**Lab Exercise 2 -Git Stash**

**Git stash** is a command in Git that temporarily saves changes you've made to your working directory but haven't yet committed. It allows you to "stash away" these changes so you can work on something else, like switching to another branch or resolving an issue, without losing your current progress.

**Key Features of git stash:**

1. **Temporarily Saves Work:** Stash your changes without committing them to the repository.
2. **Keeps a Clean Working Directory:** Returns your working directory to the last committed state, making it easy to switch branches or pull updates.
3. **Restores Changes Later:** Apply the stashed changes when you're ready to continue working on them.
4. **Supports Tracked and Untracked Files:** You can stash changes to tracked files by default, and include untracked or ignored files with specific options.

**Common Commands:**

| **Command** | **Description** |
| --- | --- |
| git stash | Stashes the changes in your working directory. |
| git stash list | Shows a list of all stashes. |
| git stash pop | Re-applies and removes the most recent stash. |

**Example Use Case:**

1. You're working on a feature, and you need to switch to another branch to fix a bug.
2. Instead of committing half-done work, you can stash your changes:

git stash

1. Switch to the other branch:

git checkout bugfix-branch

1. Once the bug is fixed, return to your original branch:

git checkout feature-branch

1. Reapply the stashed changes:

git stash pop

**When to Use git stash:**

* Switching branches without committing changes.
* Trying out quick fixes or experiments.
* Pulling updates from a remote repository that requires a clean working directory.