

---

# ★ FULL COMBINED SUMMARY OF BOTH VIDEOS (ALL IMPORTANT POINTS)

Git + GitHub Full Explanation (Easy for Beginners)

---

## 1. What is Git?

- Git is a **version control system**.
- It helps you **save different versions** of your code.
- Like "Undo" + "Backup" + "History" for code.

### Why we use Git?

- To track changes
  - To go back to old versions
  - To work in teams
  - To avoid losing code
- 

## 2. What is GitHub?

- GitHub is a **website** where we store Git projects online.
- Think of it like *Google Drive for code*.

Git = tool in your computer

GitHub = online storage for Git projects

---

## 3. Installing Git

- Download from git-scm.com
- Check installation:

```
git --version
```

---

## 4. Basic Git Configuration

First time only:

```
git config --global user.name "Your Name" git config --global user.email "your@email.com"
```

## 5. Git Workflow (Most Important)

The 3 Areas:

1. **Working Directory** – Your files
  2. **Staging Area** – Files ready to save
  3. **Repository** – Final saved version
- 

## 6. Important Git Commands (Simple Meaning)

✓ **Create Git repo**

```
git init
```

(Makes a project become a Git project)

✓ **Check status**

```
git status
```

(Shows what changed) ✓

**Add files to staging**

```
git add filename  
add .
```

(Prepare files for saving)

✓ **Save the changes**

```
git commit -m "message"
```

(Makes a permanent version snapshot)

---

## 7. Connecting with GitHub (VERY IMPORTANT)

**Steps:**

1. Create a repo on GitHub
2. Copy the repo link
3. Run:

```
git remote add origin <link>  
git branch -M main git push  
-u origin main
```

After first time:

```
git push
```

## 8. Cloning a Project

Download someone's project from GitHub:

```
git clone <link>
```

---

## 9. Branching (Most Confusing But Explained Simply)

**What is a branch?**

- A **separate line** of development
- Like creating a copy of the project to test safely

**Commands:**

Create branch: `git branch`

newbranch Switch to branch:

```
git checkout newbranch
```

Create + switch: `git`

```
checkout -b newbranch See
```

all branches: `git branch`

Delete branch:

```
git branch -d newbranch
```

---

## 10. Merging

Combine one branch with another:

```
git merge branchname
```

---

## 11. Merge Conflicts

Happens when:

- Two people edit the same line
- Git doesn't know which one to keep

You manually fix the file → save → add → commit.

---

## 12. Git Pull

Download updates from GitHub into your computer:

```
git pull
```

---

## 13. Forking (GitHub Feature)

- You copy someone's GitHub repo into your account
  - Useful for open-source contributions
- 

## 14. Pull Request (PR)

- When you want to add your changes to someone else's project
  - You send a "request"
  - They review → accept → your code joins project
- 

## 15. .gitignore

Used to ignore files (example: passwords, build files)

Create a `.gitignore` file and add:

```
node_modules/ .env
```

---

## 16. Git Log

To see history of commits:

```
git log Short version:
```

```
git log --oneline
```

---

## 17. Git Stash

Temporarily save your work without committing:

```
git stash
```

Get it back:

```
git stash pop
```

## 18. Best Practices You Should Follow

- Commit small changes
  - Use meaningful commit messages
  - Don't push directly to main
  - Create separate branches
  - Pull before push
  - Never upload passwords
- 

## ★ Final Summary (Super Short)

Concept	Meaning
Git	Tool for saving code versions
GitHub	Online place to store code
init	Start git
add	Prepare files
commit	Save version
push	Upload to GitHub
pull	Download from GitHub
clone	Download project
branch	Work separately
merge	Combine code
PR	Ask to merge your code
stash	Save work temporarily

---