GitHub Basics

1. What GitHub Is

- **Git** → a tool that tracks changes in files (version control)
- **GitHub** → a website that **stores Git repositories online**
- Think of GitHub as:
  - Google Docs **for code**
  - A **cloud backup + history + collaboration** tool

2. Core GitHub Terms

Repository (Repo)

- A **project folder** stored on GitHub
- Contains:
  - Code files
  - Documentation
  - History of changes
- Example: `my-first-java-project`

Commit

- A **saved snapshot** of your project
- Includes:
  - What changed
  - When it changed
  - Who changed it
- Always has a **message**
- Think: "Save with explanation"

Branch

- A **separate line of work**
- Used to:
  - Try features
  - Avoid breaking main code
- Default branch: `main` (or sometimes `master`)

Clone

- Makes a **local copy** of a GitHub repo on your computer
- One-time operation per repo

Push

- Sends your local commits **to GitHub**

Pull

- Brings changes **from GitHub to your computer**

Fork

- Your **own copy** of someone else's repository
- Common in open-source projects

Pull Request (PR)

- A request to:
  - Merge changes from one branch into another
- Used for:
  - Team review
  - Grading
  - Open-source contributions

3. GitHub Website

Create a GitHub Account

- Go to **github.com**
- Sign up (free)
- Choose a username (this becomes public)

Create a Repository (Website)

1. Click **+ → New repository**
2. Enter repository name
3. Choose:
   - Public (recommended for students)
4. Check:
   - **Add a README file**
5. Click **Create repository**

Upload Files (No Command Line)

1. Open the repository
2. Click **Add file → Upload files**
3. Drag and drop files
4. Add a commit message
5. Click **Commit changes**

* No Git commands required

Edit Files Online

- Click a file
- Click the **pencil icon**
- Edit text
- Commit changes

4. GitHub Using Command Line (Basic Workflow)

One-Time Setup

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

Clone a Repository

```
git clone https://github.com/username/repo-name.git
```

Creates a local folder.

Basic Daily Commands (Minimum Set)

```
git status         # check file changes
git add .          # stage all changes
git commit -m "message"   # save snapshot
git push           # upload to GitHub
```

Pull Latest Changes

```
git pull
```

Typical Workflow

1. Edit code
2. `git add .`
3. `git commit -m "Finished lab 3"`
4. `git push`

5. Beginner-Friendly GitHub Tools

GitHub Desktop (Highly Recommended)

- Official GUI from GitHub
- Buttons instead of commands
- Works on Windows and macOS

You can:

- Clone repos
- Commit changes
- Push / Pull
- Switch branches

IDE Integration

IntelliJ IDEA

- Built-in GitHub support
- Buttons for:
  - Commit
  - Push
  - Pull
- Visual diff view (see what changed)

VS Code

- Source Control panel
- Git actions via clicks
- Very beginner-friendly UI

6. README.md

- A **README.md** explains the project
- Written in **Markdown**
- Usually includes:
    - Project description
    - How to run the code
    - Author name

Example:

```
# My Java Project
This program checks whether a number is even.
```

7. Common Mistakes

- Forgetting to commit before pushing
- Editing directly in `main` without branches
- Writing vague commit messages like “update”
- Uploading compiled files (`.class`, `.exe`)

8. Simple Recommendations

- Commit **often**
- Write **clear messages**
- Push after every working change
- If confused → check `git status`