



## DevOps – Assignment 1

**Submitted by:**

- **Anil Joshi**
- **B.Tech CSE CCVT (BATCH 4)**
- **YEAR : 3rd**
- **SAP ID :500106262**

**Submitted to:**

**Dr. Prateek Raj Gautam**

# Assignment -1

**Q1. Demonstrate usage of Subversion, Mercurial with commands, explanation, and screenshots/screen recordings.**

**Answer-**

Subversion is a centralized version control system, where the repository is stored on a central server, and users check out the latest code from this server. Changes are tracked and updated with each commit.

Basic SVN Commands:

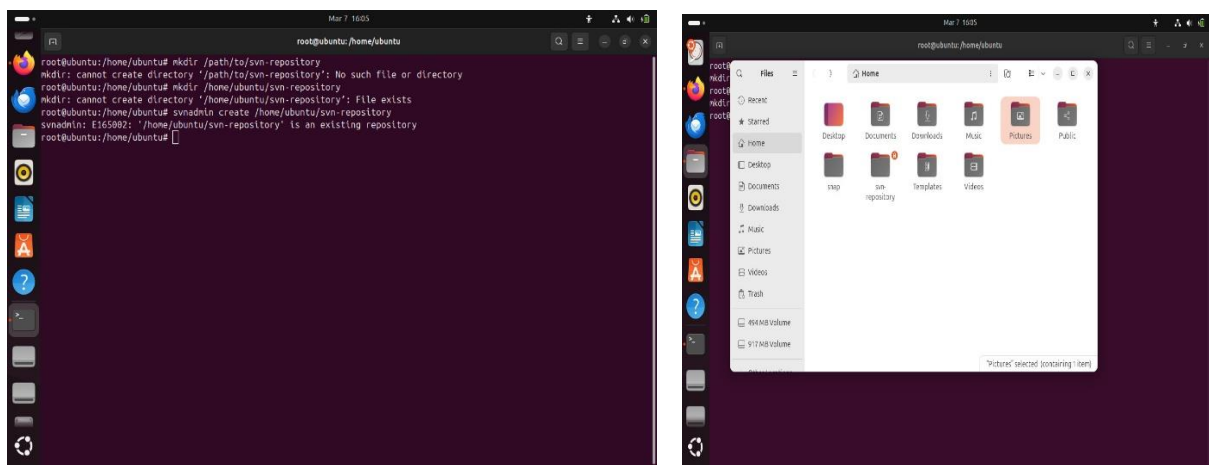
1. Checkout a repository

Command: svn checkout <repository url>

Example:

svn checkout https://example.com/svn/repository

Explanation: This command creates a local working copy of the repository. You can now make changes locally.



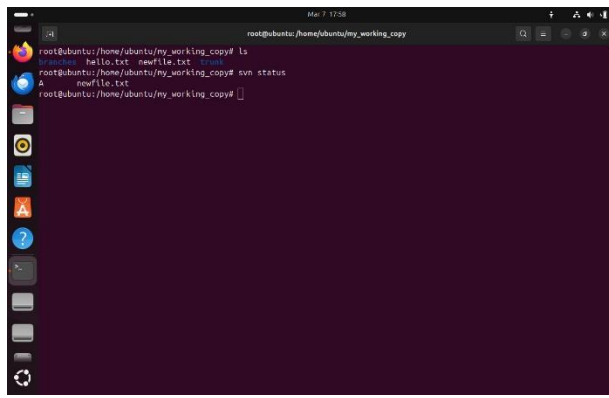
## 2. Check the status of your working copy

Command: svn status

Example:

svn status

Explanation: This shows the status of files in your working copy (e.g., modified, added, deleted, etc.).

A terminal window titled 'root@ubuntu:/home/ubuntu/my\_working\_copy' showing the output of the 'svn status' command. The output lists 'hello.txt' and 'newfile.txt' with a status of 'A', indicating they are added files. The prompt is 'root@ubuntu:/home/ubuntu/my\_working\_copy#'.

```
root@ubuntu:/home/ubuntu/my_working_copy# ls
hello.txt  newfile.txt
root@ubuntu:/home/ubuntu/my_working_copy# svn status
A      newfile.txt
root@ubuntu:/home/ubuntu/my_working_copy#
```

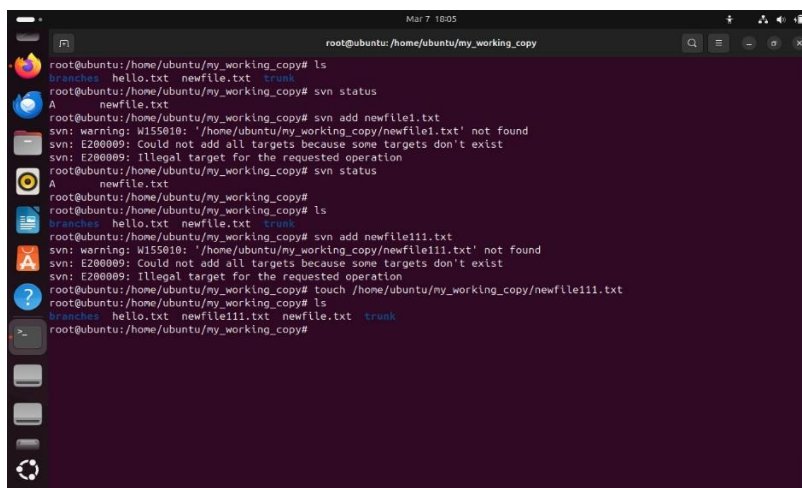
## 3. Add a new file to version control

Command: svn add <file name>

Example:

svn add new file.txt

Explanation: This adds a new file to version control, but it does not commit it yet.

A terminal window titled 'root@ubuntu:/home/ubuntu/my\_working\_copy' showing the execution of 'svn add newfile1.txt'. It displays several warning and error messages from Subversion, including 'W155010: "/home/ubuntu/my\_working\_copy/newfile1.txt' not found' and 'E200009: Could not add all targets because some targets don't exist'. The prompt is 'root@ubuntu:/home/ubuntu/my\_working\_copy#'.

```
root@ubuntu:/home/ubuntu/my_working_copy# ls
branches  hello.txt  newfile.txt  trunk
root@ubuntu:/home/ubuntu/my_working_copy# svn status
A      newfile.txt
root@ubuntu:/home/ubuntu/my_working_copy# svn add newfile1.txt
svn: warning: W155010: '/home/ubuntu/my_working_copy/newfile1.txt' not found
svn: E200009: Could not add all targets because some targets don't exist
svn: E200009: Illegal target for the requested operation
root@ubuntu:/home/ubuntu/my_working_copy# svn status
A      newfile.txt
root@ubuntu:/home/ubuntu/my_working_copy# ls
branches  hello.txt  newfile.txt  trunk
root@ubuntu:/home/ubuntu/my_working_copy# svn add newfile111.txt
svn: warning: W155010: '/home/ubuntu/my_working_copy/newfile111.txt' not found
svn: E200009: Could not add all targets because some targets don't exist
svn: E200009: Illegal target for the requested operation
root@ubuntu:/home/ubuntu/my_working_copy# touch /home/ubuntu/my_working_copy/newfile111.txt
root@ubuntu:/home/ubuntu/my_working_copy# ls
branches  hello.txt  newfile111.txt  newfile.txt  trunk
root@ubuntu:/home/ubuntu/my_working_copy#
```

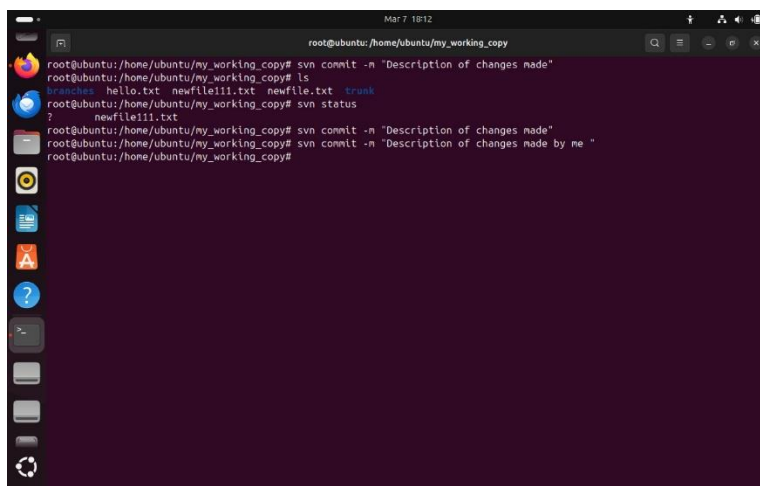
#### 4. Commit changes to the repository

Command: svn commit -m "Commit message"

Example:

svn commit -m "Added new file.txt"

Explanation: This commits your changes to the repository, and the message should describe the change made.

A terminal window titled 'Mar 7 18:12' showing a series of SVN commands and their outputs. The user is in the directory '/home/ubuntu/my\_working\_copy'. The commands and outputs are: 'svn commit -m "Description of changes made"', 'ls' showing 'hello.txt newfile111.txt newfile.txt trunk', 'svn status' showing 'newfile111.txt', and three 'svn commit -m "Description of changes made"' commands. The last command is followed by a prompt 'root@ubuntu:/home/ubuntu/my\_working\_copy#'.

```
root@ubuntu:/home/ubuntu/my_working_copy# svn commit -m "Description of changes made"
root@ubuntu:/home/ubuntu/my_working_copy# ls
hello.txt  newfile111.txt  newfile.txt  trunk
root@ubuntu:/home/ubuntu/my_working_copy# svn status
?      newfile111.txt
root@ubuntu:/home/ubuntu/my_working_copy# svn commit -m "Description of changes made"
root@ubuntu:/home/ubuntu/my_working_copy# svn commit -m "Description of changes made by me "
root@ubuntu:/home/ubuntu/my_working_copy#
```

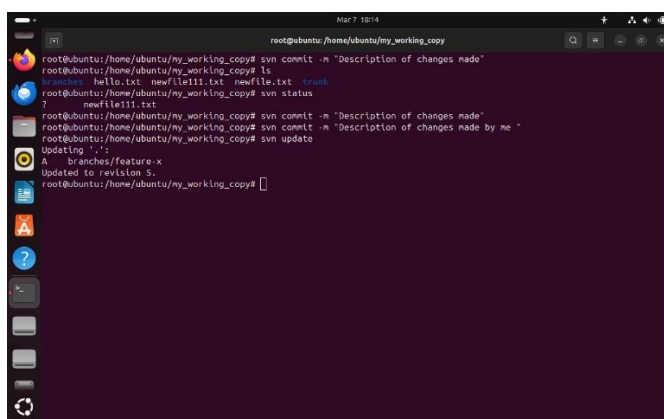
#### 5. Update your working copy

Command: svn update

Example:

svn update

Explanation: This updates your local working copy with the latest changes from the repository.

A terminal window titled 'Mar 7 18:14' showing the same sequence of SVN commands as the previous screenshot, followed by 'svn update'. The output of 'svn update' is 'Updating '.', 'A branches/feature-x', and 'Updated to revision 5.'. The prompt returns to 'root@ubuntu:/home/ubuntu/my\_working\_copy#'.

```
root@ubuntu:/home/ubuntu/my_working_copy# svn commit -m "Description of changes made"
root@ubuntu:/home/ubuntu/my_working_copy# ls
hello.txt  newfile111.txt  newfile.txt  trunk
root@ubuntu:/home/ubuntu/my_working_copy# svn status
?      newfile111.txt
root@ubuntu:/home/ubuntu/my_working_copy# svn commit -m "Description of changes made"
root@ubuntu:/home/ubuntu/my_working_copy# svn commit -m "Description of changes made by me "
root@ubuntu:/home/ubuntu/my_working_copy# svn update
Updating '.':
A   branches/feature-x
Updated to revision 5.
root@ubuntu:/home/ubuntu/my_working_copy#
```

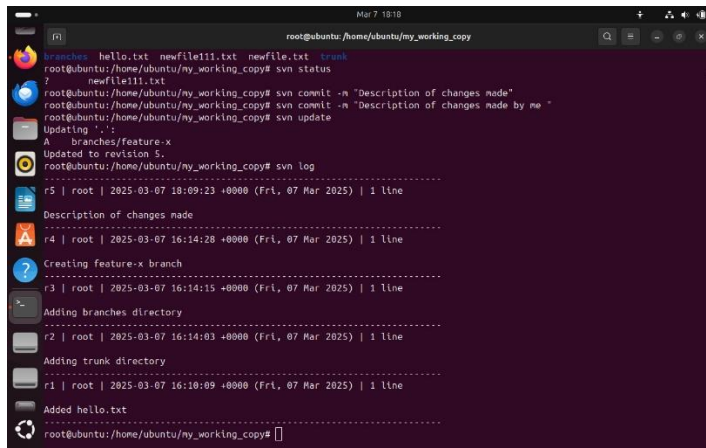
## 6.View the log of commits

Command: svn log

Example:

svn log

Explanation: This shows a history of the commits made to the repository.

A terminal window titled 'root@ubuntu: /home/ubuntu/my\_working\_copy' showing the output of the 'svn log' command. The output lists five revisions (r1 to r5) with their dates, times, and commit messages. The messages include 'Added hello.txt', 'Adding trunk directory', 'Adding branches directory', 'Creating feature-x branch', 'Description of changes made', and 'Updated to revision 5'.

```
root@ubuntu: /home/ubuntu/my_working_copy
newfile1.txt newfile11.txt newfile.txt trunk
root@ubuntu: /home/ubuntu/my_working_copy# svn status
? newfile111.txt
root@ubuntu: /home/ubuntu/my_working_copy# svn commit -m "Description of changes made"
root@ubuntu: /home/ubuntu/my_working_copy# svn commit -m "Description of changes made by me"
root@ubuntu: /home/ubuntu/my_working_copy# svn update
Updating '.':
A    branches/feature-x
Updated to revision 5.
root@ubuntu: /home/ubuntu/my_working_copy# svn log
-----
r5 | root | 2025-03-07 18:09:23 +0000 (Fri, 07 Mar 2025) | 1 line
Description of changes made
-----
r4 | root | 2025-03-07 16:14:28 +0000 (Fri, 07 Mar 2025) | 1 line
Creating feature-x branch
-----
r3 | root | 2025-03-07 16:14:15 +0000 (Fri, 07 Mar 2025) | 1 line
Adding branches directory
-----
r2 | root | 2025-03-07 16:14:03 +0000 (Fri, 07 Mar 2025) | 1 line
Adding trunk directory
-----
r1 | root | 2025-03-07 16:10:09 +0000 (Fri, 07 Mar 2025) | 1 line
Added hello.txt
-----
root@ubuntu: /home/ubuntu/my_working_copy#
```

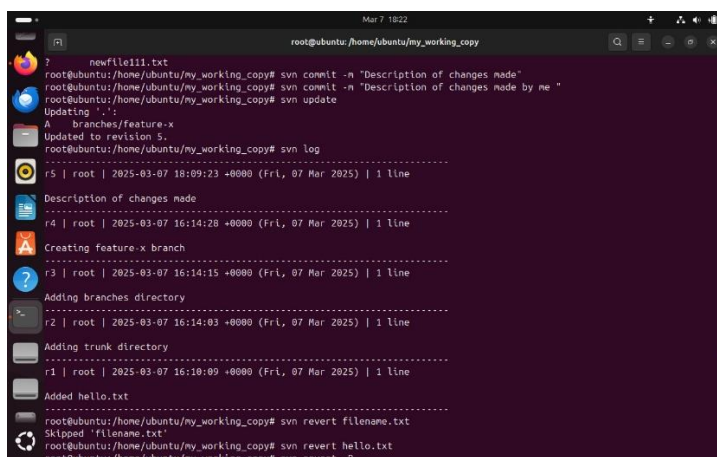
## 7.Revert changes to a file

Command: svn revert <file name>

Example:

svn revert new file.txt

Explanation: This reverts any changes made to the file since the last commit.

A terminal window titled 'root@ubuntu: /home/ubuntu/my\_working\_copy' showing the output of the 'svn revert' command. The output shows the command being executed and the file being reverted. The output also shows the same commit history as the previous screenshot, but with the addition of the 'svn revert' command and its output.

```
root@ubuntu: /home/ubuntu/my_working_copy
newfile111.txt
root@ubuntu: /home/ubuntu/my_working_copy# svn commit -m "Description of changes made"
root@ubuntu: /home/ubuntu/my_working_copy# svn commit -m "Description of changes made by me"
root@ubuntu: /home/ubuntu/my_working_copy# svn update
Updating '.':
A    branches/feature-x
Updated to revision 5.
root@ubuntu: /home/ubuntu/my_working_copy# svn log
-----
r5 | root | 2025-03-07 18:09:23 +0000 (Fri, 07 Mar 2025) | 1 line
Description of changes made
-----
r4 | root | 2025-03-07 16:14:28 +0000 (Fri, 07 Mar 2025) | 1 line
Creating feature-x branch
-----
r3 | root | 2025-03-07 16:14:15 +0000 (Fri, 07 Mar 2025) | 1 line
Adding branches directory
-----
r2 | root | 2025-03-07 16:14:03 +0000 (Fri, 07 Mar 2025) | 1 line
Adding trunk directory
-----
r1 | root | 2025-03-07 16:10:09 +0000 (Fri, 07 Mar 2025) | 1 line
Added hello.txt
-----
root@ubuntu: /home/ubuntu/my_working_copy# svn revert filename.txt
Skipped 'filename.txt'
root@ubuntu: /home/ubuntu/my_working_copy# svn revert hello.txt
root@ubuntu: /home/ubuntu/my_working_copy#
```

Mercurial (often abbreviated as hg) is a distributed version control system, similar to Git, used for tracking changes in source code during software development. Here's a step-by-step guide to demonstrate how to use Mercurial with commands and explanations.

## 1. Installation of Mercurial

First, you need to install Mercurial. On Ubuntu, you can install it using:

sudo apt update

sudo apt install mercurial

Once installed, verify the installation:

hg --version

```
ubuntu@ubuntu:~$ sudo apt update
sudo apt install mercurial
Ign:1 cdrom://Ubuntu 24.04.1 LTS _Noble Numbat_ - Release amd64 (20240827.1) noble InRelease
Hit:2 cdrom://Ubuntu 24.04.1 LTS _Noble Numbat_ - Release amd64 (20240827.1) noble Release
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:6 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:7 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
474 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  mercurial-common
Suggested packages:
  kdiff3 | kdiff3-qt | kompare | meld | tkcvs | mgdiff qct python3-mysqldb python3-openssl wish
The following NEW packages will be installed:
  mercurial mercurial-common
0 upgraded, 2 newly installed, 0 to remove and 474 not upgraded.
Need to get 3,283 kB of archives.
After this operation, 16.3 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 mercurial-common all 6.7.2-1ubuntu2.2 [2,955 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 mercurial amd64 6.7.2-1ubuntu2.2 [328 kB]
Fetched 3,283 kB in 8s (432 kB/s)
Selecting previously unselected package mercurial-common.
(Reading database ... 210996 files and directories currently installed.)
Preparing to unpack .../mercurial-common_6.7.2-1ubuntu2.2_all.deb ...
Unpacking mercurial-common (6.7.2-1ubuntu2.2) ...
Selecting previously unselected package mercurial.
```

```
mercurial-common
Suggested packages:
  kdiff3 | kdiff3-qt | kompare | meld | tkcvs | mgdiff qct python3-mysqldb python3-openssl wish
The following NEW packages will be installed:
  mercurial mercurial-common
0 upgraded, 2 newly installed, 0 to remove and 474 not upgraded.
Need to get 3,283 kB of archives.
After this operation, 16.3 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 mercurial-common all 6.7.2-1ubuntu2.2 [2,955 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 mercurial amd64 6.7.2-1ubuntu2.2 [328 kB]
Fetched 3,283 kB in 8s (432 kB/s)
Selecting previously unselected package mercurial-common.
(Reading database ... 210996 files and directories currently installed.)
Preparing to unpack .../mercurial-common_6.7.2-1ubuntu2.2_all.deb ...
Unpacking mercurial-common (6.7.2-1ubuntu2.2) ...
Selecting previously unselected package mercurial.
Preparing to unpack .../mercurial_6.7.2-1ubuntu2.2_amd64.deb ...
Unpacking mercurial (6.7.2-1ubuntu2.2) ...
Setting up mercurial-common (6.7.2-1ubuntu2.2) ...
Setting up mercurial (6.7.2-1ubuntu2.2) ...
Creating config file /etc/mercurial/hgrc.d/hgext.rc with new version
Processing triggers for man-db (2.12.0-4build2) ...
ubuntu@ubuntu:~$ hg --version
Mercurial Distributed SCM (version 6.7.2)
(see https://mercurial-scm.org for more information)

Copyright (C) 2005-2023 Olivia Mackall and others
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
ubuntu@ubuntu:~$
```

## 2. Creating a New Repository

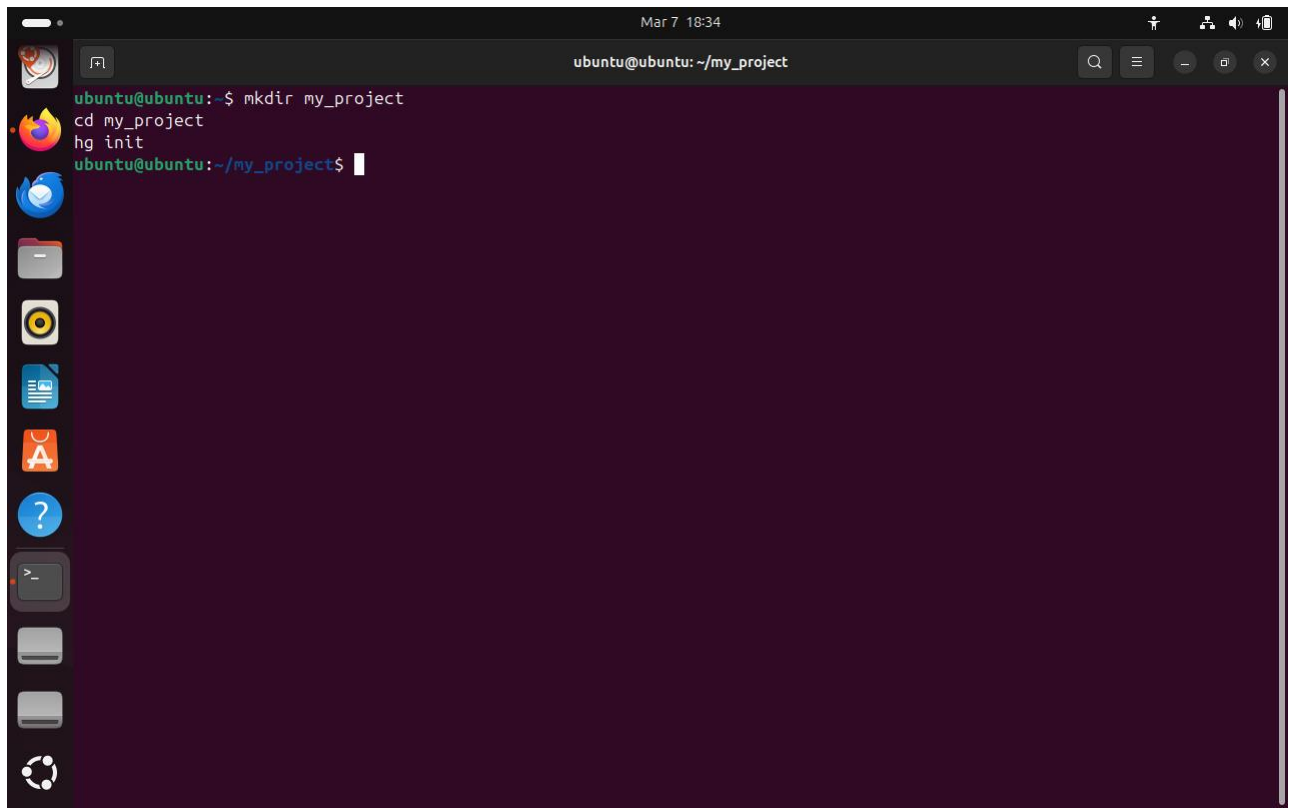
To create a new repository, navigate to your project directory and run the following command:

mkdir my\_project

cd my\_project

hg init

hg init: Initializes a new Mercurial repository in the current directory. It creates a .hg folder that stores all the versioning information and history for the repository.

A screenshot of a Linux terminal window. The window title is 'ubuntu@ubuntu: ~/my\_project'. The terminal shows the following commands and their output: 'mkdir my\_project', 'cd my\_project', and 'hg init'. The prompt changes from 'ubuntu@ubuntu:~\$' to 'ubuntu@ubuntu:~/my\_project\$' after the 'cd' command. The terminal has a dark purple background and a light blue cursor. On the left side of the terminal, there is a vertical dock with various application icons including a web browser, a file manager, and a terminal icon. The top of the window shows system status icons like network, volume, and battery, along with the date and time 'Mar 7 18:34'.





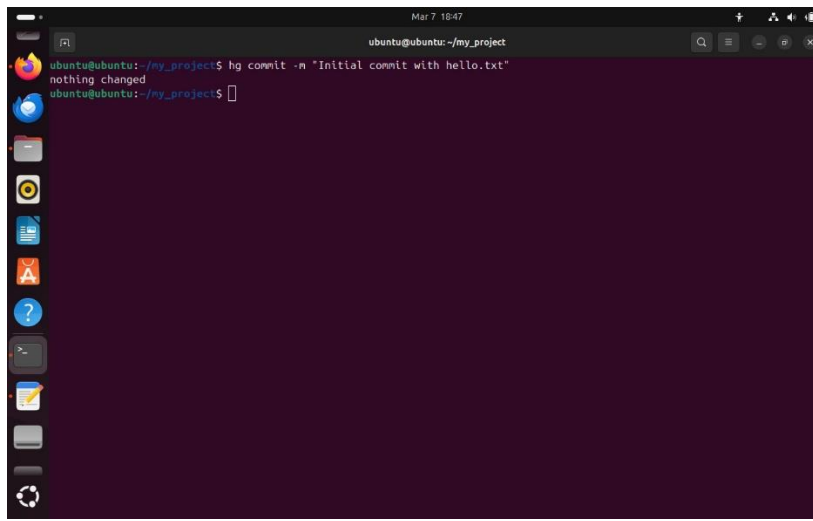
## 4. Committing Changes

After adding files, the next step is to commit your changes to the repository.  
This saves a snapshot of the current state of the files in the repository.

hg commit -m "Initial commit with hello.txt"

hg commit: Commits the changes to the repository.

-m "message": The -m option allows you to add a commit message describing the changes.

A terminal window titled 'ubuntu@ubuntu: ~/my\_project' showing the execution of the 'hg commit' command. The user enters 'hg commit -m "Initial commit with hello.txt"', and the output is 'nothing changed'. The prompt returns to 'ubuntu@ubuntu:~/my\_project\$'.

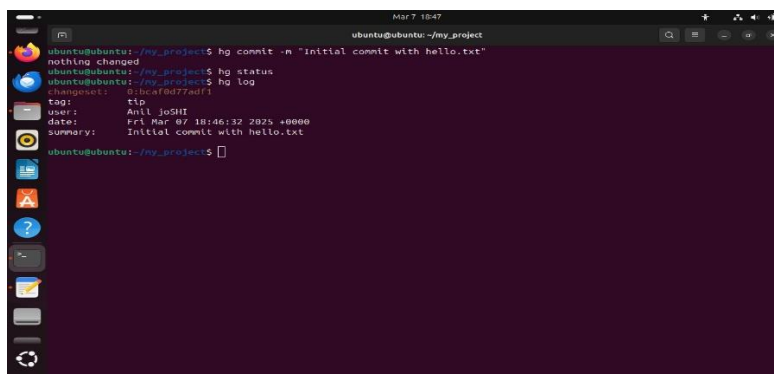
```
ubuntu@ubuntu:~/my_project$ hg commit -m "Initial commit with hello.txt"
nothing changed
ubuntu@ubuntu:~/my_project$
```

## 5. Checking the Status of the Repository

To see which files are modified, added, or deleted, use the following command:

hg status

hg status: Shows the status of files in the repository, indicating whether files are modified (M), added (A), or deleted (R).

A terminal window titled 'ubuntu@ubuntu: ~/my\_project' showing the execution of 'hg commit' followed by 'hg status' and 'hg log'. The commit was successful, and the status shows no changes. The log shows the commit details.

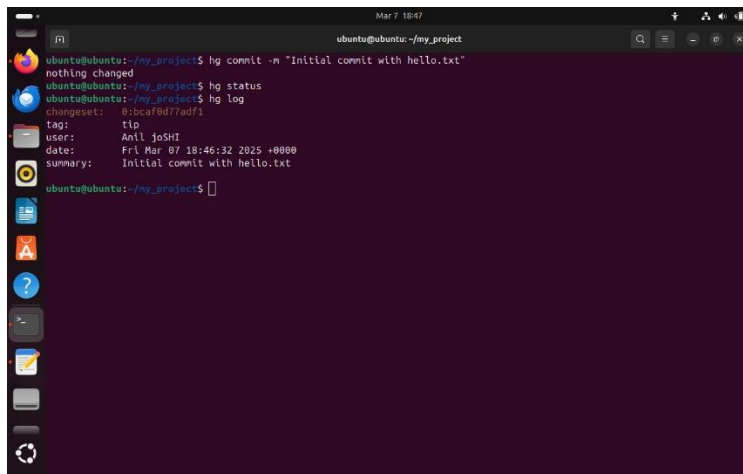
```
ubuntu@ubuntu:~/my_project$ hg commit -m "Initial commit with hello.txt"
nothing changed
ubuntu@ubuntu:~/my_project$ hg status
nothing changed
ubuntu@ubuntu:~/my_project$ hg log
changeset: 81c6aff0d77ad1
tag: tip
user: Anil Joshi
date: Fri Mar 07 18:46:32 2025 +0000
summary: Initial commit with hello.txt
ubuntu@ubuntu:~/my_project$
```

## 6. Viewing the Log of Commits

You can view the history of commits with the following command:

hg log

hg log: Displays a log of all the commits, including information about each commit such as the revision number, date, author, and commit message.

A terminal window titled 'ubuntu@ubuntu: ~/my\_project' showing the output of the 'hg log' command. The output displays commit information for a single commit: changeset 0:bcafd77adf1, tag tip, user Anil Joshi, date Fri Mar 07 18:46:32 2025 +0000, and summary 'Initial commit with hello.txt'.

```
ubuntu@ubuntu: ~/my_project$ hg commit -m "Initial commit with hello.txt"
nothing changed
ubuntu@ubuntu: ~/my_project$ hg status
ubuntu@ubuntu: ~/my_project$ hg log
changeset: 0:bcafd77adf1
tag:       tip
user:      Anil Joshi
date:      Fri Mar 07 18:46:32 2025 +0000
summary:   Initial commit with hello.txt
ubuntu@ubuntu: ~/my_project$
```

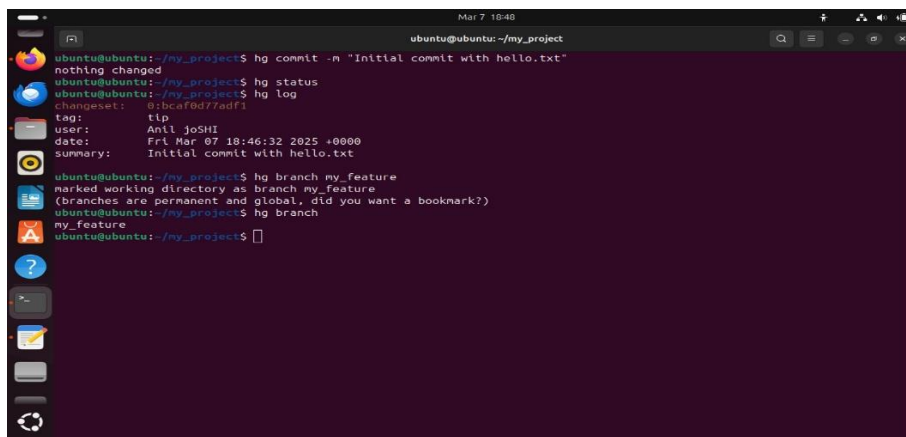
## 7. Creating a New Branch

Mercurial allows you to work on different branches of development. To create a new branch, use the following command:

hg branch my\_feature

hg branch <branch name>: Creates a new branch. This command creates a named branch in the repository.

hg branch

A terminal window titled 'ubuntu@ubuntu: ~/my\_project' showing the output of the 'hg branch my\_feature' command. The output indicates that the working directory is marked as branch 'my\_feature' and that branches are permanent and global.

```
ubuntu@ubuntu: ~/my_project$ hg commit -m "Initial commit with hello.txt"
nothing changed
ubuntu@ubuntu: ~/my_project$ hg status
ubuntu@ubuntu: ~/my_project$ hg log
changeset: 0:bcafd77adf1
tag:       tip
user:      Anil Joshi
date:      Fri Mar 07 18:46:32 2025 +0000
summary:   Initial commit with hello.txt
ubuntu@ubuntu: ~/my_project$ hg branch my_feature
marked working directory as branch my_feature
(branches are permanent and global, did you want a bookmark?)
ubuntu@ubuntu: ~/my_project$ hg branch
my_feature
ubuntu@ubuntu: ~/my_project$
```

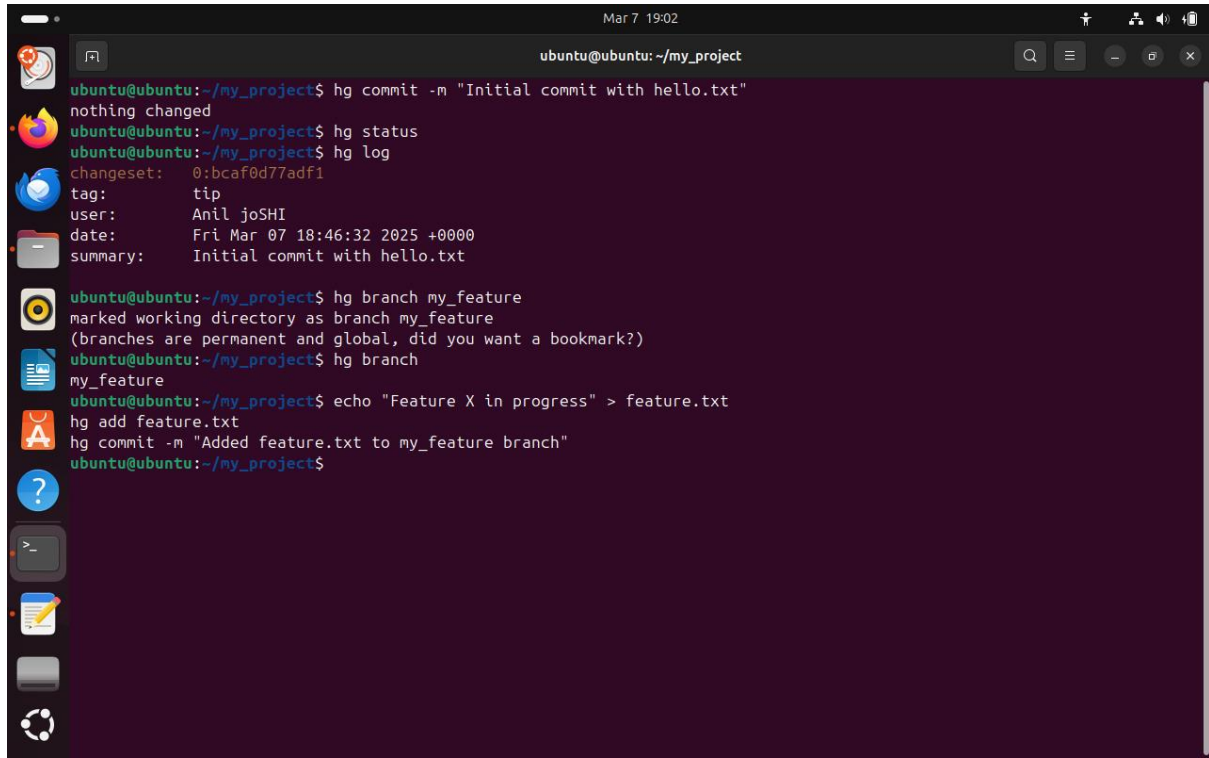
## 8. Committing on the New Branch

After creating a branch, make some changes. For example:

echo "Feature X in progress" > feature.txt

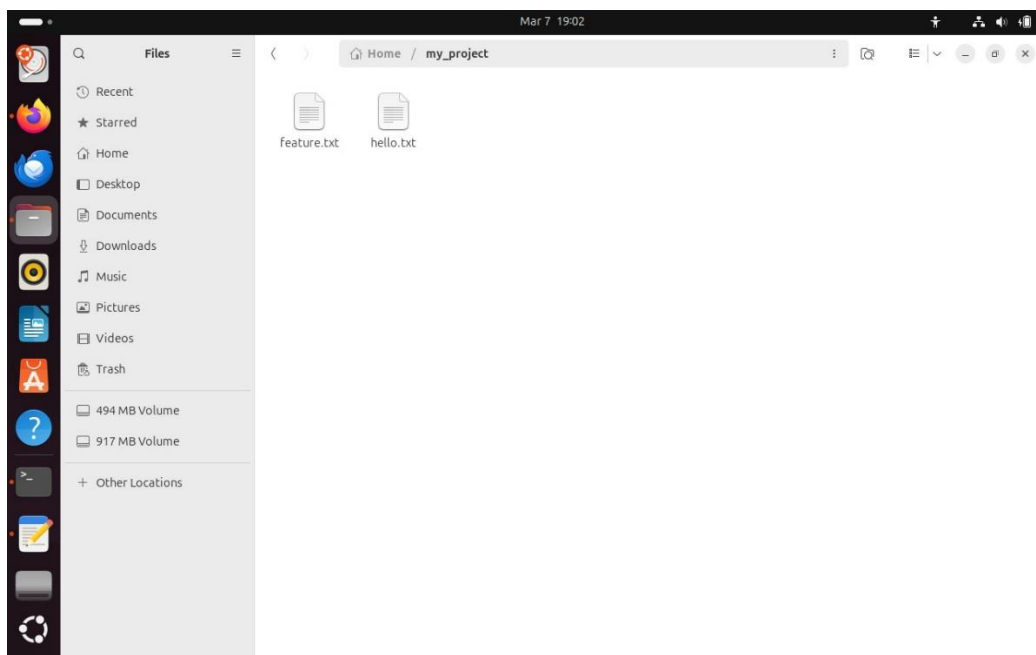
hg add feature.txt

hg commit -m "Added feature.txt to my feature branch"



```
ubuntu@ubuntu: ~/my_project
ubuntu@ubuntu:~/my_project$ hg commit -m "Initial commit with hello.txt"
nothing changed
ubuntu@ubuntu:~/my_project$ hg status
ubuntu@ubuntu:~/my_project$ hg log
changeset:  0:bcaf0d77adf1
tag:         tip
user:        Anil joSHI
date:        Fri Mar 07 18:46:32 2025 +0000
summary:     Initial commit with hello.txt

ubuntu@ubuntu:~/my_project$ hg branch my_feature
marked working directory as branch my_feature
(branches are permanent and global, did you want a bookmark?)
ubuntu@ubuntu:~/my_project$ hg branch
my_feature
ubuntu@ubuntu:~/my_project$ echo "Feature X in progress" > feature.txt
hg add feature.txt
hg commit -m "Added feature.txt to my_feature branch"
ubuntu@ubuntu:~/my_project$
```



## 9. Switching Between Branches

To switch to another branch, use the following command:

hg update default

hg update <branch name>: Switches to a different branch. In this case, we're switching back to the default branch.

## 10. Merging Branches

Once you have completed the work on a feature branch, you may want to merge it into the default branch. First, switch to the branch you want to merge into:

hg update default

Then, merge the feature branch:

hg merge my\_feature

hg merge <branch name>: Merges changes from the specified branch into the current branch. After this, you will need to commit the merge:

hg commit -m "Merged my\_feature into default"