Q) How to delete a branch in git ?

A) git branch –d <branch name>

Q) How to verify number of branches in git repository?

A) git branch –a

Q) How to revert first 10 commits in git?

A) git reset --hard HEAD~10 , if they say this answer ask how they push the changes to repository in github after reverting 🡪 answer : git push –f

Q) What is git stash?

A) If we are working on the script and we are in middle of our scripts creation and if there is any PRB that needs to be fixed first, then in order to save our scripts which we stopped in the middle, we use stash, once that PRB is delivered then we can again work on stashed chnages

Q)What is git tagging? What is the command

A)to tag specific points in history as being important 🡪 git tag <tag-name> which is light weight tag or git tag –a <tag-name> which is annonated tag

Q) when we directly do any chnages in repository in github, what command we use to make our local repo and github repo in sync?

A) If they say answer git clone its fine but ask them if u have code already in local then answer is 🡪 git fetch then we need to use git pull , then chnages will be sync to local repo from github repo

Q) How to rename a branch?

A) If you are on the branch you want to rename:

|  |  |
| --- | --- |
| 1 | git branch -m new-name |

If you are on a different branch:

|  |  |
| --- | --- |
| 1 | git branch -m old-name new-name |

Delete the old-name remote branch and push the new-name local branch.

|  |  |
| --- | --- |
| 1 | git push origin :old-name new-name |

**Create a local .gitignore**

If you create a file in your repository named *.gitignore*, Git uses it to determine which files and directories to ignore, before you make a commit.

A *.gitignore* file should be committed into your repository, in order to share the ignore rules with any other users that clone the repository.

GitHub maintains an official list of recommended *.gitignore* files for many popular operating systems, environments, and languages [in the github/gitignore public repository](https://github.com/github/gitignore).

1. In Terminal, navigate to the location of your Git repository.
2. Enter touch .gitignore to create a *.gitignore* file.

The Octocat has [a Gist containing some good rules](https://gist.github.com/octocat/9257657) to add to this file.

If you already have a file checked in, and you want to ignore it, Git **will not** ignore the file if you add a rule later. In those cases, you must untrack the file first, by running the following command in your terminal:

git rm --cached *FILENAME*

**Create a global .gitignore**

You can also create a global *.gitignore* file, which is a list of rules for ignoring files in *every* Git repository on your computer. For example, you might create the file at *~/.gitignore\_global* and add some rules to it.

1. Open Terminal.
2. Run the following command in your terminal:
3. git config --global core.excludesfile ~/.gitignore\_global

The Octocat has [a Gist containing some good rules](https://gist.github.com/octocat/9257657) to add to this file.

======================================================================================================================

git reset --hard HEAD~6

git push -f

In the above one, there will be short lived branches(branch like diagram) and long lived branches(consists of 3dots), when we scan long lib branches – it will take the quality gate from the master branch – that might be the reason why it gets failed(if master is failed)

**===========================================================================================================**

**This is used only to delete(reset) till that particular commit id**

1. # Reset the index and working tree to the desired tree
2. # Ensure you have no uncommitted changes that you want to keep
3. git reset --hard commitId
5. # Move the branch pointer back to the previous HEAD
6. git reset --soft HEAD@{1}
8. git commit -m "Revert to commitId"

==============================================================================================================

To revert particular commit id which is in between 🡪 use **git revert commitid**

Git renaming:

**1. Rename your local branch.**  
If you are on the branch you want to rename:

|  |  |
| --- | --- |
| 1 | git branch -m new-name |

If you are on a different branch:

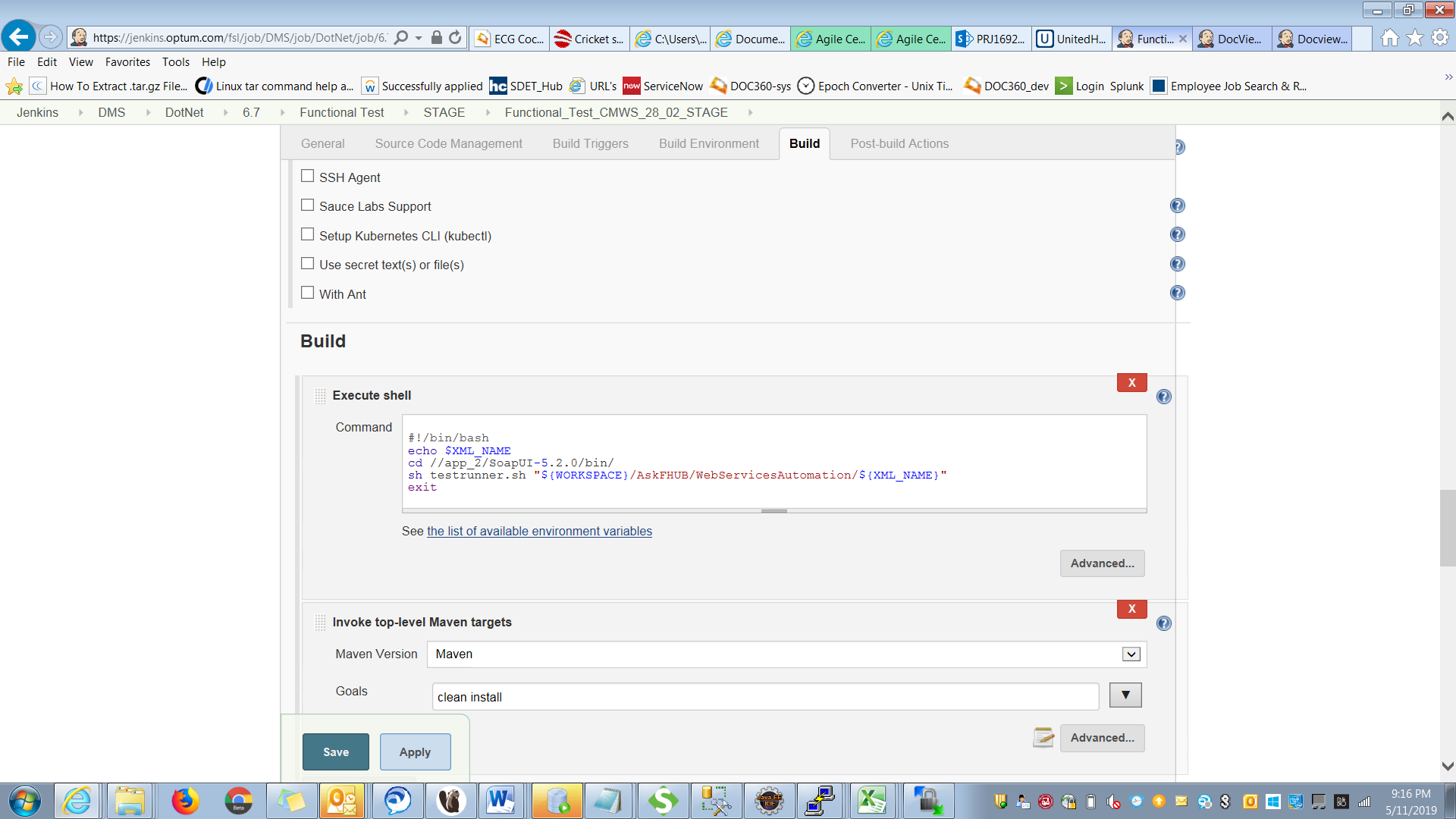
|  |  |
| --- | --- |
| 1 | git branch -m old-name new-name |

**2. Delete the old-name remote branch and push the new-name local branch.**

|  |  |
| --- | --- |
| 1 | git push origin :old-name new-name |

**3. Reset the upstream branch for the new-name local branch.**  
Switch to the branch and then:

|  |  |
| --- | --- |
| 1 | git push origin -u new-name |



-Dsonar.coverage.exclusions=\*.js