কিছু অ্যাডভান্সড গিট এর কাজ এখানে লিখলামঃ(প্রতিটি কমান্ডের শুরুতে আমি # দিব যেটা কমান্ডের সময় দেয়া যাবে না।,কেমন?)

১। আমরা যদি চাই তবে একসাথে একাধিক ফাইলও কমিট করতে পারি।

```
# git commit –m “আমরা যে কমিটটা লিখতে চাই ”file1.extension file2.extension”
```

(ফাইলগুলোর নাম লিখে তাদের এক্সটেনশনগুলো দিয়ে আমরা এই কাজটি করতে পারি, তবে অবশ্যই তাদের মাঝে স্পেস দিতে হবে(কমা “ , “ দিলে হবে না।))

২। এখন যদি আমরা চাই আমাদের পঞ্চম কমিটের বেলায় আমাদের টোটাল প্রজেক্টটা কেমন ছিল সেটা দেখতে তাহলে **git log** থেকে আমাদের ঐ কমীটের হ্যাশ ভ্যালু কপি করে নিয়ে এসে যে কমান্ডটা দিতে হবে সেটা হল

```
# git checkout hash_value
```

এটা লিখার পর আমরা লিখব **# cat file.estension** তাহলে আমরা দেখব যে ঐ সময় ফাইলটা কেমন ছিল , কি কি করেছি। এখন যদি এই ফাইলটাই আমরা আমাদের বর্তমান রিপোজিটারিতে রাখব তাহলে আমাদের লিখতে হবে(আমাদের এখানে যে হ্যাশ ভ্যালুটা ব্যবহার করছি সেটা হবে ঐ সময় থাকা কমিটের হ্যাশ ভ্যালু।)

#git checkout hash_value file_name.extension (এই কাজ করলে আমাদের হেড কমিট/ মাস্টার কমিট নোংরা হয়ে যায় তখন আমাদের আমাদের কমিট করতে হয় , তাহলে এই সময় কাকে কমিট করব? কেন যতদোষ **README.md** ঘোষ আছে না, এছাড়াও আমাদের কমিট নিয়ে নিবে যদি আমারা **README.md** কে কমিট করতে না চাই।)

আর যদি মনে করি না থাক আমাদের বর্তমান অবস্থা ভাল আছে সেক্ষেত্রে লিখব

`# git checkout master –f` (শুধুমাত্র একটি ফাইলকে বর্তমান সময়ে আনার জন্যে আমাদের –f হয়ে, তাছাড়া এটা তো আমাদের চেনা কমান্ড।)

৩।গিটে কোন ফাইলে কাজ করে সেটাতে কমিট দিলাম এরপর আবার অন্য একটা ফাইলে কাজ করলাম ,ধরে নেই যে প্রায় একই রকম কাজ করলাম এখন আমরা

কি করব আবার সেই কমিটটা লিখব, ভাই প্রোগ্রামার মানুষতো এক কাজ বারবার করি না, সেটা অ্যামেন্ড করি,আমাদের সদ্য পরিবর্তিত ফাইলটাকে যদি একটূ আগে দেয়া ফাইলটার মত কমিট দিতে চাই তাহলে লিখব (নিচের কমান্ড দেয়ার আগে আমরা **# git log** কমান্ডটি দেই এবং হেডে থাকা হ্যাশ ভ্যালুটি ও কমিটটি মনে রাখি।)

 # git commit –amend

কমান্ডটি দেয়ার পর আমরা আবার **# git log** লিখি দেখা যাবে আমাদের হ্যাশ ভ্যালু চেঞ্জ হয়েছে কিন্তু কমিটটি চেঞ্জ হয় নি। তার মানে কি? এর মানে হল কমিটটি রয়ে গেছে কিন্তু ঐ হ্যাশ ভ্যালু যে কাজের জন্য দেয়া হয়েছিল সেটি তো আমরা পরিবর্তন করলাম, (কারণ ঐ হ্যাশ ভ্যালু দিয়ে আমরা আগে একটি ফাইল কমিট করেছি আর এখন দুইটি তাই।)

৪। ধরা যাক আমরা একটা ফাইলে কিছু লিখেছি যেগুলোতে আসলে আমরা কমিট করতে চাইছি না, টেস্ট করে দেখতে চাই কি রকম হয়। আবার শুধু তাই নয় কিছুদূর লিখব তারপর আবার একটা নির্দিষ্ট যায়গায় যেতে চাই। মনে করি আমাদের **msg.txt** নামের টেক্সট ফাইল আছে। যেটাতে আমাদের লিখা ছিল “How are you? Is everything is fine?” এখন আমরাআরো কিছু লিখতে চাই কিন্তু কি কি লিখব সেটা জানি না, সেক্ষেত্রে কি হবে। এখন কিছু লিখে সেটাকে কমিট করার তো মানে হয় না, কতবার কমিট করব, আমরা সবসময় কাজ শেষেই ঐ ফাইলে কমিট করি। এখানেই লাগে stash. এখন আমরা msg.txt এ লিখলাম “ Wil you go outside now or later?” এখান আমরা যদি # git diff এই কমান্ডটা দেই তবে দেখাবে আমাদের টেক্সট ফাইলে কি কি পরিবর্তন এসেছে, কিছুক্ষণ পর মনে হল আসলে এটা লিখে ঠিক হয় নি, আবার ডিলিটও করতে ইচ্ছা করছে না। তখন আমরা লিখব # git stash. এবার **# git diff** লিখুন দেখবেন কোন পরিবর্তনের নোটিফিকেশনস নেই। এবং আমাদের msg.txt দেখুন “will you go..... later?” এই লিখাটাও নাই, মানে আগে যেরকম ছিল সে রকমই আছে। তাহলে আমাদের লিখটা গেল কই? **# git stash list** (file name) লিখলে দেখতে পারবেন কি কি পরিবর্তন এসেছে। আপনার ঐ ফাইলে কি কি পরিবর্তন এনেছেন, এখন পছন্দ মত যেটা রাখতে চান সেটির **stash@{intger_number}** টা কপি করুন, আর লিখুন **# git stash pop stash@{যে স্ট্যাশ টি রাখতে চান}** ব্যস হয়ে গেল। আপনি **#git stash pop stash@{__}** না লিখে এটাও লিখতে পারেন **#git stash apply stash@{__}**

৫। মনে করুন অনেকগুলো ফাইল তৈরী করেছেন, কিন্তু কি কি ফাইল অ্যাড করবেন বুঝতে পারছেন না, সেক্ষেত্রে লিখুন **# git clean –f –n** তাহলে দেখতে পারবেন কি কি ফাইল **git** মাধ্যমে ডিলিট করতে পারবেন। আপনি যদি এখন #git clean –f লিখেন তবে লিস্টে দেখান সব ফাইলই ডিলিট হয়ে যাবে। আর যদি নির্দিষ্ট কোন ফাইল ডিলিট করতে চান তবে লিখতে হবে **#git clean –f {file1.txt,file2.txt}**

৬। উপরের কাজটি করা যাবে যদি ফাইলগুলো অ্যাড করা না থাকে ।কিন্তু যদি অ্যাড করা থাকে কিন্তু কমিট করা না থাকে , তাহলে কি তখন **# git reset {file1.txt,file2.txt}**, ব্যস হয়ে গেল এখন ৫ নং পথ অনুসরণ করুন।

৭। **.gitignore** নামের একটা ফাইল খুলতে হবে। দেখা যায় অনেক সময় আমরা কিছু ফাইল গিটে স্টোর করতে চাই না। তখন আমাদের এই গিট ইগ্নোর ফাইলটি লাগে। এখন মনে করুন আপনার লোকাল রিপোতে **.zip** নামের ফাইল আছে, কিছু .mp3, .png এই সব এক্সটেনশনের ফাইল আছে আপনি এসব অনলাইন রিপোতে রাখতে চান না, ওকে নো সমস্যা। আমরা এখন খুব সুন্দর করে আমাদের **.gitignore** ফাইলে লিখব

```
*.zip
*.mp3
*.png
```

তাহলে হবে কি আপনার লোকাল ফোল্ডারে থাকা ঐসব এক্সটেনশনের কোন ফাইল গ্লোবালি অ্যাড করতে পারবেন না, আর আপনি চাইলে গিটইগনোর ফাইলটিও কমিট করে পুশ করে রাখতে পারেন।

যদি জোড় করে কোন ফাইল অ্যাড করতে চান তাহলে কিন্তু **#git add file_name** দিলে চলবে না। লিখতে হবে **# git add –f file_name** এখানে আমরা **–f** দ্বারা ফোর্স করা বুঝাচ্ছি।

৮।উফফ! কমিট করা ফাইল অ্যাড করা, গিট লগ দেখা, স্ট্যাটাস চেক করা এসবের জন্য কত বড় বড় কমান্ড লিখতে হয়, এটা থেকে রেহাই পাওয়ার জন্য

গিট বের করেছে গিট অ্যালিয়াস।মনে করি “git commit” না লিখে শুধুমাত্র gc লিখি আর তাতেই কাজ করে কেমন হবে , ভাল হবে না।তবে এই কাজের জন্য আমায় যেসব কাজ কররতে হবেঃ

১। সবার হোম এ এসে ব্লাংক স্পেসে রাইট ক্লিক করি (হোম বলতে বুঝাচ্ছি আমাদের কম্পিউটার অন করার পর যেটাকে আমরা ডেস্কটপ বলি।) অতঃপর **#vim ~/.myalias** নামের ফাইলটি তৈরী করি।

২। এরপর আমাদের কমান্ড সংক্ষিপ্ত করণ চলবে। প্রথমে লিখব

alias(space)(short_version)=”original_command” । মনে করি, আমাদের git commit এটি প্রচুর লিখতে হয়, তাহলে git commit না লিখে লিখতে পারি “gc” তাহলে এখন আমরা কি করব? আমরা লিখব alias gc=”git commit” । মাথায় রাখব কোন প্রকার এক্সট্রা স্পেস দেয়া চলবে না। gc এর পরেপরেই “=” লিখব এবং তারপরেই “original command” লিখব।

৩। এরপর যারা লিনাক্স চালাই তারা লিখবে **#vim ~/.bashrc** আর যারা ম্যাক ইউজ করি তাদের জন্য # **vim ~/.zshrc** লিখব। দেখা যাবে অনেক লম্বা ফাইল।একদম সবার শেষে আমরা যা সেখানে লিখব **test –r ~/.myalias && source ~/.myalias**

৪। আমাদের যদি কোন কমান্ডের জন্য এলিয়াস পরিবর্তন করতে চান তাহলে ১ নং পথ অবলম্বন করুন এবং ফাইল এডিট করুন।