Assignment

Assignment of Class (Introduction to Git)

Internship Class Assignment - 2

Name:	Ali Ahasan
Position:	Intern
Tracking Number:	J2DFEEB
Team:	InnovX Team of Business Automation Limited

Task:

- Create a Github account.
- Create a public repository in Github.
- Initialize git in your project.
- Create a branch using your first name.
- Push your code in the branch.
- Merge the branch with main branch of the repository.
- Upload your First Class Assignment code in the repository.
- Write 15 git commands and briefly discuss their use cases. Upload the . pdf file in the form.

Instructions:

1. Complete all the tasks as outlined above.
2. Submit the repository link through the provided Google Form.

15 git commands and brief discussion of their use cases

1. **git init**

Use Case: Initializes a new Git repository.

Example:

```
    git init
```

This command creates a new `.git` directory in your project, allowing you to start tracking versions of your project files.

2. **git remote add <name> <url>**

Use Case: Adds a new remote repository.

Example:

```
    git remote add upstream https://github.com/original/repo.git
```

Useful for adding a remote repository to track upstream changes.

3. **git clone <url>**

Use Case: Copies an existing Git repository from a remote location to your local machine.

Example:

```
    git clone https://github.com/user/repo.git
```

This command is used to start working on an existing project by creating a local copy of the repository.

4. **git fetch <remote>**

Use Case: Downloads objects and refs from another repository.

Example:

```
    git fetch origin
```

Fetches updated data from the remote named `origin` without merging changes.

5. **git show <commit>**

Use Case: Shows various types of objects (commits, tags, etc.) in a more detailed and readable format.

Example:

```
    git show a1b2c3d4
```

Displays the content changes and metadata for a specific commit.

6. **git commit -m "message"**

Use Case: Records changes made to the repository with a descriptive message.

Example:

```
    git commit -m "Add initial project documentation"
```

Commits the staged changes and includes a brief message explaining the changes.

7. **git push <remote> <branch>**

Use Case: Sends committed changes to a remote repository.

Example:

```
    git push origin main
```

Pushes the commits from your local `main` branch to the `main` branch on the remote named `origin`.

8. **git pull <remote> <branch>**

Use Case: Fetches and integrates changes from a remote repository into your local branch.

Example:

```
    git pull origin main
```

Updates your current local working branch with all new commits from the `main` branch on `origin`.

9. **git status**

Use Case: Displays the state of the working directory and staging area.

Example:

```
    git status
```

Shows which changes are staged, which are not, and which files are not being tracked by Git.

10. **git branch <branch-name>**

Use Case: Creates a new branch.

Example:

```
    git branch feature-login
```

```

Creates a new branch named `feature-login` without switching to it.

### 11. **git checkout <branch-name>**

Use Case: Switches branches or restores working tree files.

Example:

```
    git checkout feature-login
```

Switches to the branch `feature-login`.

### 12. **git merge <branch>**

Use Case: Merges a branch into your active branch.

Example:

```
    git merge feature-login
```

Merges the `feature-login` branch into the currently active branch.

13. **git log**

Use Case: Shows the commit history for the current branch.

Example:

```
    git log
```

Displays a list of recent commits in the current branch, including author, date, and commit message.
```

14. **git reset <file>**

Use Case: Unstages a staged file, but preserves the file contents.

Example:

```
    git reset README.md
```

Removes `README.md` from the staging area, making it no longer prepared for the next commit.

15. **git revert <commit>**

Use Case: Creates a new commit that undoes changes made in a previous commit, without altering the project history.

Example:

```
    git revert a1b2c3d
```

Useful for undoing changes and keeping history.