# Discovery 4: Learn to Work with Git

## Introduction

In this activity, you will learn to work with Git, the open-source distributed source control system. You will also learn how to manage Git repositories with GitLab, a very popular Git management software.

After completing this activity, you will be able to meet these objectives:

- Manage configuration files with Git.
- Understand basic Git workflow and how it can be useful with NSO operations.

## Job Aid

The following job aid is available to help you complete the lab activities:

- This Lab Guide

The following table contains passwords that you might need.

| Device | Username | Password |
|---|---|---|
| student-vm | student | 1234QWer |
| nso-server | student | 1234QWer |
| NSO application | admin | admin |
| GitLab | student | 1234QWer |

### Required Resources

The following resources and equipment are required for completing the activities in this lab guide:

- PC or laptop with a web browser
- Access to the internet

## Command List

The following are the most common commands that you will need:

### Linux Shell:

| Command | Comment |
|---|---|
| **source /opt/ncs/ncs-6.1/ncsrc** | Source NSO environmental variable in Docker container. |
| **ls\|ll** | Display contents of the current directory. |
| **cd** | Move directly to user home directory. |

| Command | Comment |
|---|---|
| **cd ..** | Exit out of current directory. |
| **cd test** | Move into the "test" folder, which is a subfolder of the current directory. |
| **cd /home/student** | Move into the "student" folder by specifying the direct path to it starting from the root of the directory system. |
| **ncs_cli -C** | Log in to NSO CLI directly from local server. |

### NSO CLI:

| Command | Comment |
|---|---|
| **switch cli** | Change CLI style. |
| **show ?** | Display all command options for current mode. |
| **configure** | Enter configuration mode. |
| **commit** | Commit new configuration (configuration mode only command). |
| **show configuration** | Display new configuration that has not yet been committed (configuration mode only command). |

### Makefile commands for Docker environment:

| Command | Comment |
|---|---|
| **make build** | Builds the main NSO Docker image. |
| **make testenv-start** | Starts the NSO Docker environment. |
| **make testenv-stop** | Stops the NSO Docker environment. |
| **make testenv-build** | Recompiles and reloads the NSO packages. |
| **make testenv-cli** | Enters the NSO CLI of the NSO Docker container. |
| **make testenv-shell** | Enters the Linux shell of the NSO Docker container. |
| **make dev-shell** | Enters the Linux shell of the NSO Docker development container. |

## Command Syntax Reference

This lab guide uses the following conventions for **command syntax**:

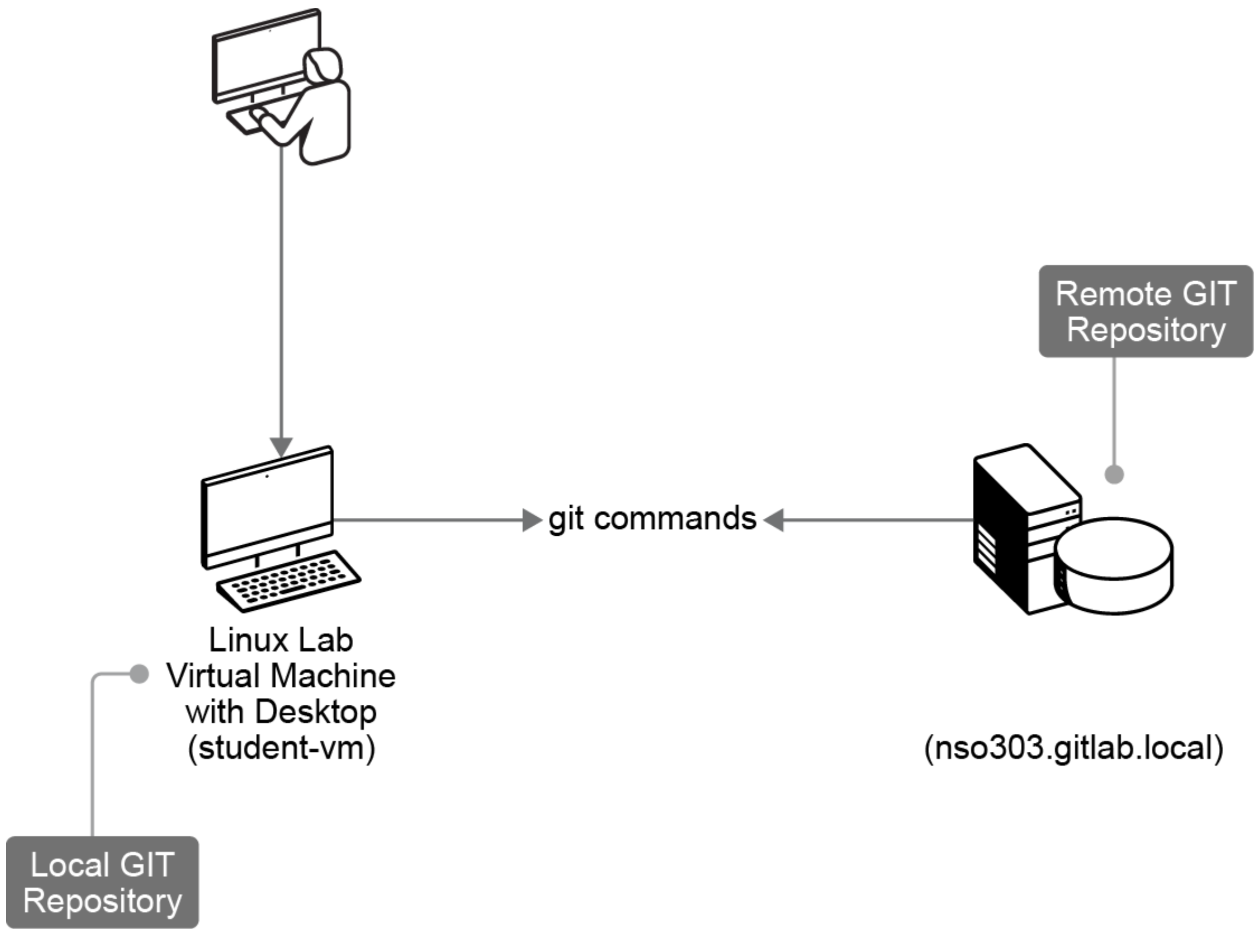
| Formatting | Description and Examples |
|---|---|
| **show running config** | Commands in steps use this formatting. |
| *Example* | Type **show running config** |
| *Example* | Use the **name** command. |
| `show running` | Commands in CLI outputs and configurations use this formatting. |

| **Formatting** | **Description and Examples** |
|---|---|
| **`config`** | |
| highlight | CLI output that is important is highlighted. |
| *Example* | `student@student-vm:~$` **`ncs --version`**<br>`        6.1` |
| *Example* | Save your current configuration as the default **startup config**.<br><br>`Router Name#` **`copy running startup`** |
| brackets ([ ]) | Indicates optional element. You can choose one of the options. |
| *Example*: | `(config-if)#` **`frame-relay lmi-type {ansi|cisco|q933a}`** |
| *italics font* | Arguments for which you supply values. |
| *Example* | Open the **ip tcp window-size** *bytes* file |
| angle brackets (<>) | In contexts that do not allow italics, arguments for which you supply values are enclosed in angle brackets [<>]. Do not type the brackets when entering the command. |
| *Example* | If the command syntax is **ping** <*ip_address*>, you enter ping *10.0.0.102* |
| string | A non-quoted set of characters. Type the characters as-is. |
| *Example* | (config)# **hostname MyRouter** |
| vertical line (\|) | Indicates that you enter one of the choices. The vertical line separates choices. Do not type the vertical line when entering the command. |
| *Example* | If the command syntax is **show ip route\|arp**, you enter either **show ip route** or **show ip arp**, but not both. |

## Lab Topology Information

Your lab session is your own personal sandbox. Whatever you do in your session will not be reflected in anyone else's session. Your lab environment is a Linux server (Student-VM) acting as a jumphost and a Linux server (NSO-server) acting as an NSO and GitLab server withing a Docker container.

## Topology



## Task 1: Setup and Clone a Remote Repository

In this task, you will connect to a remote GitLab server running as a Docker container in your lab environment and clone an already prepared project to your workstation machine.

### Activity

Complete these steps:

#### *Step 1*

Connect to the Student-VM.

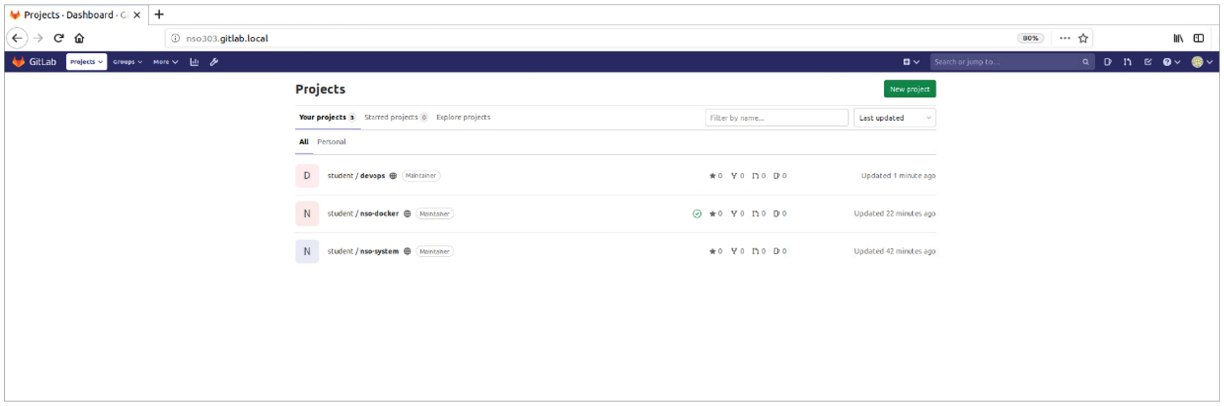
You can connect to the server either by choosing the **Student-VM** from the device list or by clicking on the **Student-VM** icon in the topology map.

#### *Step 2*

Sign in to the GitLab WebUI on *http://nso303.gitlab.local*.

Open a web browser, navigate to the GitLab WebUI and sign in with the **student** username and **1234QWer** password. The address of this user interface is *http://nso303.gitlab.local*.

Under **Projects**, you see a couple of already created projects.

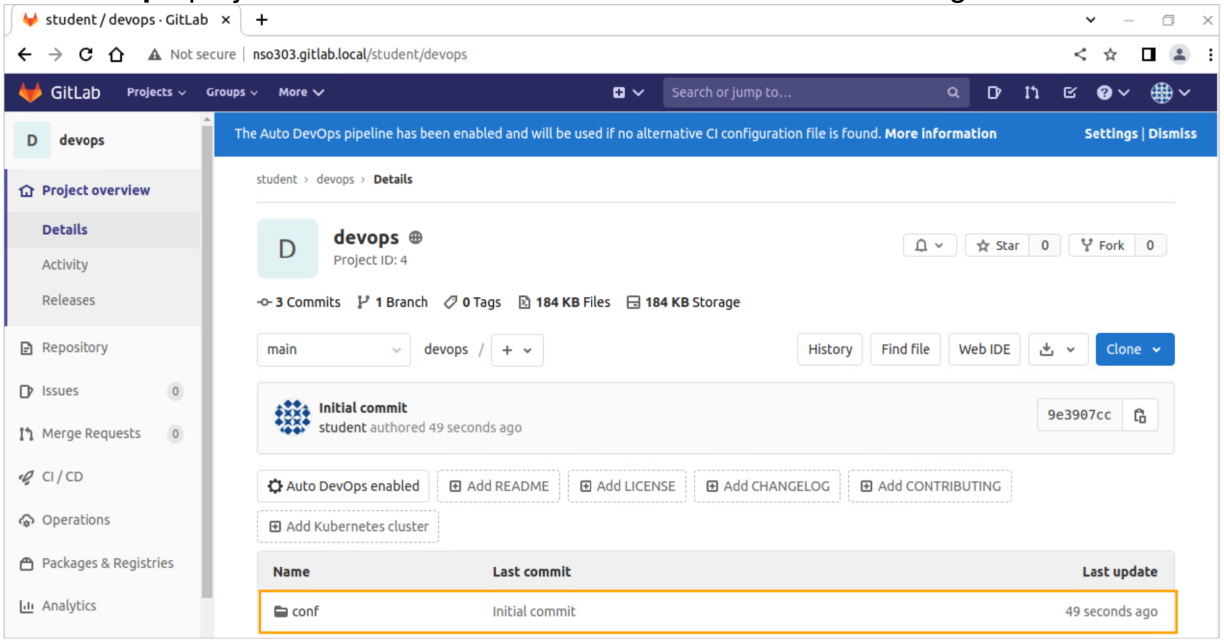


| | | |
|---|---|---|
| D | student / **devops** Maintainer | Updated 1 minute ago |
| N | student / **nso-docker** Maintainer | Updated 22 minutes ago |
| N | student / **nso-system** Maintainer | Updated 42 minutes ago |

### *Step 3*

Click the **devops** project and view files and folders that are stored there.

The **devops** project contains a **conf** folder that contains NSO configuration.



The Auto DevOps pipeline has been enabled and will be used if no alternative CI configuration file is found. **More information**

**Initial commit**
student authored 49 seconds ago
9e3907cc

| Name | Last commit | Last update |
|---|---|---|
| conf | Initial commit | 49 seconds ago |

### *Step 4*

Create a Git directory within your home folder.

Open the terminal by clicking the terminal icon on the bottom bar and create a Git directory within your home folder. Navigate to that folder.

```
student@student-vm:~$ mkdir Git
student@student-vm:~$ cd Git/
student@student-vm:~/Git$
```

### *Step 5*

Configure a name and an email address for this repository.

Use the following commands:

```
student@student-vm:~/Git$ git config --global user.name student
student@student-vm:~/Git$ git config --global user.email
student@gitlab.local
student@student-vm:~/Git$
```

You can also save your username and password with the **git config --global credential.helper store** command.

### *Step 6*

Clone the **http://nso303.gitlab.local/student/devops.git** repository to your Git folder.

Clone the repository from the GitLab server to your workstation with the **git clone <url>** command, so that you are able to work with files in your local repository. If asked for credentials, use the same **student** and **1234QWer** credentials as you did for the WebUI.

```
student@student-vm:~/Git$ git clone http://nso303.gitlab.local/student/
devops.git
Cloning into 'devops'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 4 (delta 0), reused 4 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), 2.71 KiB | 1.36 MiB/s, done.





student@student-vm:~/Git$
```

You get repository link by clicking on the **Clone** button. You will get a list of cloning options, including the location of the repository.

### *Step 7*

Verify that the contents of your devops directory is the same as on the remote repository.

List the contents of your devops directory with the **ls** command. You can see the same set of files as on the remote repository that you now have downloaded locally. You also have the .git file where everything that git needs for keeping track of your project is stored.

```
student@student-vm:~/Git$ ls -la devops/
total 16
drwxrwxr-x 4 student student 4096 Oct 10 16:16 .
drwxrwxr-x 3 student student 4096 Oct 10 16:16 ..
```

```
drwxrwxr-x 2 student student 4096 Oct 10 16:16 conf
drwxrwxr-x 8 student student 4096 Oct 10 16:16 .git
student@student-vm:~/Git$
```

***Step 8***

Review the status of the local repository with the **git status** command.

Move into the devops directory and use the **git status** command. Since no changes have been done yet, there is nothing to commit.

```
student@student-vm:~/Git$ cd devops/
student@student-vm:~/Git/devops$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
student@student-vm:~/Git/devops$
```

***Step 9***

Create and open the README.md file and add some information.

Create and open the README.md file with the **nano** command and add some information. Save the file and exit the file editor.

```
student@student-vm:~/Git/devops$ nano README.md
This repository contains backup NSO configuration.
```

***Step 10***

Use the **git status** command again and check the output.

The command tells you that there have been some changes; however, they have not been added to the staging area.

```
student@student-vm:~/Git/devops$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        README.md

nothing added to commit but untracked files present (use "git add" to
track)
student@student-vm:~/Git/devops$
```

***Step 11***

Add the changes to the repository.

To add the changes to the repository, you first need to start tracking them. To do this action, use the **git add <folder>** command. Then, use the **git status** command again.

```
student@student-vm:~/Git/devops$ git add .
student@student-vm:~/Git/devops$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   README.md

student@student-vm:~/Git/devops$
```

***Step 12***

Commit the changes to your local repository with the **git commit** command.

Specify the commit message. It should describe the changes that are included in this commit.

If you run the git status afterwards, you can see that the local copy of the main branch is ahead of the “origin/main” branch on the GitLab server for one commit. This means that the README file is currently part of your local repository but not on the GitLab remote server yet.

```
student@student-vm:~/Git/devops$ git commit -m "Adding README"
[main 5a8e31d] Adding README
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
student@student-vm:~/Git/devops$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
student@student-vm:~/Git/devops$
```

***Step 13***

Publish the commit to the remote repository.

To publish the commit, use the **git push origin main** command. Your local main branch will be synchronized with the remote one. Running the **git status** command tells you that you are back to a clean working directory.

```
student@student-vm:~/Git/devops$ git push origin main
Username for 'http://nso303.gitlab.local': student
Password for 'http://student@nso303.gitlab.local':
Counting objects: 4, done.
```

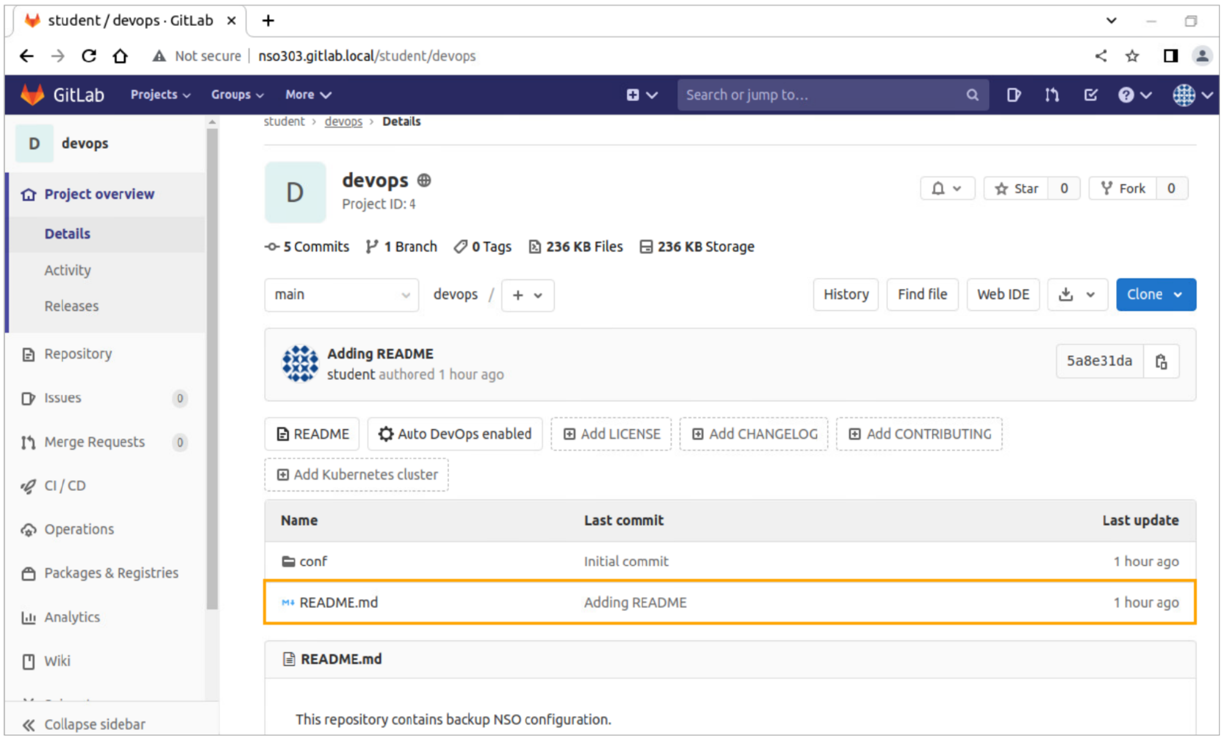
```
Delta compression using up to 2 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (3/3), 323 bytes | 323.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To http://nso303.gitlab.local/student/devops.git
   9e3907c..5a8e31d  main -> main
student@student-vm:~/Git/devops$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
student@student-vm:~/Git/devops$
```

### *Step 14*

Verify if the changes are reflected in the GitLab WebUI.

Head back to the browser and refresh the devops project page. You see that there is a new commit added and that the README file has been created.



| Name | Last commit | Last update |
|---|---|---|
| conf | Initial commit | 1 hour ago |
| README.md | Adding README | 1 hour ago |

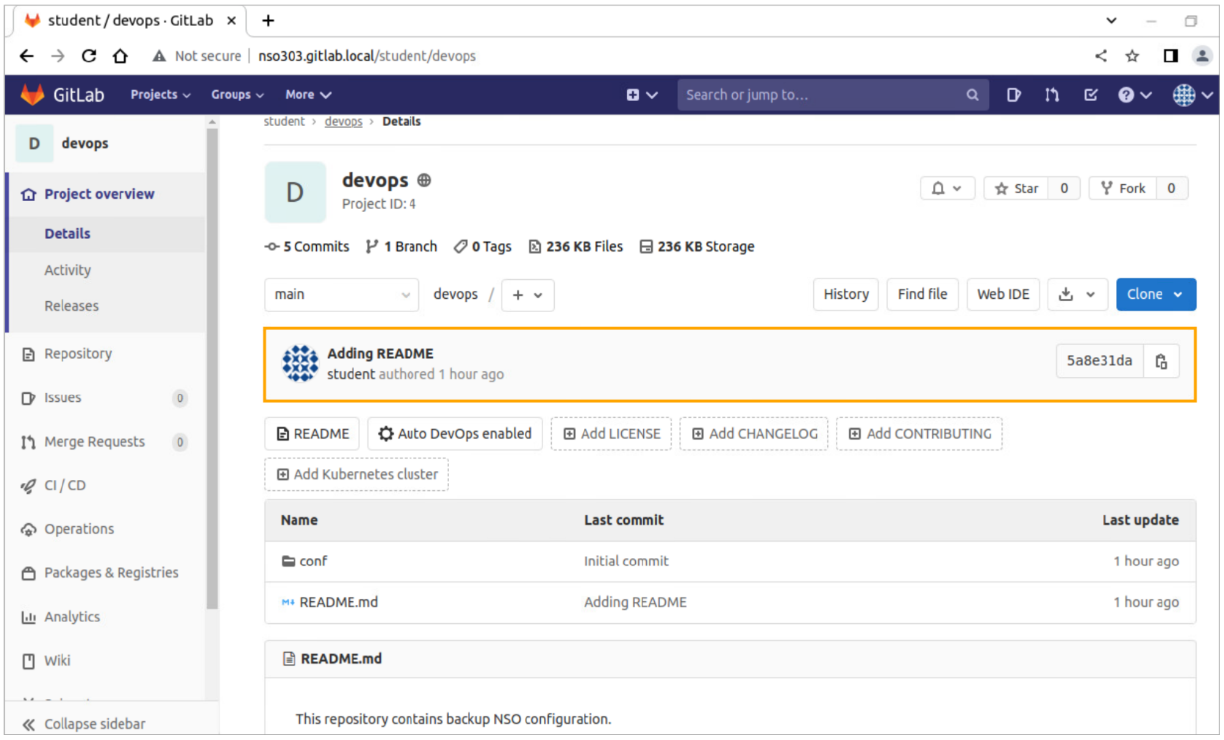
README.md

This repository contains backup NSO configuration.

### *Step 15*

Click the **Adding README** commit message and verify the changes that have been done.

Click the **Adding README** commit message.

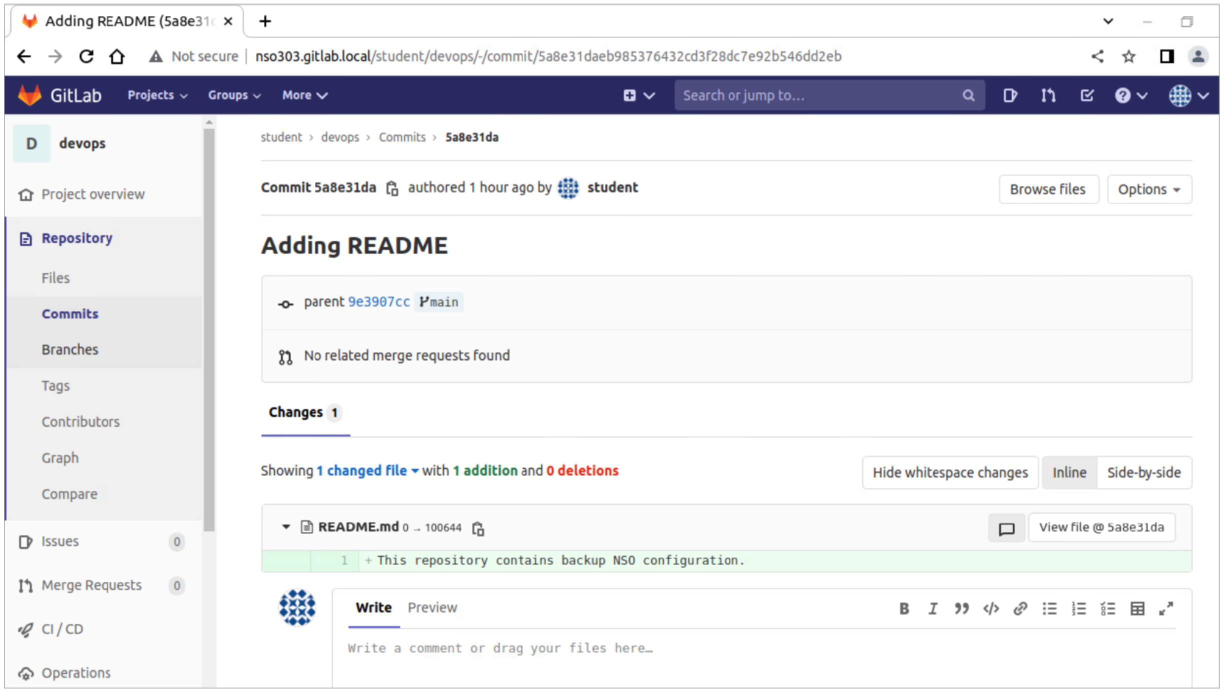

student / devops · GitLab
nso303.gitlab.local/student/devops
GitLab Projects Groups More
Search or jump to...
devops
Project overview
Details
Activity
Releases
Repository
Issues 0
Merge Requests 0
CI / CD
Operations
Packages & Registries
Analytics
Wiki
Collapse sidebar
student › devops › Details
devops
Project ID: 4
Star 0 Fork 0
5 Commits 1 Branch 0 Tags 236 KB Files 236 KB Storage
main devops / +
History Find file Web IDE Clone
Adding README
student authored 1 hour ago
5a8e31da
README Auto DevOps enabled Add LICENSE Add CHANGELOG Add CONTRIBUTING
Add Kubernetes cluster

| Name | Last commit | Last update |
|---|---|---|
| conf | Initial commit | 1 hour ago |
| README.md | Adding README | 1 hour ago |

README.md

This repository contains backup NSO configuration.

You see what changes have been made to the repository in this commit. The red highlighted lines show you the past state, and the green highlighted lines show you the new state.



Commit 5a8e31da authored 1 hour ago by student

Adding README

parent 9e3907cc main

No related merge requests found

Showing 1 changed file with 1 addition and 0 deletions

README.md 0 → 100644

```
1 + This repository contains backup NSO configuration.
```

## Step 16

Switch back to the terminal window and display the commit log for the project.

Use the **git log** command to display the commit log for the project. Each commit is marked with a unique hash value. Exit the view by typing **q**.

```
student@student-vm:~/Git/devops$ git log
commit 5a8e31daeb985376432cd3f28dc7e92b546dd2eb (HEAD -> main, origin/
main, origin/HEAD)
Author: student <student@gitlab.local>
Date:   Tue Oct 10 16:20:04 2023 +0000

    Adding README

commit 9e3907cc97263c87217bfc83a16d191b98f15873
Author: student <student@gitlab.local>
Date:   Tue Oct 10 16:12:30 2023 +0000

    Initial commit

commit 8a916d7de9032b34d8ea55ccffee022580c08914
Author: student <student@gitlab.local>
Date:   Tue Oct 10 16:09:44 2023 +0000
student@student-vm:~/Git/devops$
```

Hash of your commit might differ from the hash in the output above.

***Step 17***

Verify the changes between the previous and the latest commit by using the **git diff** command.

Use the **git diff** command to see the changes. Copy the hash of the previous commit from the previous step and compare it with HEAD, which is pointing to the latest commit.

```
student@student-vm:~/Git/devops$ git diff
8a916d7de9032b34d8ea55ccffee022580c08914 HEAD
diff --git a/README.md b/README.md
new file mode 100644
index 0000000..645ff3a
--- /dev/null
+++ b/README.md
@@ -0,0 +1 @@
+This repository contains backup NSO configuration.

student@student-vm:~/Git/devops$
```

Hash of the previous commit might differ from the hash in the output above. When specifying the hash value, it is possible to use only the first few characters from the hash instead of the entire value.

## Activity Verification

You have completed this task when you attain these results:

- You have cloned the remote GitLab repository.

- You have successfully created the README file and pushed the changes to the remote repository.

## Task 2: Edit an Existing File in the Repository

In this task, you will edit an existing file in the **git** repository: ncs.conf. You will also see how you can use GitLab to help you keep track of and revert changes.

### Activity

Complete these steps:

#### *Step 1*

Verify that the devops folder is in a clean state.

Use the **git status** command.

```
student@student-vm:~/Git/devops$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
student@student-vm:~/Git/devops$
```

#### *Step 2*

Open NSO configuration file and disable the developer and audit logs.

Open config/ncs.conf and disable the developer and audit logs. Save the file and exit the file editor.

```
student@student-vm:~/Git/devops$ nano conf/ncs.conf
...

    <developer-log>
      <enabled>false</enabled>
      <file>
        <name>./logs/devel.log</name>
        <enabled>true</enabled>
      </file>
    </developer-log>
    <developer-log-level>trace</developer-log-level>

    <audit-log>
      <enabled>false</enabled>
      <file>
        <name>./logs/audit.log</name>
        <enabled>true</enabled>
      </file>
    </audit-log>

...
```

### *Step 3*

Verify that the configuration has been changed.

Run the **git status** command to verify that the configuration has been changed. In the output, the ncs.conf file should be listed as a modified file.

```
student@student-vm:~/Git/devops$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working
directory)

        modified:   conf/ncs.conf

no changes added to commit (use "git add" and/or "git commit -a")
student@student-vm:~/Git/devops$
```

### *Step 4*

Review the detailed changes of the **ncs.conf** file.

To review what exactly was changed, run the **git diff** command. The plus (+) signs indicate what was added, while the minus (-) signs show what was removed from the file.

```
student@student-vm:~/Git/devops$ git diff
diff --git a/conf/ncs.conf b/conf/ncs.conf
index a510b51..1fdcfa2 100755
--- a/conf/ncs.conf
+++ b/conf/ncs.conf
@@ -168,7 +168,7 @@
     </ncs-log>

     <developer-log>
-      <enabled>true</enabled>
+      <enabled>false</enabled>
       <file>
         <name>./logs/devel.log</name>
         <enabled>true</enabled>
@@ -177,7 +177,7 @@
     <developer-log-level>trace</developer-log-level>

     <audit-log>
-      <enabled>true</enabled>
+      <enabled>false</enabled>
       <file>
         <name>./logs/audit.log</name>
         <enabled>true</enabled>
student@student-vm:~/Git/devops$
```

***Step 5***

Add the changes to the staging area.

To add the changes, use the **git add** command.

```
student@student-vm:~/Git/devops$ git add .
student@student-vm:~/Git/devops$
```

To remove added changes or files from the staging area, use the **git reset HEAD -- ncs.conf** command. If you want to check what is currently in the staging area, use **git diff HEAD**.

***Step 6***

Commit and push the changes to the remote repository.

To perform the commit and push, use the **git commit** and the **git push** commands.

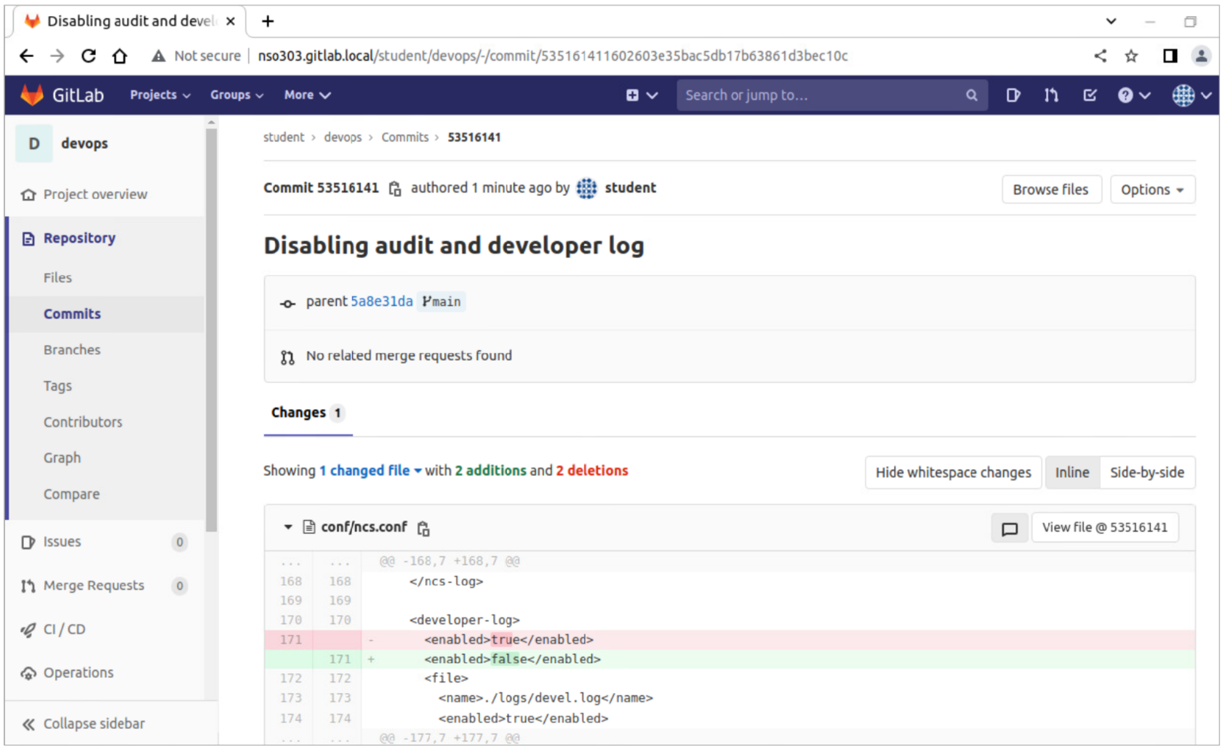
```
student@student-vm:~/Git/devops$ git commit -m "Disabling audit and
developer log"
[main 2e8c066] Disabling audit and developer log
 1 file changed, 2 insertions(+), 2 deletions(-)
student@student-vm:~/Git/devops$ git push
Username for 'http://nso303.gitlab.local': student
Password for 'http://student@nso303.gitlab.local':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 376 bytes | 376.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
To http://nso303.gitlab.local/student/devops.git
   5a8e31d..5351614  main -> main
student@student-vm:~/Git/devops$
```

The **git push** command, by default, pushes all local branches that have matching branches on the remote.

***Step 7***

Inspect the latest commit in the devops project using GitLab WebUI.

Return to the browser window and go to the devops project or select a list of commits in the navigation bar (Repository, Commits) on the left to get a list of commits. You can see that there is a new commit added. Inspect this latest commit. GitLab gives you a good overview of changes and the ability to comment on the changed lines and on the entire commit, which is useful for code reviews.



nso303.gitlab.local/student/devops/-/commit/535161411602603e35bac5db17b63861d3bec10c

student › devops › Commits › 53516141

Commit 53516141 authored 1 minute ago by student

**Disabling audit and developer log**

parent 5a8e31da main

No related merge requests found

Changes 1

Showing 1 changed file with 2 additions and 2 deletions

conf/ncs.conf

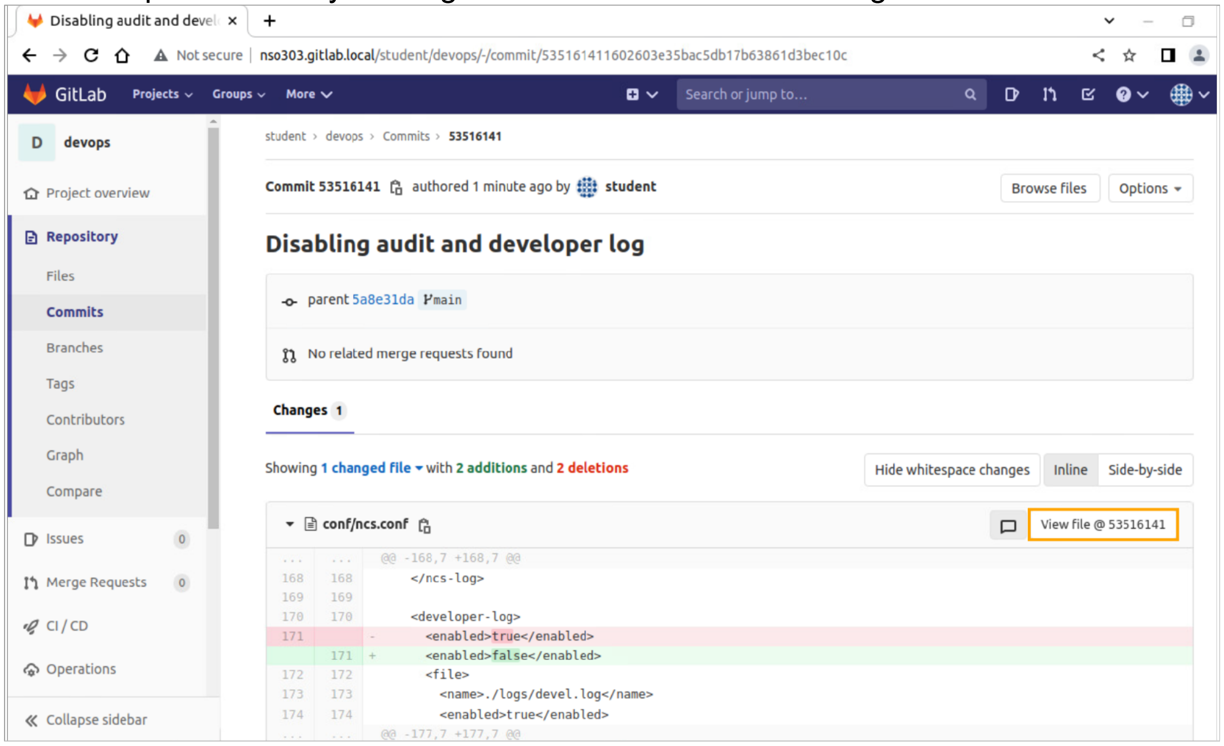
```
@@ -168,7 +168,7 @@
     </ncs-log>

     <developer-log>
-      <enabled>true</enabled>
+      <enabled>false</enabled>
       <file>
         <name>./logs/devel.log</name>
         <enabled>true</enabled>
@@ -177,7 +177,7 @@
```

## *Step 8*

Open the **ncs.conf** file in the GitLab WebUI.

You can open the file by clicking the **View file** button on the right.



nso303.gitlab.local/student/devops/-/commit/535161411602603e35bac5db17b63861d3bec10c

student › devops › Commits › 53516141

Commit 53516141 authored 1 minute ago by student

**Disabling audit and developer log**

parent 5a8e31da main

No related merge requests found

Changes 1

Showing 1 changed file with 2 additions and 2 deletions

conf/ncs.conf

View file @ 53516141

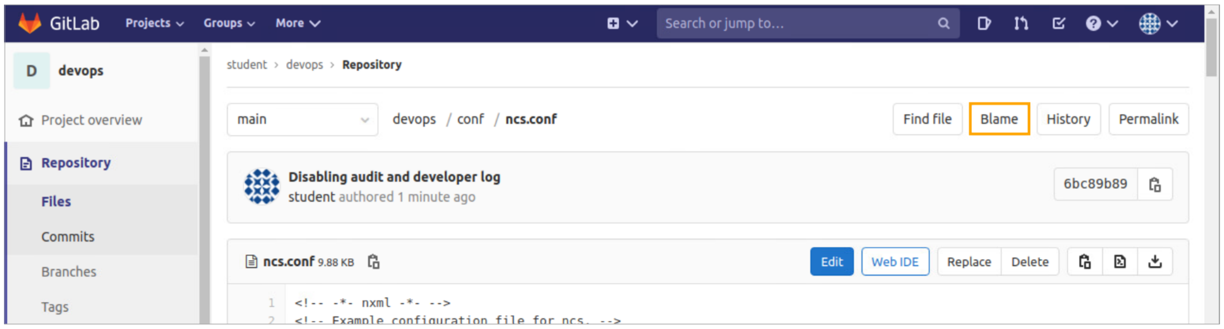
```
@@ -168,7 +168,7 @@
     </ncs-log>

     <developer-log>
-      <enabled>true</enabled>
+      <enabled>false</enabled>
       <file>
         <name>./logs/devel.log</name>
         <enabled>true</enabled>
@@ -177,7 +177,7 @@
```

## *Step 9*

Click the **Blame** button to display the authors and commits of changes to the file, line-by-line.

You will find the **Blame** button in the header of the web page on the right side.

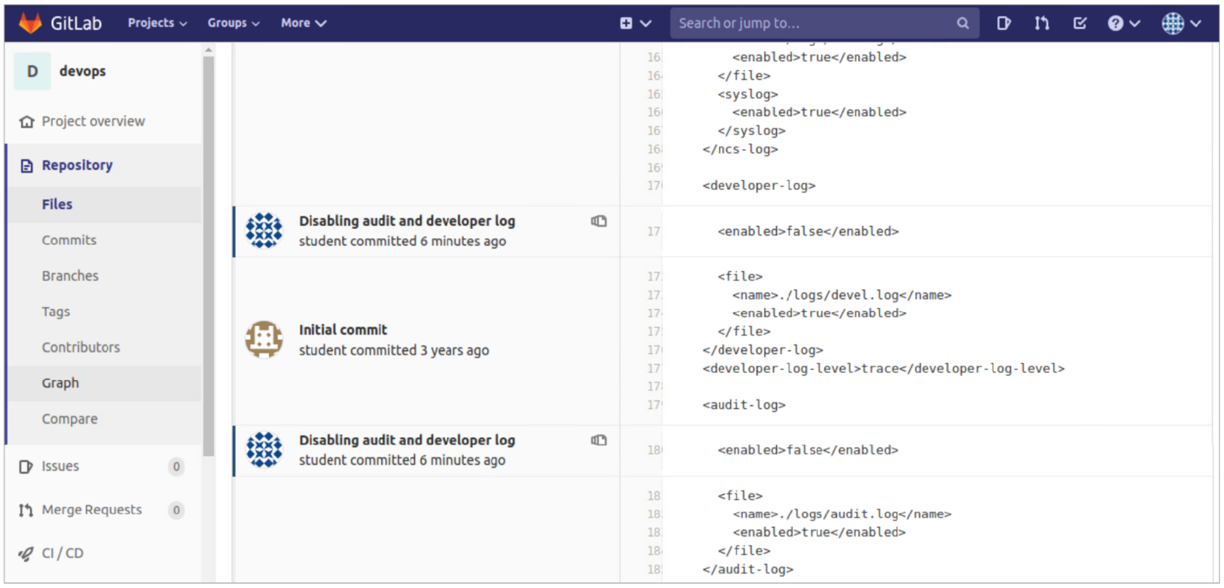


Disabling audit and developer log
student authored 1 minute ago
6bc89b89

ncs.conf 9.88 KB

```
<!-- -*- nxml -*- -->
<!-- Example configuration file for ncs. -->
```

Scroll down to find the recent changes to logging that you have made.



Disabling audit and developer log
student committed 6 minutes ago

Initial commit
student committed 3 years ago

Disabling audit and developer log
student committed 6 minutes ago

```
      <enabled>true</enabled>
    </file>
    <syslog>
      <enabled>true</enabled>
    </syslog>
  </ncs-log>

  <developer-log>
    <enabled>false</enabled>
    <file>
      <name>./logs/devel.log</name>
      <enabled>true</enabled>
    </file>
  </developer-log>
  <developer-log-level>trace</developer-log-level>

  <audit-log>
    <enabled>false</enabled>
    <file>
      <name>./logs/audit.log</name>
      <enabled>true</enabled>
    </file>
  </audit-log>
```

### *Step 10*

Return to the opened terminal window and revert the recent changes to the **ncs.conf** file.

Instead of changing the file by hand again, you can simply revert the repository back to the previous commit with the **git revert** command. If you want to go back a single commit, you can use the **git revert HEAD** command; otherwise, you need to specify the commit hash.

```
student@student-vm:~/Git/devops$ git log --oneline
5351614 (HEAD -> main, origin/main, origin/HEAD) Disabling audit and
developer log
5a8e31d Adding README
9e3907c Initial commit
student@student-vm:~/Git/devops$ git revert HEAD
```

### *Step 11*

Save the commit message that appears automatically.

To save and exit the file editor, use **CTRL+X**.

```
Revert "Disabling audit and developer log"

This reverts commit 43cc202dbcd47f8ea8ba6b095ca5700ef0de4621.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
# Your branch is ahead of 'origin/main' by 1 commit.
#   (use "git push" to publish your local commits)
#
# Changes to be committed:
#       modified:   conf/ncs.conf
#
# Untracked files:
#       main
#
```

### *Step 12*

Push the newly created commit to the remote registry.

Verify the commit message by using the **git log** command, then push the changes by using the **git push** command.

```
student@student-vm:~/Git/devops$ git log --oneline
8d111d8 (HEAD -> main) Revert "Disabling audit and developer log"
5351614 (origin/main, origin/HEAD) Disabling audit and developer log
5a8e31d Adding README
9e3907c Initial commit
student@student-vm:~/Git/devops$ git push
Username for 'http://nso303.gitlab.local': student
Password for 'http://student@nso303.gitlab.local':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 403 bytes | 403.00 KiB/s, done.
Total 4 (delta 1), reused 1 (delta 0), pack-reused 0
To http://nso303.gitlab.local/student/devops.git
   5351614..8d111d8  main -> main
student@student-vm:~/Git/devops$
```

### Activity Verification

You have completed this task when you attain these results:

- You have committed the updated ncs.conf file to the local and remote repository.
- You have reverted the changes made to ncs.conf file in the local and remote repository.

## Task 3: Merge a Package to the Repository

In this task, you will create a new feature branch from the main branch, add and commit an NSO package, and merge the feature branch back into the main branch.

## Activity

Complete these steps:

### *Step 1*

Verify that the devops folder is in a clean state.

To verify, use the **git status** command.

```
student@student-vm:~/Git/devops$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
student@student-vm:~/Git/devops$
```

### *Step 2*

Create and switch to a new branch from the current branch with the **git checkout -b** command.

Branches are very useful for development of multiple features at once or development with multiple involved developers. You can diverge from the main line of development and continue to do work without affecting that main line until your branch is ready to be included in it.

```
student@student-vm:~/Git/devops$ git checkout -b packages
Switched to a new branch 'packages'
student@student-vm:~/Git/devops$
```

### *Step 3*

Verify which branch you are currently working on.

To perform a verification, use the the **git branch** command.

```
student@student-vm:~/Git/devops$ git branch
  main
* packages
student@student-vm:~/Git/devops$
```

### *Step 4*

Create a new packages folder and copy the **hostname** package from the NSO Server **~/packages** folder there.

Use the **mkdir** command to create a folder and **scp** command with the **-r** parameter (recursive) to copy the package with all its contents.

```
student@student-vm:~/Git/devops$ mkdir packages
student@student-vm:~/Git/devops$ scp -r nso-server:/home/student/
packages/hostname packages/
hostname-template.xml                100%  326   702.0KB/s   00:00
package-meta-data.xml                100%  374     1.1MB/s   00:00
hostname.yang                        100%  506     1.7MB/s   00:00
Makefile                             100%  722     2.4MB/s   00:00
student@student-vm:~/Git/devops$
```

***Step 5***

Stage and commit the changes to the local repository.

Use the **git add** and **git commit** commands to perform the stage and commit.

```
student@student-vm:~/Git/devops$ git add .
student@student-vm:~/Git/devops$ git commit -m "Adding Hostname
package"
[packages 71ecbb2] Adding hostname package
 4 files changed, 85 insertions(+)
 create mode 100755 packages/hostname/package-meta-data.xml
 create mode 100755 packages/hostname/src/Makefile
 create mode 100755 packages/hostname/src/yang/hostname.yang
 create mode 100755 packages/hostname/templates/hostname-template.xml
student@student-vm:~/Git/devops$
```

***Step 6***

Push the local changes to the remote repository.

Since the branch “packages” was created locally, it does not yet exist on the remote repository. Use the **git push** command with the **--set-upstream** parameter to set the upstream association for any future push and pull attempts automatically.

```
student@student-vm:~/Git/devops$ git push --set-upstream origin
packages
Username for 'http://nso303.gitlab.local': student
Password for 'http://student@nso303.gitlab.local':
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 2 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (11/11), 1.66 KiB | 1.66 MiB/s, done.
Total 11 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: To create a merge request for packages, visit:
remote:   http://nso303.gitlab.local/student/devops/-/merge_requests/
new?merge_request%5Bsource_branch%5D=packages
remote:
To http://nso303.gitlab.local/student/devops.git
 * [new branch]      packages -> packages
Branch 'packages' set up to track remote branch 'packages' from
'origin'. student@student-vm:~/Git/devops$
```

***Step 7***

Change the working branch back to **main***.*

Use the **git checkout main** command.

```
student@student-vm:~/Git/devops$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
student@student-vm:~/Git/devops$
```

***Step 8***

Verify the differences between the **main** and the **packages** branch.

You can compare the two branches with the **git diff** command. The differences between the branches open in the text editor. When you are done comparing, close the text editor by pressing the **q** key.

```
student@student-vm:~/Git/devops$ git diff packages --
diff --git a/packages/hostname/package-meta-data.xml b/packages/
hostname/package-meta-data.xml
deleted file mode 100755
index 8fb7346..0000000
--- a/packages/hostname/package-meta-data.xml
+++ /dev/null
@@ -1,10 +0,0 @@
-<ncs-package xmlns="http://tail-f.com/ns/ncs-packages">
-  <name>hostname</name>
-  <package-version>1.0</package-version>
-  <description>Template-based hostname resource facing service</
description>
-  <ncs-min-version>5.3</ncs-min-version>
```

```
-
-  <!-- It's possible to add more components to the -->
-  <!-- same package, multiple services, data providers etc -->
-
-</ncs-package>
diff --git a/packages/hostname/src/Makefile b/packages/hostname/src/
Makefile
deleted file mode 100755
index cfd9d86..0000000
--- a/packages/hostname/src/Makefile
+++ /dev/null
@@ -1,30 +0,0 @@
-all: fxs
-.PHONY: all
-
-# Include standard NCS examples build definitions and rules
-include $(NCS_DIR)/src/ncs/build/include.ncs.mk
-

...
```

The **git diff** command above compares files between the checked out branch main and other branch packages, specified in the command parameters. Git assigns a minus sign (-) to the **a** version of the file and a plus sign (+) to the **b** version of the file, where **a** and **b** represent individual branches (a represents packages and b represents the main branch).

***Step 9***

Merge the **packages** branch back into the **main** branch with the **git merge** command. Git merges branches automatically if no merge conflicts occur.

If you try to merge two branches, where the same lines have been changed in both the branches since they diverged, or a file has been deleted in one branch, a merge conflict occurs. Merge conflicts must be resolved before the branches can be merged. This is done by going through all the conflicting changes and picking only a single version of those changes.

```
student@student-vm:~/Git/devops$ git merge packages
Updating 8d111d8..71ecbb2
Fast-forward
 packages/hostname/package-meta-data.xml           | 10 ++++++++++
 packages/hostname/src/Makefile                    | 30 +++++++++++++++
+++++++++++++++
 packages/hostname/src/yang/hostname.yang          | 34 +++++++++++++++
++++++++++++++++
 packages/hostname/templates/hostname-template.xml | 11 +++++++++++
 4 files changed, 85 insertions(+)
 create mode 100755 packages/hostname/package-meta-data.xml
 create mode 100755 packages/hostname/src/Makefile
 create mode 100755 packages/hostname/src/yang/hostname.yang
 create mode 100755 packages/hostname/templates/hostname-template.xml
student@student-vm:~/Git/devops$
```

Alternatively, merge requests can be opened through the GitLab WebUI and assigned to be merged (and reviewed) by someone else.

### *Step 10*

Push the merged main branch back to the remote repository.

Use the **git push origin main** command.

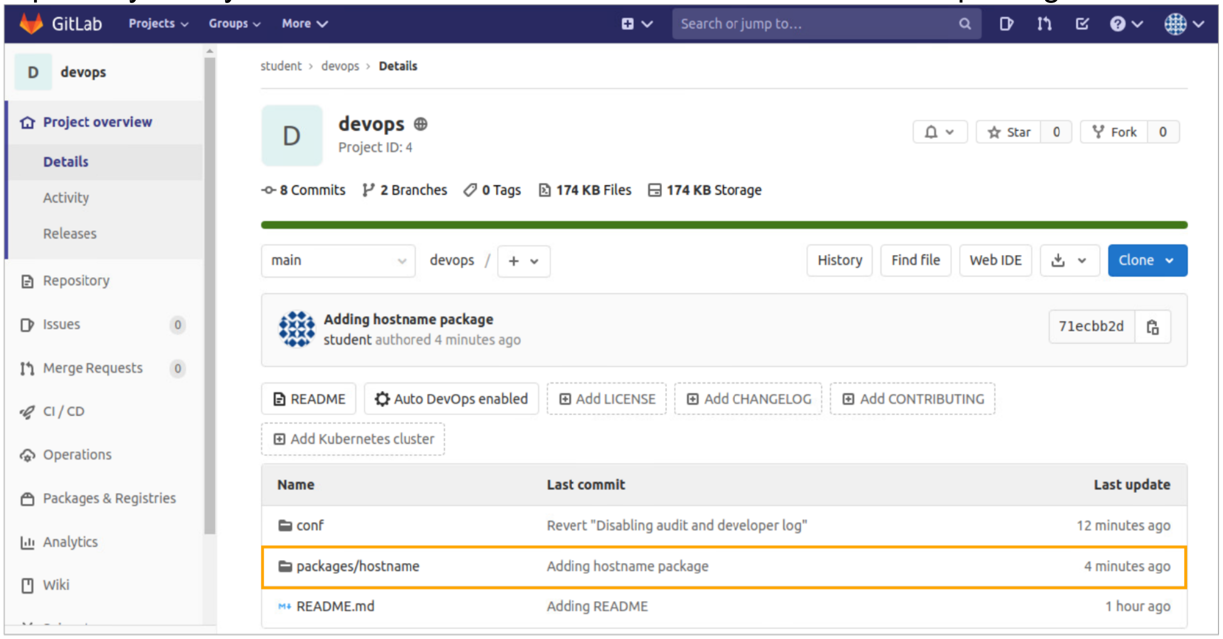
```
student@student-vm:~/Git/devops$ git push origin main
Username for 'http://nso303.gitlab.local': student
Password for 'http://student@nso303.gitlab.local':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To http://nso303.gitlab.local/student/devops.git
   8d111d8..71ecbb2  main -> main
student@student-vm:~/Git/devops$
```

By using the **git push origin main** command, you are updating only the main branch in the remote repository.

### *Step 11*

Verify that the main branch contains the hostname package by using the GitLab WebUI.

Return back to the browser with the open GitLab WebUI and refresh the devops repository. Verify that the main branch now contains the **hostname** package.



| Name | Last commit | Last update |
|---|---|---|
| conf | Revert "Disabling audit and developer log" | 12 minutes ago |
| packages/hostname | Adding hostname package | 4 minutes ago |
| README.md | Adding README | 1 hour ago |

## Activity Verification

You have completed this task when you attain these results:

- You have committed a hostname package to a new branch.

- You have successfully merged the packages branch into main branch.

You need to compare changes between two branches. Which git command will you use?

- ○ **git branch**
- ○ **git diff**
- ○ **git log**
- ○ **git status**