how to create ssh ed25519 for github in terminal

To create an **SSH ED25519 key** for GitHub in your terminal, follow these steps:

✅ Step-by-Step Guide

1. Open your terminal

Use Terminal (macOS/Linux) or Git Bash (Windows).

---

2. Generate the ED25519 SSH key

Run:

```
bash

ssh-keygen -t ed25519 -C "your_email@example.com"
```

- Replace `"your_email@example.com"` with the **email you use on GitHub**.
- The `-C` flag adds a label to the key for identification.

---

3. When prompted:

You'll see:

```
bash

Enter file in which to save the key (/home/you/.ssh/id_ed25519):
```

- Press **Enter** to accept the default location.
- Or, type a custom path (e.g., `~/.ssh/github_ed25519`) to save with a different name.

---

4. Create a passphrase (optional but recommended)

When prompted:

```
bash

Enter passphrase (empty for no passphrase):
```

- Type a **secure passphrase** (or press **Enter** for none).
- Re-enter it when asked.

---

5. Add the SSH key to the ssh-agent

Start the agent:

```
bash

eval "$(ssh-agent -s)"
```

Then add your key (default location shown):

```
bash

ssh-add ~/.ssh/id_ed25519
```

If you used a custom name:

```
bash

ssh-add ~/.ssh/github_ed25519
```

---

6. Copy the public key to your clipboard

Use:

```
bash

cat ~/.ssh/id_ed25519.pub
```

Or:

```
bash

pbcopy < ~/.ssh/id_ed25519.pub        # macOS
xclip -sel clip < ~/.ssh/id_ed25519.pub  # Linux (requires xclip)
clip < ~/.ssh/id_ed25519.pub          # Windows Git Bash
```

---

7. Add your key to GitHub

1. Go to **GitHub → Settings → SSH and GPG keys**
2. Click **"New SSH key"**
3. Give it a title (e.g., **`MacBook ED25519`**)
4. Paste the copied public key
5. Click **"Add SSH key"**

---

8. Test the connection

Run:

```
bash

ssh -T git@github.com
```

You should see:

```
bash

Hi your-username! You've successfully authenticated...
```

Let me know if you want a script to automate this!

> how to config username and email

To configure your **Git username and email**, which Git uses for commit history, follow these commands in your terminal:

✅ Set Global Git Username and Email

These settings apply to **all repositories** on your system:

```
bash

git config --global user.name "Your Name"
git config --global user.email "your_email@example.com"
```

Replace:

- `"Your Name"` with your **name**
- `"your_email@example.com"` with your **GitHub email**

🔍 Check Your Global Git Config

To verify the settings:

```
bash

git config --global --list
```

You should see something like:

```
ini

user.name=Your Name
user.email=your_email@example.com
```

🛠️ Set Username and Email for a Specific Repository Only

If you want to override your global settings **just for one repository**, do this **inside that repo folder**:

```
bash

git config user.name "Your Repo Name"
git config user.email "your_repo_email@example.com"
```

🧠 Bonus Tip: View All Git Config Levels

To see global, system, and local repo settings:

```
bash

git config --list --show-origin
```

This helps debug if there are conflicting settings.

Let me know if you'd like to **unset**, **edit**, or troubleshoot your Git config.

> Provide a list of most common commands in Bash and how to use

Here's a list of the **most common Bash commands**, grouped by purpose, with usage examples so you can understand how to use them in real scenarios.

📂 File and Directory Management

Command	Description	Example
`ls`	List directory contents	`ls -la`
`cd`	Change directory	`cd /path/to/folder`
`pwd`	Print working directory	`pwd`
`mkdir`	Create directory	`mkdir myfolder`
`rm`	Remove files/directories	`rm file.txt`, `rm -r folder/`
`touch`	Create empty file	`touch newfile.txt`
`cp`	Copy files/directories	`cp file1.txt file2.txt`
`mv`	Move/rename files	`mv old.txt new.txt`
`find`	Find files/folders	`find . -name "*.txt"`

📄 File Viewing and Editing

Command	Description	Example
`cat`	View file content	`cat file.txt`
`less` / `more`	Scroll through file content	`less file.txt`
`head`	View top lines of a file	`head -n 10 file.txt`
`tail`	View bottom lines	`tail -n 10 file.txt`
`nano`	Terminal text editor	`nano file.txt`
`vim`	Advanced terminal editor	`vim file.txt`

🔍 Searching and Filtering

Command	Description	Example
`grep`	Search text in files	`grep "search" file.txt`
`cut`	Extract sections from lines	`cut -d':' -f1 /etc/passwd`
`sort`	Sort lines	`sort names.txt`
`uniq`	Remove duplicate lines	`sort file.txt
`wc`	Count words, lines, chars	`wc -l file.txt`

🔄 File Permissions and Ownership

Command	Description	Example
`chmod`	Change file permissions	`chmod +x script.sh`
`chown`	Change file ownership	`sudo chown user:group file.txt`

🖥️ System Information

Command	Description	Example
`top`	Real-time system monitor	`top`
`htop`	Better system monitor (if installed)	`htop`
`df -h`	Disk usage	`df -h`
`du -sh`	Folder size	`du -sh folder/`
`uname -a`	Kernel/system info	`uname -a`

⚙️ Process and Job Control

Command	Description	Example
`ps`	List running processes	`ps aux`
`kill`	Kill a process by PID	`kill 1234`
`&`	Run process in background	`./script.sh &`
`jobs`	List background jobs	`jobs`

Command	Description	Example
`fg`	Bring job to foreground	`fg %1`
`bg`	Resume job in background	`bg %1`

🌐 Networking

Command	Description	Example
`ping`	Check network connection	`ping google.com`
`curl`	Make web requests	`curl https://api.github.com`
`wget`	Download files	`wget http://example.com/file.zip`
`ifconfig` or `ip a`	Network interfaces info	`ip a`

📦 Package Management (Debian-based systems like Ubuntu)

Command	Description	Example
`sudo apt update`	Update package list	`sudo apt update`
`sudo apt upgrade`	Upgrade packages	`sudo apt upgrade`
`sudo apt install`	Install package	`sudo apt install git`
`sudo apt remove`	Uninstall package	`sudo apt remove vim`

🛠️ Scripting & Misc

Command	Description	Example
`echo`	Print text	`echo "Hello World"`
`read`	Read input from user	`read name`
`date`	Show current date/time	`date`
`whoami`	Show current user	`whoami`
`history`	Show command history	`history`
`alias`	Create shortcut command	`alias ll='ls -la'`

🔒 File Compression and Archiving

Command	Description	Example
`tar`	Archive/unarchive files	`tar -czf archive.tar.gz folder/`
`zip/unzip`	Compress/uncompress	`zip files.zip file1 file2`, `unzip files.zip`

Let me know if you'd like:

- A **printable cheat sheet**

- A **beginner practice script**
- Or deeper examples for any specific command