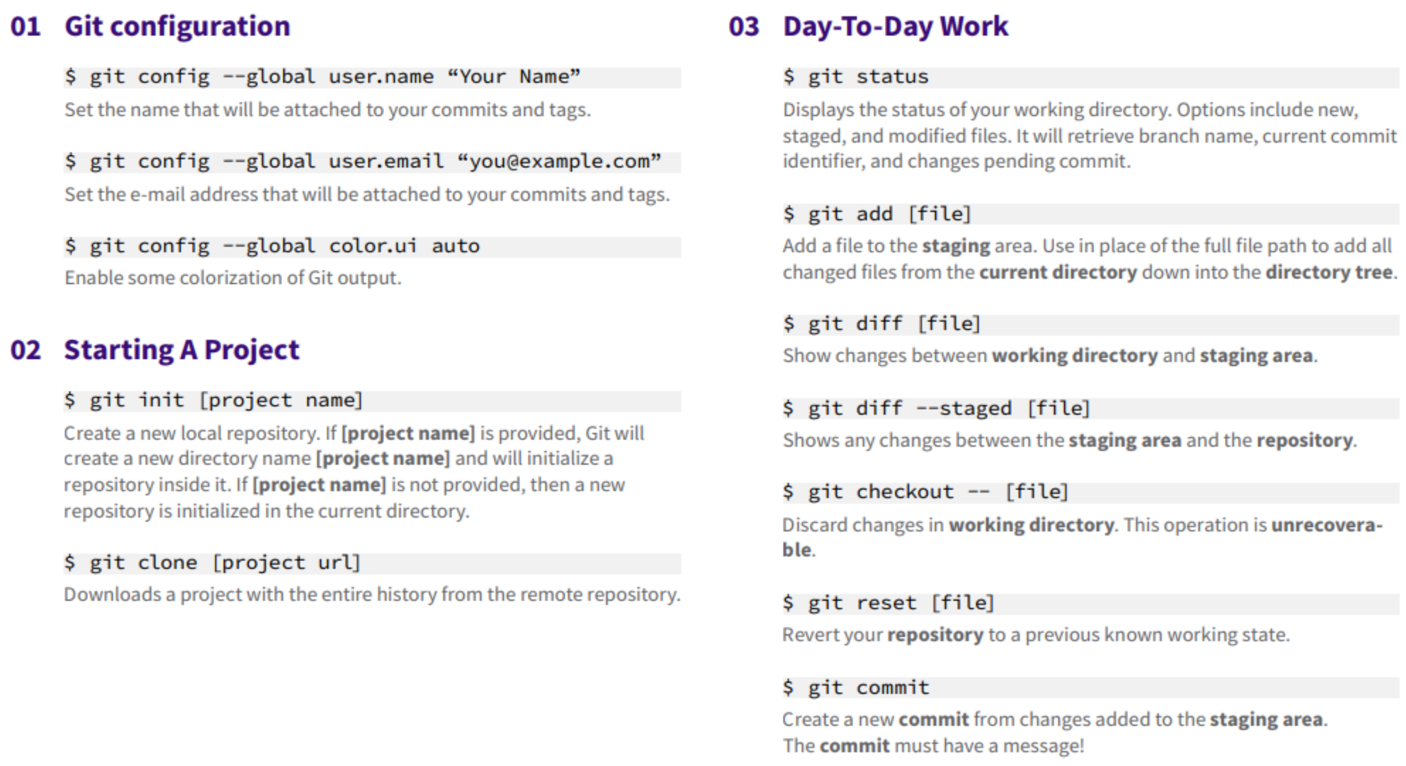
Linux Cheatsheet

By - Rushikesh S Khandare (Task - 12)

- ls - The most frequently used command in Linux to list directories
- pwd - Print working directory command in Linux
- cd - Linux command to navigate through directories
- mkdir - Command used to create directories in Linux
- mv - Move or rename files in Linux
- cp - Similar usage as mv but for copying files in Linux
- rm - Delete files or directories
- touch - Create blank/empty files
- ln - Create symbolic links (shortcuts) to other files
- cat - Display file contents on the terminal
- clear - Clear the terminal display
- echo - Print any text that follows the command
- less - Linux command to display paged outputs in the terminal
- man - Access manual pages for all Linux commands
- uname - Linux command to get basic information about the OS
- whoami - Get the active username
- tar - Command to extract and compress files in Linux
- grep - Search for a string within an output
- head - Return the specified number of lines from the top
- tail - Return the specified number of lines from the bottom
- diff - Find the difference between two files
- cmp - Allows you to check if two files are identical
- comm - Combines the functionality of diff and cmp
- sort - Linux command to sort the content of a file while outputting
- export - Export environment variables in Linux
- zip - Zip files in Linux
- unzip - Unzip files in Linux
- ssh - Secure Shell command in Linux
- service - Linux command to start and stop services
- ps - Display active processes
- kill and killall - Kill active processes by process ID or name
- df - Display disk filesystem information
- mount - Mount file systems in Linux
- chmod - Command to change file permissions
- chown - Command for granting ownership of files or folders
- ifconfig - Display network interfaces and IP addresses
- traceroute - Trace all the network hops to reach the destination
- wget - Direct download files from the internet

- ufw - Firewall command
- iptables - Base firewall for all other firewall utilities to interface with
- apt, pacman, yum, rpm - Package managers depending on the distro
- sudo - Command to escalate privileges in Linux
- cal - View a command-line calendar
- alias - Create custom shortcuts for your regularly used commands
- dd - Majorly used for creating bootable USB sticks
- whereis - Locate the binary, source, and manual pages for a command
- what is - Find what a command is used for
- top - View active processes live with their system usage
- useradd and usermod - Add new user or change existing users data
- passwd - Create or update passwords for existing users

Git and Github Cheatsheet



01 Git configuration

```
$ git config --global user.name “Your Name”
```
Set the name that will be attached to your commits and tags.

```
$ git config --global user.email “you@example.com”
```
Set the e-mail address that will be attached to your commits and tags.

```
$ git config --global color.ui auto
```
Enable some colorization of Git output.

02 Starting A Project

```
$ git init [project name]
```
Create a new local repository. If **[project name]** is provided, Git will create a new directory name **[project name]** and will initialize a repository inside it. If **[project name]** is not provided, then a new repository is initialized in the current directory.

```
$ git clone [project url]
```
Downloads a project with the entire history from the remote repository.

03 Day-To-Day Work

```
$ git status
```
Displays the status of your working directory. Options include new, staged, and modified files. It will retrieve branch name, current commit identifier, and changes pending commit.

```
$ git add [file]
```
Add a file to the **staging** area. Use in place of the full file path to add all changed files from the **current directory** down into the **directory tree**.

```
$ git diff [file]
```
Show changes between **working directory** and **staging area**.

```
$ git diff --staged [file]
```
Shows any changes between the **staging area** and the **repository**.

```
$ git checkout -- [file]
```
Discard changes in **working directory**. This operation is **unrecovera-ble**.

```
$ git reset [file]
```
Revert your **repository** to a previous known working state.

```
$ git commit
```
Create a new **commit** from changes added to the **staging area**. The **commit** must have a message!

```
$ git rm [file]
```
Remove file from **working directory** and **staging area**.

```
$ git stash
```
Put current changes in your **working directory** into **stash** for later use.

```
$ git stash pop
```
Apply stored **stash** content into **working directory**, and clear **stash**.

```
$ git stash drop
```
Delete a specific **stash** from all your previous **stashes**.

04 Git branching model

```
$ git branch [-a]
```
List all local branches in repository. With **-a**: show all branches (with remote).

```
$ git branch [branch_name]
```
Create new branch, referencing the current **HEAD**.

```
$ git checkout [-b][branch_name]
```
Switch **working directory** to the specified branch. With **-b**: Git will create the specified branch if it does not exist.

```
$ git merge [from name]
```
Join specified **[from name]** branch into your current branch (the one you are on currently).

```
$ git branch -d [name]
```
Remove selected branch, if it is already merged into any other.
-D instead of **-d** forces deletion.

05 Review your work

```
$ git log [-n count]
```
List commit history of current branch. **-n count** limits list to last **n** commits.

```
$ git log --oneline --graph --decorate
```
An overview with reference labels and history graph. One commit per line.

```
$ git log ref..
```
List commits that are present on the current branch and not merged into **ref**. A **ref** can be a branch name or a tag name.

```
$ git log ..ref
```
List commit that are present on **ref** and not merged into current branch.

```
$ git reflog
```
List operations (e.g. checkouts or commits) made on local repository.