Week 2 Part 1 Assignment

Deadline: Tuesday, 11:59 PM

Part 1: Project Setup and Git Commands

Task:

For this assignment, you'll be working with Git and GitHub through the terminal (no GitHub web interface). The main goal is to create a new repository, push the project files, and document the commands used. Here's how to set up and execute the project:

Git Commands Documentation:

1. Create a New GitHub Repository (Week2-FullName):
    - Create a new GitHub repository called Week2-FullName (replace "FullName" with your full name).
    - Go to your GitHub account, create a new repository with the desired name, and leave it public.
    - Copy the repository URL from the GitHub page after the repository is created.
2. Clone the Repository to Your Local Machine:
    - After the repository is created, use the following command to clone it to your local machine:

```
git clone <repo-url>
```

Replace <repo-url> with the actual URL of your new GitHub repository (it will look something like https://github.com/yourusername/Week2-FullName.git).

3. Add Files to the Staging Area:
    - After you modify or add files to the project, use either of these commands to add all files to the staging area:

```
git add .
git add *
```

This will prepare the changes for committing.

4. Commit the Changes:
    - When you're ready to save the changes, commit them with a clear message explaining what you have done. For example:

```
git commit -m "Completed Week 2 assignment: Added Excel management features."
```

5. Push Changes to GitHub:
    - After committing your changes, push the changes to your GitHub repository:

```
git push
```

This will update your repository with the latest changes.

Basic Terminal Commands:

Command	Description
cd Week2-FullName	Change directory to the project folder Week2-FullName
ls	List files and folders in the current directory (use dir on Windows)
mkdir new_directory_name	Create a new folder with the specified name inside the project directory
python app.py	Run a Python script named app.py
pip install pandas openpyxl streamlit	Install required libraries: pandas, openpyxl, and streamlit

Part 2: Streamlit Excel Data Management Web Application

Task:

Create a Streamlit web application that manages Excel files (including reading, modifying, adding, and deleting data). This app should allow the user to interact with files stored in a folder named "All Data - FullName". The app should allow users to perform the following actions:

Requirements:

1. **Display All Excel Files:**
    a. Load all Excel files from the folder named "All Data - FullName".
    b. List all the available Excel files for the user and allow them to select one to open.
2. **File Operations:**
    a. Create a New Excel File:
        i. Provide a button to create a new Excel file inside the folder "All Data - FullName".
    b. Open Existing Excel File:
        i. Provide an interface to open and view the content of any selected Excel file.
    c. Add/Edit/Delete Rows:
        i. Allow the user to interact with the displayed data (add, modify, or delete rows).
        ii. Provide buttons to add a new row, edit an existing row, or delete a row.
        iii. Provide confirmation for actions like deletion.
    d. Delete Table:
        i. Allow users to delete the entire table (file), with a confirmation message before proceeding.
3. **Specific Excel File Creation:**

Once the user is done with file operations, allow them to create a new Excel file named **"Warehouse Number One"**.

4. In this file, include the following columns:
    a. Product
    b. Quantity
    c. Amount
    d. Weight
    e. Product Serial Number
    f. Product Supplier
5. Users should be able to manually input data into these columns for testing purposes.

Libraries & Tools Required:

- Streamlit:
  To build the user interface (UI) for the web app.
- Pandas:
  For handling data and manipulating Excel files.
- Openpyxl:
  For working with Excel files.

Steps to Implement the Streamlit Web Application:

- **Create a Streamlit App (app.py):**
  - Create an app using the Streamlit library.
  - Implement functionality to load and display the list of Excel files.
- **Display Data from the Excel File:**
  - Allow the user to open an Excel file and display the data inside a table.
- **Interactivity:**
  - Implement buttons for adding, editing, and deleting data.
  - Use st.button() for actions and st.selectbox() for selecting files and actions.
- **Message Confirmation:**
  - Use st.confirm() or similar Streamlit functionalities to confirm user actions like deletion and addition.

Submission Instructions:

1. **GitHub Repository:**
   - Make sure your GitHub repository is named Week2-YourName (replace "YourName" with your full name).
   - Push all code and documents related to the assignment to the repository.
2. **Documentation:**
   - Document your code line by line either in the README.md file or in a separate document (PDF or video walkthrough).
   - Include any necessary instructions for running your Streamlit app.

General Instructions:

- **Collaboration:**
  You may collaborate with others, but the work must be original.
- **External Libraries:**
  You are allowed to use any external libraries as long as they are relevant to the task and make the implementation easier.

Example of the Excel File ("Warehouse Number One")

Once users create the new file, the columns will look like this:

Product	Quantity	Amount	Weight	Product Serial Number	Product Supplier
Screwdrivers	10	50	200g	12345	Supplier A
Product 2	5	25	300g	67890	Supplier B

Evaluation Criteria

Criteria	Description	Points
App Functionality	The app loads and interacts with Excel files (add, edit, delete).	40%
User Interface & Interactivity	Clear buttons, input fields, and confirmation messages.	30%
Code Documentation & Structure	The code is well-organized, with clear comments and structure.	30%
Total		100%

We look forward to seeing your projects. Good luck!