**Git commands**

**Configure user information for all local repositories**

$ git config --global user.name "[name]"

Sets the name you want atached to your commit transactions

$ git config --global user.email "[email address]"

Sets the email you want atached to your commit transactions

**CREATE REPOSITORIES**

Start a new repository or obtain one from an existing URL

$ gitinit [project-name] (We are not using this )

Creates a new local repository with the specified name

$ git clone [url]

Downloads a project and its entire version history

$git rebase -i SHA1

$git rebase upstream/<branch\_name>

**MAKE CHANGES**

Review edits and craf a commit transaction

$ git status

Lists all new or modified files to be commited

$ git add [file]

Snapshots the file in preparation for versioning

$ git reset [file]

Unstages the file, but preserve its contents

$ git diff

$ git commit -am "[descriptive message]"

Records file snapshots permanently in version history

$git commit --amend -m "New commit message"

**GROUP CHANGES**

Name a series of commits and combine completed efforts

$ git branch

Lists all local branches in the current repository

$ git branch [branch-name]

Creates a new branch

$ git checkout [branch-name]

Switches to the specified branch and updates the working directory

$ git checkout -b [name\_of\_your\_new\_branch]

$git checkout -b newbranch SHA1

Create the branch on your local machine and switch in this branch

$ git merge [branch]

Combines the specified branch’s history into the current branch

$ git branch -d [branch-name]

Deletes the specified branch

git checkout [branch-name]

git cherry-pick SHA1

**REFACTOR FILENAMES**

Relocate and remove versioned files

$ git rm [file]

Deletes the file from the working directory and stages the deletion

$ git mv [file-original] [file-renamed]

Changes the file name and prepares it for commit

**SAVE FRAGMENTS**

Shelve and restore incomplete changes

$ git stash

Temporarily stores all modified tracked files

$ git stash list

Lists all stashed changesets

$ git stash pop

Restores the most recently stashed files

$ git stash drop

Discards the most recently stashed changeset

REVIEW HISTORY

Browse and inspect the evolution of project files

$ git log

Lists version history for the current branch

$ git log --follow [file]

Lists version history for a file, including renames

$ git diff [first-branch]...[second-branch]

Shows content differences between two branches

$ git show [commit]

Outputs metadata and content changes of the specified commit

IMP: git diff --name-only SHA1 and SHA2

Erase mistakes and craf replacement history

$ git reset [commit]

Undoes all commits afer [commit], preserving changes locally

$ git reset --hard [commit]

Discards all history and changes back to the specified commit

SYNCHRONIZE CHANGES

Register a repository bookmark and exchange version history

$ git fetch [bookmark]

Downloads all history from the repository bookmark

$ git merge [bookmark]/[branch]

Combines bookmark’s branch into current local branch

$ git push [alias] [branch]

Uploads all local branch commits to GitHub

$ git pull

Downloads bookmark history and incorporates changes

ssh-keygen -t rsa – for keygen

$ git show-branch –list –show the all branches

Git squash sha – for ignore unwanted commits (latest commits into one)

Git pull upstream branch name– get the new updated information