Creating a Repository in GitHub

1. Sign Up with GitHub
   https://github.com/

   Read the guide
   https://guides.github.com/activities/hello-world/
   https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control

2. Download and Install the Desktop GitHub
   https://desktop.github.com/

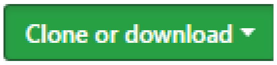
3. Create a Repository MyFirstProject on GitHub.com using + options
   Avoid spaces while creating the Repository
   Check the Initialize the repository with a README
   Choose Git Ignore –None/Windows

4. Create/upload a text file to the GitHub using UI, Commit directly to the master branch

5. Modify the above text file by adding a few lines of text and commit the changes to the master branch
   Check the changes are saved or not

Cloning an existing Repository from GitHub Desktop

1. Copy the URL from GitHub UI by clicking the clone or download button

2. Clone the below repository into your Repository Using Import your Project to GitHub from + options
   https://github.com/jalatechnologies/HelloWorld.git

3. Clone the same repository into your GitHub Desktop downloaded

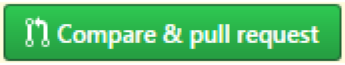
4. Create a Branch Branch-1 on GitHub Desktop and do some modification to the text file

Check the changes are reflected on the GitHub origin/master repository

5. Commit the changes to the Branch-1 on GitHub Desktop
   Check the changes are reflected on the GitHub origin/master repository
   Notice that there are NO local changes to be committed in GitHub Desktop

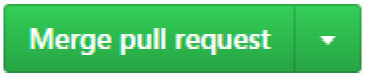
6. Push the changes done in local branch to the origin/master using Push option from Repositories menu
   Check the changes are saved to the origin/master, changes are not saved

7. Notice that the branch Branch-1 in GitHub and a button to compare and Pull Request



8. Create a Pull request by clicking the compare and Pull request
   Check the changes are not saved to origin/master

9. Need to merge the changes into origin/master using Merge Pull Request



Check all the changes done in Branch-1 are successfully saved to origin/master

10. The branch Branch-1 is no longer needed, So delete the branch Branch-1
    Go to branches and delete the branches that are merged into origin/master

Git- The Command line

1. Install Git on windows
   https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
   https://git-scm.com/downloads

2. Set your Identity
   The first thing you should do when you install Git is to set your user name and email address

```
$ git config --global user.name "JALA Technologies", need to
include quotes

$ git config –global user.email jalatechnologies.com
```

3. Configure the editor

```
$ git config --global core.editor emacs
```

4. Check your settings

```
$ git config –list
```

5. Check the specific key value

```
$ git config user.name
```

6. Getting help

```
$ git help <verb>

$ man git-<verb>

Ex: $ git help config
```

7. Initializing a repository
   Navigate to project repositories folder D:\GitRepos
   and type

```
$ git init
```

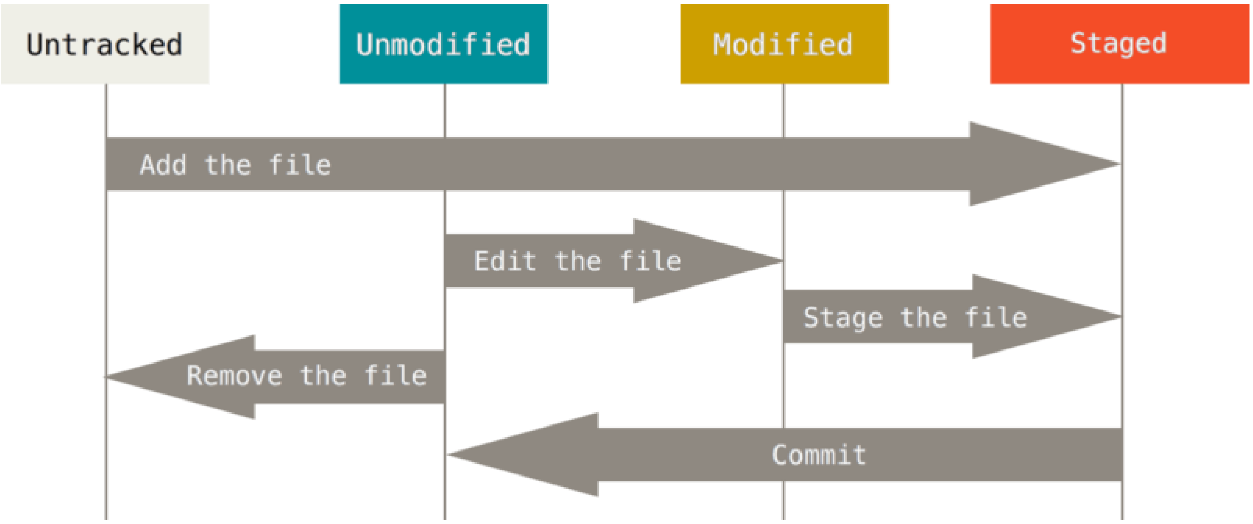
8. Cloning an existing repository
   Go to GitHub and click the below button for the Repository URL

```
$ git clone https://github.com/jalatechnologies/HelloWorld.git

This command should clone the repository on to your computer
```

9. Making changes to the cloned repository
Some basics to know
Tracked files: Files that Git knows
Untracked files: Files that Git doesn’t know

As you edit files, Git sees them as modified, because you’ve changed them since your last commit. As you work, you selectively stage these modified files and then commit all those staged changes, and the cycle repeats



10. Checking the status of your files

```
$ git status
```

11. Tracking new files
    If you add any files to the cloned repository, it is untracked by default. Run git status command to check if the file is tracked or untracked

```
$ git add FILE NAME
```

12. Staging modified files
    Again git add is used to stage. It is a multipurpose command

```
$ git add FILE NAME
```

13. Committing the changes

```
$ git commit –m "First commit from the command line"
```

14. Viewing the Commit history

```
$ git log
```

15. Showing your remotes

```
$ git remote

$ git remote -v
```

16. Fetching and Pulling from your remotes

```
$ git fetch <remote URL> or

$ git fetch origin- fetches any new work that has been pushed to that server since
you cloned (or last fetched from) it
```

17. Pushing to your remotes

```
$ git push origin master
```

18. Creating a new Branch

```
   $ git branch BRANCHNAME
```

19. Switching Branches

```
$ git checkout testing
```

20. Basic merging

```
$ git checkout master

$ git merge workingBranch-Do remember to switch to the branch the
changes should go into
```

21. Listing the branches

```
$ git branch
```