REIT6811 Tutorial 6

Comprehensive Data Handling in Research

Please ensure you have read the tutorial and understood its instructions beforehand. For this tutorial, you are required to read the GitHub Basics in Appendix A and complete steps 1 and 2 before attending the tutorial.

Tutorials are evaluated for each student; therefore, all students are required to participate. Every student should be actively involved in discussions among your groups and participate when your tutor discusses the answers. The marking rubric for tutorials is on Blackboard.

Scenario

You are part of a research team conducting a study on "Using Generative AI Tools - Boon or Bane." Your team has generated a large volume of data and documents through the course of this research. These include:

- Literature Review documents: Journal and conference articles, books, newspaper articles, etc.
- Quantitative Analysis: Survey data, survey questions, analysis files (Python scripts, csv, excel files), survey analysis report.
- Qualitative Analysis: several interview transcripts, interview protocols, consent forms, analysis/insights report, and data visualisations.
- Drafts and Reports: Draft research proposals, conference papers, and final reports.
- Additional Materials: Information sheets, photos, and other media files.

Activity 1 (10 mins)

In your groups, discuss the following questions about the above scenario. Write your answers **in your journals**:

1. **Storage and Accessibility:** Explain why it is important to store and organise your files efficiently. *(2-3 sentences)*
2. **File Storage Methods**: Explain your methods for storing the different types of data in the scenario. Specify which data types may require special access controls. *(2-3 sentences)*

Activity 2 (75 min)

GitHub is a powerful platform that combines version control with collaborative tools, making it ideal for managing and sharing code, documents, and data in both academic and professional settings. At its core, GitHub uses Git, an open-source system to track changes in files, allowing users to manage projects with precision. It offers features like issues, pull

requests, and project boards for effective collaboration. With free access to public and private repositories, GitHub simplifies managing and sharing work.

Referring to the above scenario, your task is to organise the materials efficiently, implement version control, and prepare them for collaborative work using GitHub.

Steps:

<u>If you are an entry-level GitHub user, please skip step 6. If you are an experienced GitHub user, please try to finish step 6.</u>

1. **Organise the Research Data**: Create a folder structure that categorises all documents, scripts, and data files.
2. **Create a GitHub Repository**: Sign up or log in to GitHub and create a new repository named *`Generative_AI_Research_SID'*. (replace SID with your 8- digit student ID).
3. **Download and Install GitHub Desktop**: Install the GitHub Desktop client on your computer. Alternatively, you may use the terminal for Git operations.
4. **Syncing and Backup**: Upload the organised folder structure to the repository. Ensure that each folder and file is correctly named and categorised as per your organisation plan.
5. **Commit and Push**: As you upload files, practice writing meaningful commit messages that describe the changes made (e.g., "Added cleaned survey data and initial analysis scripts"). After committing the changes of your repository, you can push those changes to the GitHub remote side.
6. **Implement Version Control and Collaborative Teamwork**:
    i. Find two other students in your session to form a project group including three members.
    ii. One of the members will be the code reviewer and the other two members will be the code developers.
    iii. The code reviewer will create a .txt file in his/her repository named "Project_logbook". In the .txt file, write a few sentences.
    iv. The code developers will fork and clone the code reviewer's repository to their local machine.
    v. Each of the code developers will create a new branch in their forked local repository. Please use an informative name for the new branch.
    vi. Each of the code developers will modify the content in "Project_logbook.txt". Please note that at this step, the code developers need to modify different parts of the .txt file, i.e., change the content in different lines.
    vii. After implementing the changes, both code developers will commit their changes and then create a pull request, respectively.

viii. The code reviewer will receive the pull requests from the code developers, review their changes, and then merge changes from their branches into the main branch of the code reviewer's repository.
ix. After merging the branches, the code reviewer will fetch origin (this should be done automatically in GitHub desktop. If it does not, you will do the fetch manually) and then pull origin to make the changes apply to the main branch.
x. Repeat steps vi-vii but the two code developers will change the same part of "Project_logbook.txt", e.g., change the content in the same line.
xi. After receiving the pull requests from the code developers, the code reviewer will see a red warning showing the branch has conflicts that must be resolved. The code reviewer will resolve the conflict (e.g., selecting changes from one developer or make the reviewer's own change), commit the changes for resolving the conflicts, push, and then merge the branches.

7. **Documentation:** Create a `README.md` file in your repository that outlines the structure of the project, how to navigate the files, and instructions for future collaborators on how to contribute.

In your journals, answer the following questions:

1. Why is it important to have a good naming convention? *You may want to consider the information at https://datamanagement.hms.harvard.edu/plan-design/file-naming-conventions (2-3 sentences)*
2. Provide screenshots of:
   a. A well-organised GitHub repository containing all research data, scripts, and documents with your username.
   b. A clear history that tracks the development of the research project. (Hint: 'Activity' tab)
   c. The `README.md` file providing an overview of the repository and guidelines for collaboration.

Activity 3 (optional)

UQ has several tools and resources to help researchers handle and store their data effectively and with integrity.

One of these is the UQ Library's Research Data Management Plan Checklist:
https://guides.library.uq.edu.au/for-researchers/research-data-management-plan/checklist

Read through the checklist and

1. Identify which checklist points you would be most comfortable with implementing (e.g. identifying file types, backing up and copying files, versioning software/datasets etc)

2. Identify which checklist points are least familiar to you (e.g. data access controls, policies for long-term data retention, metadata creation and documentation, metadata standards, data sharing and embargo periods etc)
3. Choose one checklist point you identified as unfamiliar and research it (using the UQ library guides or other authoritative sources) to understand its significance and requirements for effective implementation.

Another important tool at UQ is the Research Data Manager (UQ RDM), a system to store, manage and share research data. Using the RDM can significantly simplify a Research Data Management Plan for a project. The UQ library's RDM guide can be found here: https://guides.library.uq.edu.au/for-researchers/uq-research-data-manager

Using the UQ library's RDM guide, try to determine which checklist points in the Research Data Management Plan Checklist are automated by the RDM, or made significantly easier when using the RDM.

Appendix A. Instructions for GitHub Basics

1. Creating a GitHub Account (complete before tutorial)

- **Sign Up:** Visit https://github.com/ and sign up for a free account. You'll need to provide a username, email address, and password. After signing up, confirm your email to activate the account.

2. Getting Started with GitHub Desktop (complete before tutorial)

- **Download GitHub Desktop:** Go to https://desktop.github.com/ and download the application for your operating system (Windows or macOS).
- **Installation:** Follow the instructions to install GitHub Desktop on your computer.
- **Login:** Open GitHub Desktop and log in with the GitHub account you just created.

3. Creating a Repository

- **New Repository:** After logging in, click on the "+" icon in the top-right corner of the GitHub page and select "New repository."
- **Repository Name:** Give your repository a clear, descriptive name that reflects your course or project.
- **Visibility:** Choose whether to make the repository public (visible to everyone) or private (only you and your collaborators can see it).
- **Initialize Repository:** Add a README file so you could put more detailed information about your repository later.
- **Create repository:** Click the "Create repository" button to finish setting it up.

4. Adding Files to Your Repository

- Navigate to your newly created repository.
- Click the "Add file" button, then select "Upload files."
- Drag and drop your files into the box or click to choose files from your computer.
- **Commit Changes:** After adding the files, scroll down and add a commit message describing what the files are. Then click "Commit changes" to save them in your repository.

5. Syncing Your Work

- **Cloning the Repository:** To work on your files locally, you'll need to clone your repository. In GitHub Desktop, click "Clone a repository from the Internet…" and select the repository you created. This will download the repository to your computer.
- **Making Changes:** Open the files in your repository, make changes, save them, and return to GitHub Desktop.
- **Committing Changes:** After making changes, you'll see them listed in GitHub Desktop. Write a short description of what you changed and click "Commit to main" to save the changes locally.
- **Pushing to GitHub:** Click "Push origin" to upload your changes to the GitHub website. This keeps your online repository up to date.

6. Adding Collaborators

- **Invite a Collaborator:** To allow others to contribute to your repository, go to your repository on https://github.com/. Click on “Settings,” then “Collaborators.”
- **Search for User:** Enter the GitHub username or email of the person you want to invite.
- **Set Permissions:** Choose the level of access they should have (e.g., Read, Write, Admin). Send the invitation, and your collaborator will receive a request to join the repository.

7. Tracking Issues

- **Create an Issue:** In your GitHub repository, go to the “Issues” tab and click “New issue.”
- **Describe the Issue:** Write a clear description of the issue, including any relevant details or steps needed to resolve it. You can assign it to yourself or a collaborator.
- **Label and Milestone:** Add labels (e.g., bug, enhancement) to categorize the issue, and optionally link it to a milestone if it’s part of a larger project.
- **Tracking Progress:** Once the issue is created, you and your collaborators can comment on it, track its progress, and close it when it’s resolved.