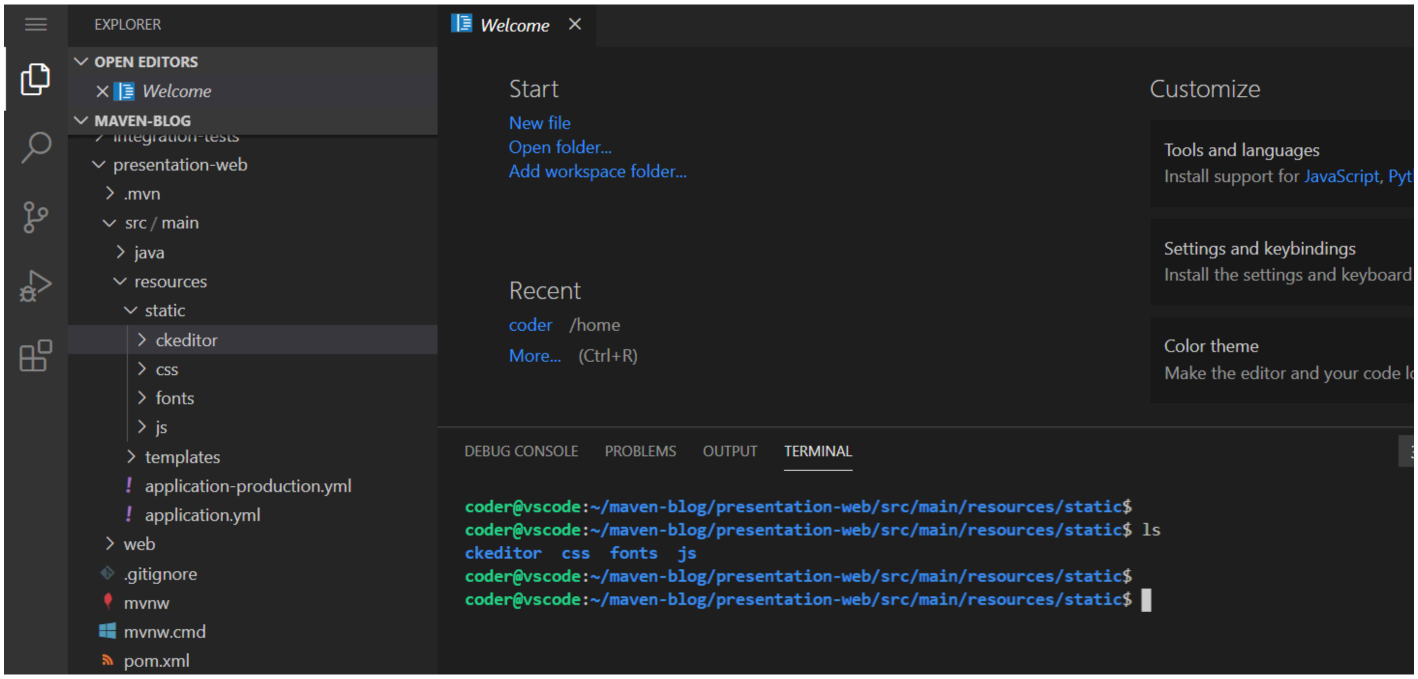
Step 5: Open a terminal and navigate to the home directory then clone the backup repository which contains the patched code version of these libraries. (On VS Code server)

Commands:
```
cd
git clone http://gitlab/root/backup.git
ls
```



```
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$ cd
coder@vscode:~$ git clone http://gitlab/root/backup.git
Cloning into 'backup'...
remote: Enumerating objects: 3192, done.
remote: Counting objects: 100% (3192/3192), done.
remote: Compressing objects: 100% (2911/2911), done.
remote: Total 3192 (delta 144), reused 3192 (delta 144), pack-reused 0
Receiving objects: 100% (3192/3192), 3.14 MiB | 35.70 MiB/s, done.
Resolving deltas: 100% (144/144), done.
coder@vscode:~$
coder@vscode:~$ ls
backup  maven-blog
coder@vscode:~$
```

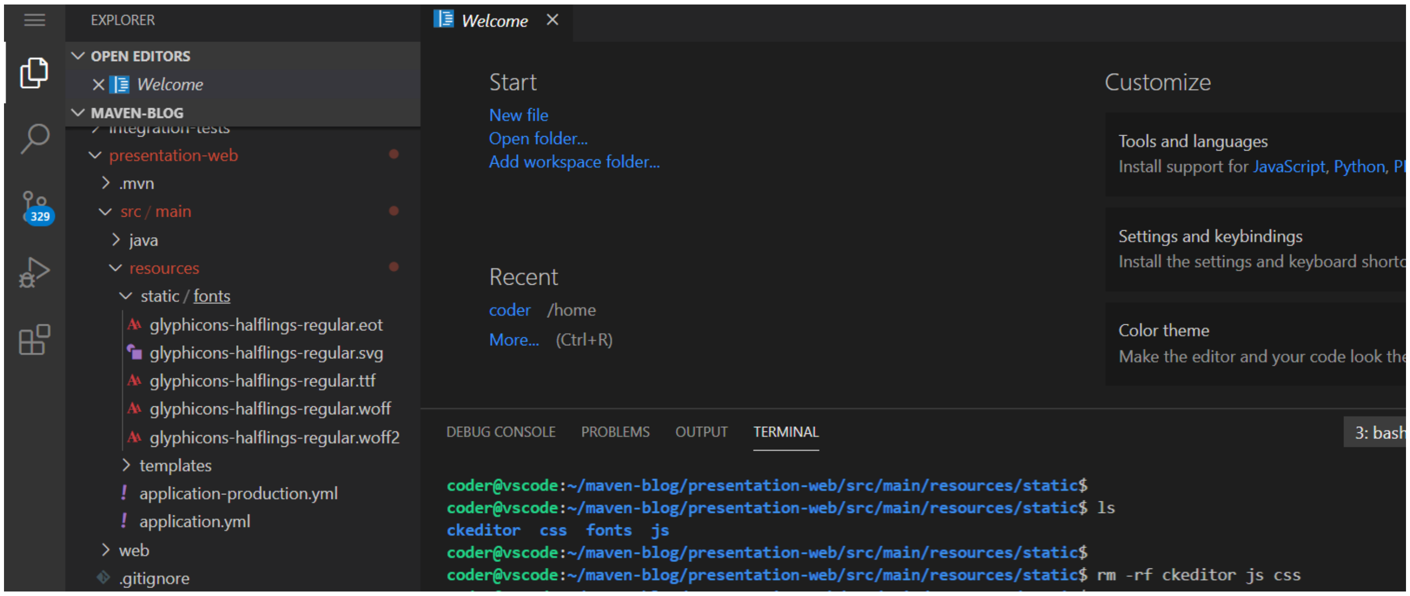
Step 6: Navigate to the directory where static resources of the website are stored such as ckeditor and bootstrap (~/maven-blog/presentation-web/src/main/resources/static)



```
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$ ls
ckeditor  css  fonts  js
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
```

Step 7: Delete the ckeditor directory with ‘js’ and ‘css’ directory.

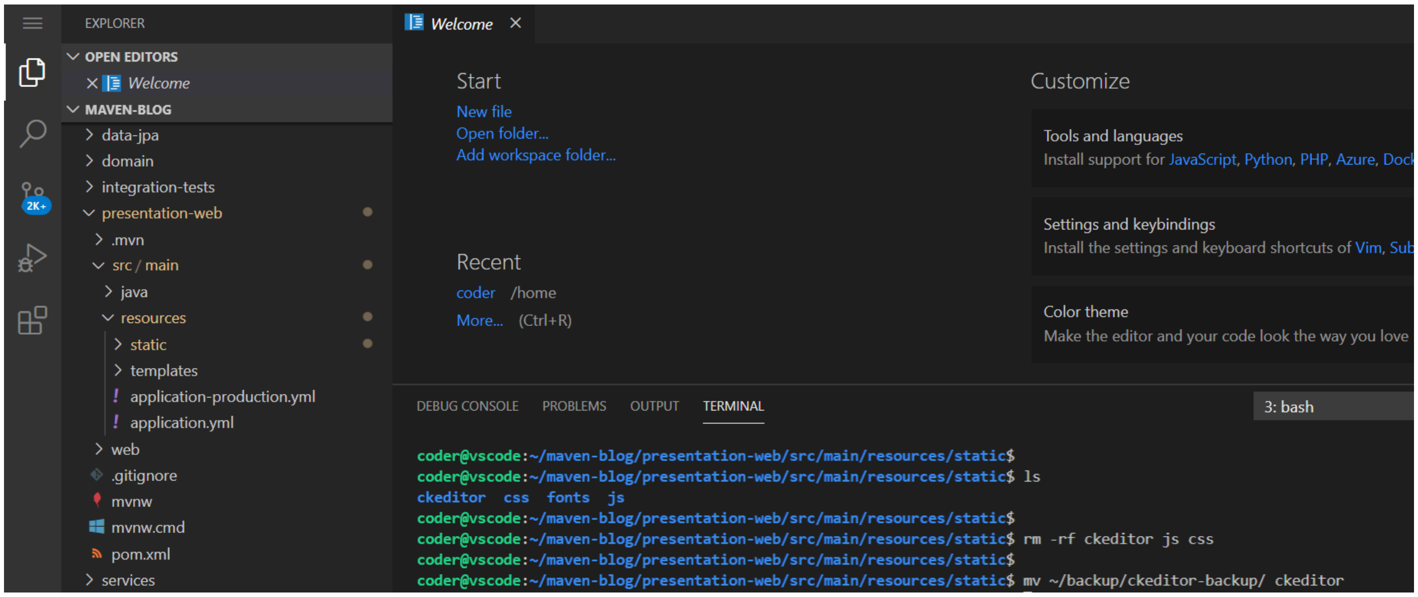
Command: rm -rf ckeditor js css



```
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$ ls
ckeditor  css  fonts  js
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$ rm -rf ckeditor js css
```

Step 8: Move the ckeditor-backup directory from the ‘backup’ directory and save it as ‘ckeditor’ in the current directory.

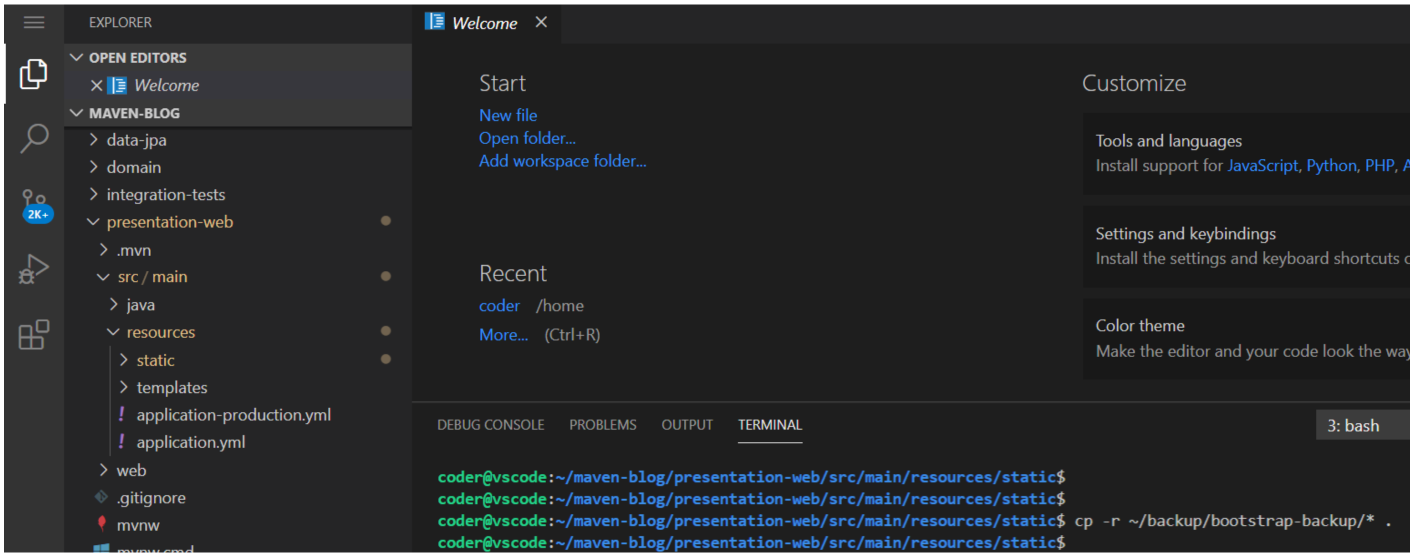
Command: mv ~/backup/ckeditor-backup/ ckeditor



```
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$ ls
ckeditor  css  fonts  js
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$ rm -rf ckeditor js css
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$ mv ~/backup/ckeditor-backup/ ckeditor
```

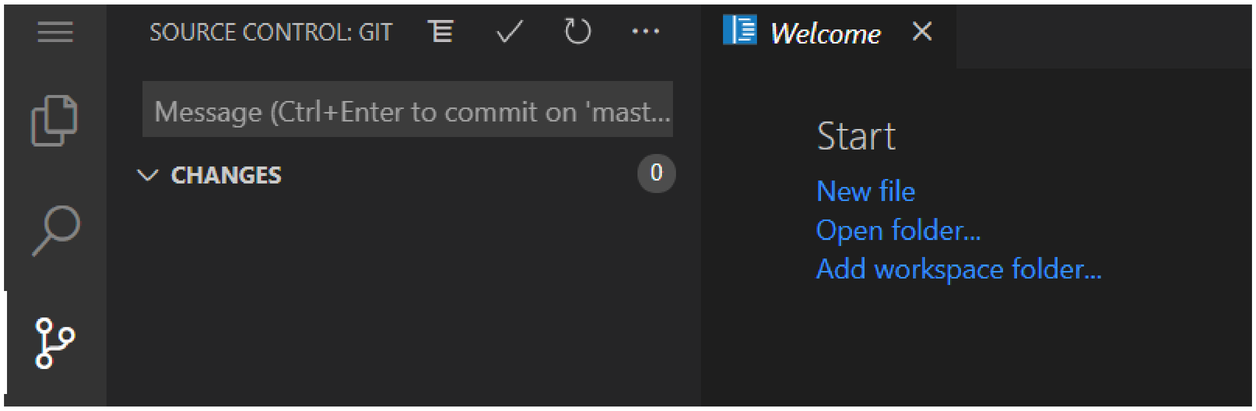
Step 9: Copy the files stored inside bootstrap-backup located inside the ‘backup’ directory.

Command: cp -r ~/backup/bootstrap-backup/* .

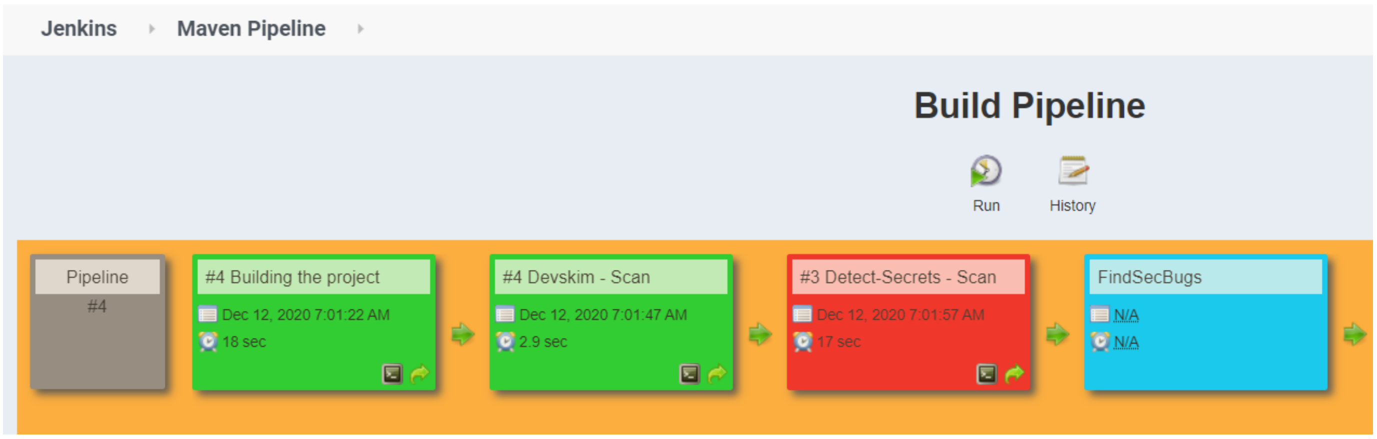


```
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$ cp -r ~/backup/bootstrap-backup/* .
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static$
```

Step 10: Commit and push all the changes to the remote repository.
(Same as Step 10 onwards of DevSkim issue section)

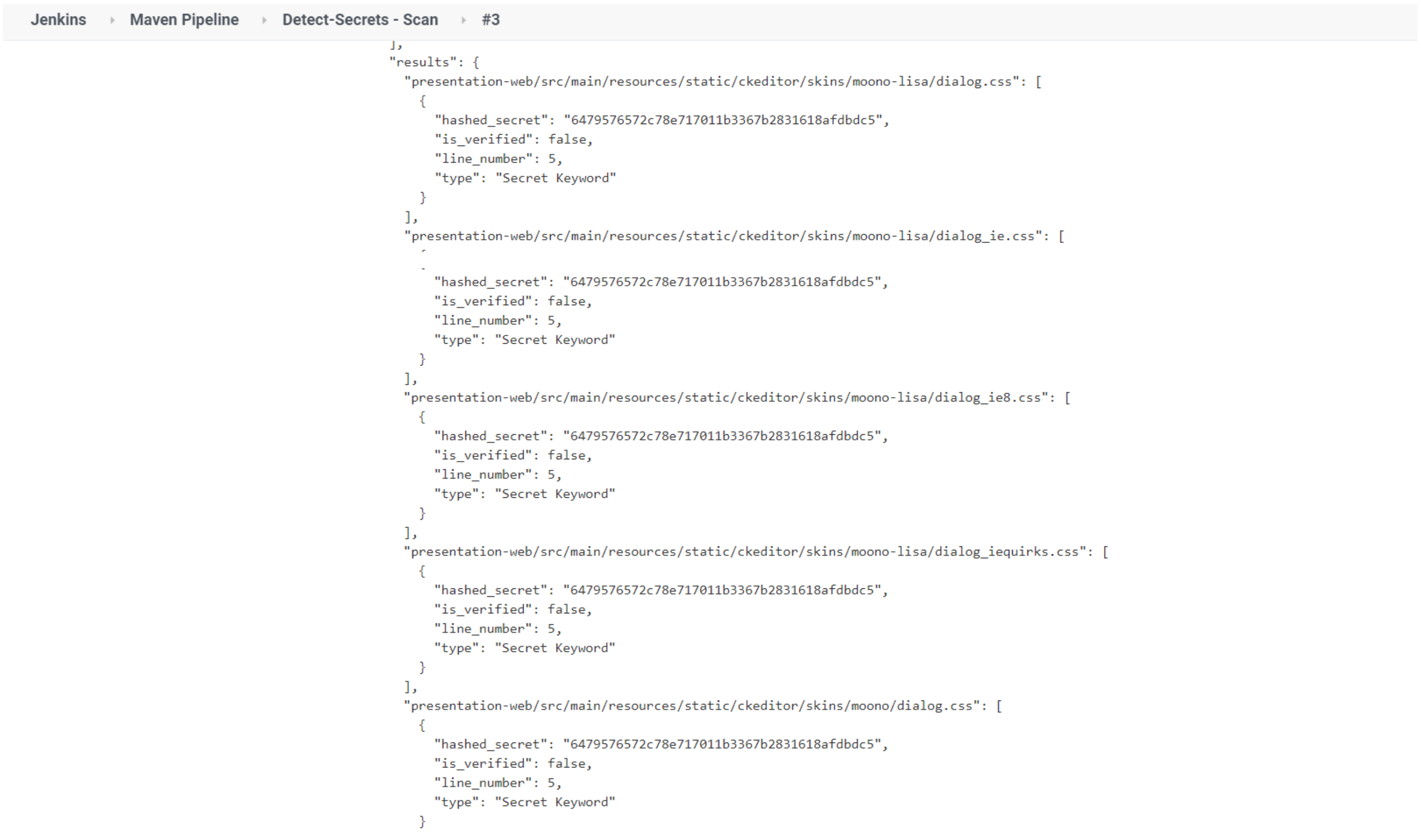


Step 11: Check the pipeline to see any changes.

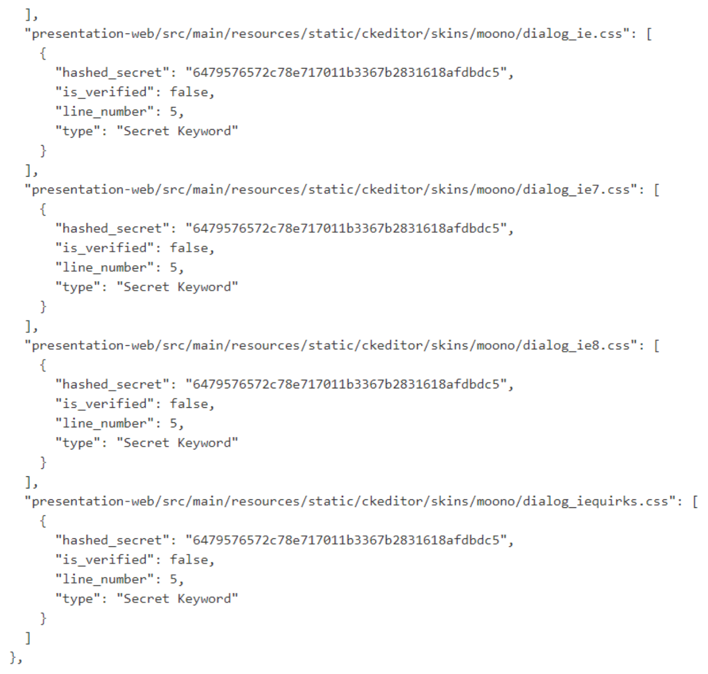

Jenkins › Maven Pipeline ›

Build Pipeline

The same strings are flagged by detect-secrets again, Check the console logs of Detect-Secrets in order to see what files are flagged.


Jenkins › Maven Pipeline › Detect-Secrets - Scan › #3

```
],
"results": {
  "presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/dialog.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/dialog_ie.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/dialog_ie8.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/dialog_iequirks.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono/dialog.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
```

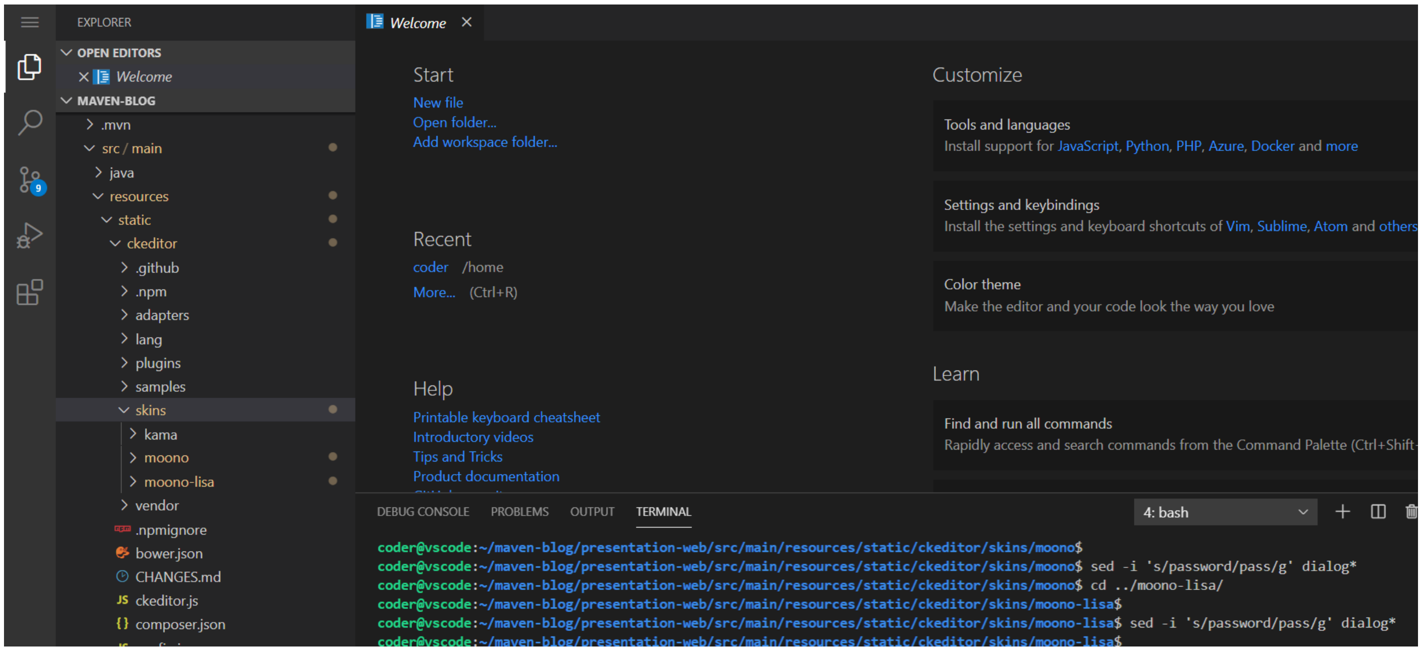


```
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono/dialog_ie.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono/dialog_ie7.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono/dialog_ie8.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ],
  "presentation-web/src/main/resources/static/ckeditor/skins/moono/dialog_iequirks.css": [
    {
      "hashed_secret": "6479576572c78e717011b3367b2831618afdbdc5",
      "is_verified": false,
      "line_number": 5,
      "type": "Secret Keyword"
    }
  ]
},
```

Step 12: Navigate inside the skins directory and replace the ‘password’ string from the files.

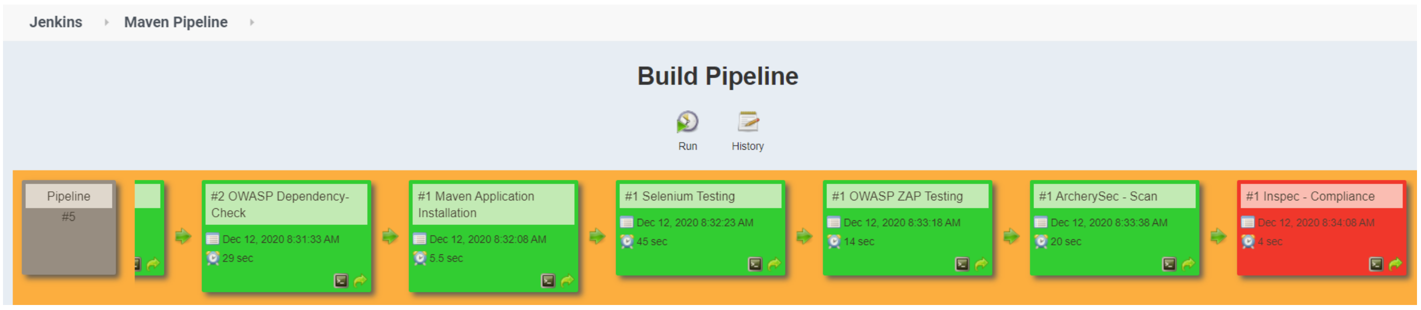
Commands:

```
cd ~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa/
sed -i 's/password/pass/g' dialog*
cd ~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono/
sed -i 's/password/pass/g' dialog*
```

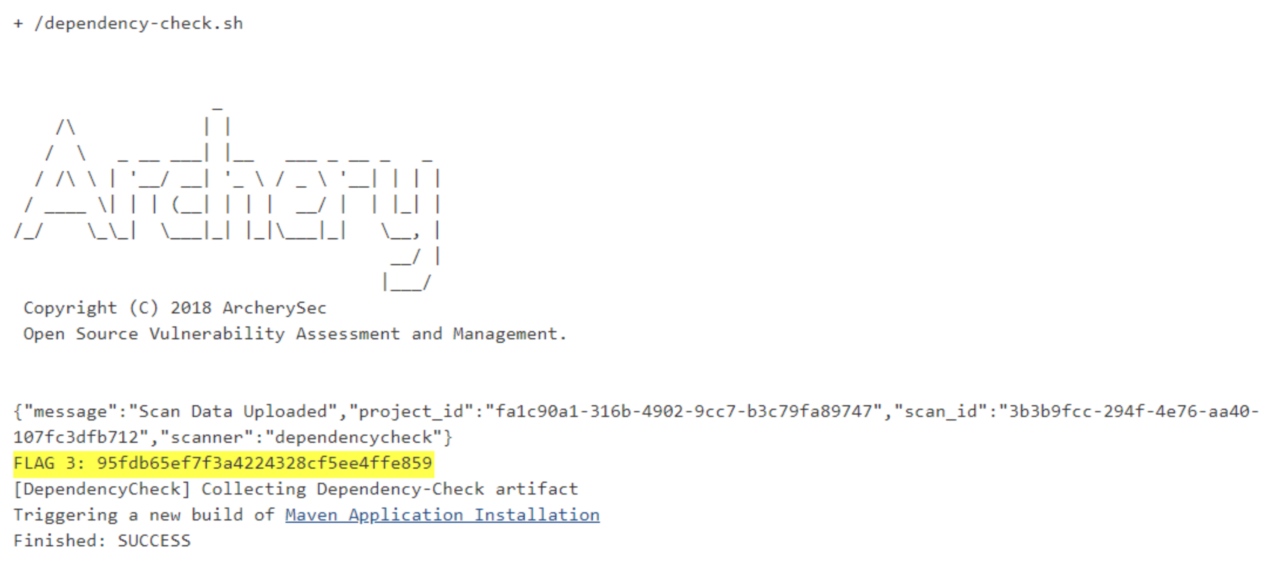


```
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono$ sed -i 's/password/pass/g' dialog*
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono$ cd ../moono-lisa/
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$ sed -i 's/password/pass/g' dialog*
coder@vscode:~/maven-blog/presentation-web/src/main/resources/static/ckeditor/skins/moono-lisa$
```

Step 13: Commit and push all the changes to the remote repository. Check the changes in the pipeline
(Same as Step 10 onwards of the DevSkim issue section)



The OWASP Dependency-Check stage has passed successfully. The logs can be checked from clicking on the “OWASP Dependency-Check” and opening the console output.

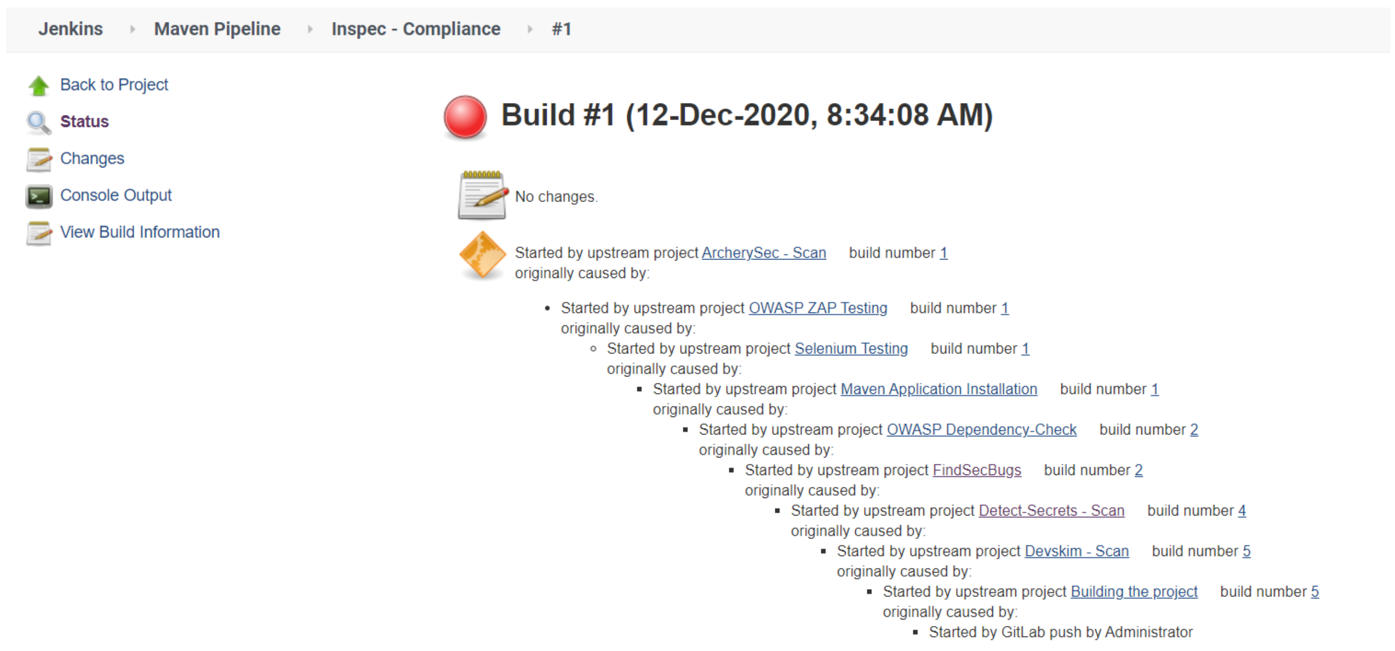


```
+ /dependency-check.sh

                    _
    /\             | |
   /  \   _ __ ___| |__   ___ _ __ _   _
  / /\ \ | '__/ __| '_ \ / _ \ '__| | | |
 / ____ \| | | (__| | | |  __/ |  | |_| |
/_/    \_\_|  \___|_| |_|\___|_|   \__, |
                                    __/ |
                                   |___/
 Copyright (C) 2018 ArcherySec
 Open Source Vulnerability Assessment and Management.


{"message":"Scan Data Uploaded","project_id":"fa1c90a1-316b-4902-9cc7-b3c79fa89747","scan_id":"3b3b9fcc-294f-4e76-aa40-
107fc3dfb712","scanner":"dependencycheck"}
FLAG 3: 95fdb65ef7f3a4224328cf5ee4ffe859
[DependencyCheck] Collecting Dependency-Check artifact
Triggering a new build of Maven Application Installation
Finished: SUCCESS
```

FLAG 3: 95fdb65ef7f3a4224328cf5ee4ffe859

Inspec Issue

Step 1: Click on the ‘Inspec’ to check the job build page.



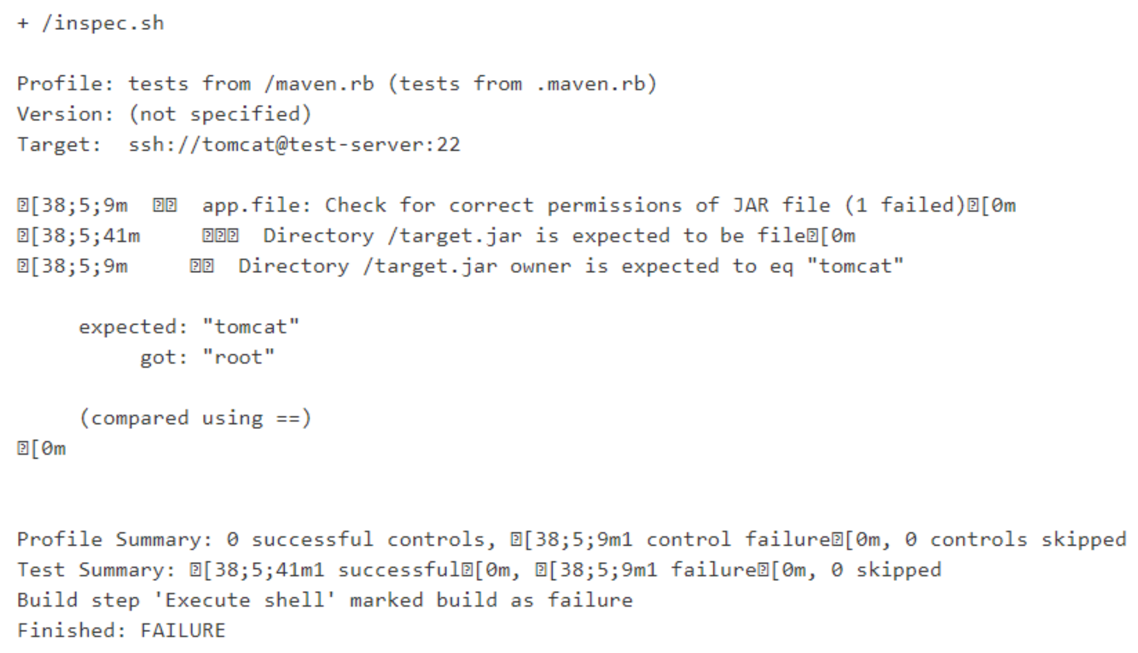
Build #1 (12-Dec-2020, 8:34:08 AM)

No changes.

Started by upstream project ArcherySec - Scan build number 1
originally caused by:

- Started by upstream project OWASP ZAP Testing build number 1
  originally caused by:
  - Started by upstream project Selenium Testing build number 1
    originally caused by:
    - Started by upstream project Maven Application Installation build number 1
      originally caused by:
      - Started by upstream project OWASP Dependency-Check build number 2
        originally caused by:
        - Started by upstream project FindSecBugs build number 2
          originally caused by:
          - Started by upstream project Detect-Secrets - Scan build number 4
            originally caused by:
            - Started by upstream project Devskim - Scan build number 5
              originally caused by:
              - Started by upstream project Building the project build number 5
                originally caused by:
                - Started by GitLab push by Administrator

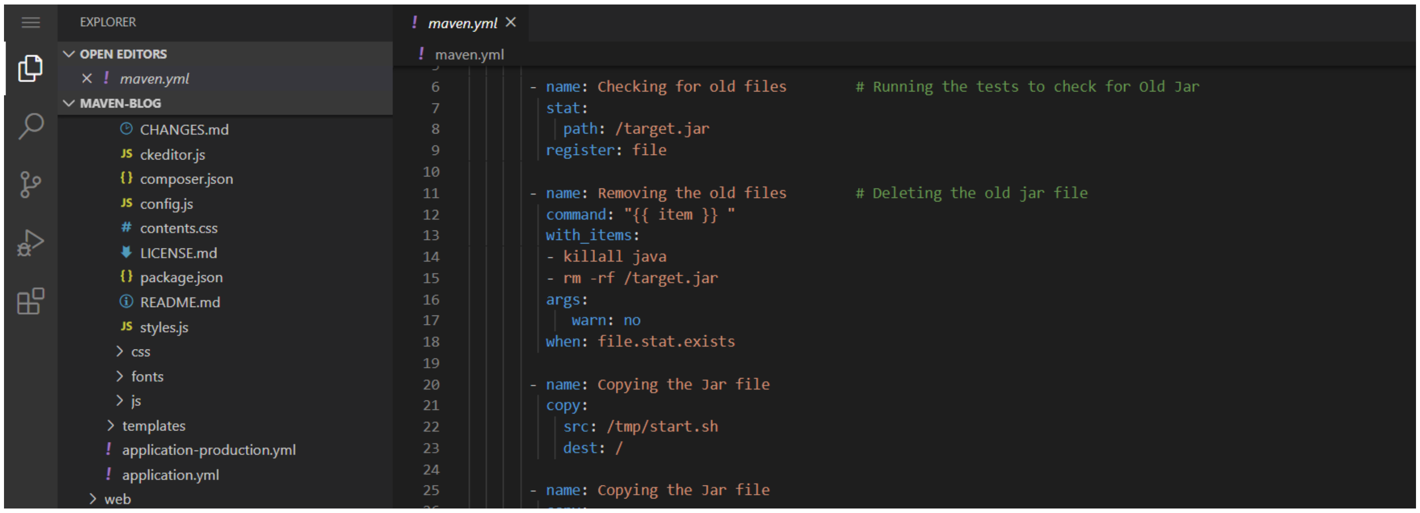
Step 2: Click on the “Console Output” to check the issues found in Inspec.

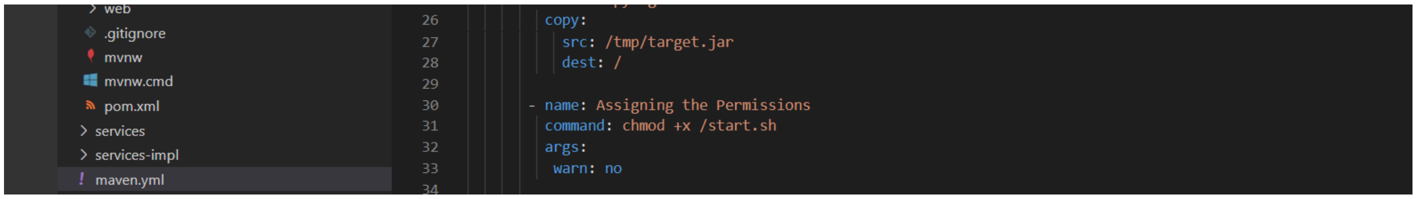


```
+ /inspec.sh

Profile: tests from /maven.rb (tests from .maven.rb)
Version: (not specified)
Target:  ssh://tomcat@test-server:22

[38;5;9m    app.file: Check for correct permissions of JAR file (1 failed)[0m
[38;5;41m        Directory /target.jar is expected to be file[0m
[38;5;9m        Directory /target.jar owner is expected to eq "tomcat"

     expected: "tomcat"
          got: "root"

     (compared using ==)
[0m


Profile Summary: 0 successful controls, [38;5;9m1 control failure[0m, 0 controls skipped
Test Summary: [38;5;41m1 successful[0m, [38;5;9m1 failure[0m, 0 skipped
Build step 'Execute shell' marked build as failure
Finished: FAILURE
```

The file /target.jar in the remote server needs to be owned by user tomcat but is currently owned by user root.

Step 3: Open the maven.yml file which is used by ansible to deploy the application on the remote server. (On VS Code server)


```
- name: Checking for old files        # Running the tests to check for Old Jar
  stat:
    path: /target.jar
  register: file

- name: Removing the old files        # Deleting the old jar file
  command: "{{ item }} "
  with_items:
  - killall java
  - rm -rf /target.jar
  args:
     warn: no
  when: file.stat.exists

- name: Copying the Jar file
  copy:
    src: /tmp/start.sh
    dest: /

- name: Copying the Jar file
```


```
  copy:
    src: /tmp/target.jar
    dest: /

- name: Assigning the Permissions
  command: chmod +x /start.sh
  args:
   warn: no
```

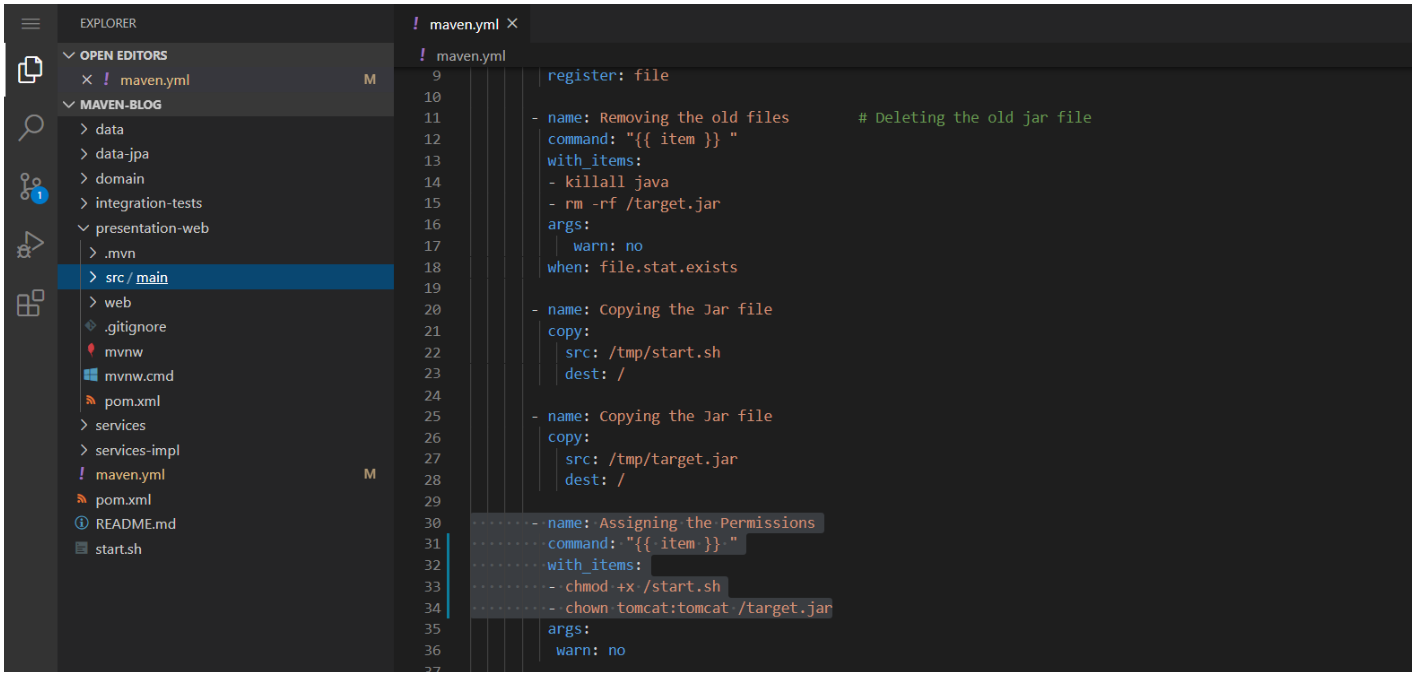
Step 4: Place command to create and set the ownership on the target.jar file.

From:

```
- name: Assigning the Permissions
  command: chmod +x /start.sh
  args:
   warn: no
```

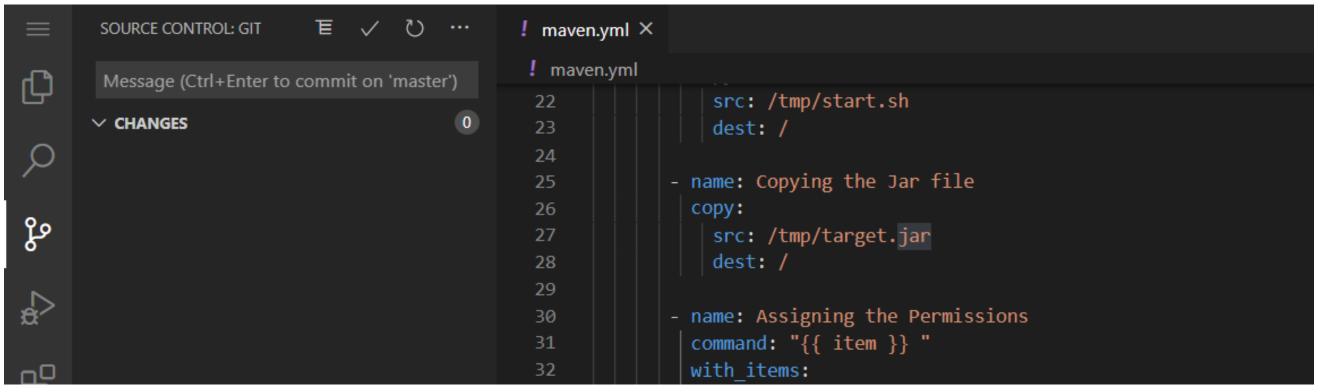
To:

```
 - name: Assigning the Permissions
   command: "{{ item }} "
   with_items:
   - chmod +x /start.sh
   - chown tomcat:tomcat /target.jar
   args:
    warn: no
```



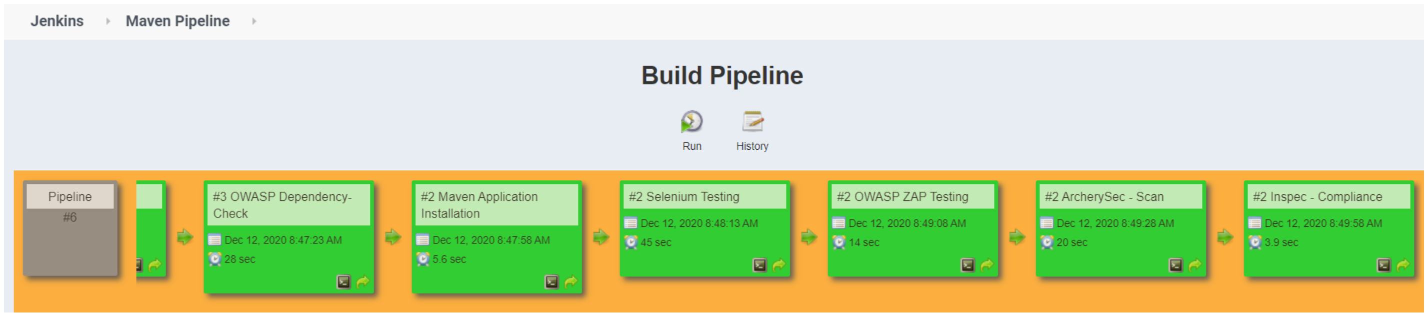
```
    register: file

  - name: Removing the old files       # Deleting the old jar file
    command: "{{ item }} "
    with_items:
    - killall java
    - rm -rf /target.jar
    args:
       warn: no
    when: file.stat.exists

  - name: Copying the Jar file
    copy:
      src: /tmp/start.sh
      dest: /

  - name: Copying the Jar file
    copy:
      src: /tmp/target.jar
      dest: /

  - name: Assigning the Permissions
    command: "{{ item }} "
    with_items:
    - chmod +x /start.sh
    - chown tomcat:tomcat /target.jar
    args:
     warn: no
```

Commit the changes and push the files to the remote repository.
(Same as Step 10 onwards of the DevSkim issue section)



```
      src: /tmp/start.sh
      dest: /

  - name: Copying the Jar file
    copy:
      src: /tmp/target.jar
      dest: /

  - name: Assigning the Permissions
    command: "{{ item }} "
    with_items:
```

Step 5: Check the pipeline for the changes


Jenkins › Maven Pipeline ›

Build Pipeline

Run History

Pipeline #6	#3 OWASP Dependency-Check	#2 Maven Application Installation	#2 Selenium Testing	#2 OWASP ZAP Testing	#2 ArcherySec - Scan	#2 Inspec - Compliance
	Dec 12, 2020 8:47:23 AM	Dec 12, 2020 8:47:58 AM	Dec 12, 2020 8:48:13 AM	Dec 12, 2020 8:49:08 AM	Dec 12, 2020 8:49:28 AM	Dec 12, 2020 8:49:58 AM
	28 sec	5.6 sec	45 sec	14 sec	20 sec	3.9 sec

The Inspec stage has passed successfully. The logs can be checked from clicking on the "Inspec" and opening the console output.

```
+ /inspec.sh

Profile: tests from /maven.rb (tests from .maven.rb)
Version: (not specified)
Target:  ssh://tomcat@test-server:22

⍰[38;5;41m  ⍰⍰⍰  app.file: Check for correct permissions of JAR file⍰[0m
⍰[38;5;41m     ⍰⍰⍰  Directory /target.jar is expected to be file⍰[0m
⍰[38;5;41m     ⍰⍰⍰  Directory /target.jar owner is expected to eq "tomcat"⍰[0m


Profile Summary: ⍰[38;5;41m1 successful control⍰[0m, 0 control failures, 0 controls skipped
Test Summary: ⍰[38;5;41m2 successful⍰[0m, 0 failures, 0 skipped
FLAG 4: e371c83cd17a24349be7c87fb309bdaa
Finished: SUCCESS
```

FLAG 4: e371c83cd17a24349be7c87fb309bdaa

Learning

Working on a simple DevSecOps pipeline consisting of different components to fix the issues present in the pipeline