MF_MOCK Project Summary

Static Website Development with Git & GitHub

1. Project Setup

Created the foundational structure for the static website:

• Created **MF_MOCK** folder
• Created **index.html** using the **!** shortcut for HTML boilerplate
• Created **js/main.js** with JavaScript to dynamically render content
• Learned VS Code shortcuts: Emmet, commenting (**Ctrl+/**), cursor modes

2. Built the Website

Implemented a JavaScript-driven static website:

• Wrote HTML shell with ** **
• Wrote JavaScript to inject content dynamically into the DOM
• Tested in browser — ✓ **Success!**

Sample JavaScript code:

```
const app = document.getElementById("app");
app.innerHTML = content;
```

3. Git Version Control

Set up version control with some troubleshooting:

Initial Issues:

• Accidentally ran **git init** in wrong folder (PROJECTS instead of MF_MOCK)
• Encountered ownership warnings

Solutions Applied:

• Fixed ownership: **git config --global --add safe.directory [path]**
• Navigated to correct folder: **cd MF_MOCK**
• Initialized Git properly: **git init**
• Staged files: **git add .**
• Created first commit: **git commit -m "Initial commit - basic static site"**

4. GitHub CLI Installation & Setup

Installed and configured GitHub CLI for streamlined workflows:

• Installed GitHub CLI: **winget install --id GitHub.cli**
• Restarted VS Code to recognize the **gh** command
• Authenticated with GitHub: **gh auth login** (via web browser)

5. Created GitHub Repository

Used GitHub CLI to create and push the repository:

• Created repo: **gh repo create MF_MOCK --public --source=. --remote=origin --push**
• Fixed duplicate **origin** remote issue
• Successfully pushed to GitHub: **git push -u origin main**
• Repository URL: **https://github.com/Park-Piao/MF_MOCK**

6. Key Concepts Learned

Git Remote (origin)

• **origin** is a nickname for your GitHub repository URL
• It's a shortcut so you don't have to type the full URL every time
• Standard convention for the main remote repository

CLI Tools & Commands

• Each tool (Git, GitHub CLI, Netlify CLI, npm) has its own command syntax
• CLI isn't a programming language — it's a way to interact with tools via terminal
• Learn commands as you need them using **--help** flags and documentation

Learning Approach

• **Learn by doing** is the recommended approach for most developers
• Use built-in help: **git --help**, **gh --help**
• Reference documentation and search when needed
• Even experienced developers look up commands regularly

7. Final Project Structure

```
MF_MOCK/ ■■■ index.html ■■■ js/ ■ ■■■ main.js ■■■ ShortCut.txt
```

8. Essential Commands Reference

Git Commands:

• **git init** — Initialize a new Git repository
• **git add .** — Stage all files for commit
• **git commit -m "message"** — Create a commit with a message
• **git remote add origin [url]** — Add remote repository
• **git push -u origin main** — Push to remote and set upstream
• **git remote -v** — View configured remotes

GitHub CLI Commands:

• **gh auth login** — Authenticate with GitHub
• **gh repo create [name] --public** — Create a new public repository

VS Code Shortcuts:

• **!** + Tab — Generate HTML boilerplate
• **Ctrl+/** — Toggle comment

• **Insert** key — Toggle between insert and overtype mode
• **Ctrl+Space** — Trigger IntelliSense autocomplete

9. Learning Resources

Git: https://git-scm.com/docs
GitHub CLI: https://cli.github.com/manual/
Netlify CLI: https://docs.netlify.com/cli/get-started/
Emmet (HTML shortcuts): https://docs.emmet.io/cheat-sheet/

10. Next Steps

• **Deploy to Netlify** — Make the website live on the internet
• Add CSS styling to enhance the appearance
• Implement more JavaScript features (navigation, forms, etc.)
• Learn about continuous deployment (auto-deploy on git push)

—

Summary generated on February 1, 2026