Linux History and philosophy

Linux began in 1991 with Linus Torvalds creating a free kernel, combining it with GNU tools for a full OS, driven by the philosophy of free and open-source software (FOSS), emphasizing collaboration, transparency, user freedom (to use, study, modify, share), and a decentralized approach, growing from a hobby into powering servers, Android, supercomputers, and more, contrasting proprietary models.

History of Linux

- 1991: Finnish student Linus Torvalds started developing a Unix-like kernel as a personal project, inspired by the MINIX OS, aiming for a free alternative to proprietary systems.
- Early Development: Torvalds released his kernel's source code, and it quickly grew through global developer contributions, combining with the GNU Project's free tools to form complete operating systems (distributions like Debian, Red Hat).
- Licensing Shift: Initially restricted, the kernel adopted the GNU General Public License (GPL) in 1992.
- Growth & Adoption: Linux gained traction in servers, embedded systems and became a stable, secure alternative to Windows/macOS.

Linux Philosophy

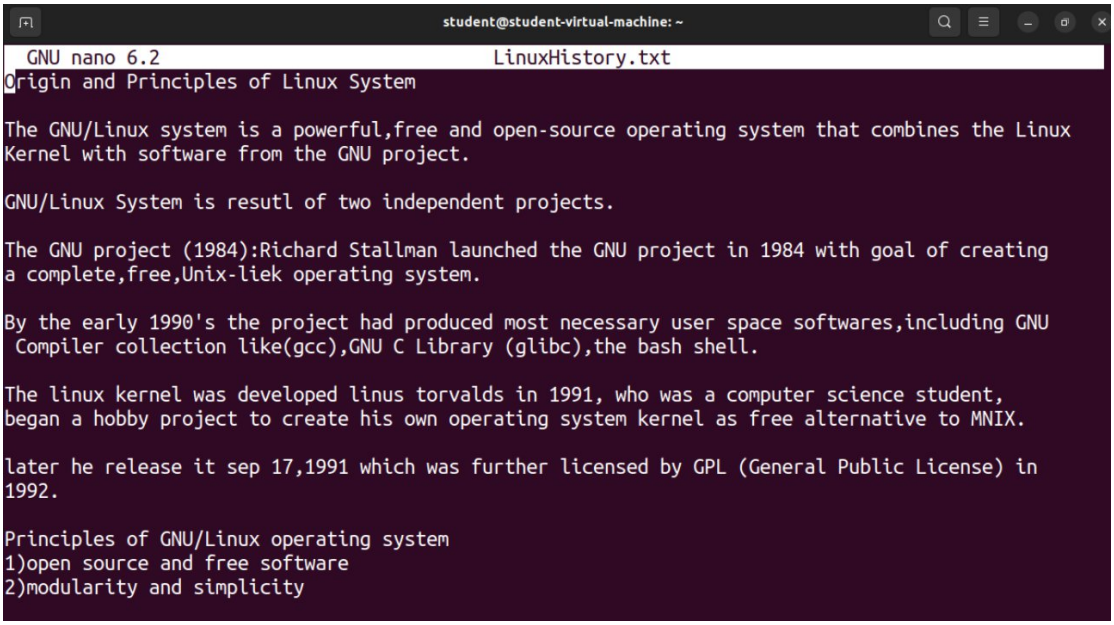
- Free & Open Source (FOSS)
- User Freedom
- Collaboration & Transparency
- Decentralization
- Customization & Control

Task-1

1) **write a brief history of linux**

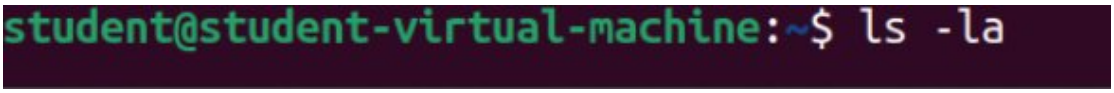
```
student@student-virtual-machine:~$ nano LinuxHistory.txt
```

- Creating a LinuxHistory.txt file and editing it in nano.

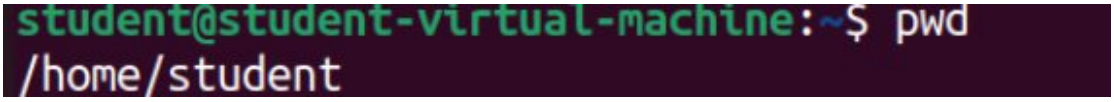


Task-2

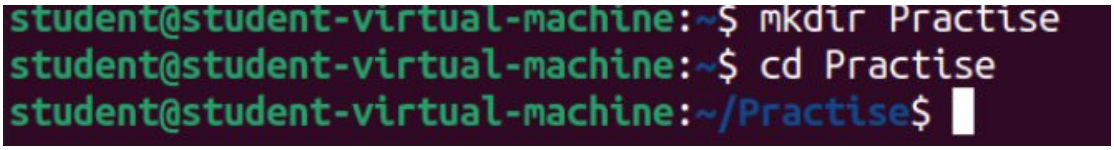
2) **File Navigation and Directory Structure**



- ls – cmd used to display the files in a directory
- -l - flag used to list in vertical format
- -a – used to display the hidden files.



- pwd – cmd used to print the correct working directory.
- realpath – cmd also used to view the abspath of a file/folder.



- mkdir – command use to create a make directory
- -p – flag used to create a non parent directory

Task-3

3) **File management commands to organize files and directories**

```
student@student-virtual-machine:~$ touch Practise/sample.txt
```

touch – cmd used to create any file in a directory or folder.

```
student@student-virtual-machine:~$ cp Practise/sample.txt Practise/duplicate.txt
```

cp- cmd used to copy the file from one directory/folder to another directory or folder or within same directory or folder.

```
student@student-virtual-machine:~/Practise$ ls -lh
total 0
-rw-rw-r-- 1 student student 0 Dec 17 15:04 duplicate.txt
-rw-rw-r-- 1 student student 0 Dec 17 15:02 sample.txt
```

rm – command used to delete a file,folder,directory.

```
student@student-virtual-machine:~/Practise$ rm duplicate.txt
```

Task-4

Use grep cmd to find all the instance of “Linux” in LinuxHistory.txt

and redirect output to LinuxInstances.txt.

#creating file LinuxHistory.txt.

```
student@student-virtual-machine:~$ touch LinuxHistory.txt
student@student-virtual-machine:~$ cat >> LinuxHistory.txt
Linux kernel devloped by linus torvald
Linux is distributed under GNU License
Linux comes with different versions popularly known as Distribution("Distros") like fedora,ubuntu,arch-linux,kali-linux
Linux is renowned for being robust and secure
Linux dominates cloud computing.student@student-virtual-machine:~$
```

Use grep to filter all line that contain instance of Linux.

```
Linux dominates cloud computing.student@student-virtual-machine:~$ grep -o "Linux" LinuxHistory.txt > LinuxInstance
student@student-virtual-machine:~$ cat LinuxInstances.txt
Linux
Linux
Linux
Linux
Linux
Linux
Linux
Linux
Linux
Linux
```

Task-5

Permissions and Ownership

#View current permissions for sample.txt

```
student@student-virtual-machine:~$ touch sample.txt
student@student-virtual-machine:~$ ls -hil sample.txt
9439973 -rw-rw-r-- 1 student student 0 Dec 20 16:21 sample.txt
```

#change the permission of sample.txt to read only for the owner and no permission for others.

```
student@student-virtual-machine:~$ chmod 400  sample.txt
student@student-virtual-machine:~$ ls -hil sample.txt
9439973 -r-------- 1 student student 0 Dec 20 16:21 sample.txt
```

#change the ownership of LinuxHistory.txt to another user.

```
student@student-virtual-machine:~$ sudo chown student1 LinuxHistory.txt
[sudo] password for student:
student@student-virtual-machine:~$ ls -lih LinuxHistory.txt
9439978 -rw-rw-r-- 1 student1 student 0 Dec 20 16:28 LinuxHistory.txt
```

Task -6

Remote System Access

#use ssh to connect to remote system

```
student@student-virtual-machine:~$ ssh -i "sarabesh-test-server.pem" ec2-user@35.154.250.9
The authenticity of host '35.154.250.9 (35.154.250.9)' can't be established.
ED25519 key fingerprint is SHA256:umyjrxDxMQG5L0STJg/QqwSMWmnqkkzxPsnupuBJkj4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '35.154.250.9' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-192-168-0-147 ~]$ sudo apt upgrade
```

#once connected use scp to copy LinuxHistory.txt from local machine to remote system.

```
student@student-virtual-machine:~$ scp -i "sarabesh-test-server.pem" LinuxHistory.txt ec2-user@35.154.250.9:/home/ec2-user/
LinuxHistory.txt                                              100%    0     0.0KB/s   00:00
```

```
student@student-virtual-machine:~$ ssh -i "sarabesh-test-server.pem" ec2-user@35.154.250.9
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Dec 20 19:14:13 2025 from 103.195.247.230
[ec2-user@ip-192-168-0-147 ~]$ ls -lhi
total 0
8567532 -rw-rw-r--. 1 ec2-user ec2-user 0 Dec 20 19:36 LinuxHistory.txt
[ec2-user@ip-192-168-0-147 ~]$ rm LinuxHistory.txt
[ec2-user@ip-192-168-0-147 ~]$ Connection to 35.154.250.9 closed by remote host.
Connection to 35.154.250.9 closed.
```

Assignment -1

Text Processing and Automation - Write a bash script that uses grep, sed, and awk to list all login attemtps to a linux system, extract attempted user and error messages to a separate file.

```
student@student-virtual-machine:~$ sudo grep -E "Accepted password|Accepted
      publickey|Failed passwod|/var/log/auth.log  \
      | sed 's/invalid user //g" \
      | awk '
           /Accepted/ {
              status-="SUCCESS"
              user=$4
           }  ip=$6
           /Failed/ {
              status="FAILED"
              user=$4
           }  ip=$6
        } printf"%-7s %-15s $s\n", status, user, ip '"
      }  output.txt

student@student-virtual-machine:~$ cat output.txt
SUCCESS    student        192.168.1.10     192.168.1.10
FAILED     student        203.0.113.45     203.0.113.45
FAILED     admin          198.51.100.22    198.51.100.22
SUCCESS    student        192.168.1.15     192.168.1.15
FAILED     test           203.0.113.89     203.0.113.89
```

Assignment-2

Version Control with Git - Initialize a new Git repository, simulate making changes to a project by adding text files, and commit those changes. Create a new branch, make further changes in this branch, and merge it back to the main branch, resolving a simulated merge conflict.

#created a new git local repository and pushed changes to the remote repository.

```
HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent
$ git init
Initialized empty Git repository in E:/rag_agent/.git/

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (main)
$ git add .
```

```
HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (main)
$ git commit -m "Initial commit"
[main (root-commit) 6186653] Initial commit
 20 files changed, 1070 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 agents/__init__.py
 create mode 100644 agents/rag_agent.py
 create mode 100644 data/original.txt
 create mode 100644 data/sample.txt
 create mode 100644 data/validated.txt
 create mode 100644 groq_agent/__init__.py
 create mode 100644 groq_agent/agent.py
 create mode 100644 main.py
 create mode 100644 rag_agents/__init__.py
 create mode 100644 rag_agents/agent.py
 create mode 100644 requirements.txt
 create mode 100644 sequential_agent/__init__.py
 create mode 100644 sequential_agent/agent.py
 create mode 100644 sequential_agent/groq.py
 create mode 100644 sequential_agent/subagents/__init__.py
 create mode 100644 sequential_agent/subagents/file_processor/__init__.py
 create mode 100644 sequential_agent/subagents/file_processor/agent.py
 create mode 100644 sequential_agent/subagents/validator/__init__.py
 create mode 100644 sequential_agent/subagents/validator/agent.py
```

```
HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (main)
$ git push -u origin main
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 8 threads
Compressing objects: 100% (22/22), done.
Writing objects: 100% (29/29), 14.07 KiB | 1.41 MiB/s, done.
Total 29 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Sarabesh-13/Rag_Agent.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

#now create another branch and made some changes, then merged changes to the main branch also.

```
HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (main)
$ git branch -M test

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (test)
$ git checkout test
Already on 'test'
Your branch is up to date with 'origin/main'.

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (test)
$ git fetch

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (test)
$ git status
On branch test
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)

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (test)
$ git branch
* test
```

```
HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (test)
$ git add README.md

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (test)
$ git commit -m "added README.md file in test branch"
[test f504e6d] added README.md file in test branch
 1 file changed, 20 insertions(+)
 create mode 100644 README.md

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (test)
$ git push -u origin test
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 869 bytes | 869.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'test' on GitHub by visiting:
remote:      https://github.com/Sarabesh-13/Rag_Agent/pull/new/test
remote:
To https://github.com/Sarabesh-13/Rag_Agent.git
 * [new branch]      test -> test
branch 'test' set up to track 'origin/test'.
```

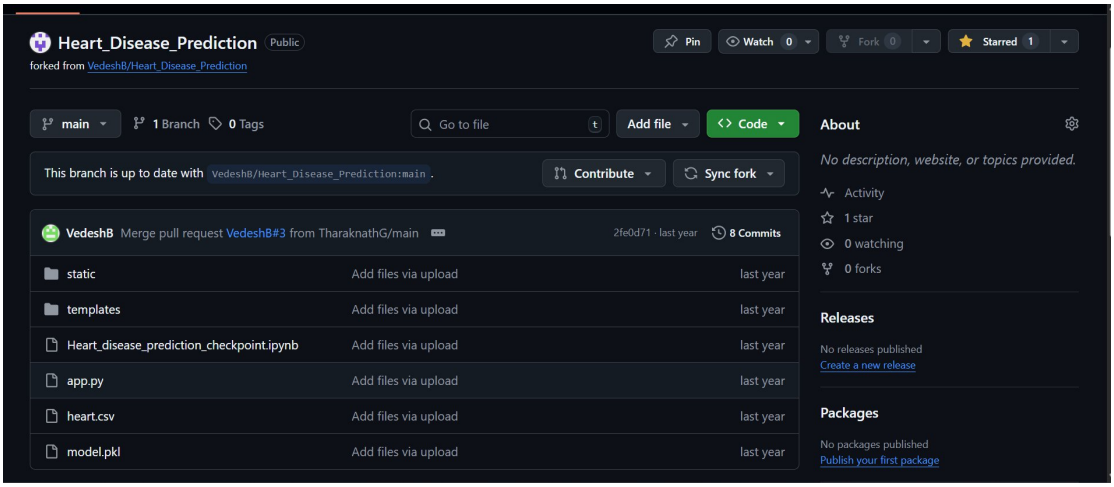
```
HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (main)
$ git merge test
Already up to date.

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (main)
$ git status
On branch main
nothing to commit, working tree clean

HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (main)
$ git commit -m "merged changes from test with main branch"
On branch main
nothing to commit, working tree clean
```

```
HP@DESKTOP-N0ID8BU MINGW64 /e/rag_agent (main)
$ git push -u origin main
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Sarabesh-13/Rag_Agent.git
   6186653..f504e6d  main -> main
branch 'main' set up to track 'origin/main'.
```

Assignment-3

Collaborative Development Using Git - Fork an existing repository (this can be a simulated action if no actual repository is available), clone it locally, and demonstrate managing updates from the original repository. Create a pull request to the original repository with changes made in the forked repository, and outline a simple code review process.

#fork an existing repository



#cloning forked repository

This forked repository has link to the original repository we can make pull request to original repository and keep forked repo up to date.

```
HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction
$ git clone git@github.com:Sarabesh-13/Heart_Disease_Prediction.git
Cloning into 'Heart_Disease_Prediction'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 27 (delta 8), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (27/27), 212.96 KiB | 356.00 KiB/s, done.
Resolving deltas: 100% (8/8), done.
```

#add original repository as upstream

```
HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ git remote add upstream https://github.com/VedeshB/Heart_Disease_Prediction.git
```

```
HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ git remote -v
origin  git@github.com:Sarabesh-13/Heart_Disease_Prediction.git (fetch)
origin  git@github.com:Sarabesh-13/Heart_Disease_Prediction.git (push)
upstream        https://github.com/VedeshB/Heart_Disease_Prediction.git (fetch)
upstream        https://github.com/VedeshB/Heart_Disease_Prediction.git (push)
```

#make changes to fork repository

```
HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ 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)

HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ git add .

HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ git commit -m "first commit in forked repository"
[main 5e7b2c1] first commit in forked repository
 1 file changed, 48 insertions(+)
 create mode 100644 README.md
```

```
HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.33 KiB | 1.33 MiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:Sarabesh-13/Heart_Disease_Prediction.git
   2fe0d71..5e7b2c1  main -> main
branch 'main' set up to track 'origin/main'.
```

#manages changes from original repository

```
HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ git fetch upstream
From https://github.com/VedeshB/Heart_Disease_Prediction
 * [new branch]      main       -> upstream/main
```

```
HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ git merge upstream/main
Already up to date.
```

#create pull request to original repository

```
HP@DESKTOP-N0ID8BU MINGW64 /e/heart_disease_prediction/Heart_Disease_Prediction (main)
$ git merge upstream/main
Already up to date.
```

#simple outline code review process

1) Forked repository in GitHub. which has access to make only pull request original repository and makes the forked repo up to date with the upstream/original repository,
2) Clone the forked repo and make some changes and push to main branch .these changes are reflected to only forked repository.
3) We will get to know as updates if there is a new changes in main branch of original repository.
   At that time we can first use git fetch to know about changes to original/main repo and then make pull request to origin main branch and make push request to remote fork main branch to keep code up to date.