**Git Task**

1. **Difference between Git and Github**

**Git Definition**: **Git is an extremely popular version control system that is at the heart of a wide variety of high-profile projects. Git is installed and maintained on your local system (rather than in the cloud) and gives you a self-contained record of your ongoing programming versions.**

**Github Definition:GitHub is designed as a Git repository hosting service.It’s an online database that allows you to keep track of and share your Git version control projects outside of your local computer/server. Unlike Git, GitHub is exclusively cloud-based.**

****

1. **Git Workflow**

**A Git Workflow is a recipe or recommendation for how to use Git to accomplish work in a consistent and productive manner. Git workflows encourage users to leverage Git effectively and consistently. Git offers a lot of flexibility in how users manage changes. Given Git's focus on flexibility, there is no standardized process on how to interact with Git.**

**Here are the few steps:**

**1.Create a "repository" with a git hosting tool (like GitHub, Bitbucket)2. Clone the repository to your local machine.Add a file to your local repo and "commit" (save) the changes with some message**

**3. "Push" your latest changes to your master branch**

**4. Make a change to your file with a git hosting tool and commit**

**5. "Pull" the latest changes to your local machine**

**6. Create a "branch" (testing), make a change, commit the**

**change**

**7. Open a "pull request" (propose changes to the master branch)**

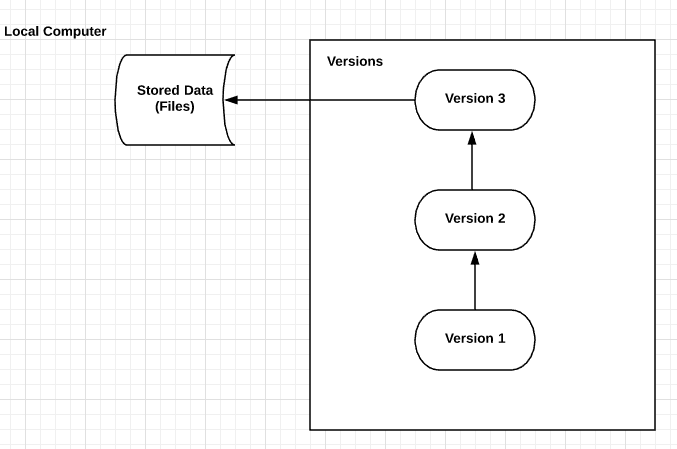
**8. "Merge" your branch to the master branch**

1. **Types of Version Control System**

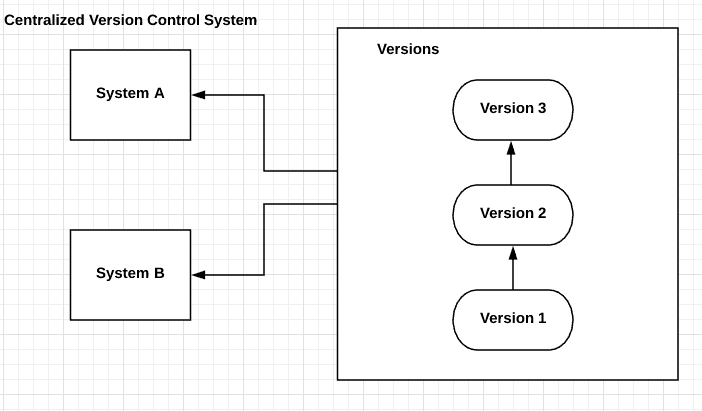
**Version Control System: Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.Version Control Systems are process management systems which maintain changes recorded in a file or set of files over period of time. Each change is maintained as a version. Users can track specific versions later.**

**There are three types of Version Control System**

**1. Local Version Control System: Local version control system maintains track of files within the local system. This approach is very common and simple. This type is also error prone which means the chances of accidentally writing to the wrong file is higher.**

****

**2.Centralized Version Control System : In this approach, all the changes in the files are tracked under the centralized server. The centralized server includes all the information of versioned files, and list of clients that check out files from that central place.**

****

**3.Distributed Version Control System:Distributed version control systems come into picture to overcome the drawback of centralized version control systems. The clients completely clone the repository including its full history. If any server dies, any of the client repositories can be copied on to the server which help restore the server.**

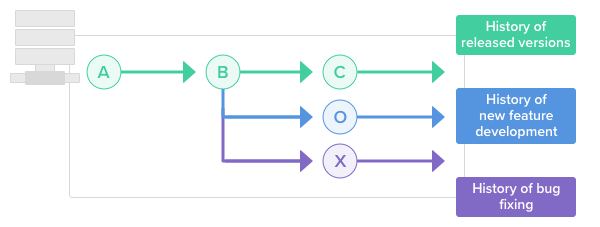
1. **Branching concept in Git**

**A Git branch is essentially an independent line of development. You can take advantage of branching when working on new features or bug fixes because it isolates your work from that of other team members.Different branches can be merged into any one branch as long as they belong to the same repository.**

**Syntax:**

**Git branch**

**Creating a new branch does not change the repository; it simply points out the commit**

****

1. **Forking Workflow in Git**

**The Forking Workflow is fundamentally different from other popular Git workflows. Instead of using a single server-side repository to act as the “central” codebase, it gives every developer their own server-side repository. This means that each contributor has not one, but two Git repositories: a private local one and a public server-side one. The Forking Workflow is most often seen in public open source projects.The main advantage of the Forking Workflow is that contributions can be integrated without the need for everybody to push to a single central repository.**

**The following is a step-by-step example of this workflow:**

1. **A developer 'forks' an 'official' server-side repository. This creates their own server-side copy.**
2. **The new server-side copy is cloned to their local system.**
3. **A Git remote path for the 'official' repository is added to the local clone.**
4. **A new local feature branch is created.**
5. **The developer makes changes on the new branch.**
6. **New commits are created for the changes.**
7. **The branch gets pushed to the developer's own server-side copy.**
8. **The developer opens a pull request from the new branch to the 'official' repository.**
9. **The pull request gets approved for merge and is merged into the original server-side repository**

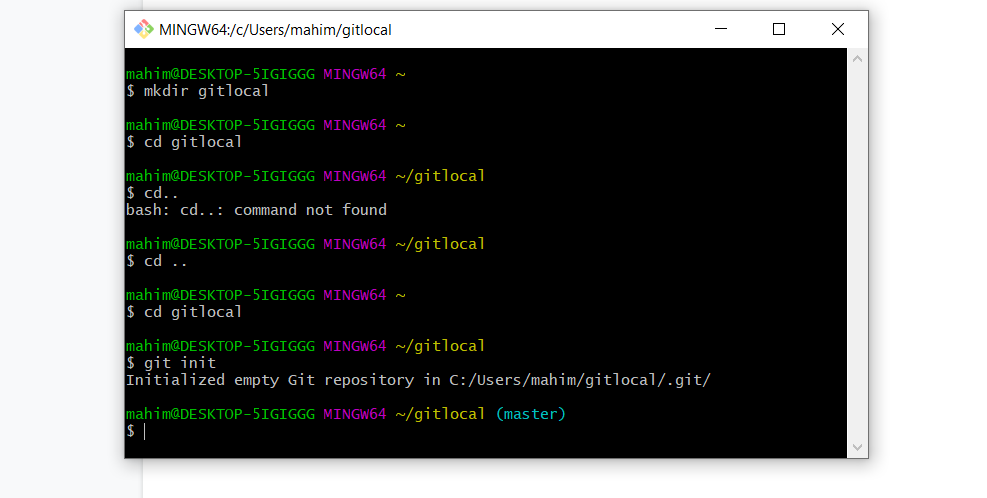
**Practical Task**

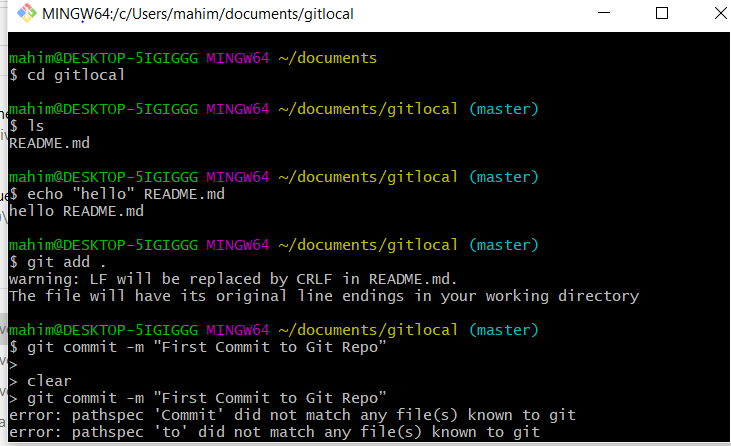
**1.Initialize an empty Git repository on your local machine with the name**

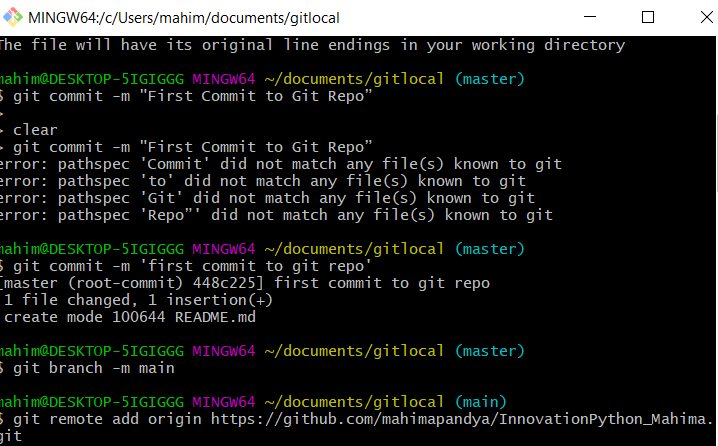
**“gitlocal” and make a README.md file in that directory which should contain**

**your name as a heading and a hello**

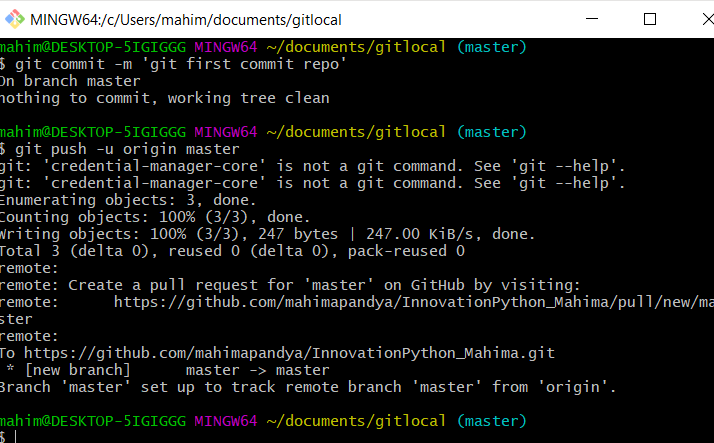
**message(https://www.makeareadme.com)**

****

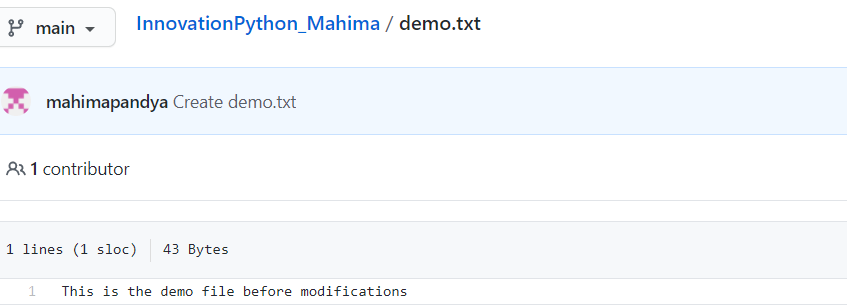




**2. Now check the status of your git directory and push all the files in that directory to your GitHub repo which you have made in the first step. With a massage “First Commit to Git Repo”**



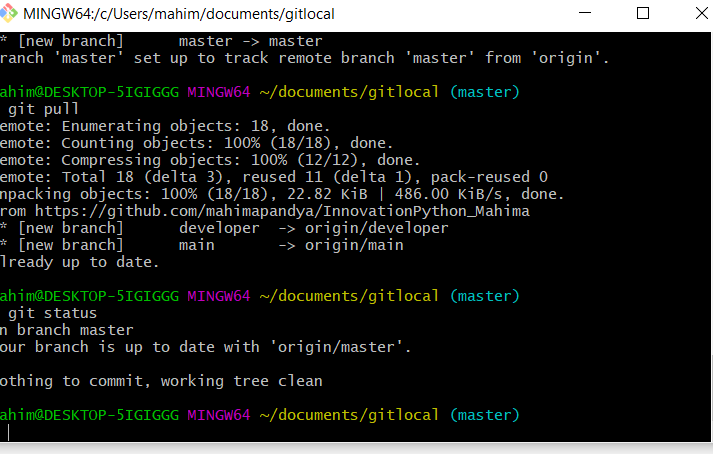
**3.Now add a file to your Github repo named “demo.txt” from the github console with content : “This is the demo file before modifications”**

****

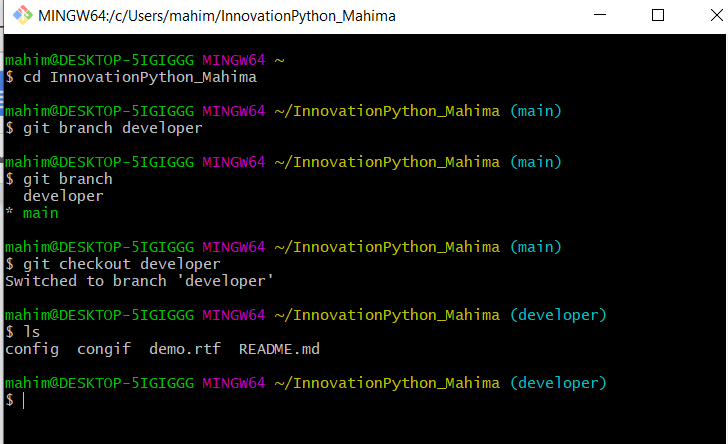
**4.** **Pull the changes in your git repo to your local machine git directory named**

**“gitlocal” and check the status for the modifications done in that repo. This time**

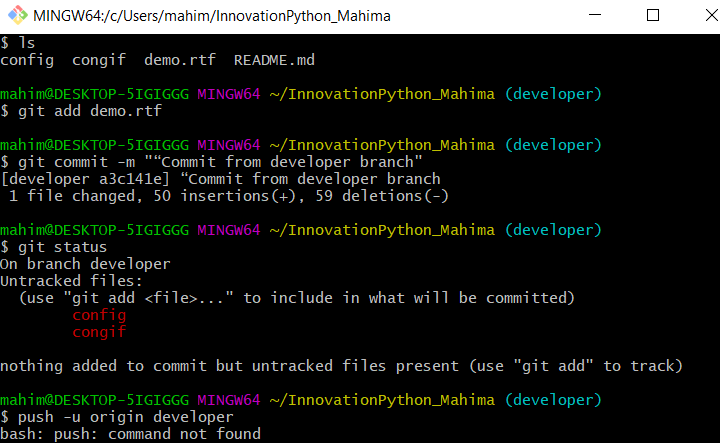
**the demo file should be visible in your local machine.**

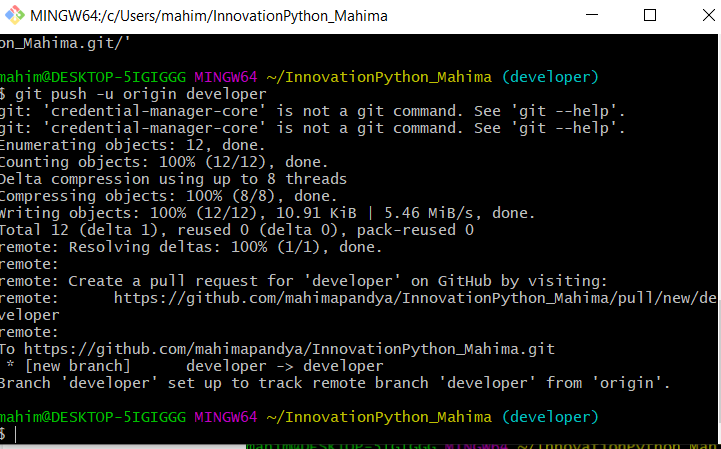


**5.● Now make a new branch in your local machine with the name “developer”.**

****

**5. Edit that demo file and write some content in that eg: “This is a demo file after modification” and push the modifications to your GitHub repo from the developer branch with a commit message “Commit from developer branch”.**

****

****