How to get started with GITHUB?

1. Create account on **github.com**
2. Install git in your local machine
   Download git from **https://git-scm.com/downloads**
3. Go to your command prompt and type -> **git version/git --version**
4. Create a public repository on **github.com**
5. In your local machine make one folder & copy your project folder inside that
6. Inside the same folder create one "README.md" file
7. Go to your project directory & open cmd
8. Type your first command as -> **git init** (this would initialize your github repository in your local machine)
9. Once your git repository is initialized, save your github email & password using ->
   a. **git config user.name "your username"**
   b. **git config user.email "your email"**
10. Now for uploading your folder on github follow these steps:
    a. In the same cmd, add your all project files using this command -> **git add .**
    b. Once the files are added into your git repo, save these changes using -> **git commit -m "files added"**
    c. For checking the git status use -> **git status**
    d. Git status command is basically used for checking whether you are in staging area or in unstage (means whether you have added your files or not/ committed your files or not)
    e. Now, for adding your github repo link, use -> **git remote add origin "github repo link"**
    f. Now, once all files are added & committed, upload these files in your public repository using -> **git push origin master**
    g. Then, check your github repository.
11. Now, once you have added your project on github & you did some changes in your local machine, then to reflect those changes in github, use:
    a. **git status** (this would let you know how many files are changed or added)
    b. git add .
    c. **git status** (this would let you know that you have added the files but not committed)
    d. git commit -m "changes in the file are uploaded"
    e. **git status** (this would let you know that nothing to commit, everything is done)
    f. git push origin master
12. Now, in team collaboration if your teammates are changing some files or adding some files then to reflect those changes in your local machine use ->
    a. **git pull origin master** (this line would update your local machine code same as github repository)
    b. In team collaboration, this command is very much important for avoiding merge conflicts.

    c. **So, whenever working in the team, try to first pull everything from github & then start adding your files.**

13. Now, in case you want to copy the github repository, use ->
    a. **git clone "repository link"** (this would download the zip file in your local machine)
14. Now, in case you are working in a team & you are dividing your work, make your own branch on github & upload your code there & once you are done then merge your branch with the master branch
    a. For making your branch create branch using -> **git branch "branch-name"**
    b. For checking different branches & you are currently on which branch use -> **git branch**
    c. Say, you are on master branch & want to switch to another branch then use -> **git checkout "branch-name"**
    d. Now, add your files in your branch & then commit & push.
    e. Now, say you are done with your work & wants to merge it with master branch then use ->
        i. go to your master branch -> **git checkout master**
        ii. merge it other branch -> **git merge "branch-name"**
        iii. push the changes remotely -> **git push origin master**
15. Once branches are merged into master branch then delete the branches using ->
    a. **git branch -d "branch-name"** (this would locally delete the branch)
    b. **git push origin --delete "branch-name"** (this would remotely delete the branch on github)
16. Why do merge conflicts happen?
    a. take one file test.txt with the content "abc" and create one branch as t1
    b. On master branch, push this file
    c. Then edit the file as
        "abc"
        "abcdef"
    d. After editing commit it on branch t1 & don't push
    e. Then again edit the file as (here "abcdef would not be reflected because you have not push & it is committed on branch t1")
        "abc"
        "def"
    f. After editing commit it on master branch & don't push
    g. Now, when someone tries to merge these files, they would be getting a merge conflict, how to resolve it?
        i. use git status for checking in which file the conflict has occurred
        ii. then check the content of that file using command -> type filename
        iii. You would see something like this:

```
D:\merge-conflicts-test>type test.txt
"abc"
<<<<<<< HEAD
"abcdef"
=======
"abcdef"
"def"
>>>>>>> master
```

iv. Now what is this added in the file?

“==” indicates the center of conflict

Content between <<<HEAD and “==” is the one exists in the current branch

Content between “==” and >>>MASTER is the one exists in the merging branch

v. Now, to resolve this conflict, open the file & remove “==”, “<<<HEAD”,”>>>MASTER” (don’t remove content)

vi. Once edited, add the file again & then commit you would be able to commit (the conflict would be resolved)

Reference:

1. https://www.atlassian.com/git/tutorials/using-branches/merge-conflicts#:~:text=Git%20commands%20that%20can%20help%20resolve%20merge%20conflicts&text=The%20status%20command%20is%20in,will%20help%20identify%20conflicted%20files.&text=Passing%20the%20%2D%2Dmerge%20argument,conflict%20between%20the%20merging%20branches.
2. https://thenewstack.io/dont-mess-with-the-master-working-with-branches-in-git-and-github/
3. https://thenewstack.io/tutorial-git-for-absolutely-everyone/

Example for branches:

```
D:\sample>git branch t2

D:\sample>git branch
* master
  t2

D:\sample>git checkout t2
Switched to branch 't2'
M       t1.txt

D:\sample>git status
On branch t2
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   t1.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\sample>git add t1.txt

D:\sample>git commit -m "t1 updated"
[t2 675b1b5] t1 updated
 1 file changed, 1 insertion(+), 1 deletion(-)

D:\sample>git branch
  master
* t2

D:\sample>git push origin t2
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 310 bytes | 31.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 't2' on GitHub by visiting:
remote:      https://github.com/jayatanwani/sample/pull/new/t2
remote:
```

```
To https://github.com/jayatanwani/sample.git
 * [new branch]      t2 -> t2

D:\sample>git checkout master
Switched to branch 'master'

D:\sample>git merge t2
Updating dd03a53..675b1b5
Fast-forward
 t1.txt | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)

D:\sample>git push origin master
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/jayatanwani/sample.git
   dd03a53..675b1b5  master -> master

D:\sample>git branch
* master
  t2

D:\sample>git branch -d t2
Deleted branch t2 (was 675b1b5).

D:\sample>git push origin --delete t2
To https://github.com/jayatanwani/sample.git
 - [deleted]
```

Example for merge conflicts & how to resolve?

Example 1:

```
D:\merge-conflicts-test>git init
Initialized empty Git repository in D:/merge-conflicts-test/.git/

D:\merge-conflicts-test>git add .

D:\merge-conflicts-test>git commit -m "added"
[master (root-commit) 0dcd086] added
 1 file changed, 1 insertion(+)
 create mode 100644 test.txt

D:\merge-conflicts-test>git branch t1

D:\merge-conflicts-test>git branch
* master
  t1

D:\merge-conflicts-test>git remote add origin https://github.com/jayatanwani/merge-test.git

D:\merge-conflicts-test>git push origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 215 bytes | 107.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/jayatanwani/merge-test.git
 * [new branch]      master -> master

D:\merge-conflicts-test>type test.txt
"abc"

D:\merge-conflicts-test>git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\merge-conflicts-test>git checkout t1
```

```
Switched to branch 't1'
M       test.txt

D:\merge-conflicts-test>git add test.txt

D:\merge-conflicts-test>git commit -m "t1 merge"
[t1 c452d94] t1 merge
 1 file changed, 2 insertions(+), 1 deletion(-)

D:\merge-conflicts-test>git checkout master
Switched to branch 'master'

D:\merge-conflicts-test>git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\merge-conflicts-test>git add test.txt

D:\merge-conflicts-test>git commit -m "master merge"
[master 0b6048e] master merge
 1 file changed, 3 insertions(+), 1 deletion(-)


D:\merge-conflicts-test>git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 259 bytes | 259.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/jayatanwani/merge-test.git
   0dcd086..0b6048e  master -> master

D:\merge-conflicts-test>git push origin t1
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 253 bytes | 84.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 't1' on GitHub by visiting:
```

```
remote:      https://github.com/jayatanwani/merge-test/pull/new/t1
remote:
To https://github.com/jayatanwani/merge-test.git
 * [new branch]      t1 -> t1


D:\merge-conflicts-test>git checkout t1
Switched to branch 't1'

D:\merge-conflicts-test>git merge master
Auto-merging test.txt
CONFLICT (content): Merge conflict in test.txt
Automatic merge failed; fix conflicts and then commit the result.

D:\merge-conflicts-test>git status
On branch t1
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)

        both modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\merge-conflicts-test>type test.txt
"abc"
<<<<<<< HEAD
"abcdef"
=======
"abcdef"
"def"
>>>>>>> master

D:\merge-conflicts-test>git add test.txt

D:\merge-conflicts-test>git commit -m "conflict resolved"
[t1 e621d1a] conflict resolved
```

Example 2:

```
D:\merge-conflicts-test>git branch
* master

D:\merge-conflicts-test>type test.txt
"a"
D:\merge-conflicts-test>git branch test

D:\merge-conflicts-test>git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\merge-conflicts-test>git add .

D:\merge-conflicts-test>git commit -m "a master added"
[master 2ea2dfb] a master added
 1 file changed, 1 insertion(+), 1 deletion(-)

D:\merge-conflicts-test>git checkout test
Switched to branch 'test'

D:\merge-conflicts-test>git status
On branch test
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\merge-conflicts-test>git add .

D:\merge-conflicts-test>git commit -m "b test added"
[test b43285f] b test added
 1 file changed, 2 insertions(+), 1 deletion(-)
```

```
D:\merge-conflicts-test>git checkout master
Switched to branch 'master'

D:\merge-conflicts-test>git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\merge-conflicts-test>git add .

D:\merge-conflicts-test>git commit -m "c master added"
[master 98d1269] c master added
 1 file changed, 2 insertions(+), 1 deletion(-)

D:\merge-conflicts-test>git checkout test
Switched to branch 'test'

D:\merge-conflicts-test>git status
On branch test
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\merge-conflicts-test>git add .

D:\merge-conflicts-test>git commit -m "d test added"
[test 990021c] d test added
 1 file changed, 2 insertions(+), 1 deletion(-)

D:\merge-conflicts-test>git checkout master
Switched to branch 'master'

D:\merge-conflicts-test>git merge test
Auto-merging test.txt
CONFLICT (content): Merge conflict in test.txt
```

```
Automatic merge failed; fix conflicts and then commit the result.

D:\merge-conflicts-test>git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)

        both modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")

D:\merge-conflicts-test>type test.txt
"a master"
<<<<<<< HEAD
"c master"
=======
"b test"
"d test"
>>>>>>> test

D:\merge-conflicts-test>git add .

D:\merge-conflicts-test>git commit -m "conflict resolved"
[master 4f0e0a9] conflict resolved

D:\merge-conflicts-test>git merge test
Already up to date.

D:\merge-conflicts-test>git push origin master
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 4 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (15/15), 1.13 KiB | 192.00 KiB/s, done.
Total 15 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/jayatanwani/merge-test.git
   c23a606..4f0e0a9  master -> master
```