## **Chapter Five: Change Management**

### **5.1 Version Control Tool Used**

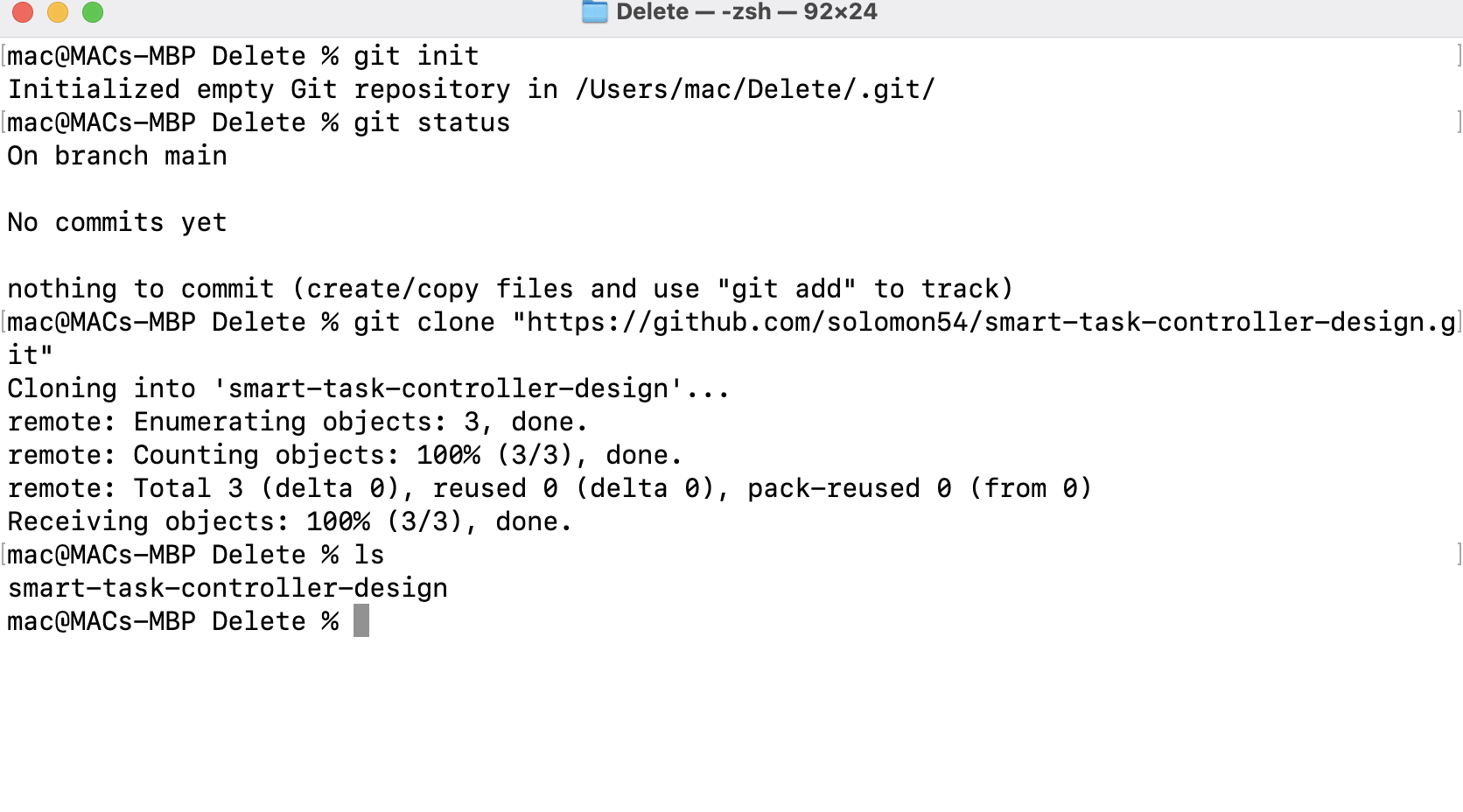
The project employs **Git**, a distributed version control system, hosted on **GitHub** for collaborative development and version tracking. This tool enables multiple developers to work simultaneously, merge changes, and maintain a history of all modifications.

### **5.2 Git Workflow and Commands**

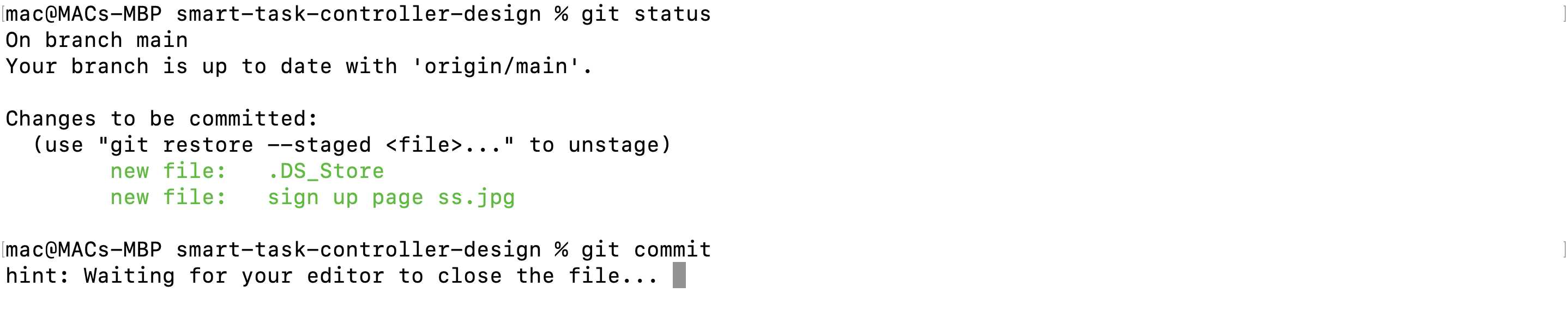
The Git workflow ensures systematic change management with the following commands:

|  |  |
| --- | --- |
| **Command** | **Description** |
| git init | Initializes a new Git repository in the project directory, marking the starting point for version control. |
| git add <file> | Stages specified files for the next commit, preparing changes for inclusion. |
| git commit -m "msg" | Commits the staged changes to the local repository with a descriptive message (e.g., "Add User class"). |
| git push | Uploads local commits to the remote GitHub repository, synchronizing changes. |
| git pull | Downloads the latest changes from the remote repository to the local copy, ensuring alignment. |

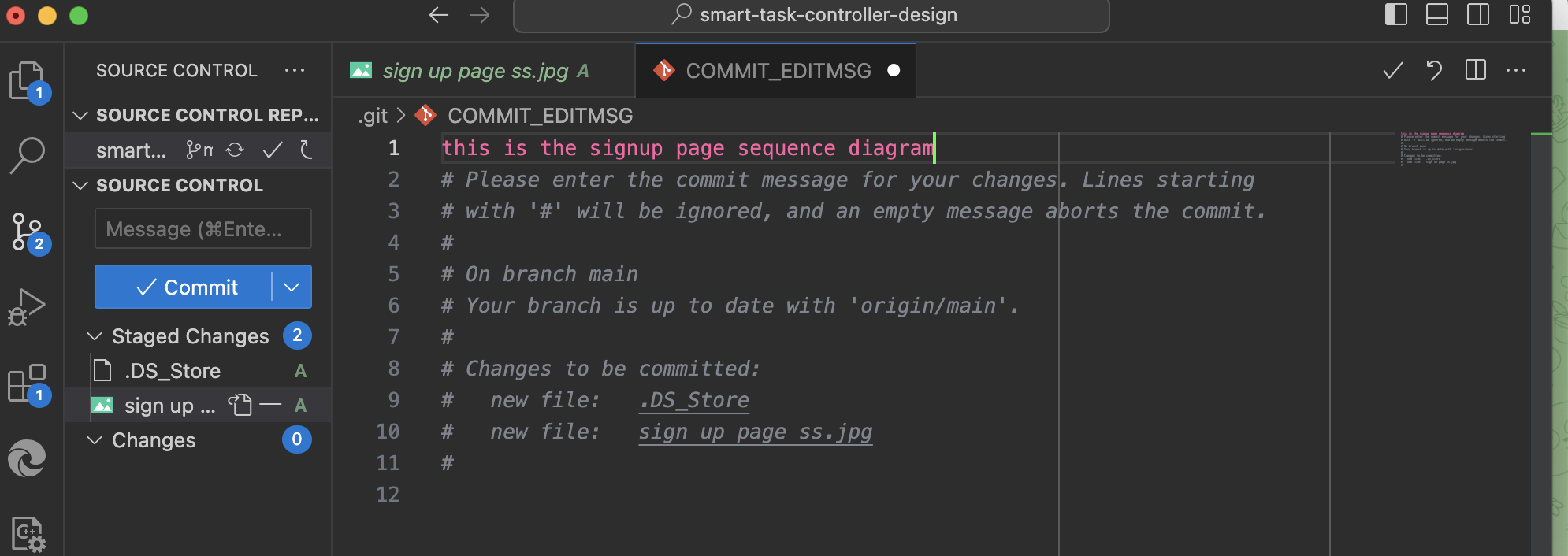
### **5.3 Git Snapshots**

To demonstrate version control in action, screenshots of key Git activities are included, such as commit logs, branch creation, and pull requests on GitHub. These visuals provide evidence of the project's development history and collaboration efforts.

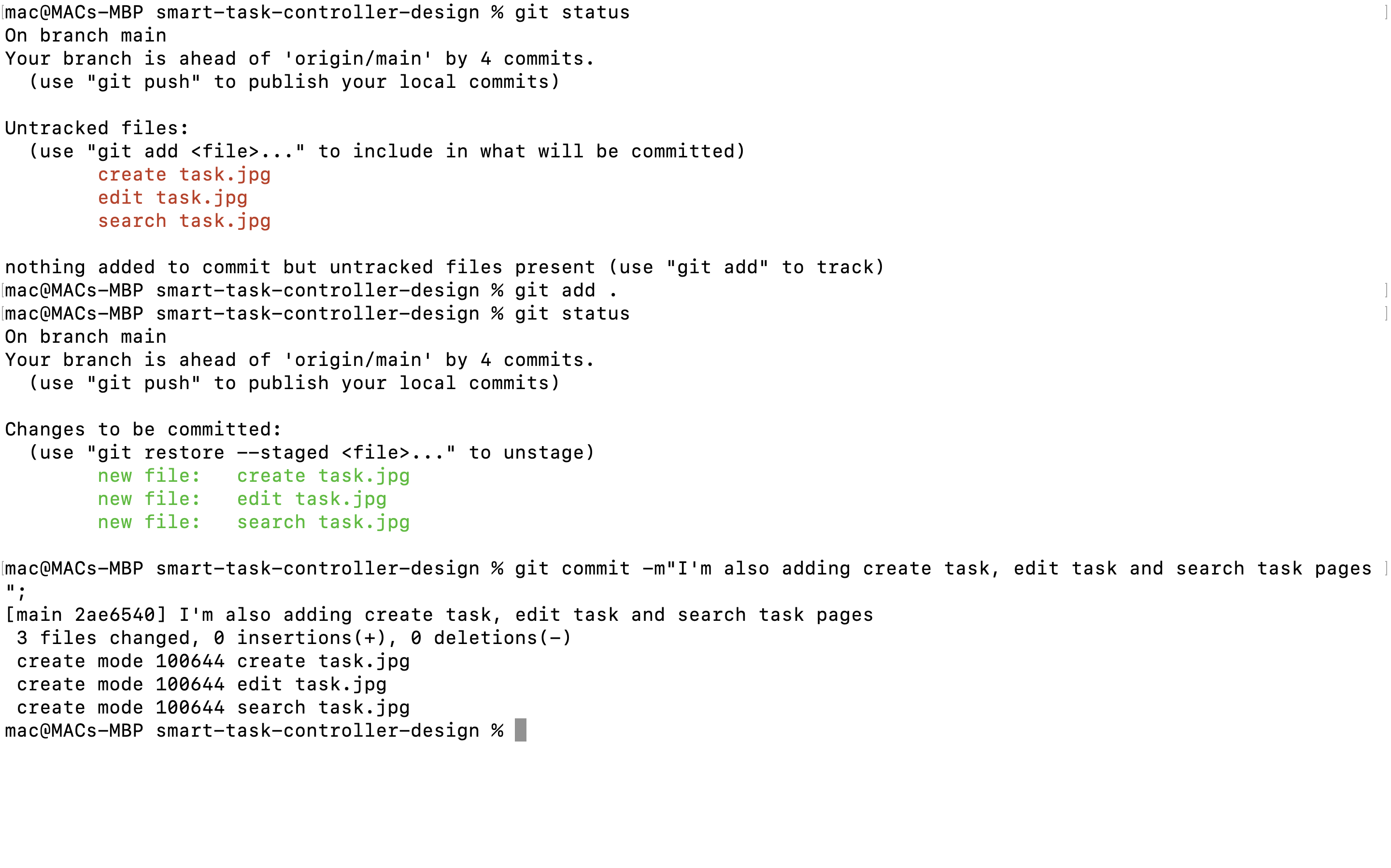
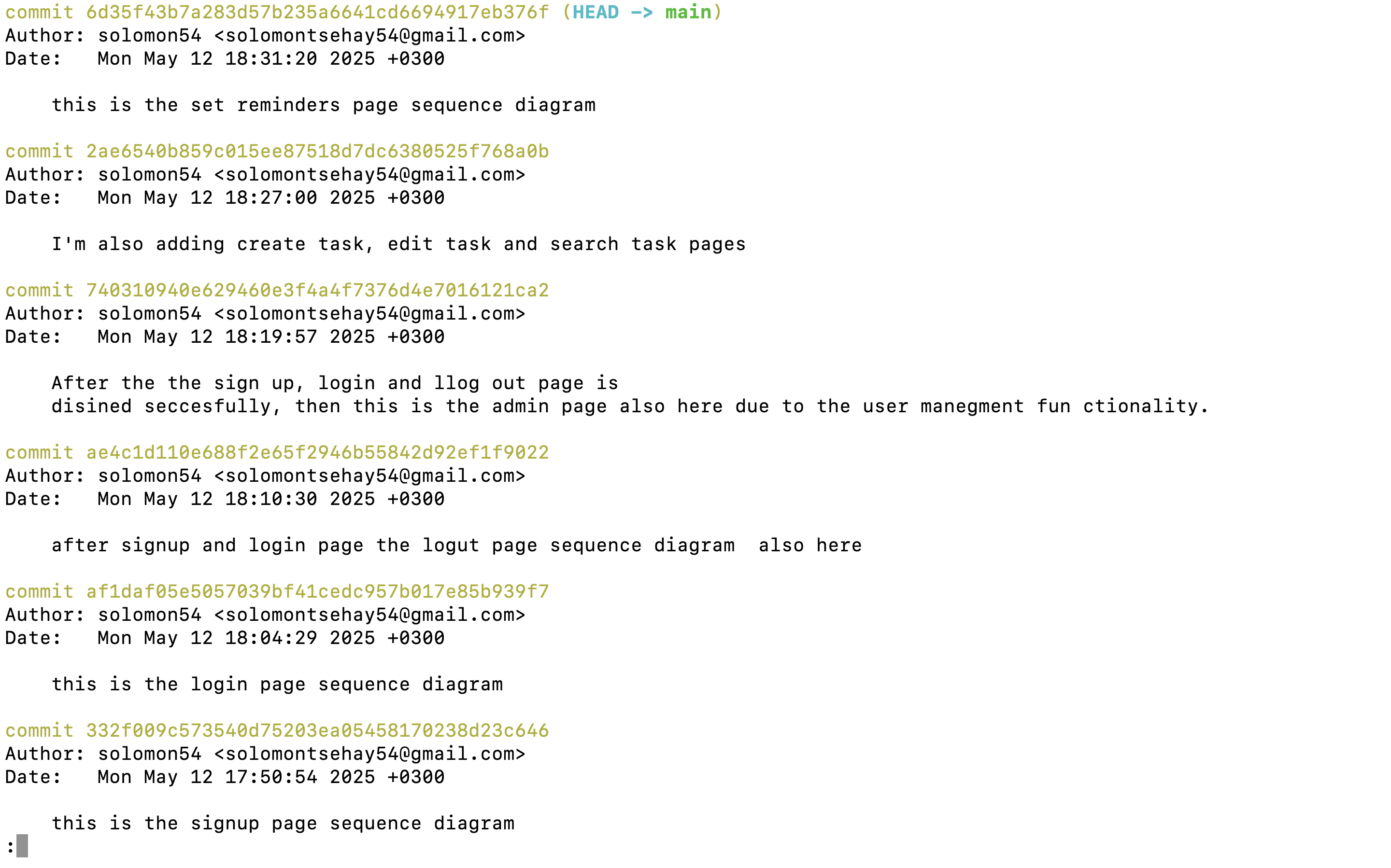
git init



### **git add**

****

**git commit**

****

**git log**

### **5.4 Benefits**

* **Track History**: Every change is logged, allowing reversion to previous states if needed.
* **Easy Collaboration**: Multiple developers can work concurrently with merge conflict resolution.
* **Revert Changes**: Mistakes can be undone using commit history.
* **Branch-Based Development**: Features can be developed in isolation before merging into the main branch.