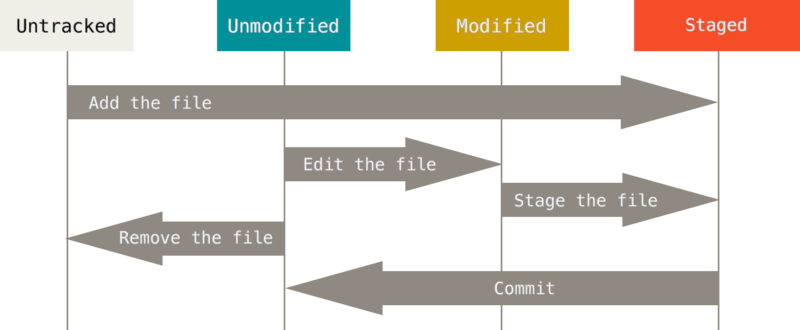
Introduction to Git:

Git is a distributed version control system, used to manage projects. Now, what does that mean? So, let’s begin with first understanding, what is Git? Let us imagine that, in a certain university, a group of 4 computer science students are given a project in C++. They must complete the project within one month and present it. So, they start working on the project, and while working on it they encounter certain problems. While working on their project they came up with certain updates to the project and decided to add those. After adding those updates they realized that their end product was a mess. So, their main issue was to get back to their stable version of the project. So, how do we solve such an issues? This is where Git comes into play. All the above problems can be solved using Git to maintain the project. We all know about the undo feature available across the various platforms and software, but it is available until a certain session is going on, once the session is over, we can’t undo. So, Git is also something similar, it provides the undo feature for a longer time, even if the session is over. So, the group’s problem to get back to the original version can be easily solved using Git. This in brief is Git.





The life-cycle of your project files

Now, moving to the next section, the three stages of Git. Every project under the distributed version control system Git, goes through three stages — Modified, Staged, and Committed.

• Modified:

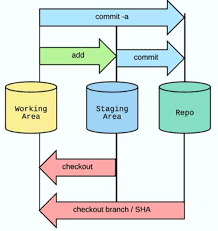
So, going back to our example, we saw that, those students have messed up their project. So, under Git we can create a repository, a repository is basically a kind of a directory that contains all the files related to your code. So, in the repository we can write code, and maintain it. Once, the code is written, anyone willing to make some modification can make those changes in their own remote repository. A remote repository is a local copy (one that you create on your local machine) of the original project that is being maintained via Git. So, basically you can make changes to your copy of the project without hampering the original code. This is called Modification, i.e. making some additions to the original project.

• Staged:

In the previous section we learnt about modification, now let’s look at the second stage in Git, i.e. staged. So, we saw that we can make changes to the project without hampering the original version, but how do we apply those changes to our remote repository? So, we use the commands in the Git command line — git add. So, this command tracks the new changes and pushes it to the staging area. So, staging area is place prior to the actual implementation of changes, i.e. this area contains all the added files that contain new code, which are ready to be joined to the remote repository. So, all the new files are first pushed to the staging area. This can also be understood with an analogy. We all must have participated in a race or some sort of athletic event. Before, the race begins, we hear three words, Get, Set…, Go! Now, we can think of ‘Set’ as the staging area. So, this is an indication that make yourself ready, as the race is about to start. Similarly, staging area is a place where all the new files are finally ready to be joined to the remote repository.

• Commit:

This is the final stage, as this stage finally applies the new changes to the remote repository. Looking at the previous analogy, this is your ‘Go’ position. So, a commit is a set of new files that are being added to a project as part of the modification. Each commit represents the changes made to project in the past, with the details about the time at which commit was made and the author of the code. So, finally when you make a commit, and it gets committed, then this simply means that you have successfully applied a certain modification to the code.



Reference: <https://medium.com/opendev-blog/the-three-stages-of-git-16565bfa67e5>