**Day-18,17-Apr-2025,Thursday**

**Topic:** **Version control using GIT**

* **Git-Global information tracker**
* **Git is worked on client side**
* **Github is worked on Server side**
* **Github build a solution on the top of the git**

## VCS-version Control System

**The version control system has 2 responsibilities:**

1. **Sharing of code**
2. **Versioning of code**

**The version control system has divided into 3 types**

1. **Centralized Version Control System Centralised**
2. **Source Version Control System**
3. **GIT(Distributed}**

**Distributed :It has multiple clones.if one goes down we can acess another**

**Centralised:we can use only one clone,if one failed entire failed**

**Fork-already prepared-one in use**

**Fork Make your own copy of someone else’s repo.**

**Bit-bucket /git-lab/github –all are same**

**Here's a list of general Git Bash command prompt commands — not just Git-specific ones, but also ones you'd use to navigate and manage files/folders in Git Bash (which is like a mini Linux terminal for Windows).**

**📁 Directory & File Navigation**

| **Command** | **Description** |
| --- | --- |
| **pwd** | **Print the current working directory** |
| **ls** | **List files and directories** |
| **ls -la** | **List all files (including hidden ones) with details** |
| **cd foldername/** | **Change directory** |
| **cd ..** | **Go back one directory** |
| **clear** | **Clear the terminal screen** |
| **exit** | **Close Git Bash** |

**📂 File & Folder Management**

| **Command** | **Description** |
| --- | --- |
| **mkdir foldername** | **Create a new folder** |
| **touch filename.txt** | **Create a new empty file** |
| **rm filename.txt** | **Delete a file** |
| **rm -r foldername/** | **Delete a folder and its contents** |
| **mv oldname.txt newname.txt** | **Rename or move a file** |
| **cp file1.txt file2.txt** | **Copy a file** |
| **cp -r folder1 folder2** | **Copy a folder recursively** |

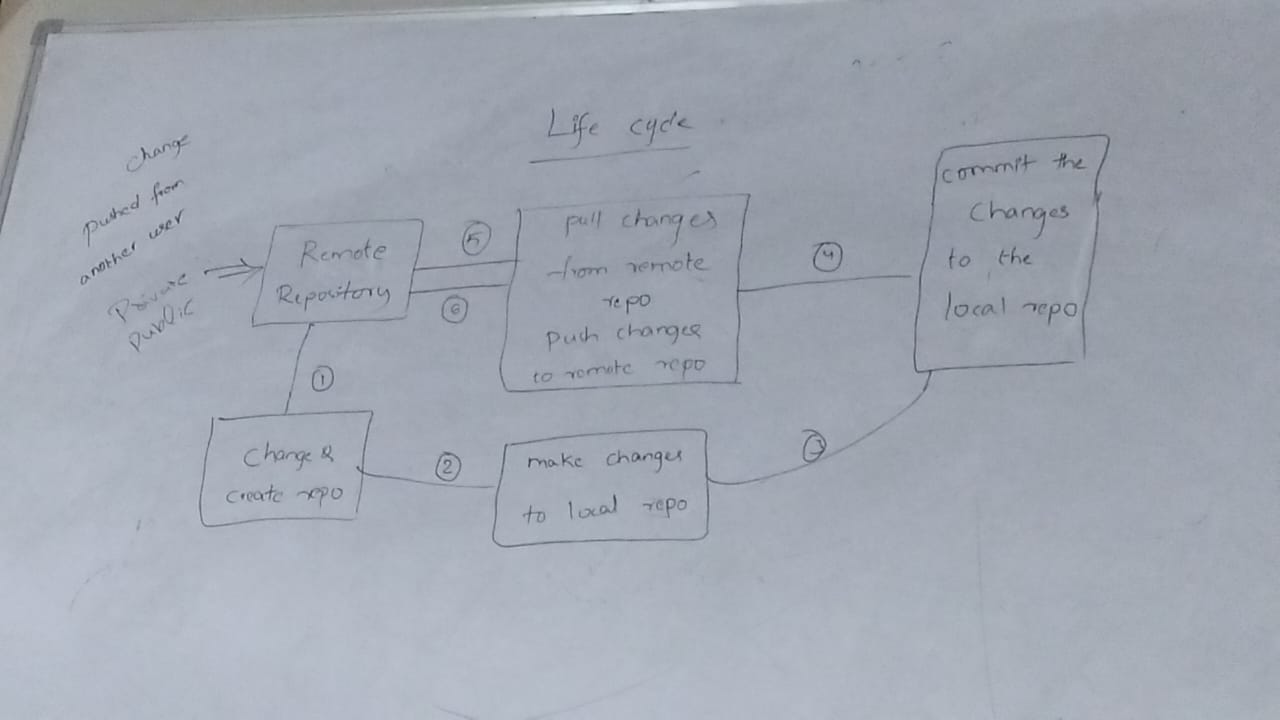
**📄 Viewing & Editing Files**

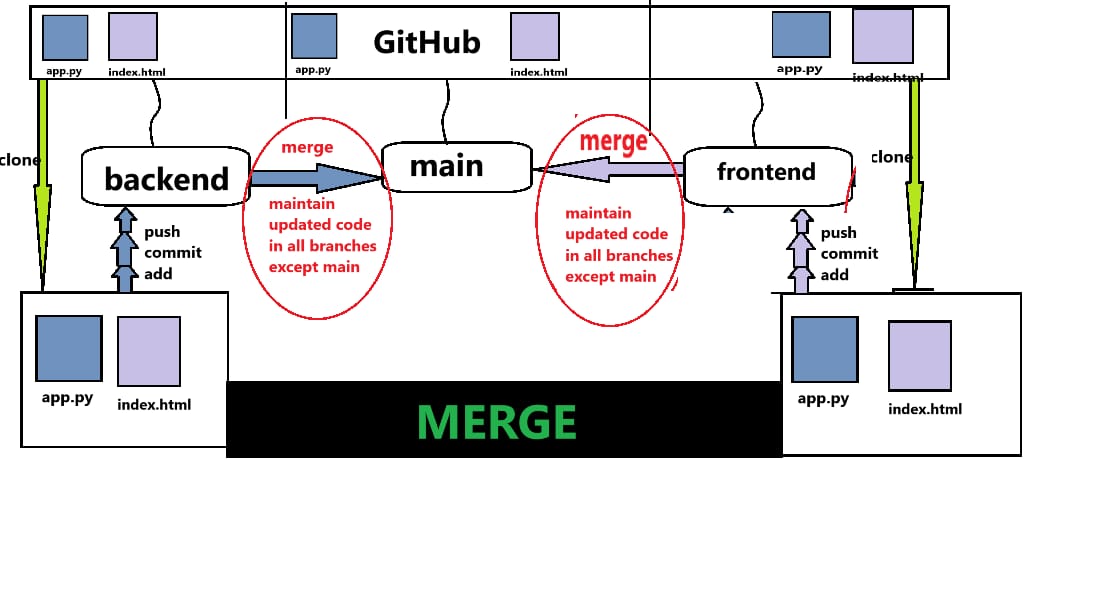
| **Command** | **Description** |
| --- | --- |
| **cat filename.txt** | **View file content** |
| **nano filename.txt** | **Open file in a terminal editor (if installed)** |
| **echo "text" > file.txt** | **Write text to file (overwrite)** |
| **echo "text" >> file.txt** | **Append text to file** |

**🔐 Permissions (Advanced, optional on Windows)**

| **Command** | **Description** |
| --- | --- |
| **chmod +x script.sh** | **Make a script executable** |
| **chmod 755 file** | **Change file permissions** |

* **Hook-we can prevent some uneven events**
* **Config-is used to config the git credentials like,tls and certifications**
* **GBS-git branching strategy(for any organization the primary goal is to ensure that the customer get releases on time.therefore,get delivery on time for that we are maintaining the branching strategy)**
* **The biggest project in github: Kubernetes or k8 project**
* **Branch-it is separation**
* **Master Branch-the branch that allows update and upto date is called as Master branch**
* **Fututre Branch-nothing but whenever people want to implement new changes to your currenct function is called future branch.**





**Fault tolerance—adding new feature to the existing feature**

**Ex: y1 y2**

**3L 7L**

**3L+4L**

**In this example in 1st year it is 3LPA,in 2nd year it is 7LPA,here 4LPA is added to 3LPA.So it become 7LPA.**

**Here the initial is 3 LPA is not effected ,just 4LPA is added .so it is called fault tolerance.**

**Release branch:**

**The code need to be checked,so here the checked is placed or available here.**