Lecture - 9

# Gemini Code Assist Course

- → Why version control matters
- → Initializing the git repository
- → Checking repository status
- → Creating a .gitignore file
- → First Commit
- → Viewing Commit log
- → Connecting to github
- → Authentication with github
- → Pushing code to Remote
- → Hand-on cycle
- → Takeaway - Not just code, but also automate workflows

→ Why version control matters

- → Essential for saving work in a professional, collaborative way
- → Tracks history, enables teamwork, and prevents accidental loss of progress

→ Initializing the repository

- → Use Gemini CLI to run "git init"
- → Converts your local project folder into a git repository
- → important step for tracking code changes

Please Initialize a new Git repository in this project ← Prompt

→ Checking repository status

- → Run "git status" via gemini CLI to see tracked/Untracked files
- → Reveals unnecessary files (__pycache__, .DS_Store, Expenses.db) that shouldn't be committed

Please check the status of our git repository ← Prompt

→ Creating a .gitignore file

- Add rules to exclude env. specific or temporary files
- Keeps repository clean and professional

Please create a .gitignore file for this project. Git should ignore __pycache__, .DS_Store and the database file expenses.db ← Prompt

→ First Commit

- Stage all project files
- Ask Gemini to generate a nice and professional commit message (e.g., "feat: Initial setup of flask app")
- Commit the snapshot to create the first version in history

Lets do our first git commit. Please stage all the untracked project files, analyse the staged changes and accordingly create a professional commit message summarising the initial setup of our application and CRUD changes and then commit the changes with the same message ← Prompt

→ Viewing Commit log -

- Run "git log" to confirm the initial commit is recorded
- Ensures repository is tracking changes properly

Show the log of our first commit ← Prompt

→ Connecting to Github -

- Create a new empty repo on Github
- Link local repo to Github using
  < git remote add origin <repo-url> >
- Gemini can handle this linking automatically if prompted

I have created a new fresh repository on github for this project. Please Link our local repository to it as the origin. The remote URL is <repo-url> ← Prompt

→ Authentication

- Github requires authentication before pushing code
- Options:
  - SSH Keys
  - PAT
  - gh auth login

→ Pushing code to remote

- Use Gemini CLI to push the main branch to g'thub
- local commits are now synced and stored securely in cloud

Please push the "main" branch to the remote "origin" repository on github ← Prompt

→ Hands-on Cycle

- Add a new feature
- Cycle → (status → stage → Commit → Push)
- The workflow forms the backbone of modern software development

→ Takeaway -

- Automates workflows
- Natural language prompts into reliable git commands
- Makes version control faster, simpler and more intuitive