**Day 2**: **15th May 2025**

Today's activities included installation tasks, environment setup, Java installation, and an oral presentation, all of which were completed successfully.

**Complete Git Commands Reference Guide**

**Initial Setup & Repository Creation**

**1. Create Folder & Add Files**

mkdir project-name # Create a new folder

cd project-name # Navigate into the folder

touch filename.txt # Create a new file (Linux/Mac)

# OR use your text editor to create files

**What it does:** Sets up your project directory structure  
**When to use:** At the start of any new project

**2. Set Path/Location**

cd /path/to/your/project

# Examples:

cd C:/Users/YourName/Projects/MyProject # Windows

cd ~/Documents/Projects/MyProject # Mac/Linux

**What it does:** Changes your current directory to the project location  
**When to use:** Navigate to your project folder before running Git commands

**3. Initialize Git Repository**

git init

**What it does:** Creates a new Git repository in the current directory (creates hidden .git folder)  
**When to use:** Once per project, when you want to start tracking changes with Git

**Basic Git Workflow Commands**

**4. Add Files to Staging Area**

git add filename.txt # Add specific file

git add . # Add all files in current directory

git add \*.js # Add all JavaScript files

**What it does:** Stages files for commit (prepares them to be saved)  
**When to use:** Before committing changes, to select which files to include

**5. Check Repository Status**

git status

**What it does:** Shows which files are staged, modified, or untracked  
**When to use:** Frequently, to see current state of your repository

**6. Commit Changes**

git commit -m "your commit message"

# Example:

git commit -m "Add user authentication feature"

**What it does:** Saves staged changes to repository history with a descriptive message  
**When to use:** After staging files, to create a snapshot of your changes

**User Configuration**

**7. Set Global User Email**

git config --global user.email "your.email@example.com"

**What it does:** Sets your email for all Git repositories on your computer  
**When to use:** Once during initial Git setup

**8. Set Global Username**

git config --global user.name "Your Full Name"

**What it does:** Sets your name for all Git repositories on your computer  
**When to use:** Once during initial Git setup

**9. View Configuration**

git config --global --list # View all global settings

git config user.name # View specific setting

**Repository History & Information**

**10. View Commit History**

git log # Full commit history

git log --oneline # Condensed view

git log --graph # Visual branch representation

**What it does:** Shows chronological list of commits with details  
**When to use:** To review project history or find specific commits

**11. Clone Repository**

git clone https://github.com/username/repository-name.git

# Example:

git clone https://github.com/achantsa/Atlas\_Day\_2.git

**What it does:** Downloads a complete copy of a remote repository to your computer  
**When to use:** To work on existing projects or contribute to others' projects

**Remote Repository Management**

**12. Get Remote URL**

git config --get remote.origin.url

# Alternative:

git remote -v

**What it does:** Shows the URL of your remote repository  
**When to use:** To verify which remote repository you're connected to

**13. Branch Management**

git branch -m master # Rename current branch to 'master'

git branch -m main # Rename current branch to 'main'

git branch -m old-name new-name # Rename specific branch

**What it does:** Renames branches in your repository  
**When to use:** To follow naming conventions or organize your branches

**14. Add Remote Repository**

git remote add origin https://github.com/username/repository-name.git

# Your example:

git remote add origin https://github.com/achantsa/Atlas\_Day\_2.git

**What it does:** Links your local repository to a remote repository  
**When to use:** After creating a local repo that you want to push to GitHub

**15. Push Changes to Remote**

git push origin main # Push to main branch

git push origin master # Push to master branch

git push -u origin main # Push and set upstream tracking

**What it does:** Uploads your local commits to the remote repository  
**When to use:** To share your changes with others or backup your work

**16. Pull Changes from Remote**

git pull origin main # Pull from main branch

git pull origin master # Pull from master branch

git pull # Pull from tracked branch

**What it does:** Downloads and merges changes from remote repository  
**When to use:** To get latest changes from collaborators

**Advanced Commands**

**17. Show Commit Details**

git show # Show latest commit details

git show commit-hash # Show specific commit

**What it does:** Displays detailed information about commits including changes made  
**When to use:** To examine what changes were made in specific commits

**18. View Differences**

git diff # Show unstaged changes

git diff --staged # Show staged changes

git diff commit1 commit2 # Compare two commits

**What it does:** Shows line-by-line differences between versions  
**When to use:** To review changes before committing or compare versions

**19. Reset Changes**

git reset filename.txt # Unstage specific file

git reset --soft HEAD~1 # Undo last commit, keep changes staged

git reset --hard HEAD~1 # Undo last commit, discard changes

**What it does:** Undoes changes or moves repository to previous state  
**When to use:** To fix mistakes or undo unwanted changes  
**⚠️ Warning:** --hard permanently deletes changes

**20. Revert Commits**

git revert commit-hash # Create new commit that undoes specified commit

git revert HEAD # Revert last commit

**What it does:** Creates a new commit that reverses changes from a previous commit  
**When to use:** To safely undo changes without rewriting history

**Common Workflow Example**

Here's a typical workflow combining these commands:

# 1. Set up new project

mkdir my-project

cd my-project

git init

# 2. Configure Git (do this once)

git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"

# 3. Create and add files

touch README.md

git add README.md

git commit -m "Initial commit"

# 4. Connect to remote repository

git remote add origin https://github.com/yourusername/my-project.git

git push -u origin main

# 5. Regular workflow

# Make changes to files

git add .

git status

git commit -m "Describe your changes"

git push origin main

**Additional Important Commands**

**Branch Operations**

git branch # List all branches

git branch branch-name # Create new branch

git checkout branch-name # Switch to branch

git checkout -b new-branch # Create and switch to new branch

git merge branch-name # Merge branch into current branch

**Useful Shortcuts**

git add -A # Add all changes (including deletions)

git commit -am "message" # Add and commit in one command

git push # Push to default remote/branch (after setting upstream)

**Important Notes:**

* Always run git status to understand your repository state
* Use descriptive commit messages
* The git config commands only need to be run once globally
* Pull before pushing to avoid conflicts