Lab Experience Journal/Summary

June 14, 2025

What was done

https://github.com/imid12/jupyter-exploration.

This lab session provided a hands-on introduction to essential tools for modern data science and software development: **GitHub** for version control and **Jupyter Notebooks** for interactive computing. I've had a functional GitHub account for a while now.

I was able to **create a new repository.** I navigated to my GitHub profile and initiated the creation of a new repository, giving it a descriptive name ("jupyter-exploration."). I chose to initialize it with a README file, which is a common practice to provide immediate context for anyone viewing the repository. This repository would serve as the central hub for managing my lab work and tracking changes.

For the **Jupyter Notebook** component, I had two primary options: using an online environment or a local installation. For this lab, I opted for an **online environment**, specifically **Google Colaboratory (Colab)**. This choice eliminated the need for local setup and allowed me to quickly access a pre-configured Jupyter environment directly through my web browser. If I had chosen a local installation, I would have followed the instructions to install Anaconda or Miniconda, which bundles Jupyter Notebook along with other essential Python libraries.

Within the Jupyter Notebook environment (Colab in my case), I began by performing **basic operations**. This typically involved creating new cells and experimenting with fundamental Python code. I would have started with simple arithmetic operations (e.g., 2 + 2), printing text (print("Hello, World!")), and defining basic variables. I also practiced creating markdown cells to add descriptive text and explanations to my notebook, enhancing its readability and documentation. A key part of the process was saving my notebook frequently to ensure my progress was preserved.

What You Learned

This lab was a fundamental introduction to two incredibly powerful and widely used tools. My experience with **version control using GitHub** was particularly insightful. I learned that GitHub isn't just a place to store code; it's a robust system for tracking changes, collaborating with others, and managing different versions of a project. The concept of a **repository** as a project's central hub, and the importance of **commits** for saving discrete changes, became clear. While I didn't delve into complex branching and merging in this basic lab, understanding how to push my local changes to a remote repository on GitHub established a critical workflow for saving and sharing my work. It highlighted the value of having a historical record of all modifications, making it easy to revert to previous versions if needed.

My experience with **interactive computing using Jupyter Notebooks** truly transformed how I approach coding and data exploration. The cell-based structure, where I could execute code snippets independently and see the output immediately, was a game-changer. This interactivity makes it incredibly efficient for experimenting with code, visualizing data, and documenting the entire analytical process in one cohesive document. Unlike traditional script files, Jupyter Notebooks allow for a narrative flow, combining code, output, and explanatory text (using Markdown) seamlessly. This makes them ideal for reproducible research, teaching, and sharing analysis.

One of the initial challenges I faced was simply getting comfortable with the workflow of saving changes in the Jupyter Notebook and then ensuring those changes were properly reflected and pushed to my GitHub repository. This involved understanding the "save" functionality within Jupyter and then the git add, git commit, and git push commands (or their graphical equivalents if using a desktop client). While seemingly straightforward, ensuring everything was synced correctly took a few attempts. I overcame this by diligently following the instructions, paying close attention to the order of operations, and verifying the changes directly on my GitHub repository webpage. Another minor hurdle was remembering the syntax for markdown cells versus code cells in Jupyter, but a quick reference to basic markdown syntax helped resolve that.

Questions or Comments

This lab provided an excellent foundation for using GitHub and Jupyter Notebooks. I'm curious about the best practices for **managing larger projects** with many files across multiple repositories on GitHub. Also, for Jupyter Notebooks, what are the most effective ways to **manage dependencies and virtual environments** when working on complex data science projects locally?

The online environment for Jupyter Notebooks (like Colab) was incredibly convenient for this introductory lab. I also looked at using GPUs within Colab. For more advanced or production-level work, is a local installation generally preferred, and what are the primary benefits of going that route over cloud-based options?
