GIT Internal Demo		
Commands (Terminal One)	**Commands (Terminal Two)**	**Description**
Create a empty GIT Repo		
git init git-internals-demo		
ls		
ls -la	tree .git	Tree command
		git init option. explain about the default branch.
Add one file and create the first commit.		
touch README.md	viddy -n 0.5 tree .git	
		Explain the GIT workflow : working directory -> staging area -> Local Repo -> Remote
ls		
git status		
git add README.md		Some object is created in .git directory.
		Explain the different object available in GIT. 1. Blob : Stored the original content 2. Tree : stored the information about directries & files. 3. Commit : Create complete snapshot of your project.
		Now we will try to check Blob object.
git show HASH		Explain about the HASH
git commit -m "Added the README.md file."	1	Its created two more objects. -> tree & commit
		Now we will try to check Tree & commit object.
git show HASH		IT is tree object
git ls-tree HASH		It will contains permissions, object type, object adress, file name
git show HASH		IT is commit object - Explain the details
git show --pretty=raw 7beb87		Explain the output Tell History about the GIT
git log		
		Also show same things in GIT GUI tool

As a end user we never think about tree & blob. We only talk about snapshot/commit like we have to back to particular, we have merge two snapshot like that.		
cat .git/refs/heads/master		Ref is nothing but the pointers.
Add one more commit for understanding.		
vi README.md		# GIT Internals Demo
git status		
git add .		One Blob object is created.
git commit -m "Updated the README.md file"	2	Two Object created -> Tree & Commit.
git logs		Show git commits
git show --pretty=raw commit-HASH		Ask people about the commands.
Explain Object pointing flow. (Immutable)		
git ls-tree tree-HASH		
git show blob-HASH		
Understand the GIT reset with multiple file changes		
mkdir lib		
touch lib/cal.py		
touch hello.py ping.py		Open with sublime write hello program
git add .		Numbers files+directories = Number of Blob Object
		By mistake you add one unrequire file and now we have to remove it from GIT staging area
git diff		Explain the GIT diff command.
git reset ping.py		ping.py file will be remove from GIT staging are.
git diff		
git diff --staged		
git commit -m "Added multiple files."	3	It will create two tree object and one commit Object
git add ping.py		
git commit -m "Added ping.py file."	4	
git log		Also show with GIT GUI Tool
Explain About the GIT Rebase		

vi ping.py		def main(): print("Ping to Google") main()
git add ping.py		
git commit -m "Updated the ping.py file."		
git log		Also show with GIT GUI tool
git rebase -i HEAD~4		Explain it about
git log		Also show with GIT GUI tool
Explain About the GIT Reflog		
git reflog		
git reset ID		
git reflog		Added one more entry
Bitbucket Demo		