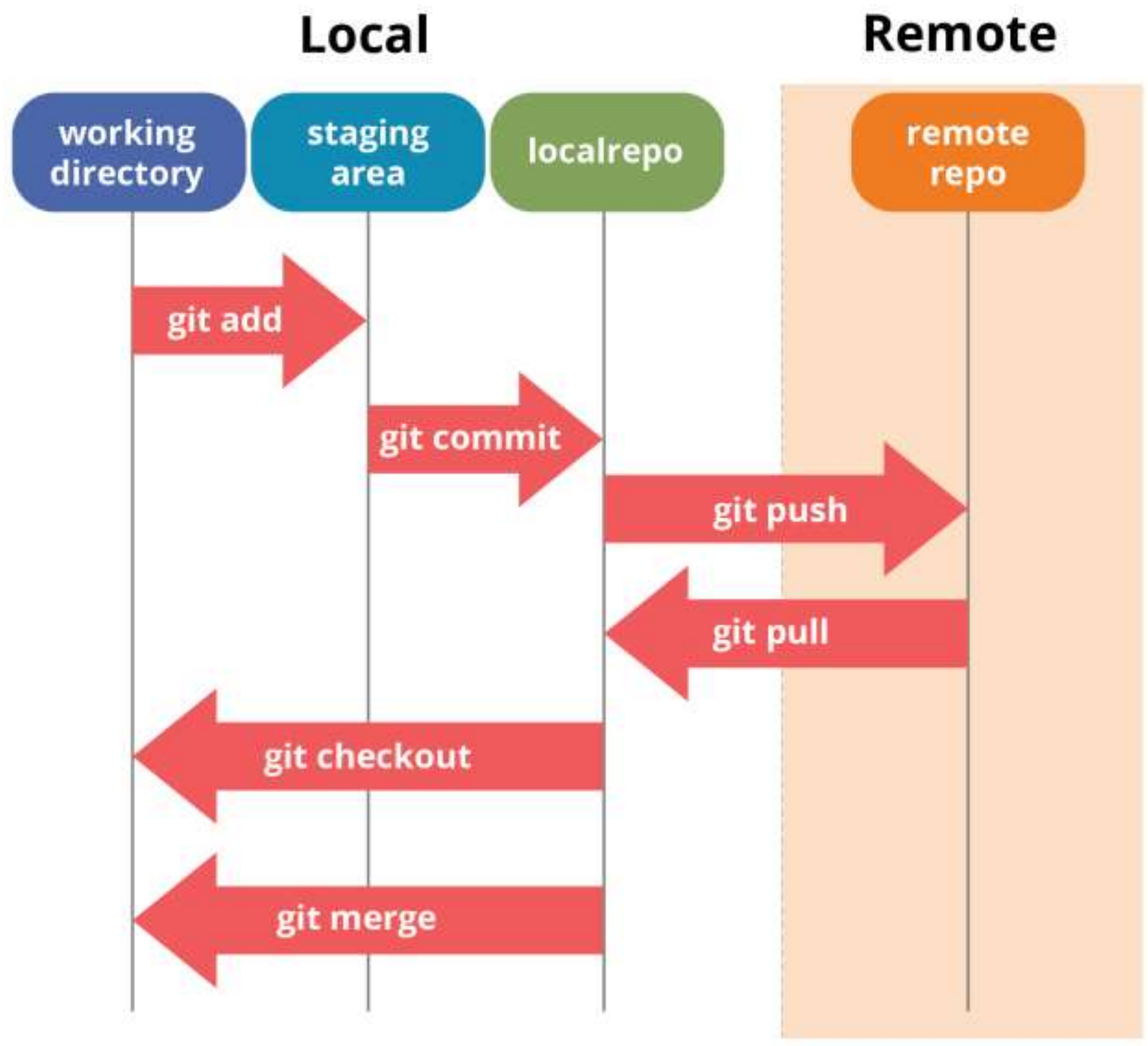
1. Introduction to Git

Linus Torvalds is the founder of Git. He created the open-source version control system (VCS) in 2005. Torvalds is also the creator of the Linux kernel operating system.

Git is a distributed version control system used to track changes in code during software development. It enables multiple developers to collaborate, manage code versions, and maintain the integrity of a project.

Key Features of Git:

- **Version Control**: Keeps a history of every change made to a project.
- **Branching and Merging**: Allows developers to work on features in isolation and merge them back into the main branch.
- **Collaboration**: Facilitates teamwork and simplifies code sharing.
- **Speed**: Operates quickly and efficiently.



---

2. Install Git on Windows

1. **Download Git**: Visit git-scm.com and download the latest version of Git for Windows.
2. **Install Git**:
    - Run the installer and choose the default options unless customization is needed.
    - During installation, configure:
        - **Default Editor**: Select your preferred text editor (e.g., VS Code, Notepad++).
        - **Adjust PATH Environment**: Select "Use Git from the command line and 3rd-party software."
3. **Verify Installation**:

- Open Command Prompt or Git Bash.
- Run: `git --version` to ensure Git is installed correctly.

---

3. GitHub

GitHub is a cloud-based platform that hosts **Git repositories**, providing tools for version control and collaboration.

Features of GitHub:

- Hosting of repositories (public and private).
- Collaboration tools (issues, pull requests, discussions).
- Integration with **CI/CD** tools.
- Secure code management.

---

4. Git Commands

Basic Commands:

- **git init:** Initialize a new Git repository.
- **git clone** <URL>: Clone an existing repository.
- **git add** <file>: Stage changes for commit.
- **git commit -m "message":** Commit staged changes with a message.
- **git status:** Show the status of the working directory.
- **git log:** View commit history.

Branching and Collaboration:

- **git branch:** List, create, or delete branches.
- **git checkout <branch>:** Switch to another branch.
- **git merge <branch>**: Merge another branch into the current branch.
- **git pull:** Fetch and merge changes from a remote repository.
- **git push:** Push changes to a remote repository.

---

5. Git vs. GitHub

Feature	Git	GitHub
Purpose	Version control system for tracking code changes.	Cloud-based hosting for Git repositories.
Installation	Installed locally on your machine.	Accessed via a web browser or API.
Features	Branching, merging, version tracking.	Collaboration, pull requests, issue tracking.

---

6. GitLab

GitLab is an open-source **DevOps** platform that integrates version control, **CI/CD** pipelines, and project management. It provides similar features to GitHub but is often preferred for its **self-hosting** capabilities.

GitLab Features:

- Integrated **CI/CD.**
- Issue tracking.
- Code reviews.
- Advanced security features.

---

7. Git Clone and Git Push Commands

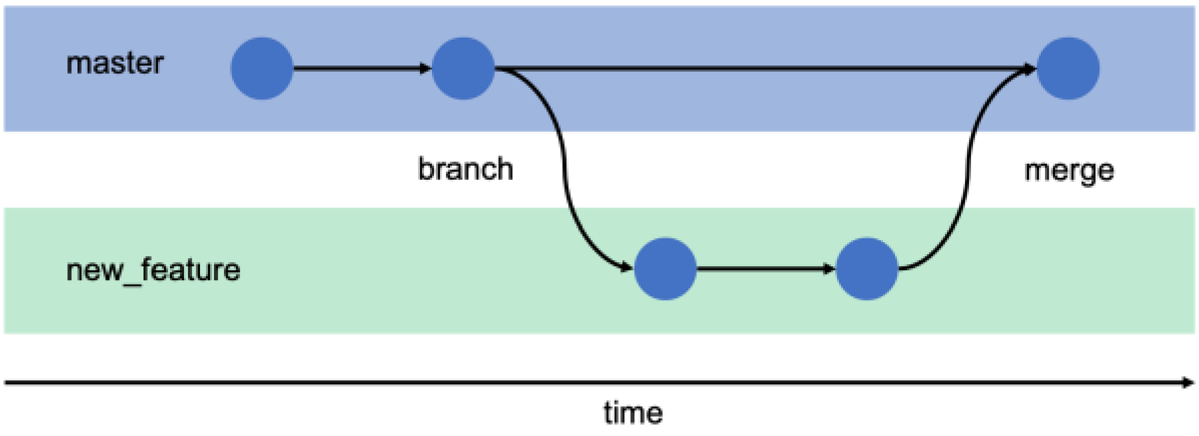
- **Git Clone**: Used to copy an existing remote repository to your local machine.
- **git clone <repository_url>**
- **Git Push**: Uploads local commits to the remote repository.
- **git push origin <branch_name>**

---

8. Git History and Git Pull Commands

- **Git History**: View the commit history of your repository.
- **git log**
- **git log --oneline** # Compact view
- **Git Pull**: Fetches changes from the remote repository and merges them into the current branch.
- **git pull origin <branch_name>**

9. Branching and Merging



- **Branching**: Create a separate branch to develop new features without affecting the main branch.
- **git branch <branch_name>** # Create a new branch
- **git checkout <branch_name>** # Switch to the branch
- **Merging**: Combine changes from one branch into another.
- **git merge <branch_name>**

10. Resolve Merge Conflicts in Git

Merge conflicts occur when changes in two branches conflict during a merge.

Steps to Resolve Conflicts:

1. Identify conflicting files (Git will highlight them).
2. Open the conflicting files and manually edit them to resolve issues.
3. Stage the resolved files:
4. **git add <file>**
5. Commit the resolution:
6. **git commit -m "Resolved merge conflict"**