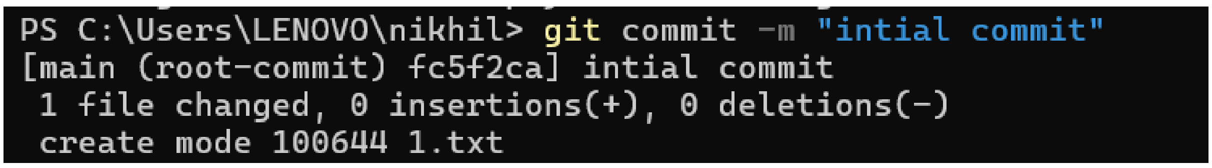
Git Commands Executed and Observations

- **git commit Command:**



```
PS C:\Users\LENOVO\nikhil> git commit -m "intial commit"
[main (root-commit) fc5f2ca] intial commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
```

Explanation:

git commit is used to save the staged changes in your local repository as a snapshot. It records the changes along with a message describing what was modified. Each commit creates a unique version in the project history that we can track or revert later.

1. Authentication Methods in Git: SSH Keys and HTTPS

- In Git-based version control systems like GitHub, authentication is required to securely connect a local machine to a remote repository. Two commonly used authentication methods are HTTPS and SSH.
- HTTPS (Hyper Text Transfer Protocol Secure) uses username and password or a Personal Access Token (PAT) for authentication. It is simple to set up and widely used, especially for beginners.
- SSH (Secure Shell) uses cryptographic key pairs (public and private keys) to establish a secure and password-less connection between the local machine and the remote repository. It is more secure and commonly used in professional DevOps environments.

A) SSH Authentication

SSH (Secure Shell) uses a pair of cryptographic keys:

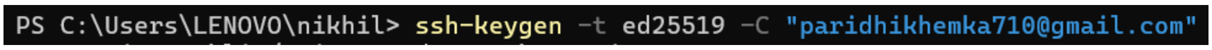
- Public Key (shared with GitHub)
- Private Key (kept secret on our system)

This allows secure, password-less login.

Steps to Create SSH Key:

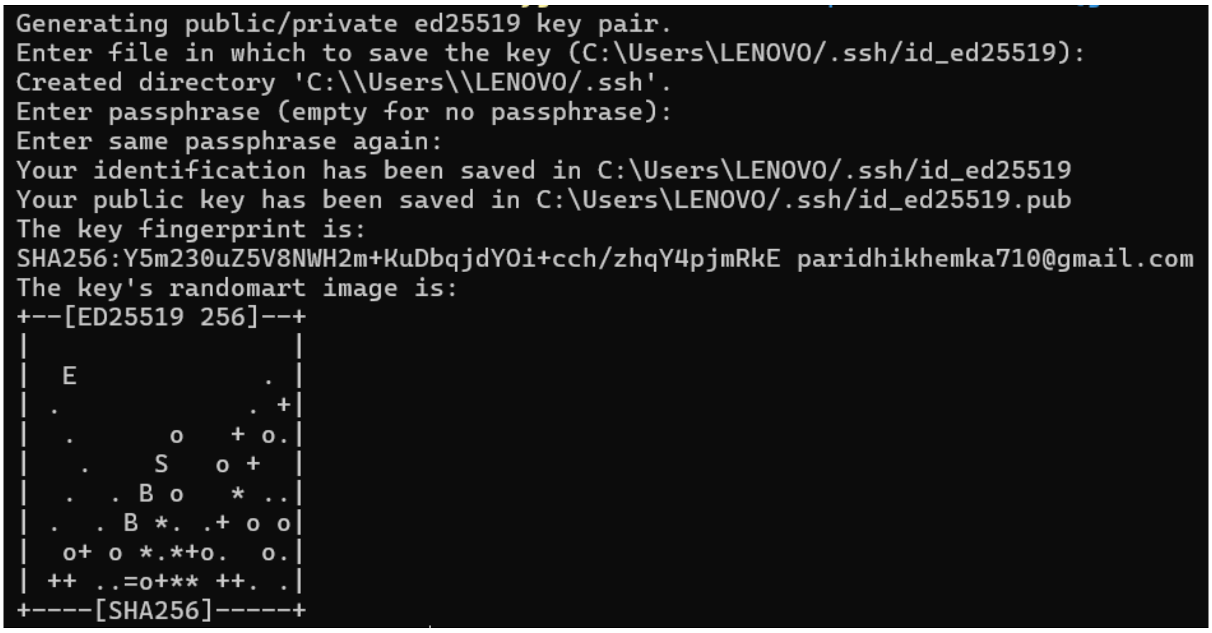
- **Step 1: Generate SSH Key**

  Command:



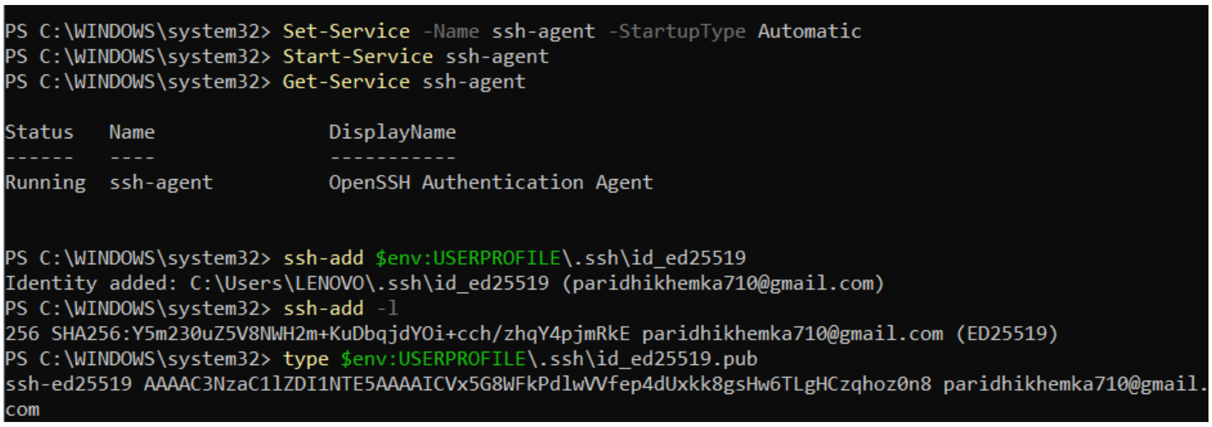
```
PS C:\Users\LENOVO\nikhil> ssh-keygen -t ed25519 -C "paridhikhemka710@gmail.com"
```

  Output:



```
Generating public/private ed25519 key pair.
Enter file in which to save the key (C:\Users\LENOVO/.ssh/id_ed25519):
Created directory 'C:\\Users\\LENOVO/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\LENOVO/.ssh/id_ed25519
Your public key has been saved in C:\Users\LENOVO/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:Y5m230uZ5V8NWH2m+KuDbqjdYOi+cch/zhqY4pjmRkE paridhikhemka710@gmail.com
The key's randomart image is:
+--[ED25519 256]--+
|                 |
|  E            . |
| .            . +|
|  .      o   + o.|
|   .    S   o +  |
|  .  . B o   * ..|
| .  . B *. .+ o o|
|  o+ o *.*+o.  o.|
| ++ ..=o+** ++. .|
+----[SHA256]-----+
```

- **Step 2: Start SSH Agent**



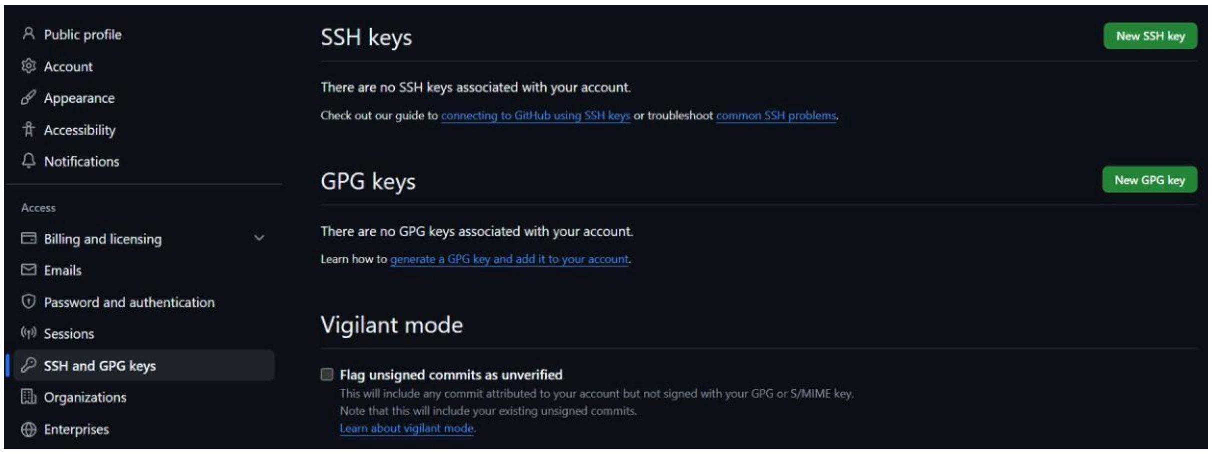
```
PS C:\WINDOWS\system32> Set-Service -Name ssh-agent -StartupType Automatic
PS C:\WINDOWS\system32> Start-Service ssh-agent
PS C:\WINDOWS\system32> Get-Service ssh-agent

Status   Name               DisplayName
------   ----               -----------
Running  ssh-agent          OpenSSH Authentication Agent


PS C:\WINDOWS\system32> ssh-add $env:USERPROFILE\.ssh\id_ed25519
Identity added: C:\Users\LENOVO\.ssh\id_ed25519 (paridhikhemka710@gmail.com)
PS C:\WINDOWS\system32> ssh-add -l
256 SHA256:Y5m230uZ5V8NWH2m+KuDbqjdYOi+cch/zhqY4pjmRkE paridhikhemka710@gmail.com (ED25519)
PS C:\WINDOWS\system32> type $env:USERPROFILE\.ssh\id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAICVx5G8WFkPdlwVVfep4dUxkk8gsHw6TLgHCzqhoz0n8 paridhikhemka710@gmail.
com
```

- **Step 5: Add SSH Key to GitHub**

    a. GitHub → Settings
    b. SSH and GPG keys
    c. New SSH Key
    d. Paste the copied key
    e. Save



SSH keys

There are no SSH keys associated with your account.

Check out our guide to connecting to GitHub using SSH keys or troubleshoot common SSH problems.

GPG keys

There are no GPG keys associated with your account.

Learn how to generate a GPG key and add it to your account.

Vigilant mode

Flag unsigned commits as unverified

This will include any commit attributed to your account but not signed with your GPG or S/MIME key.
Note that this will include your existing unsigned commits.
Learn about vigilant mode.

- **Step 6: Test SSH Connection**

Command:

```
PS C:\WINDOWS\system32> ssh -T git@github.com
```

Output:

```
Hi Paridhi3012! You've successfully authenticated, but GitHub does not provide shell access.
```

B) HTTPS Authentication in Git (Using Personal Access Token)

HTTPS (Hyper Text Transfer Protocol Secure) is a secure method to connect a local Git repository with a remote repository (like GitHub). Instead of a password, GitHub now requires a Personal Access Token (PAT) for authentication.

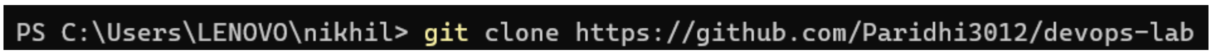
<u>Steps to Use HTTPS Authentication:</u>

- **Step 1: Generate Personal Access Token (PAT):**

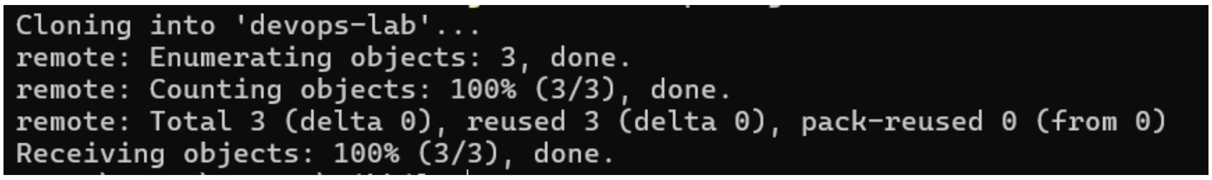
    a. Go to GitHub
    b. Click Profile Picture → Settings
    c. Scroll down → Developer Settings
    d. Click Personal Access Tokens
    e. Click Tokens (classic)
    f. Click Generate new token (classic)

g. Give name (Example: DevOpsLabToken)
h. Select Expiration
i. Select scope → ✓ repo
j. Click Generate Token

- **Step 2: Clone Using HTTPS**



```
PS C:\Users\LENOVO\nikhil> git clone https://github.com/Paridhi3012/devops-lab
```



```
Cloning into 'devops-lab'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```

2. Result

In this experiment, both HTTPS and SSH authentication methods were successfully implemented to connect a local Git repository with a remote GitHub repository. Using the HTTPS method, the repository was cloned and authenticated using a Personal Access Token (PAT). Changes were staged and committed successfully, confirming proper authentication and synchronization in the local repository.

Similarly, SSH authentication was configured by generating an SSH key pair, adding the public key to the GitHub account, and verifying the connection using the ssh -T command. The successful authentication message confirmed secure key-based access.

3. Conclusion

This experiment demonstrated the implementation of two secure Git authentication methods: HTTPS using a Personal Access Token and SSH using public-private key encryption. HTTPS authentication is simple to configure and suitable for beginners, while SSH provides enhanced security and password-less authentication, making it more suitable for professional and DevOps environments.