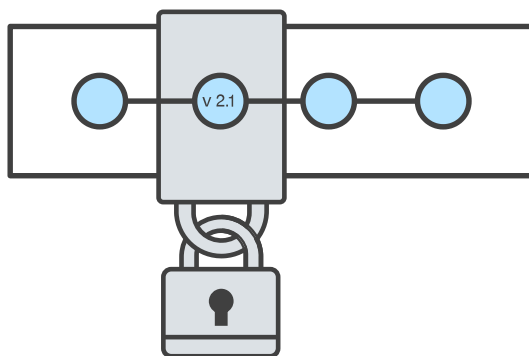


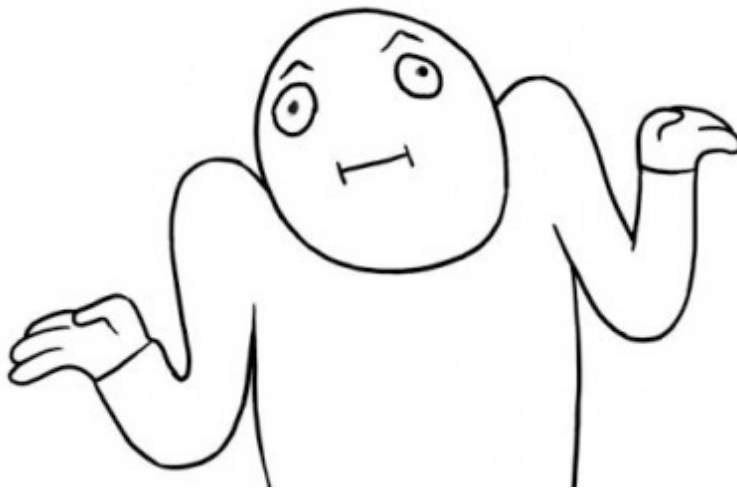
Agenda



1. What is version control?

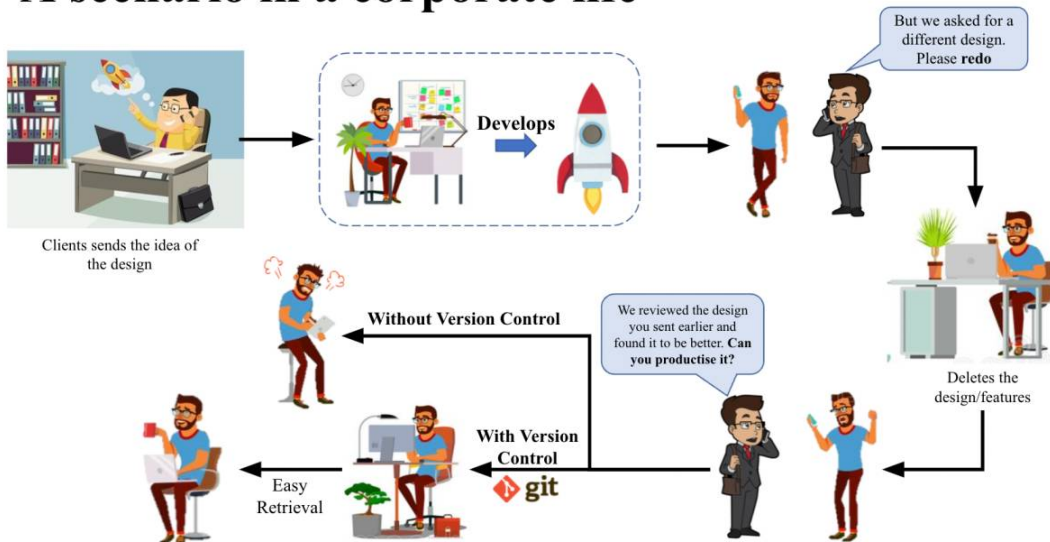


- A size of developer written code can be from 1kb to 100s and 1000s of files, millions of lines of code.
- As time passes it becomes super difficult to track who is making change & where?
- Even a single character mistake can break the entire software.
- Developers come and go, code stays forever in the company.
- I don't know or don't remember when a change was done in this file / folder is not something that a software team / company can afford.



- Hence to track changes in the source code of program from the beginning to years, we need some powerful tool that show us the history of the entire code.

A scenario in a corporate life



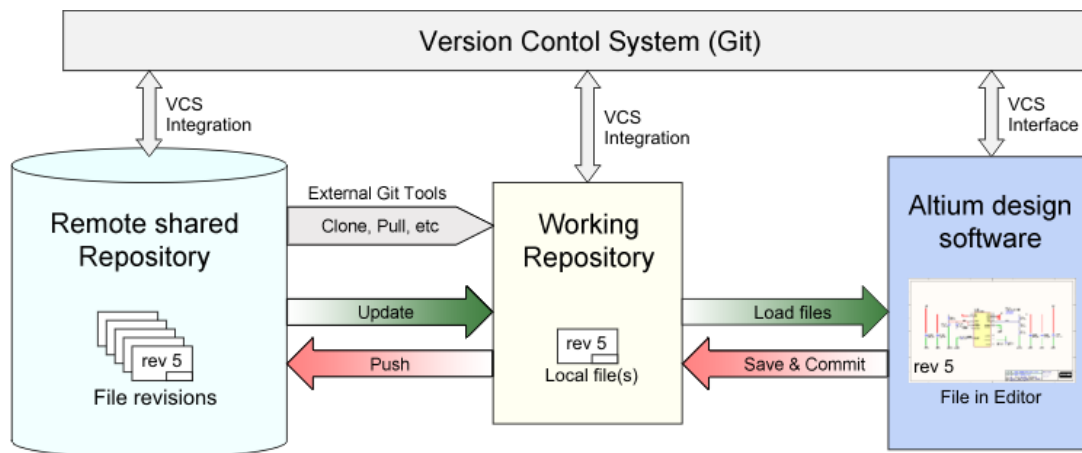
Here comes the hero of our today's session -

Version Control System

- Version control systems are software tools that help software teams manage changes to source code over time.
- As development environments have accelerated, version control systems help software teams work faster and smarter.

Benefits of version control systems

Using version control software is a best practice for high performing developers. Version control also helps developers move faster and allows teams preserve efficiency and agility as the team scales to include more developers.



Version Control Systems (VCS) have seen great improvements over the past few decades and some are better than others. VCS are sometimes known as SCM (Source Code Management) tools or RCS (Revision Control System). One of the most popular VCS tools in use today is called Git. Git is a *Distributed* VCS, a category known as DVCS, more on that later. Like many of the most popular VCS systems available today, Git is free and open source. Regardless of what they are called, or which system is used, the primary benefits you should expect from version control are as follows.

1. **Complete the long-term change history of every file.** This means every change made by many individuals over the years. Changes include the creation and deletion of files as well as edits to their contents. Different VCS tools differ in how well they handle the renaming and moving of files. This history should also include the author, date and written notes on the purpose of each change. Having the complete history enables going back to previous versions to help in root cause analysis for bugs and it is crucial when needing to fix problems in older versions of software. If the software is being actively worked on, almost everything can be considered an "older version" of the software.
2. **Branching and merging.** Having team members work concurrently is a no-brainer, but even individuals working on their own can benefit from the ability to work on independent streams of change. Creating a "branch" in VCS tools keeps multiple streams of work independent from each other while also providing the facility to merge that work back together, enabling developers to verify that the changes on each branch do not conflict. Many software teams adopt a practice of branching for each feature or perhaps branching for each release, or both. There are many different workflows that teams can choose from when they decide how to make use of branching and merging facilities in VCS.
3. **Traceability.** Being able to trace each change made to the software and connect it to project management and bug tracking software such as Jira, and being able to annotate each change with a message describing the purpose and intent of the change can help not only with the root cause analysis and other forensics. Having the annotated history of the code at your fingertips when you are reading the code, trying to understand what it is doing and why it is so designed can enable developers to make correct and harmonious changes that are in accord with the intended long-term design of the system. This can be especially important for working effectively with legacy code and is crucial in enabling developers to estimate future work with any accuracy.

While it is possible to develop software without using any version control, doing so subjects the project to a huge risk that no professional team would be advised to accept. So the question is not whether to use version control but which version control system to use.

There are many choices, but here we are going to focus on just one, Git.

2. Git and Github



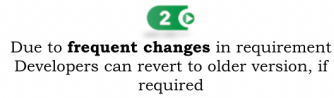
Open Source Distributed Version Control System

- **Control System:** This signifies Git to be a **control** tracker, i.e. a place to **store content**.
- **Version Control System:** Helps in maintaining history of what changes have happened by referring to them as **versions**.
- **Distributed Version Control System:** This enables the all the versions to be stored in server and local drive of a developer
- **Open Source:** Freely available and may be redistributed and modified.



Why is Git needed

While working on end-to-end operationalizing of ML/AI product any one of the below scenarios might arise and can be handled by Git



- Git is a open source Version Control System - [Developed in 2005 by Linus Torvalds \(Creator of Linux OS\)](#)
- Giving a simple example - Bhim UPI is a technology & PhonePe, GooglePay, BharatPe are build on top of it. Same way Git is open source technology and [Bitbucket \(An Atlassian product\)](#), [Github \(A Microsoft company\)](#), [Gitlab](#), [Cloud Source Repositories \(A Google product\)](#) etc are build on top of Git.
- They store our code on their cloud (+ use Git internally) and show us the full history of our code. In return we pay them some charges.

Today, most famous cloud repository is Github, You find all most all open source projects there only. like

[React](#), [Node JS](#), [MySQL](#), [VS Code IDE](#), [Redux \(A state management library\)](#), [Linux](#), [Angular](#), [Vue JS](#) etc.

3. Git Installation & Configuration

- Visit following page to install git on windows / mac / linux machine
- <https://git-scm.com/downloads>

Git Installation Guide (Windows)

- Use same email id in local git configuration.

```
git config --global user.name <<your name>>
git config --global user.email <<your email id>>
```



- Verify this email & username

```
git config - -get user.email
git config - -get user.name
```



GitHub

Github Sign up Guide

- Signup on github.com using your email id.
- Create a "hello-world" repo on github & add a README.md file will work on this repo itself.

4. Git commands

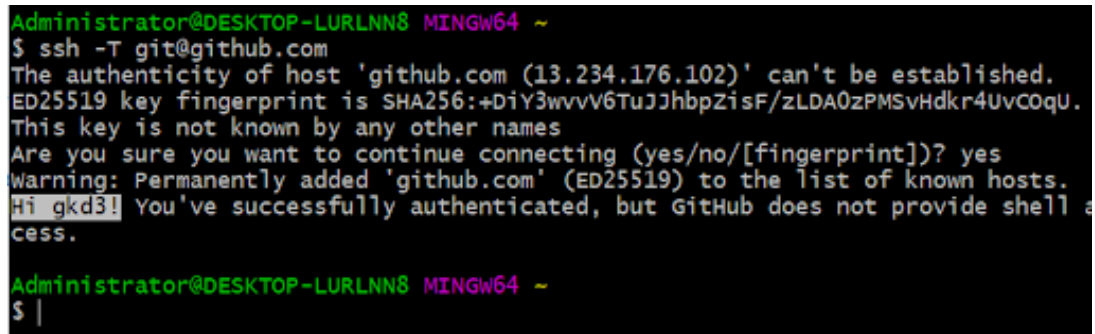
Git SSH (or Secure Shell) When we submit changes from local computer to repo in cloud (github.com), we need to provide user id and password prove our identity. This is required each time when we submit the changes or download the updated code from cloud. To get rid of from this problem

There is a solution called called SSH connection. It is a network protocol for safely encrypting any data pushed from a computer to a server over the Internet.

[Generate SSH key & save on Github](#)

Reference

- Generate SSH Key on your computer - [Github doc link](#)
- Save SSH key on Github.com - [Github doc link](#)
- Verify the SSH connection - [Github doc link](#)
- Following image shows what it looks if SSH connection is successfully established.

A terminal window with a black background and green text. The prompt is 'Administrator@DESKTOP-LURLNN8 MINGW64 ~'. The user enters '\$ ssh -T git@github.com'. The output shows a warning about the host's authenticity, a fingerprint, and a confirmation to add the host to the known hosts list. The user responds 'yes'. The output then says 'Warning: Permanently added \'github.com\' (ED25519) to the list of known hosts.' followed by 'Hi gkd3! You've successfully authenticated, but GitHub does not provide shell access.' The prompt returns to '\$ |'.

Usage

- There are 3 places from where we can run the Git commands
 - Git bash
 - Command line tool (CMD in window, terminal in Linux / Mac OS)
 - Terminal within the VS Code.
- It doesn't matter from where you run the git commands, Output of that command (text we see after command's execution) and effect of the command on cloud([github.com](#)) will be the same across these 3 places.

Demo

Download "hello-world" repository from github (created previously)

```
git clone https://github.com/gkd3/hello-world.git
cd hello-world
```



- Create index.html file with content & push that to git repository in github.

```
git add index.html
```



- Save the changes in local git with proper message. (A message should indicate what changes we have done)

```
git commit -m "Added index.html file"
```



- Now check the current status of the repository

```
git status
```



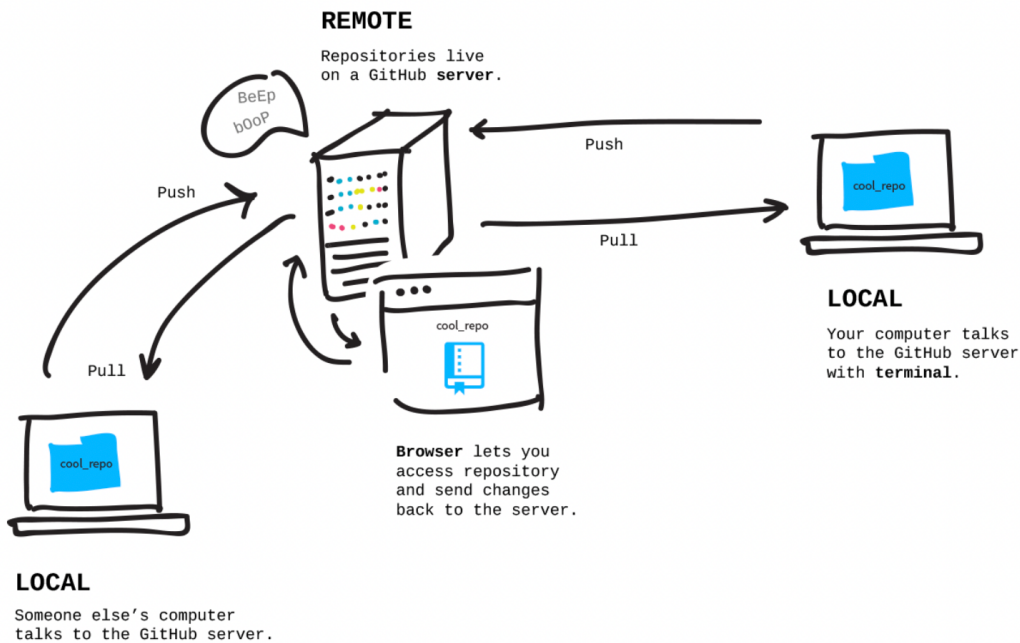
- Now submit the saved change to cloud (github.com)

git push



- Demo of this can be found [here](#)

[Add new file on local and push on Github](#)



Describes how pull & push work

How Git works in organizations

Mostly when you make the change, it should be reviewed by someone. Your changes can become source of truth for all developers once it is verified by someone. To that we need to follow these steps

- Create a separate branch (git checkout -b <>)
- Make changes in that branch (git add & git commit)
- Save it on cloud (git push)
- Raise PR request (This will be done on github.com)
- Get it approved (This will be done by reviewer of your code)
- Merge your branch to the main/master branch. (This will be done by mostly you)
- On your computer - navigate to main branch and download the updated code (git checkout main && git pull)

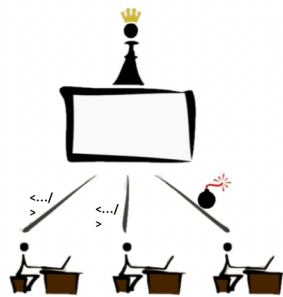
[Create branch & PR Merge Pull](#)

Github Desktop

[Github Desktop](#)

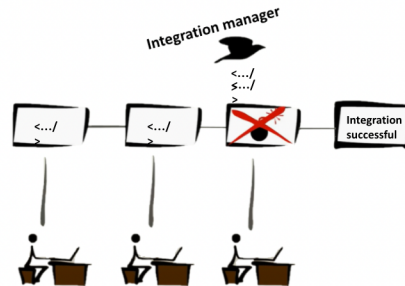
- One of the best & fast GUI tool to view code differences & perform git operation on your local computer
- As you are learning the git, It is recommended that you perform all operations via git commands only
- Use Github Desktop only to view code changes before pushing it to the cloud

Why Git is “Distributed”? Why not centralized?



Centralized

- One wrong **commit** can lead to entire collapse of the builds.
- Merge is **painful** 📄💥



Distributed

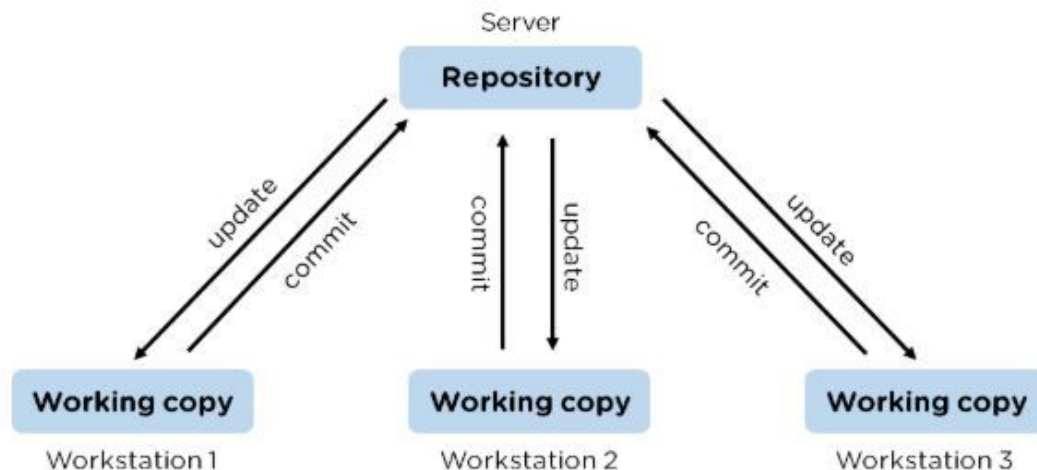
- Integration manager **ONLY** picks up the successful commits so that the integration is successful
- Merging is **breeze** 📄👌

Now with all the excitement that Git can change way of how codes are operationalized, let us get started to install and use Git.
But before that let us choose VS Code as the IDE for a hands-on experience of Git through an IDE

Interview Questions

What do you understand by the term 'Version Control System'?

A version control system (VCS) records all the changes made to a file or set of data, so a specific version may be called later if needed. This helps ensure that all team members are working on the latest version of the file



What is Git?

Git is a version control system for tracking changes in computer files and is used to help coordinate work among several people on a project who tracking progress over time. In other words, it's a tool that facilitates source code management in software development.

Git favors both programmers and non-technical users by keeping track of their project files. It enables multiple users to work together and handles large projects efficiently.

Name a few Git commands with their function.

- Git config - Configure the username and email address
- Git add - Add one or more files to the staging area
- Git diff - View the changes made to the file
- Git init - Initialize an empty Git repository
- Git commit - Commit changes to head but not to the remote repository.

Thank You