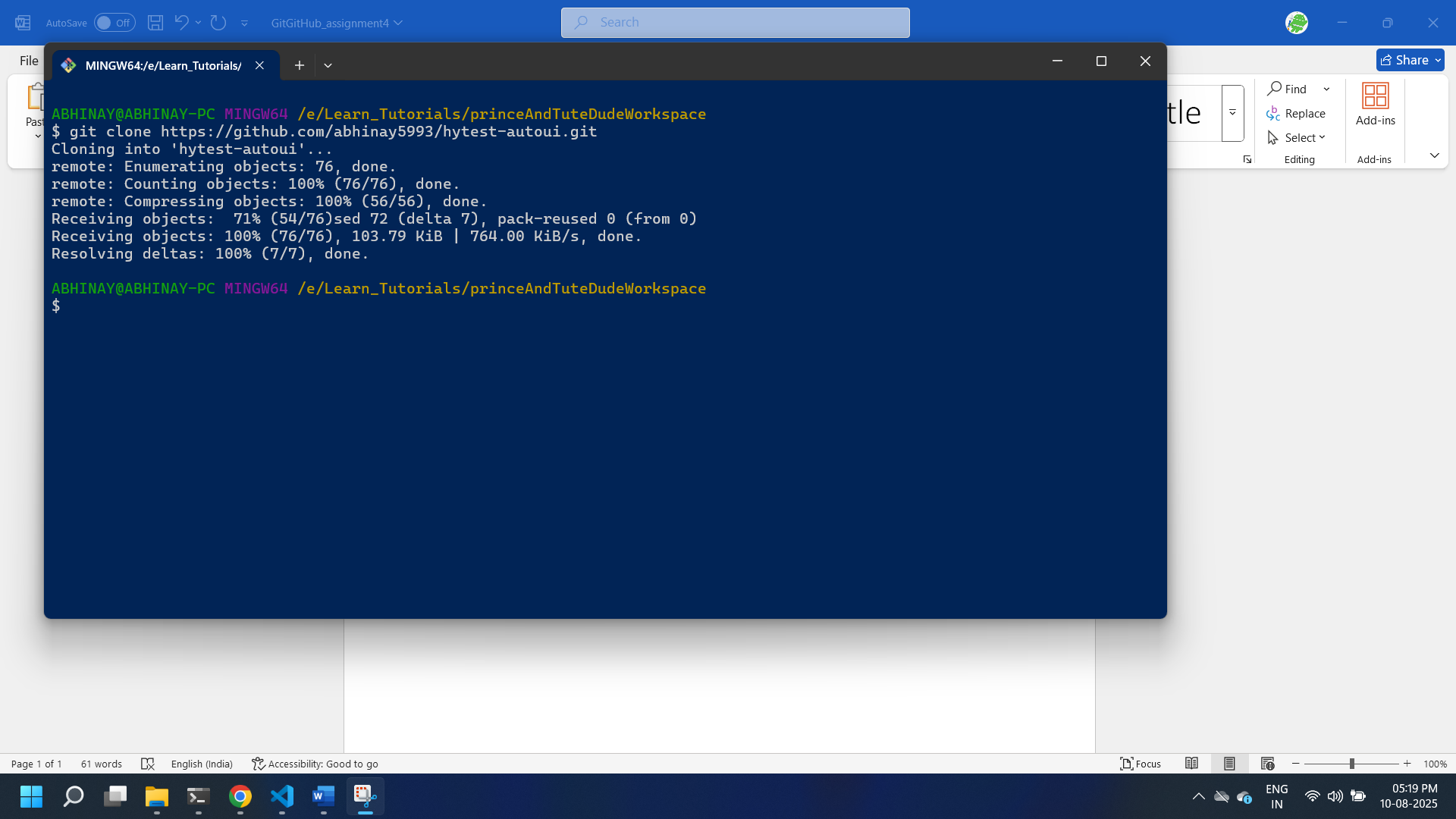
**Task – 1:**

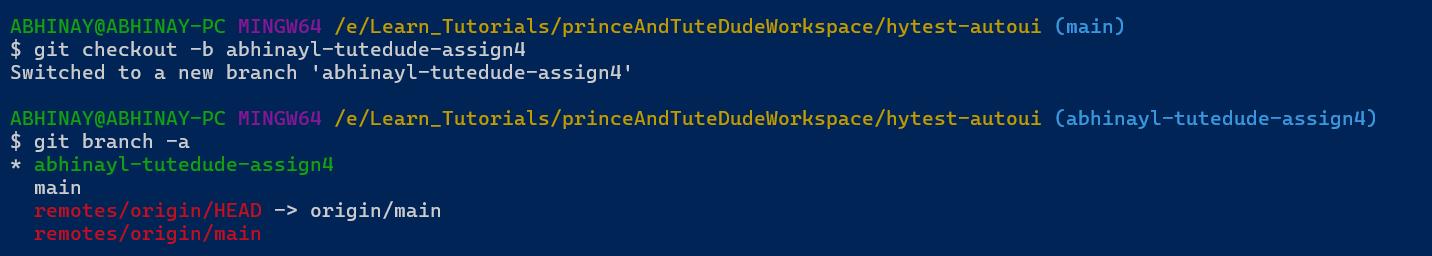
**Create a new GitHub repository –**

[https://github.com/abhinay5993/**hytest-autoui**.git](https://github.com/abhinay5993/hytest-autoui.git)

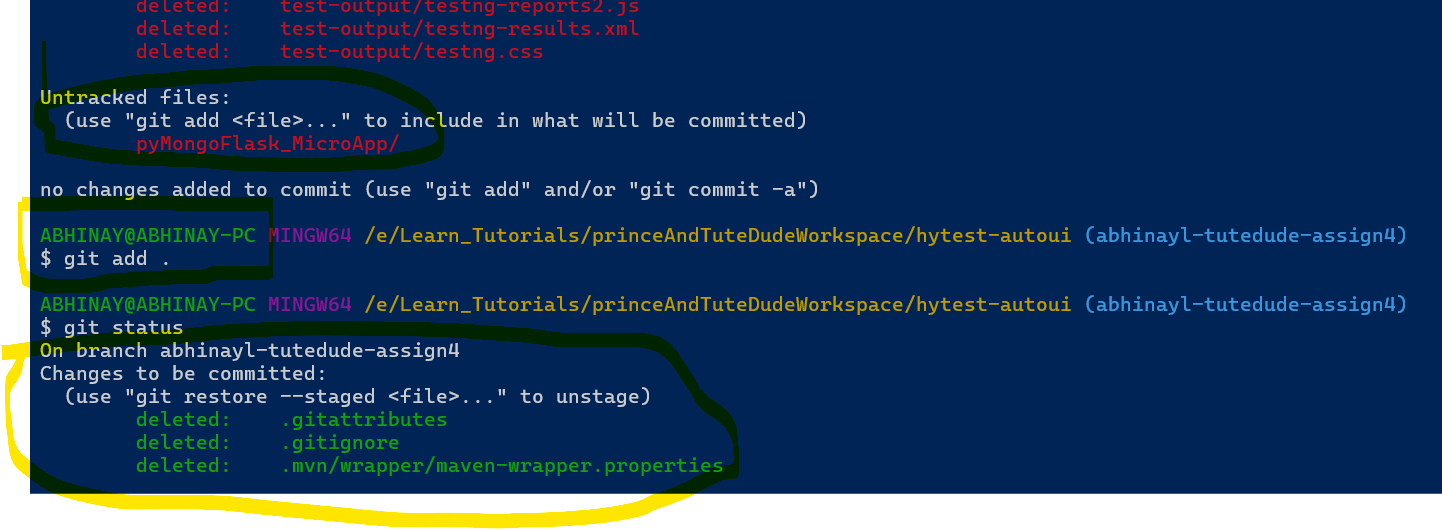
* **Clone the repository to your local machine using SSH (generate an SSH key if needed, add the public key to your GitHub account) :**

****

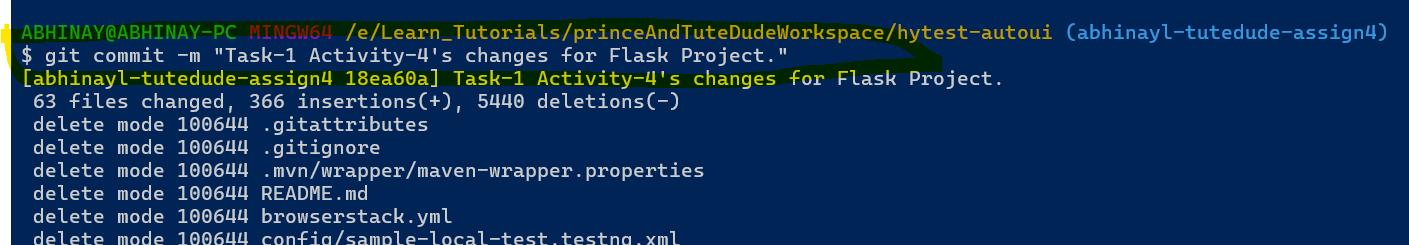
* **Create a new branch named after your username (e.g., Tutedude):**

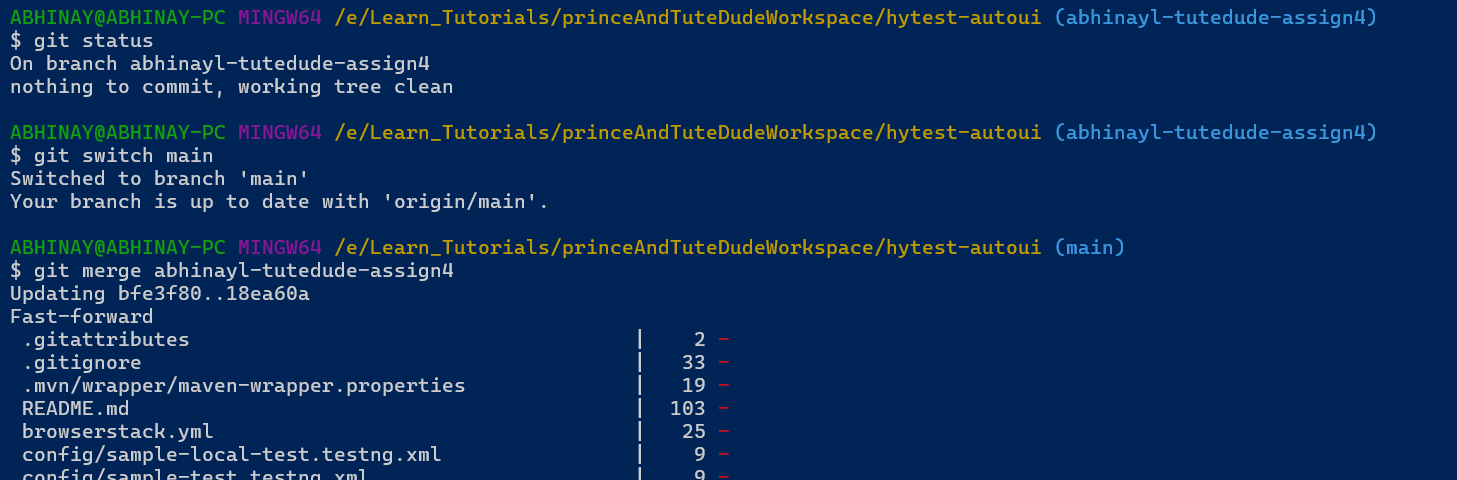
****

* **Add your Flask project files to this branch :**

****

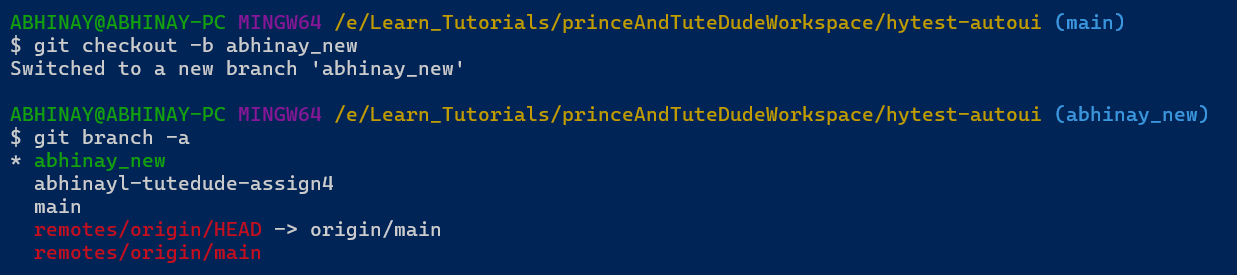
* **Commit the changes and merge the branch into the main branch :**

****

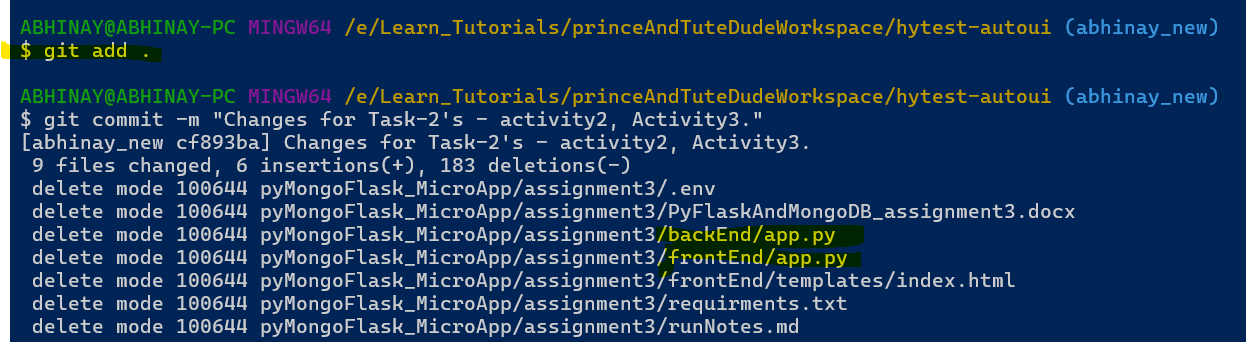
****

**Task – 2:**

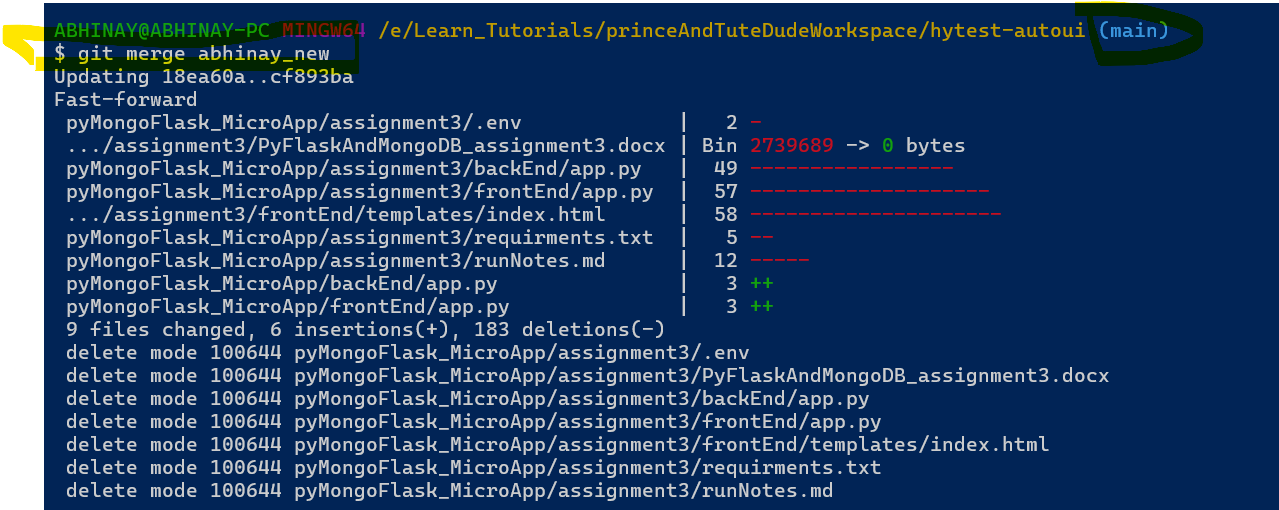
**Create a new branch named <your\_name>\_new (e.g., Tutedude\_new) :**

****

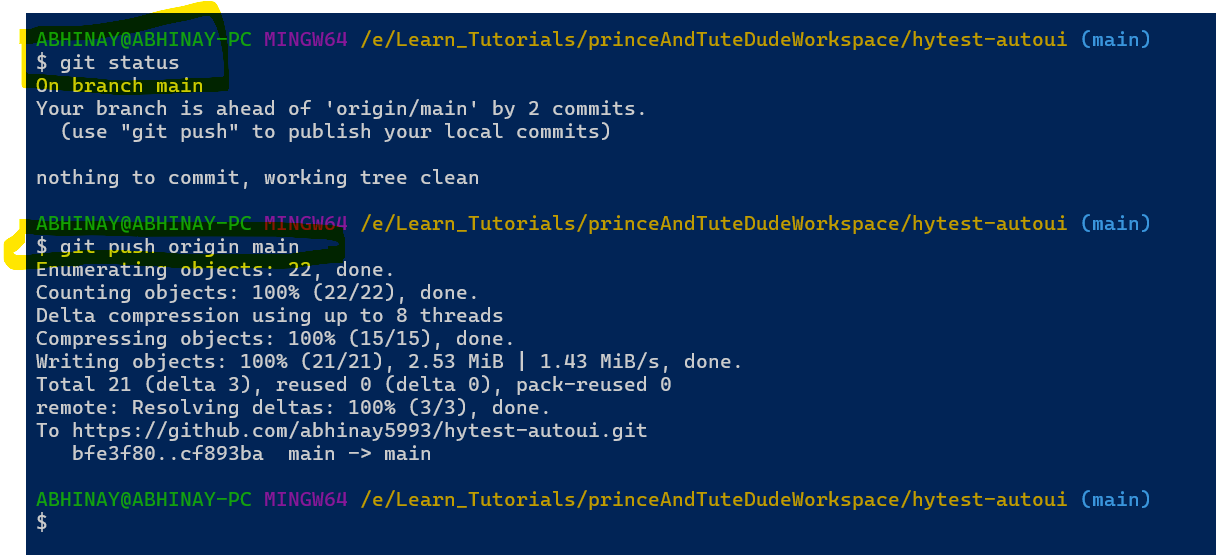
* **Update the content of the JSON file used for the /api route in this branch:**

****

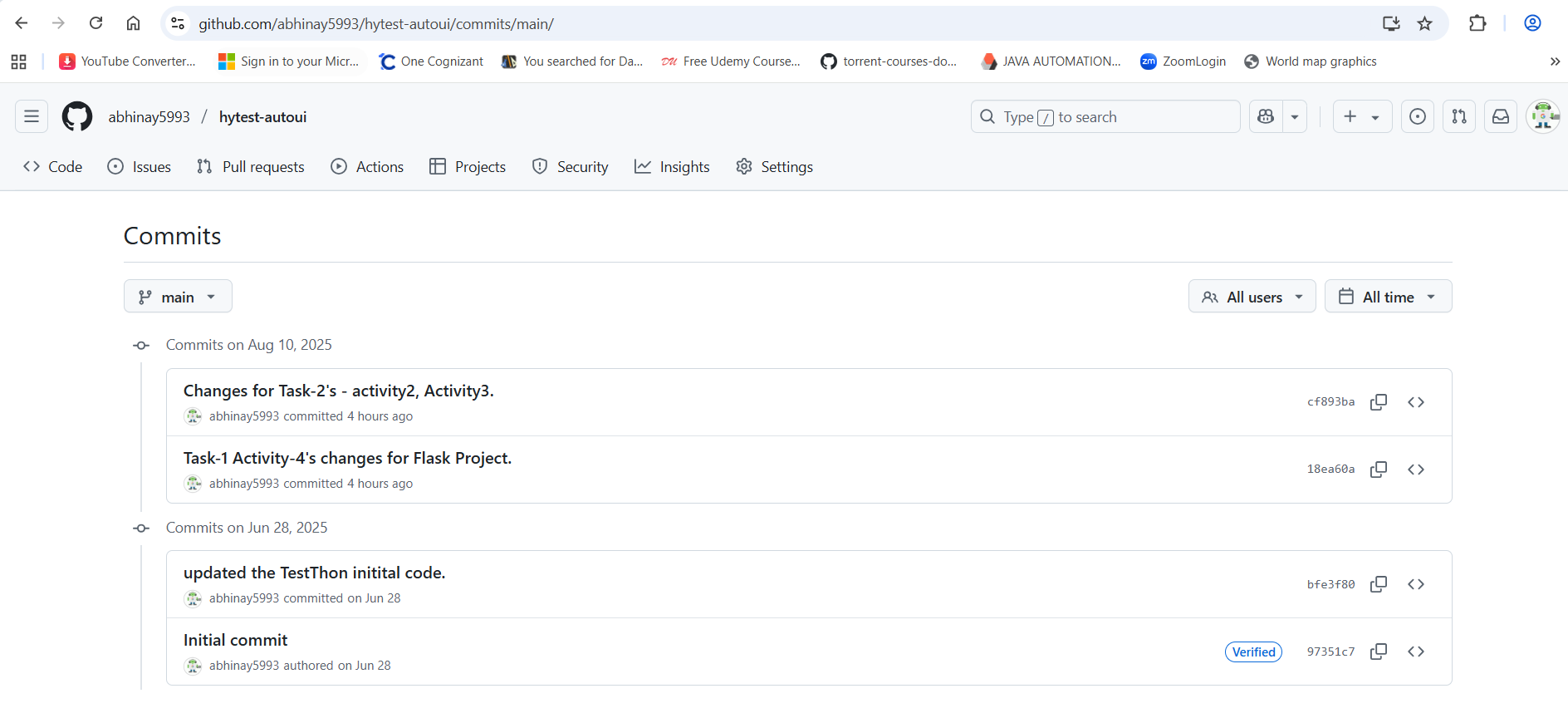
* **Merge the <your\_name>\_new branch into the main branch.**
* **If there are conflicts during the merge, resolve them by accepting the changes from the <your\_name>\_new branch. 🡪 able to merge without any conflicts:**

****

* **Add the resolved changes to the staging area, commit them, and push the updates to the remote repository:**

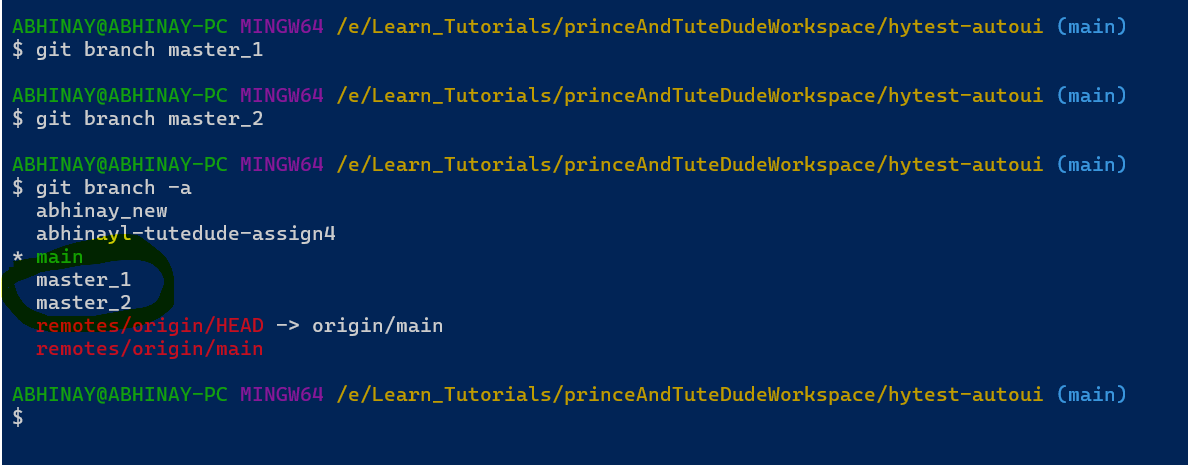
****

**GitHub Commit Logs:**

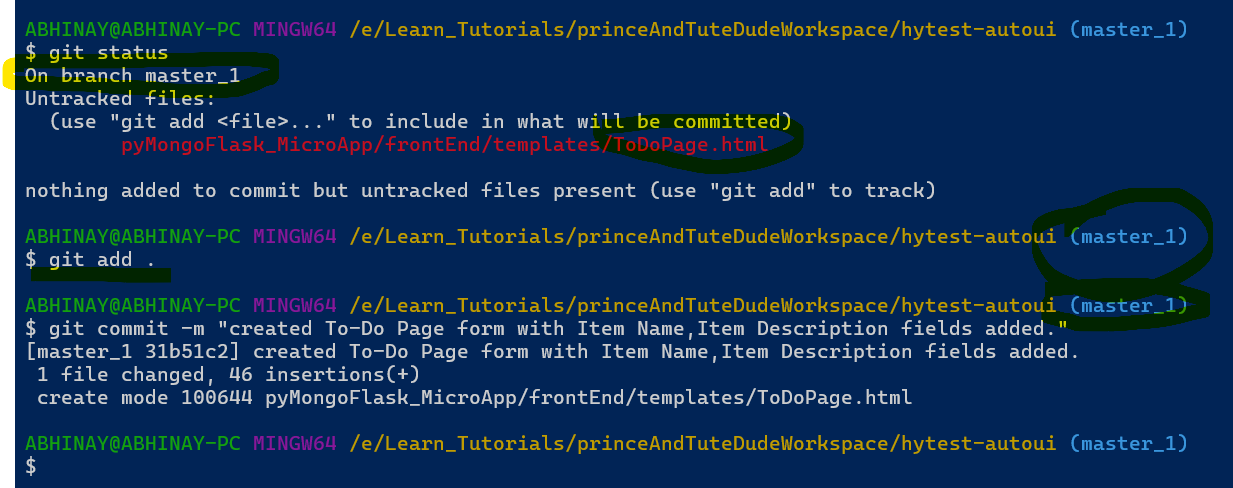
****

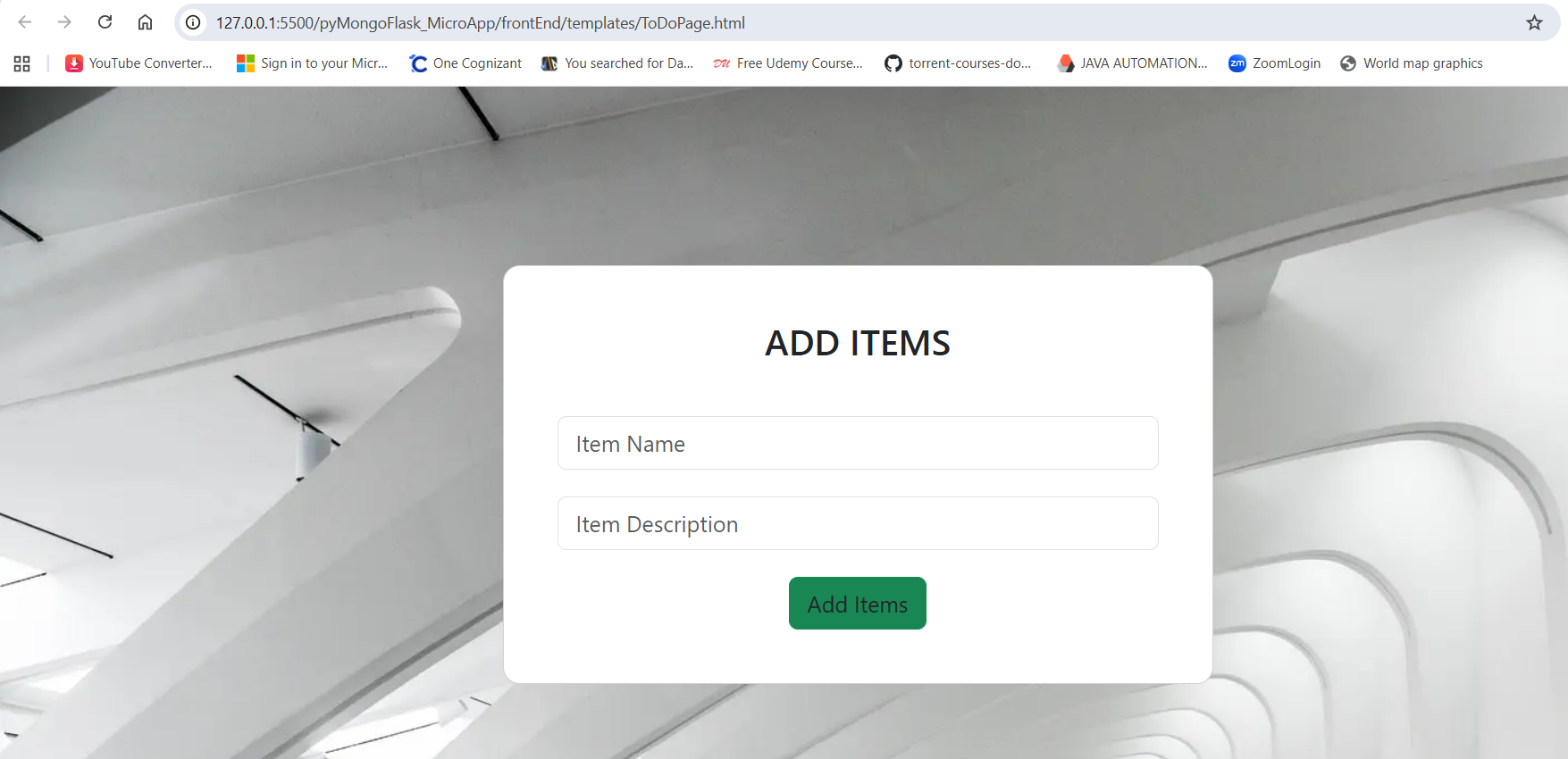
**3. Task – 3: Branch Creation:**

* **Create two branches: master\_1 and master\_2 from the main branch :**

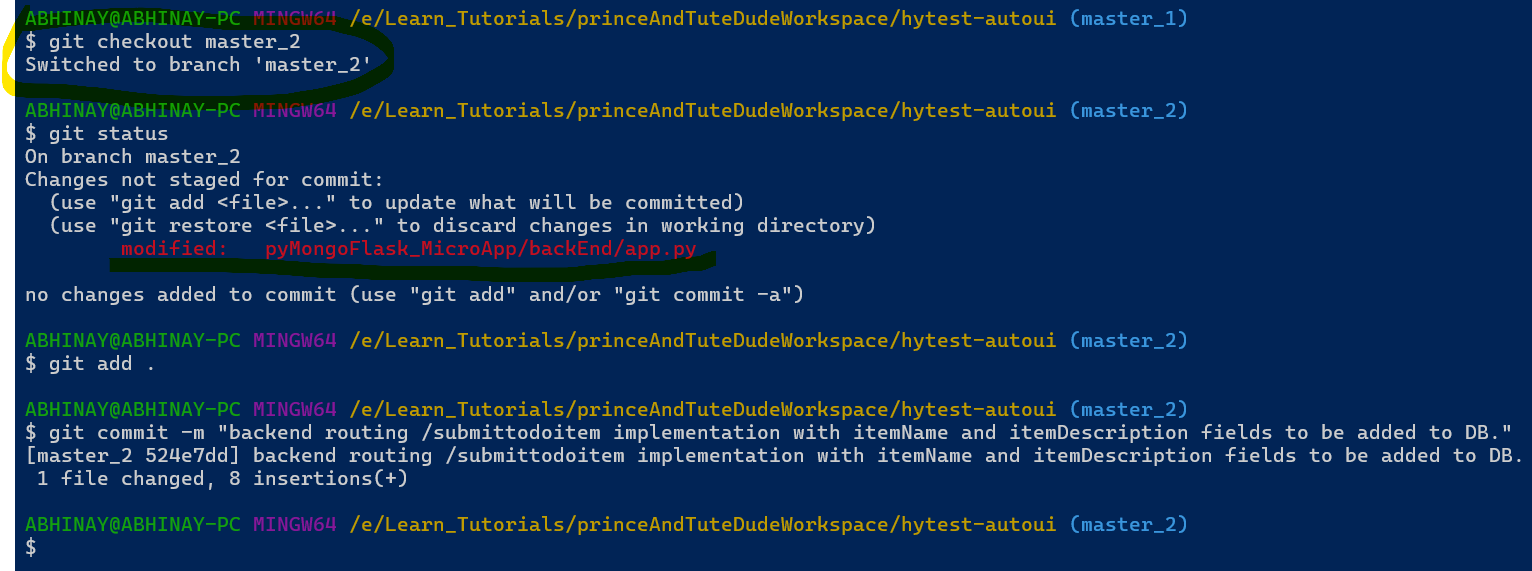
****

* **Feature Development in master\_1:**
* **In the master\_1 branch, create a To-Do Page in the frontend -**
  + **The page should contain a form with the following fields:**
    - **Item Name**
    - **Item Description**

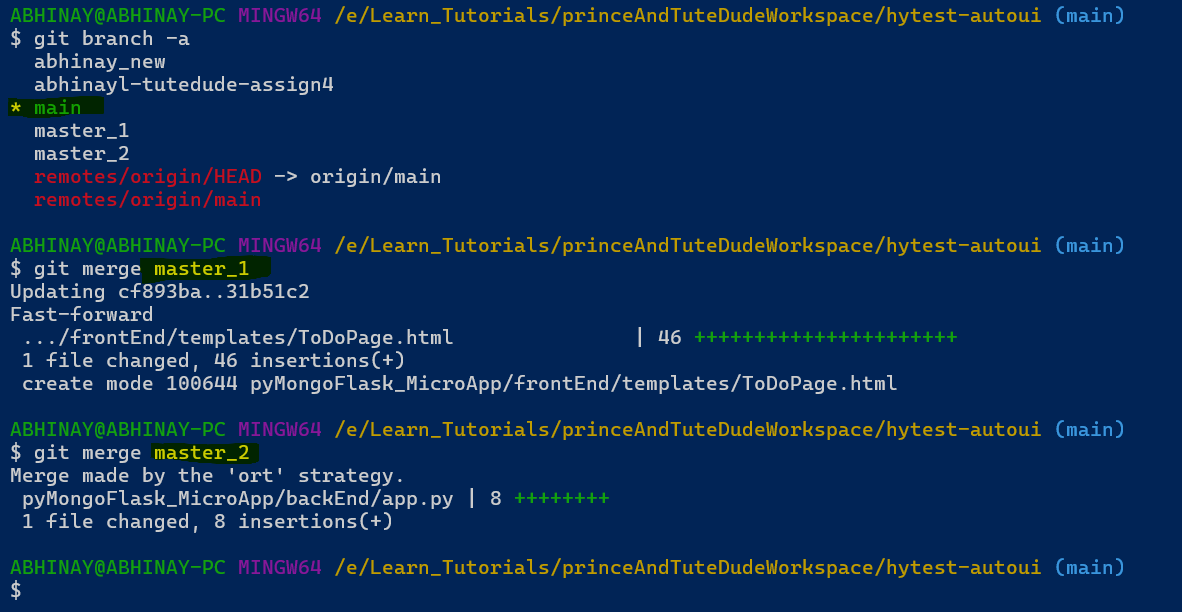
****

****

* **Backend API in master\_2:**
* **In the master\_2 branch, create a backend route named /submittodoitem.**
* **This route will:**
  + **Accept itemName and itemDescription via a POST request.**
  + **Store these details in a MongoDB database.**

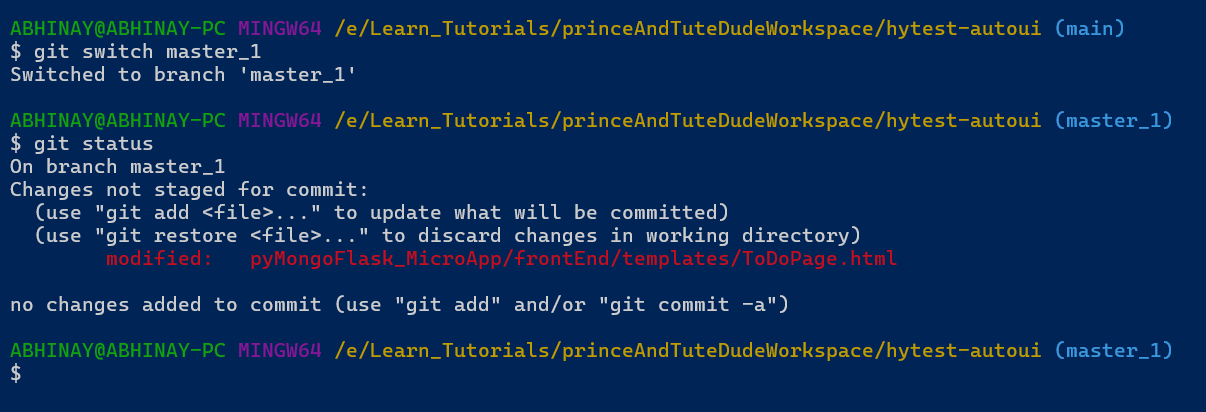
****

* **Merging Changes: Merge the changes from both master\_1 and master\_2 into the main branch without any merge conflicts.**

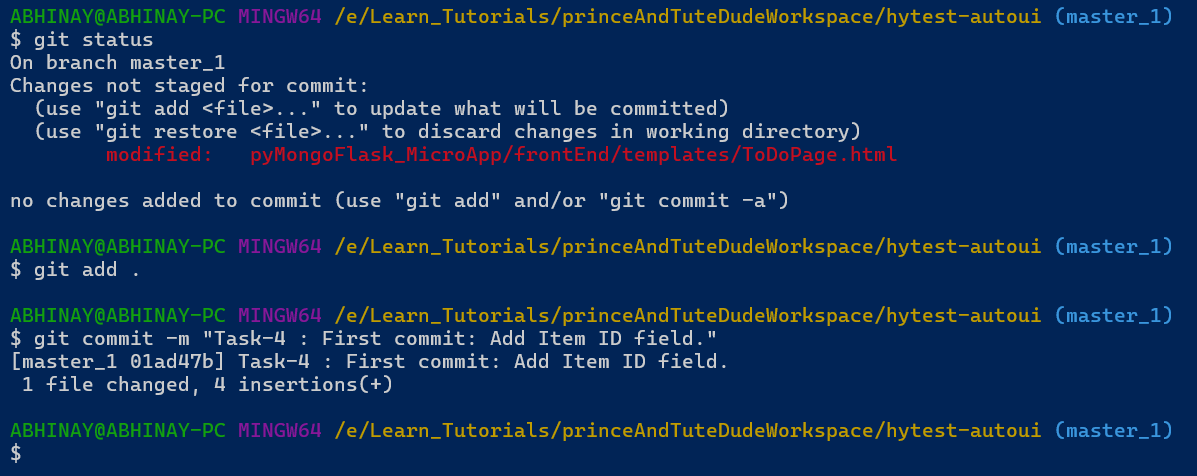
****

**4. Task – 4: Enhancing the To-Do Form in master\_1:**

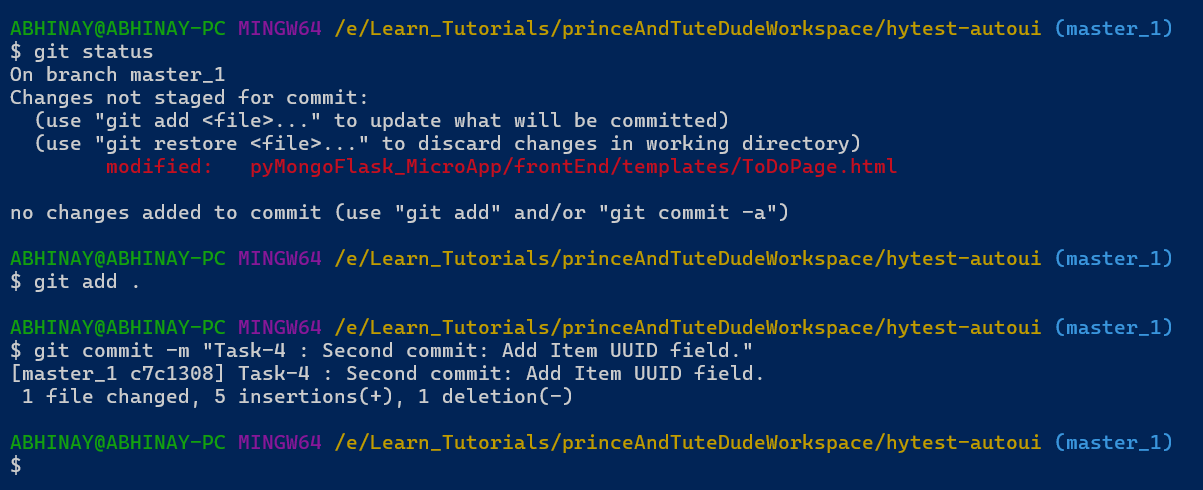
* **In the master\_1 branch, add the following fields to the To-Do form:**
  + **Item ID**
  + **Item UUID**
  + **Item Hash**

****

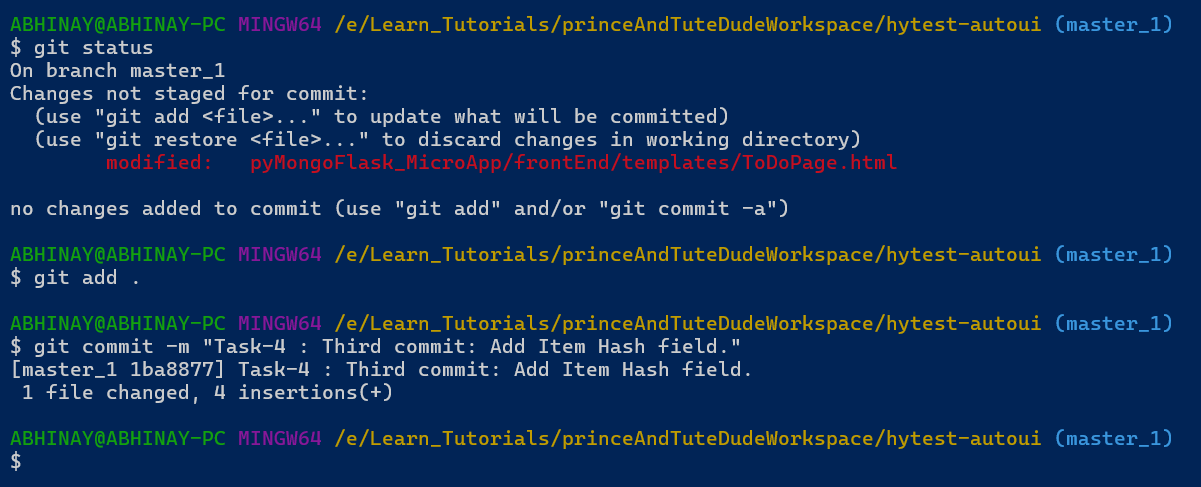
* **Committing in Sequence:**
* **Add and commit each field separately in the following order:**
* **First commit: Add Item ID field.**

****

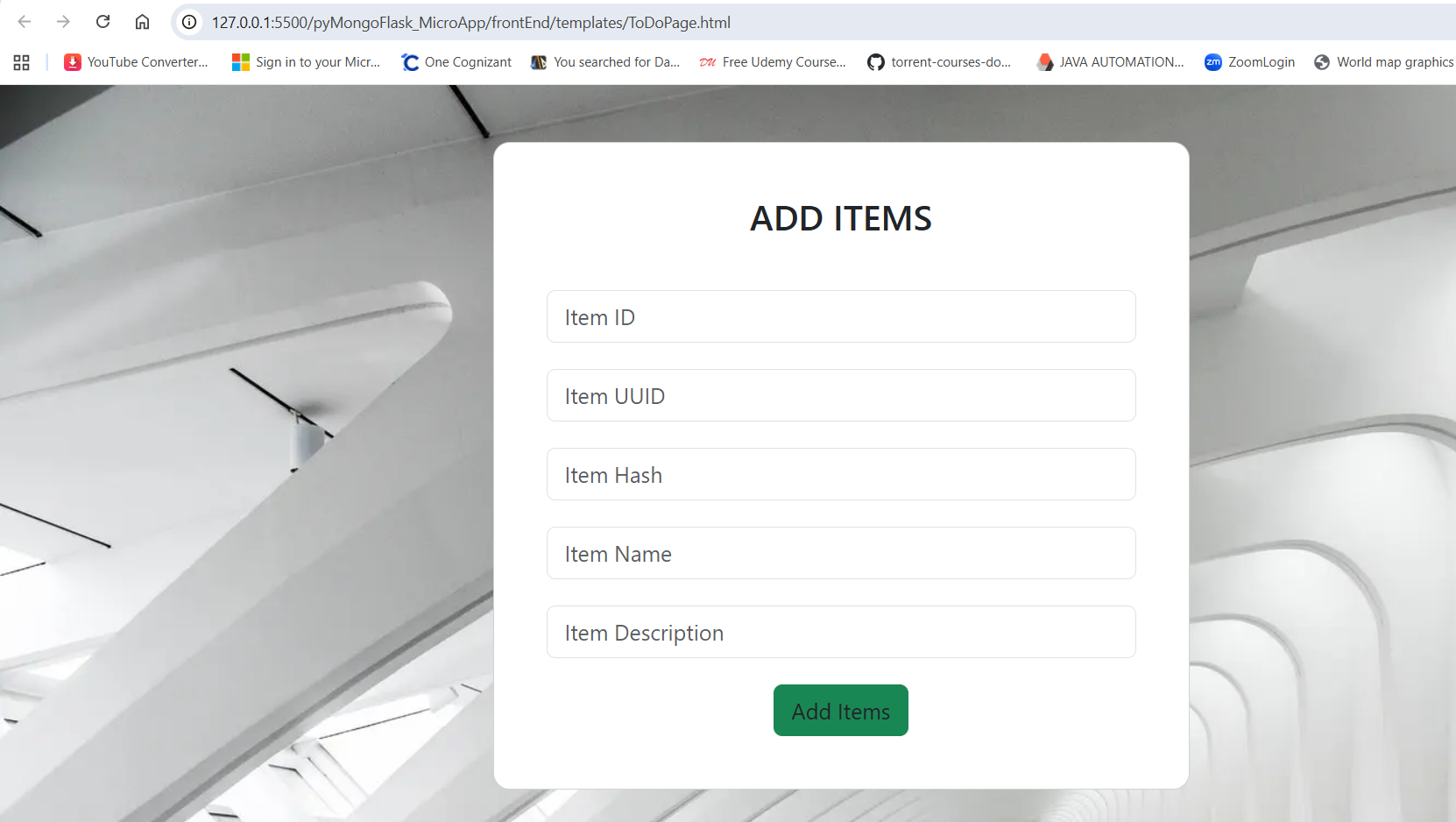
* **Second commit: Add Item UUID field.**

****

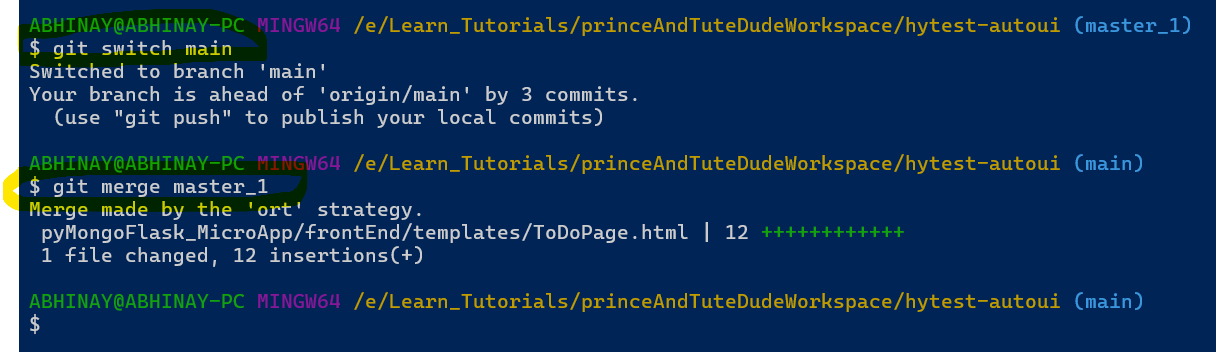
* **Third commit: Add Item Hash field.**

****

**Updated View UI Page:**

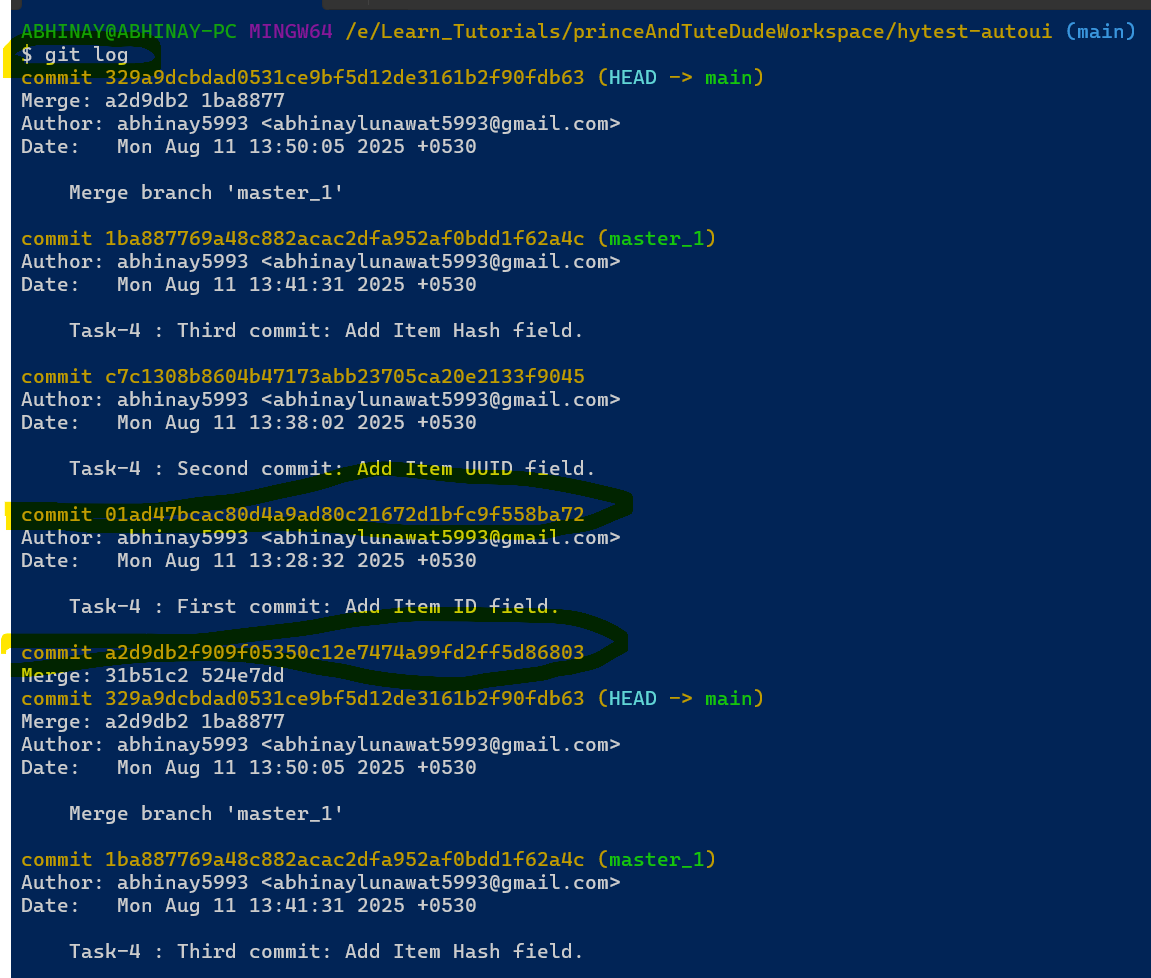
****

* **Merging to main:**
* **Merge the master\_1 branch into the main branch.**

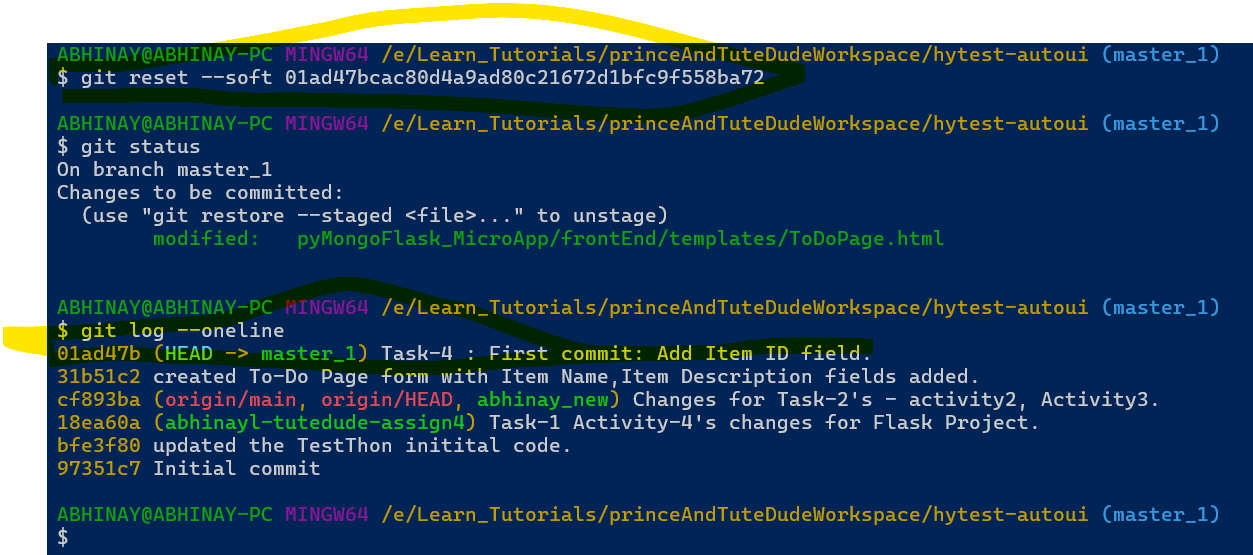
****

**5. Task – 5: Git Reset and Commit Deletion:**

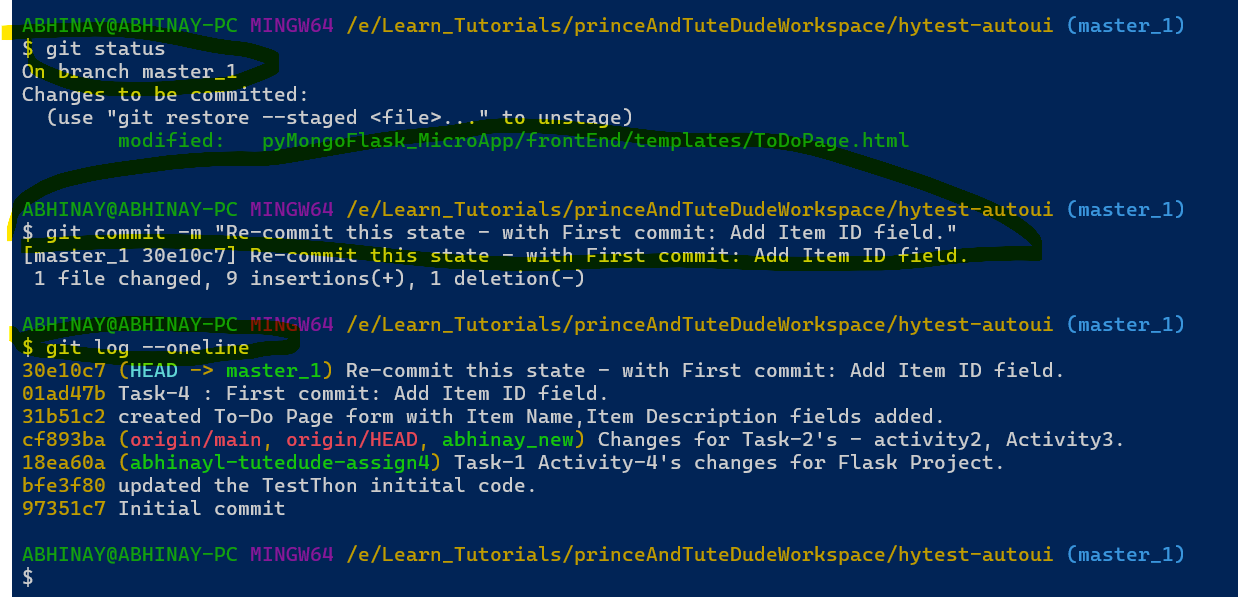
* **In the main branch, use Git Reset to roll back to the commit where only the Item ID field was added.**

****

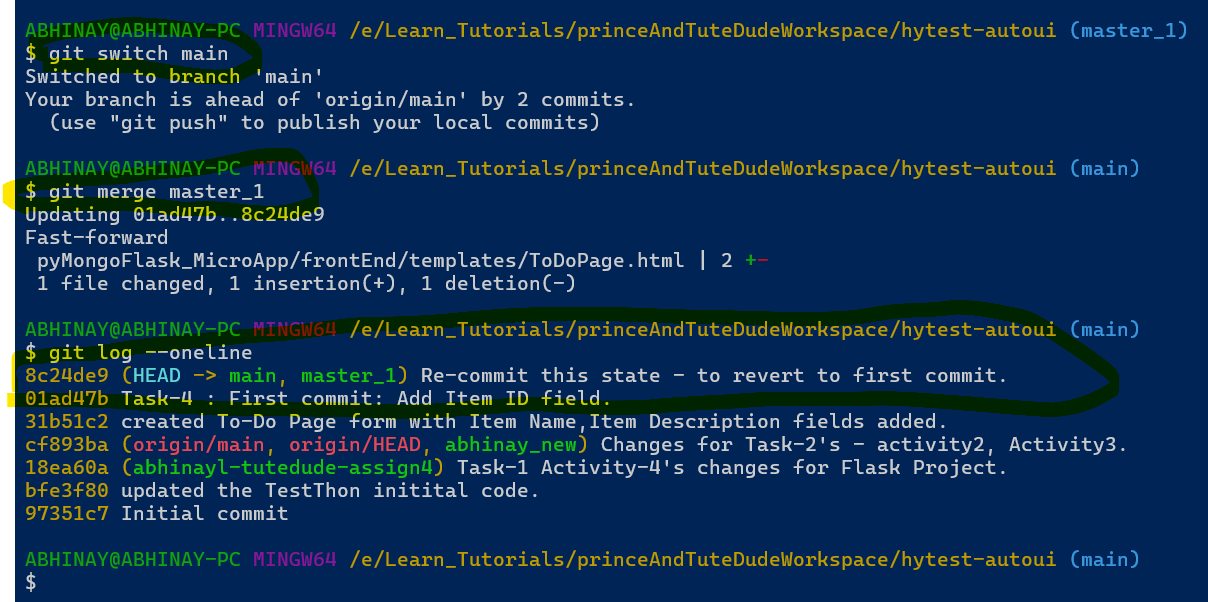
* **Use git reset --soft to ensure changes remain staged:**

****

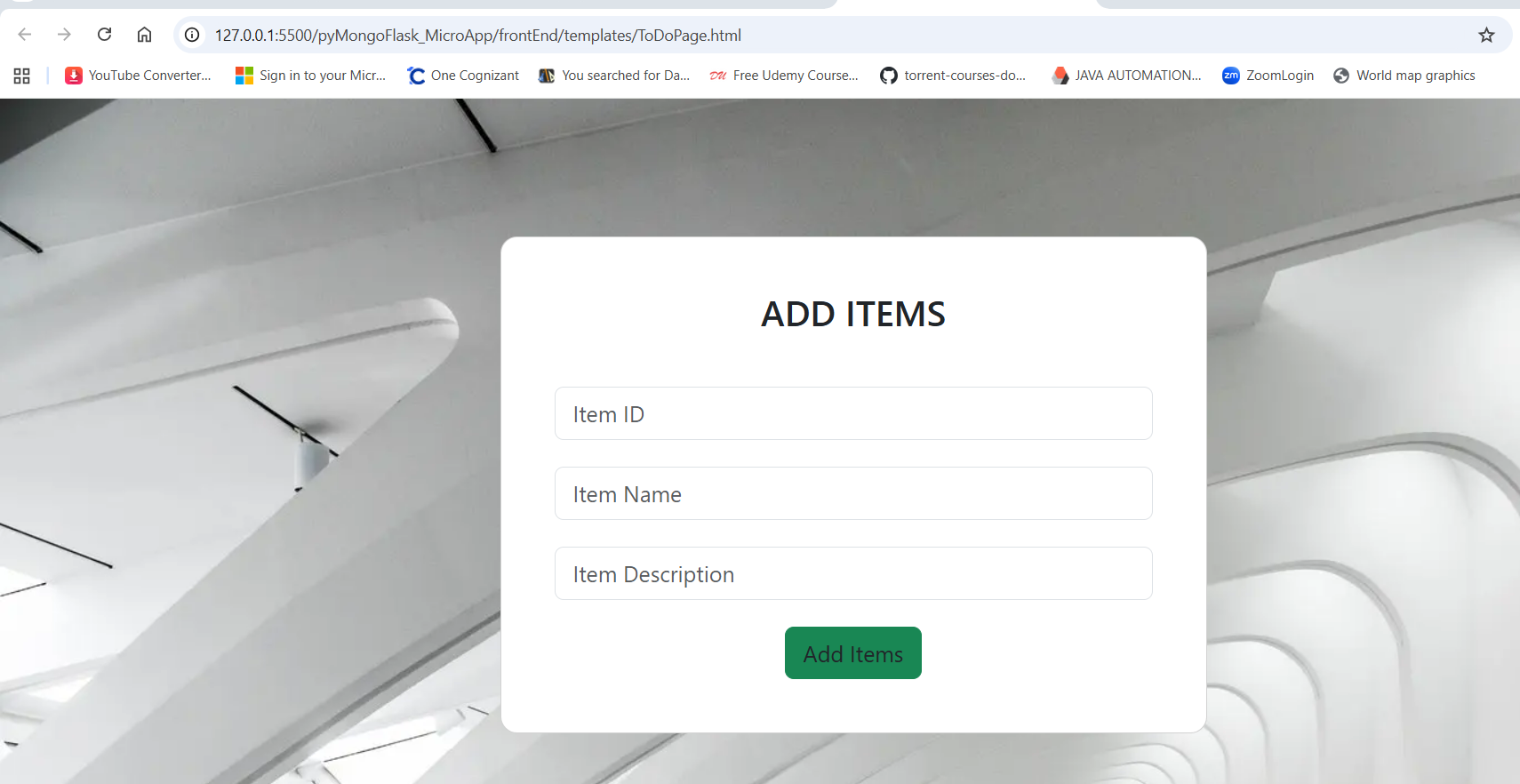
* **Re-commit this state to the main branch.**

****

* **Merge this updated state to the main branch.**

****

**Post Merge UI View:**

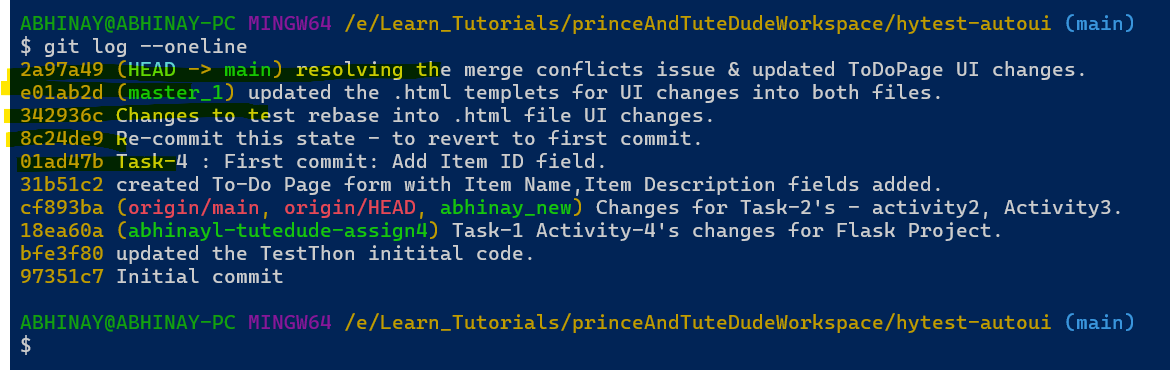
****

**6. Task – 6: Rebasing Changes:**

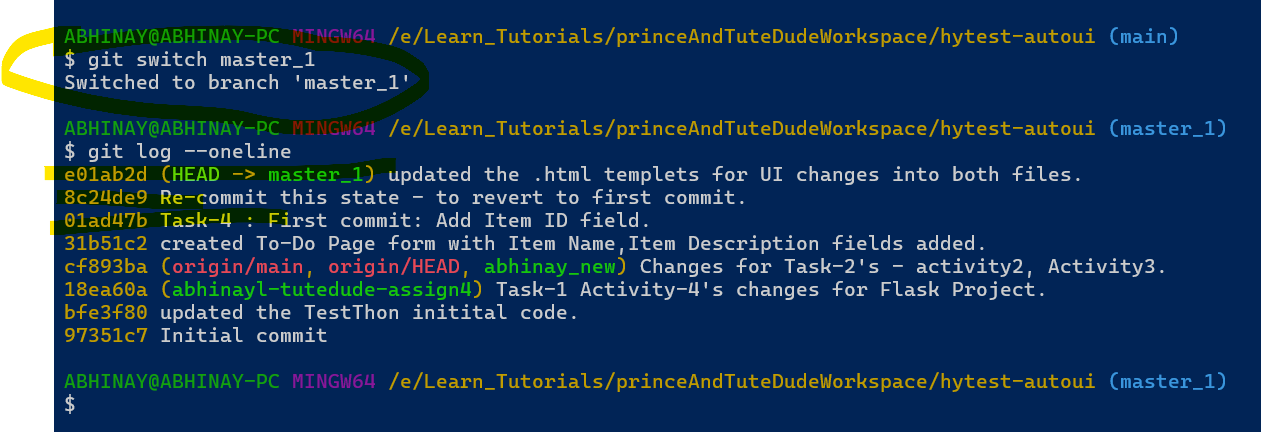
* **Rebase the updated changes in the main branch to the master\_1 branch.  
  Clarification:**
* **During rebasing, preserve individual commits to maintain the commit history for each change (i.e., do not squash commits).**

**Use git rebase main master\_1 to integrate changes from the main branch back into the master\_1 branch.**

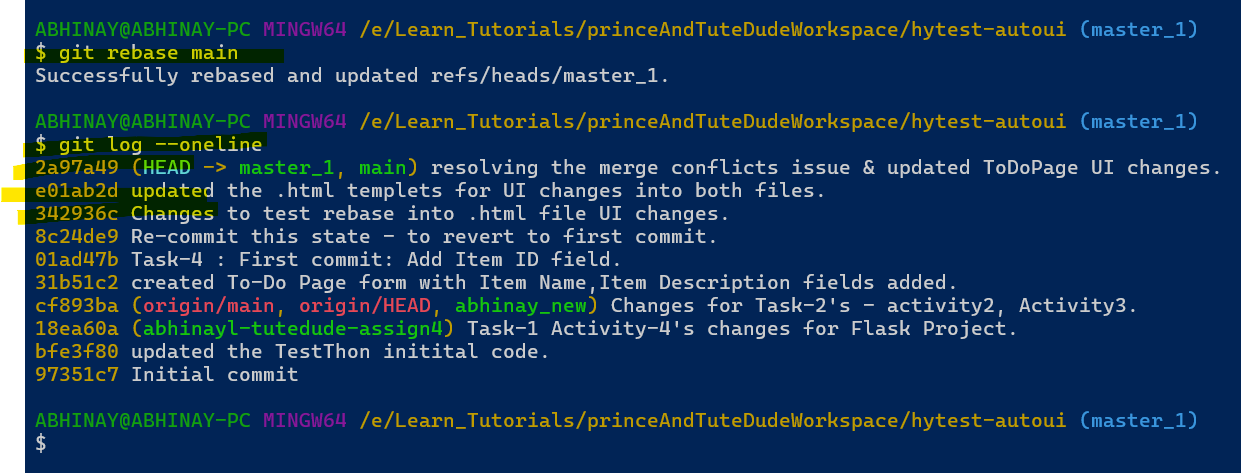
**Before Rebase ‘main’ branch logs :**

****

**Before Rebase ‘master\_1’ branch logs :**

****

**Rebase Operation into ‘master\_1’ branch :**

****

**Thanks,**

**Abhinay Lunawat**

**Abhinaylunawat5993@gmail.com**