ConverSight

Git Integration

Enhancing Collaboration within ConverSight Notebooks.

Whitepaper

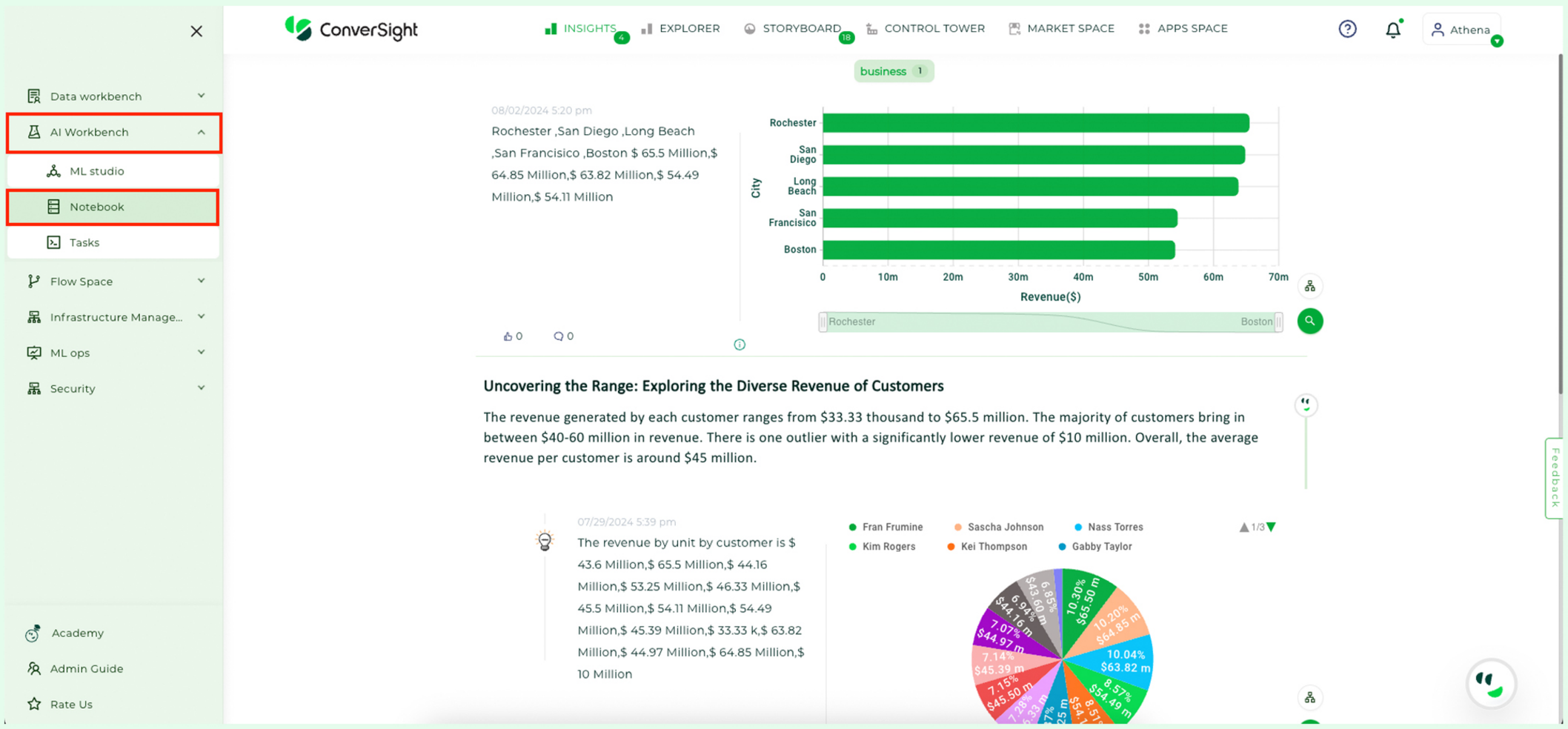
Table of Contents

I. Introduction to Git Integration

The convergence of ConverSight Notebooks with Git integration marks a significant milestone in the realm of collaborative development and version control. With this seamless integration, users are empowered with a robust solution for managing their projects efficiently, facilitating secure cloud storage and enabling effortless access from any device or location. This whitepaper aims to explain the integration of Git within ConverSight Notebooks, providing a comprehensive guide to harnessing its functionalities for streamlined workflow management and enhanced collaboration.

2. Getting Started

To access Notebook in the ConverSight platform, navigate to the Configuration section and choose **'Notebook'** under **'AI Workbench.'**



ConverSight offers two default Base Notebooks: The **'Standard ConverSight Notebook'** and the **'Machine Learning Notebook.'**

The **'Standard ConverSight Notebook'** contains essential packages for executing tasks, flows, CS Apps and performing fundamental analytical operations.

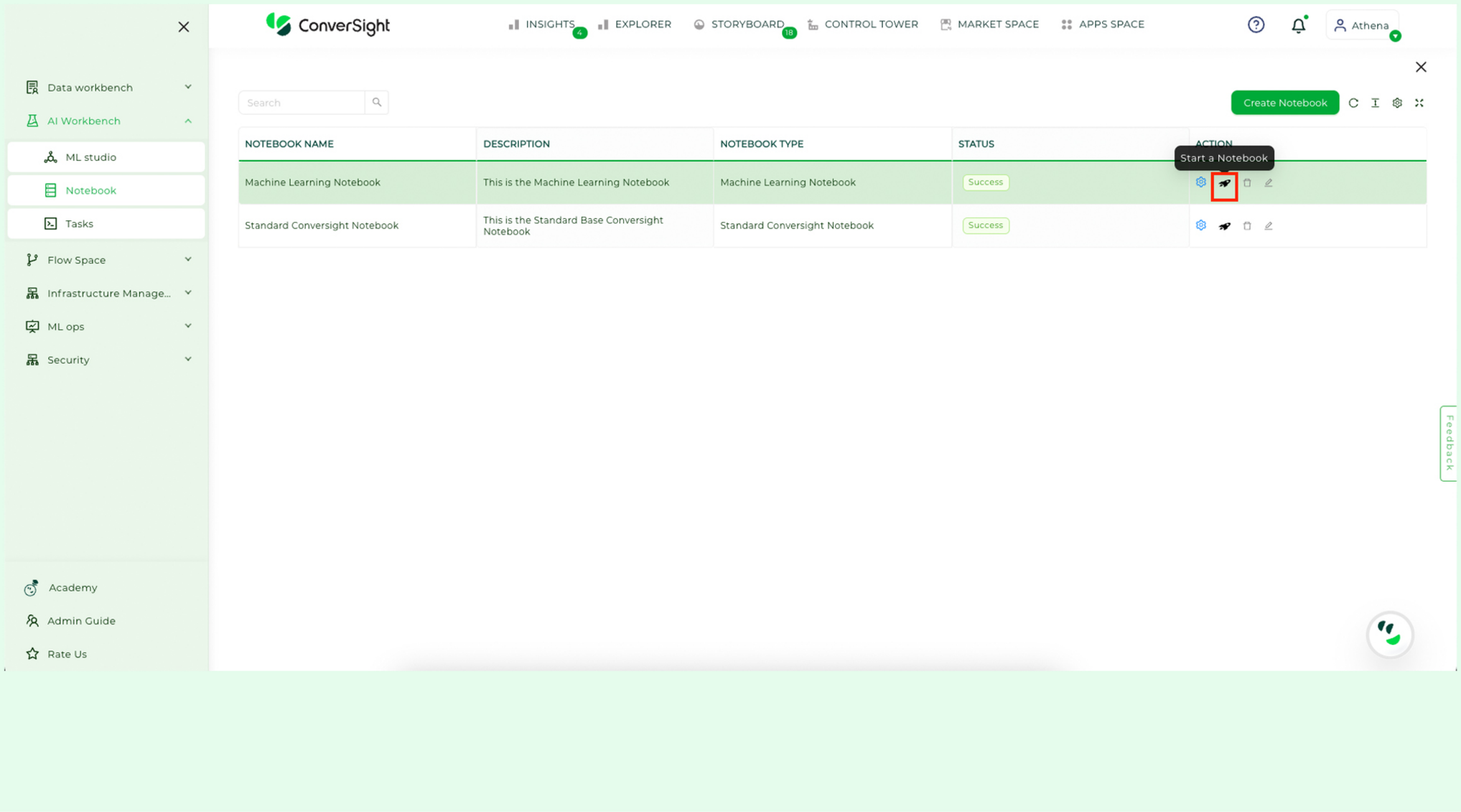
The **'Machine Learning Notebook'** is tailored to meet the needs of users requiring advanced analytics and machine learning capabilities, providing a comprehensive set of packages.

Users can customize notebooks within ConverSight by adding either custom or ConverSight verified packages to the Standard ConverSight Notebook or Machine Learning Notebooks, resulting in the extension of the Base Notebooks.

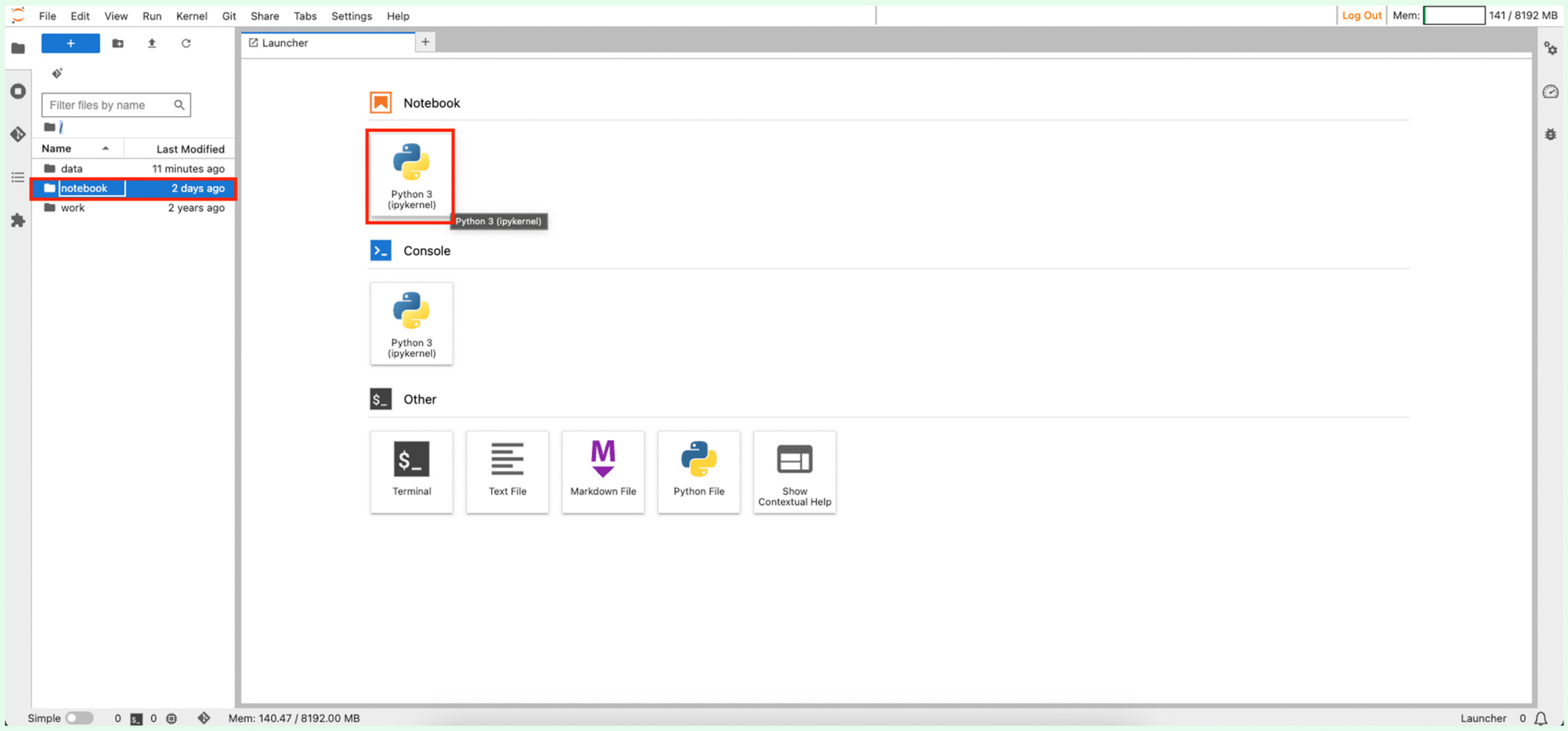
Note

Editing or deleting the default Standard ConverSight Notebook and Machine Learning Notebook is restricted.

To initiate the Notebook, you can simply click on the **‘Rocket’** icon located in the **‘Action’** field of the associated Notebook.



To begin a new workflow, users need to access the designated **‘notebook’** folder and initiate a workflow by choosing a preferred kernel. The Notebook functions within a Python environment, equipped with pre-installed packages to ensure smooth execution.



Note

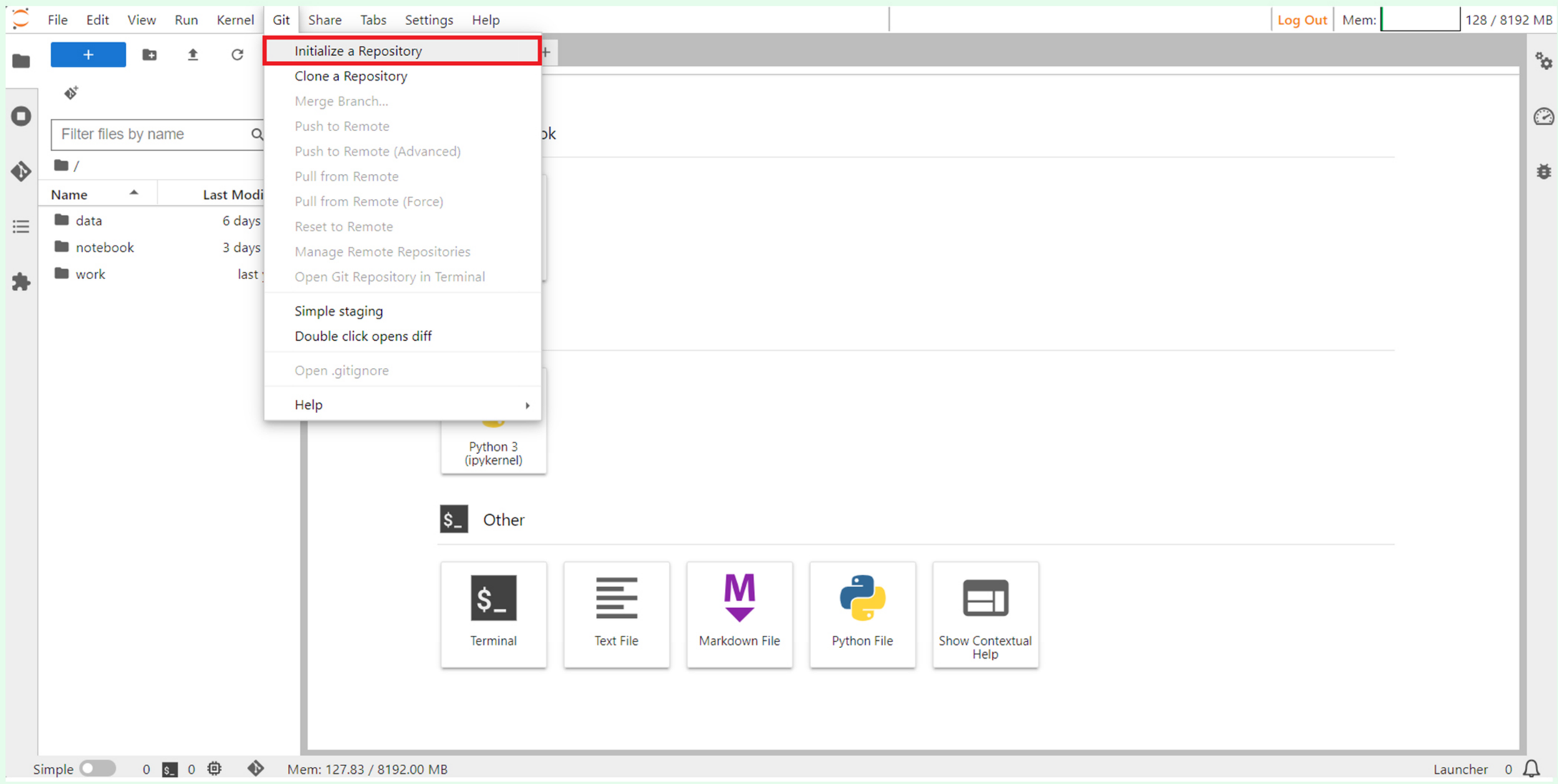
The authentication token in the Notebook expires after 4 hours. Nevertheless, the installed packages remain usable until the server is stopped or the Notebook is closed by the user.

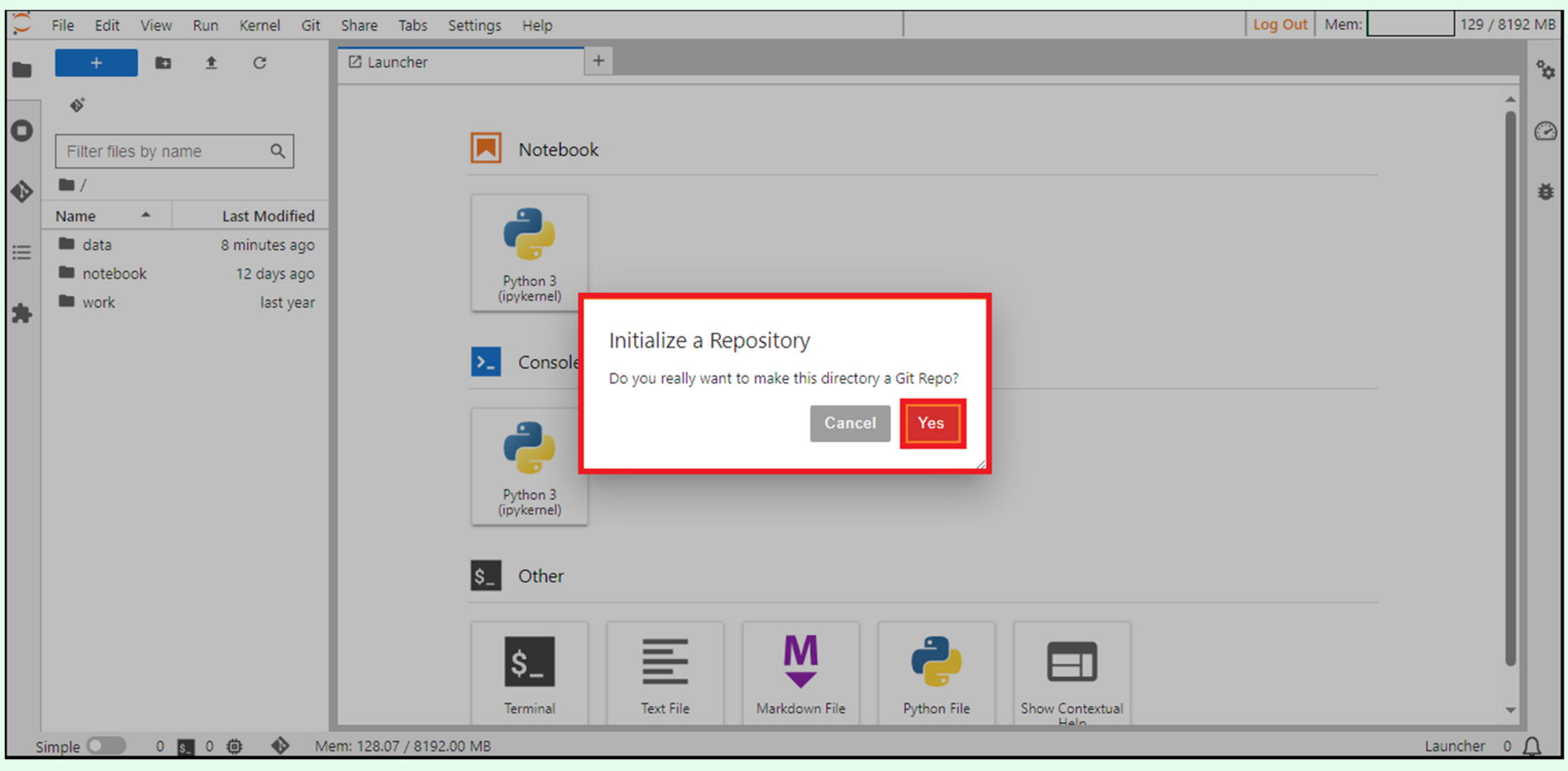
3. Repository Management

3.1 Initializing a Repository

Initializing a repository is the process of preparing a folder or directory to use Git's version control system. This step establishes the necessary structures and metadata within the selected directory, enabling Git to track changes made to files within it. Once initialized, the repository serves as a central hub for managing project versions, facilitating collaboration among team members and preserving the history of file modifications.

To initialize a repository within ConverSight Notebooks, choose the **'Initialize a Repository'** option from the **'Git'** menu. A prompt will then ask if you want to designate this directory as a Git repository. Click **'Yes'** to proceed.



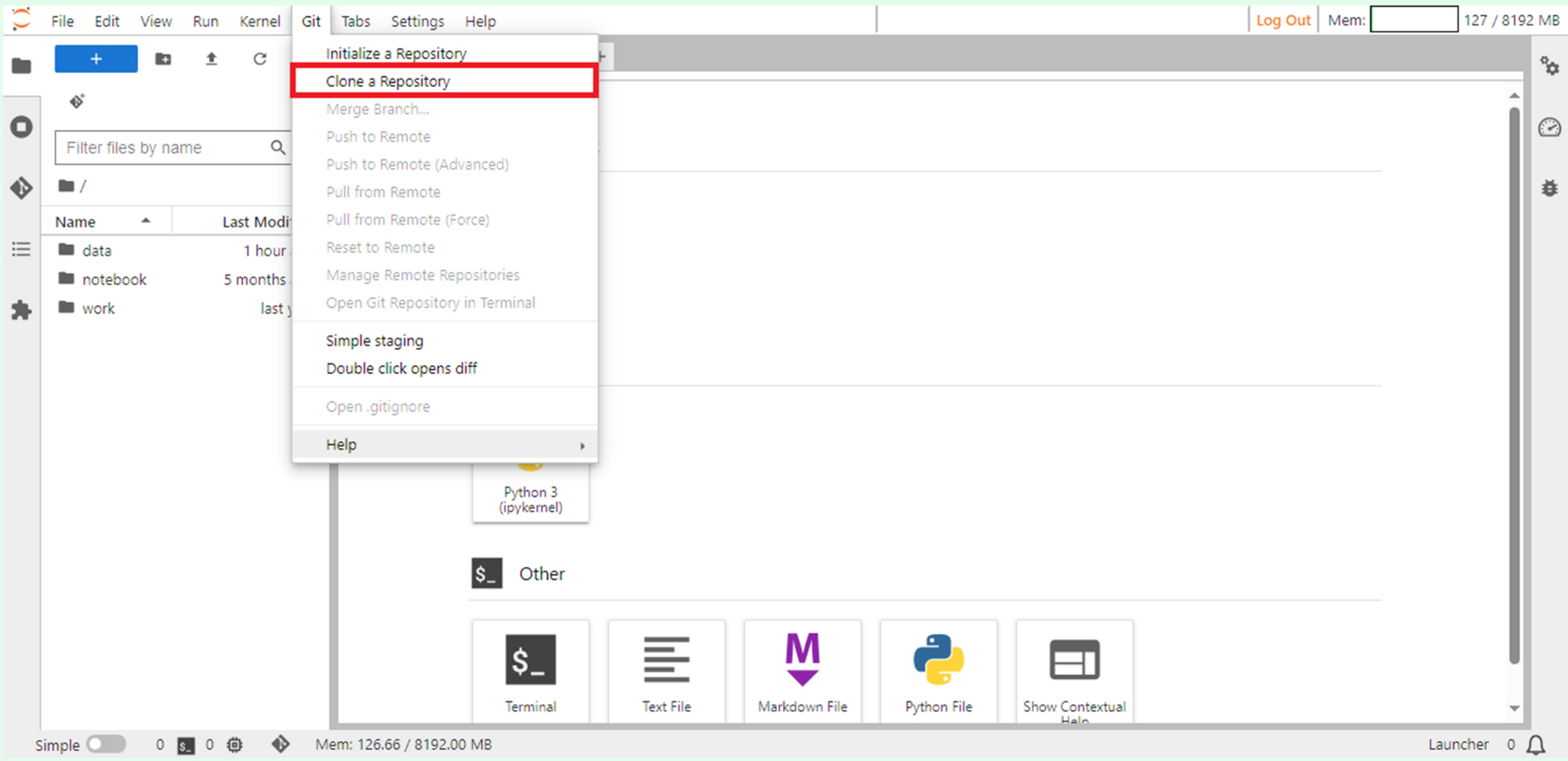


Once initialized, the directory serves as a local repository, granting you access to view the repository's current state, current branch, manage changes, make commits and explore the commit history. This foundational step sets the stage for efficient version control and collaborative development within ConverSight Notebooks.

3.2 Cloning a Repository

Cloning a repository in Git within JupyterLab in the ConverSight platform involves duplicating a remote repository onto your local machine. This process enables users to work on the repository's files independently, without altering the original version.

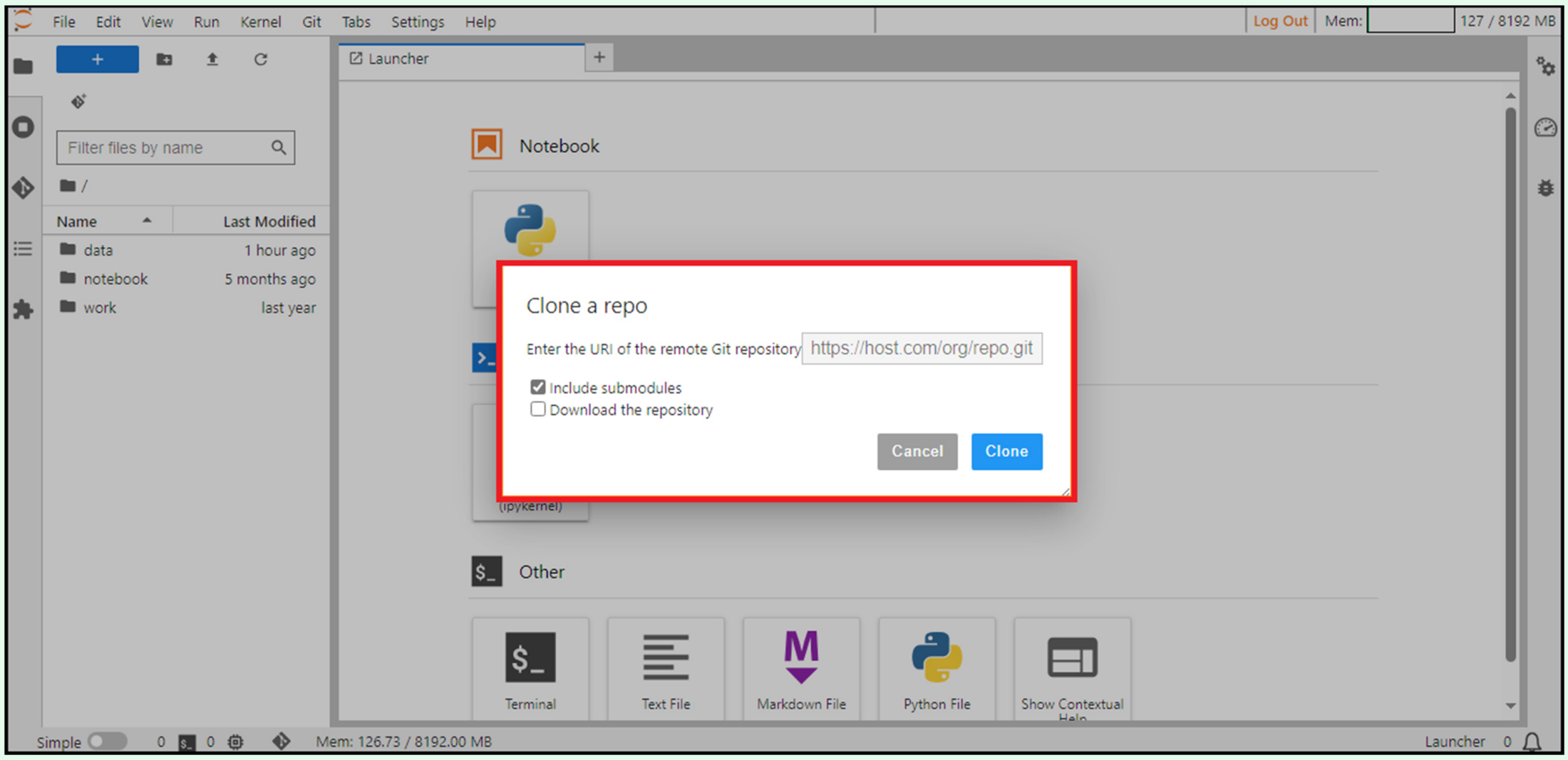
In ConverSight, users can clone any Git repository into Jupyter Notebook by selecting the **'Clone a Repository'** option from the **'Git'** menu. Once cloned, you can access and edit the repository's contents locally, facilitating collaborative development and version tracking.



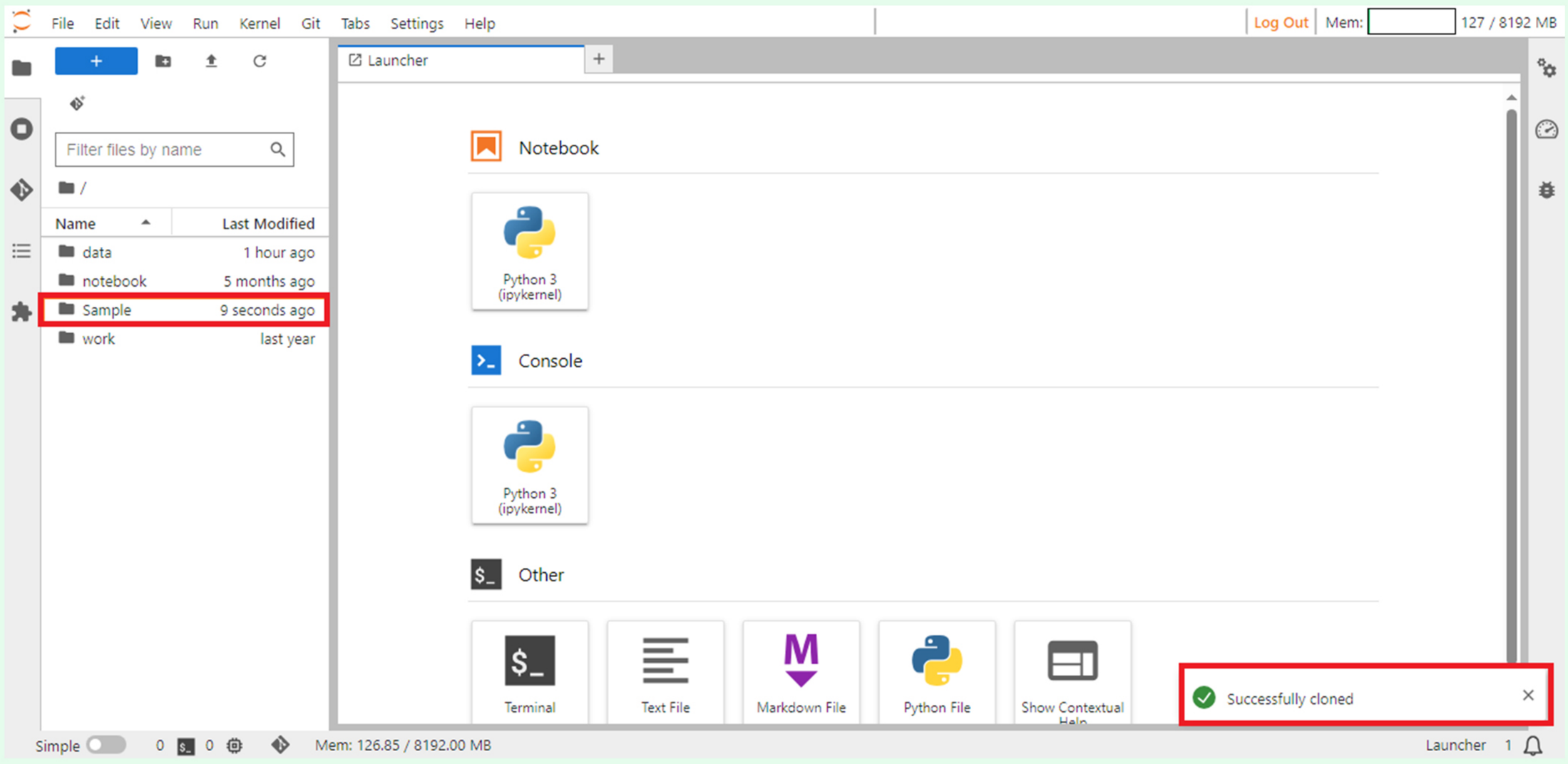
Choosing the **'Clone a Repository'** option will lead to a page where users need to input the required information to start the cloning process.

ARGUMENT	DESCRIPTION
URL	Enter the URL of the remote Git repository that needs to be cloned.
Include Submodules	A Git submodule is an embedded repository within another Git repository. Select the checkbox if you want to include them.
Download Repository	Selecting this checkbox allows you to download an entire repository.

Once you have entered all the details click on the **‘Clone’** button.



A popup notification will appear, indicating that the cloning process was successful. You will observe the addition of a new folder (in this instance, **Sample**).

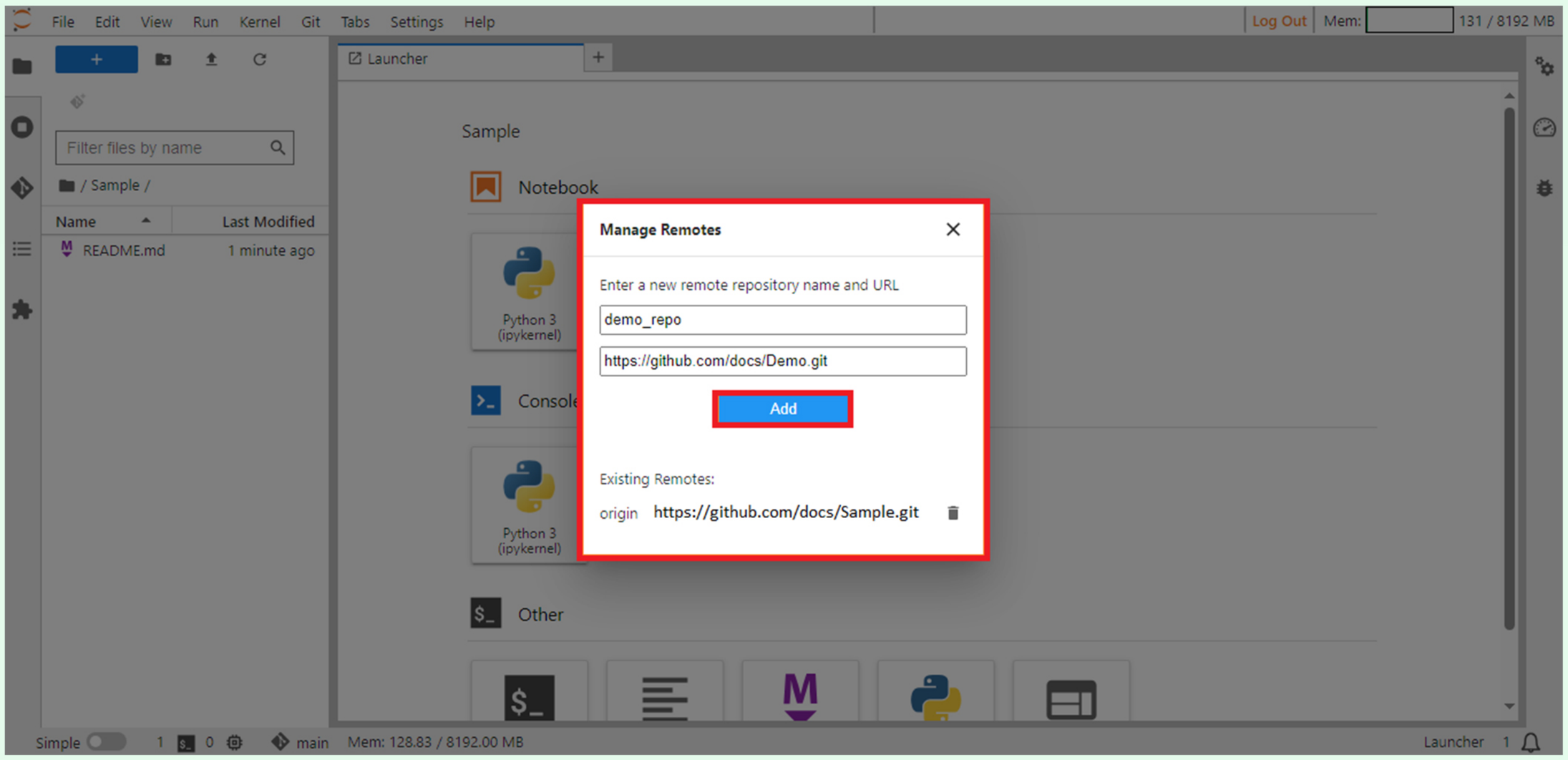


3.3 Managing Remote Repositories

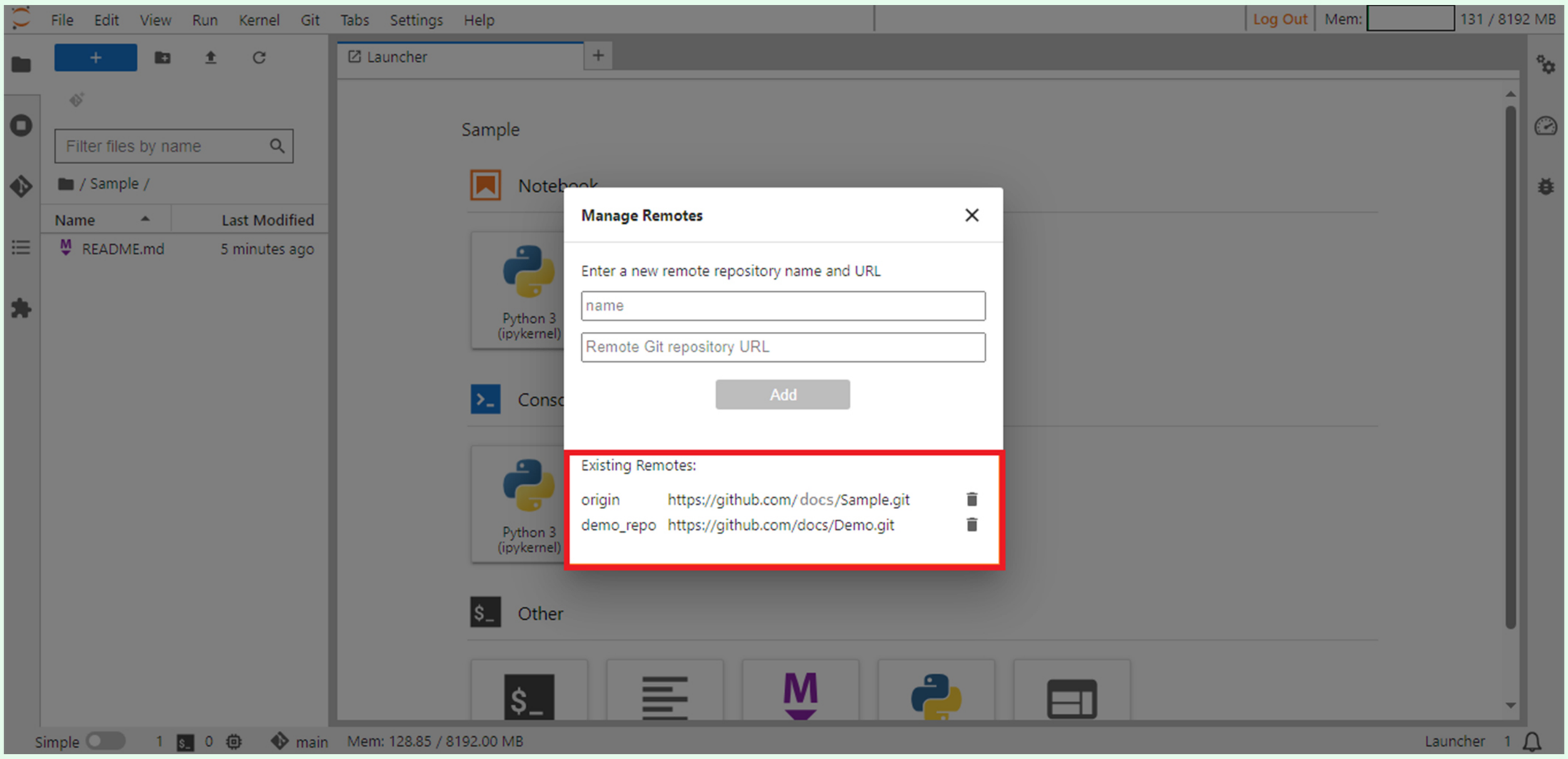
Within ConverSight Notebooks, users can efficiently manage their remote repositories through the intuitive interface. To add a new repository, navigate to the **'Manage Remote Repositories'** option under the **'Git'** menu.

Adding a Repository

To add a new repository, provide the name and the URL of the remote Git repository and click **'Add'** to complete the process.

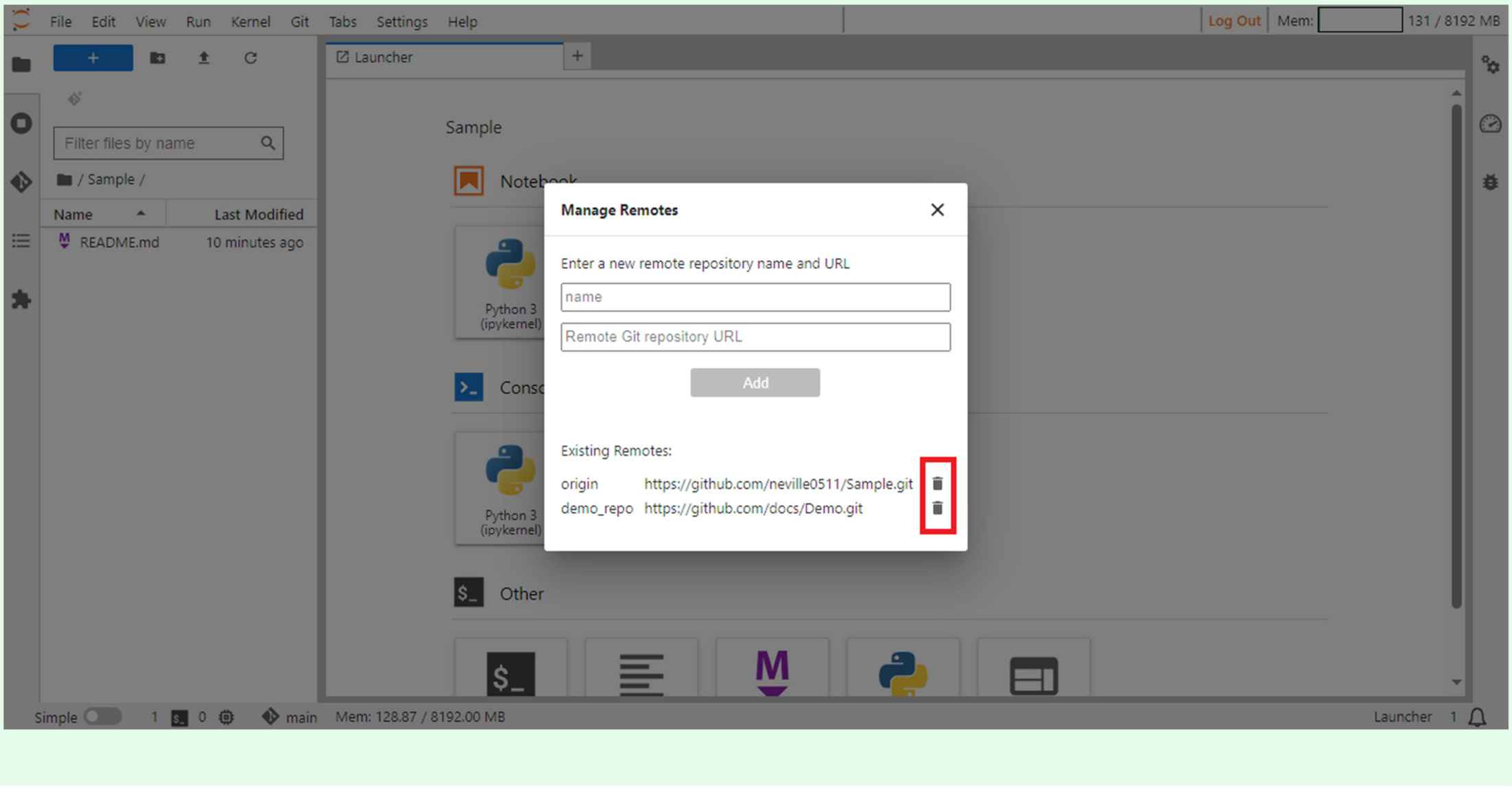


The newly added repositories will be displayed as shown.



Deleting a Repository

To remove an existing repository, click on the **'Delete'** icon next to the respective repository.



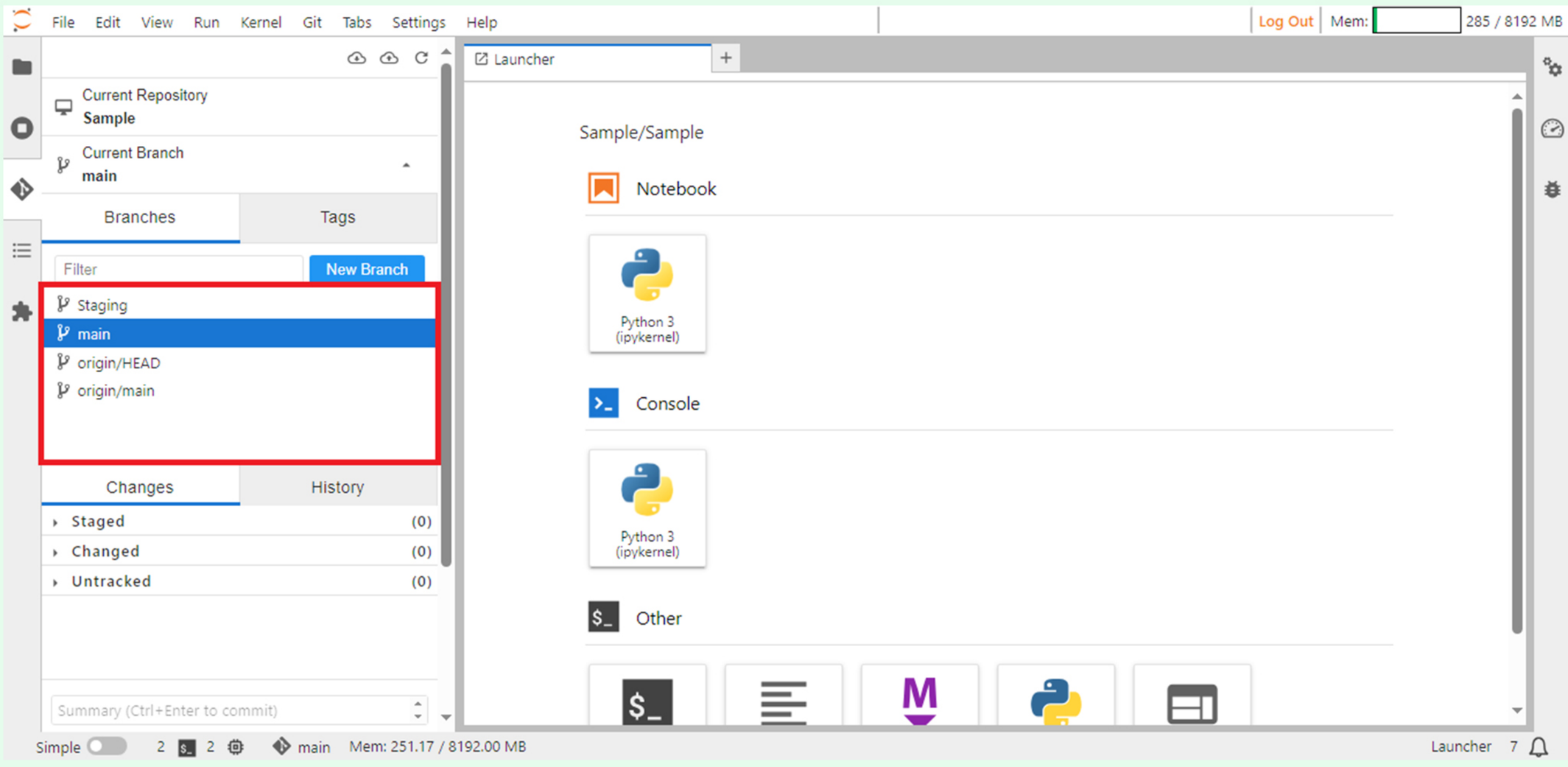
4. Branch Operations

4.1 Merging Branches

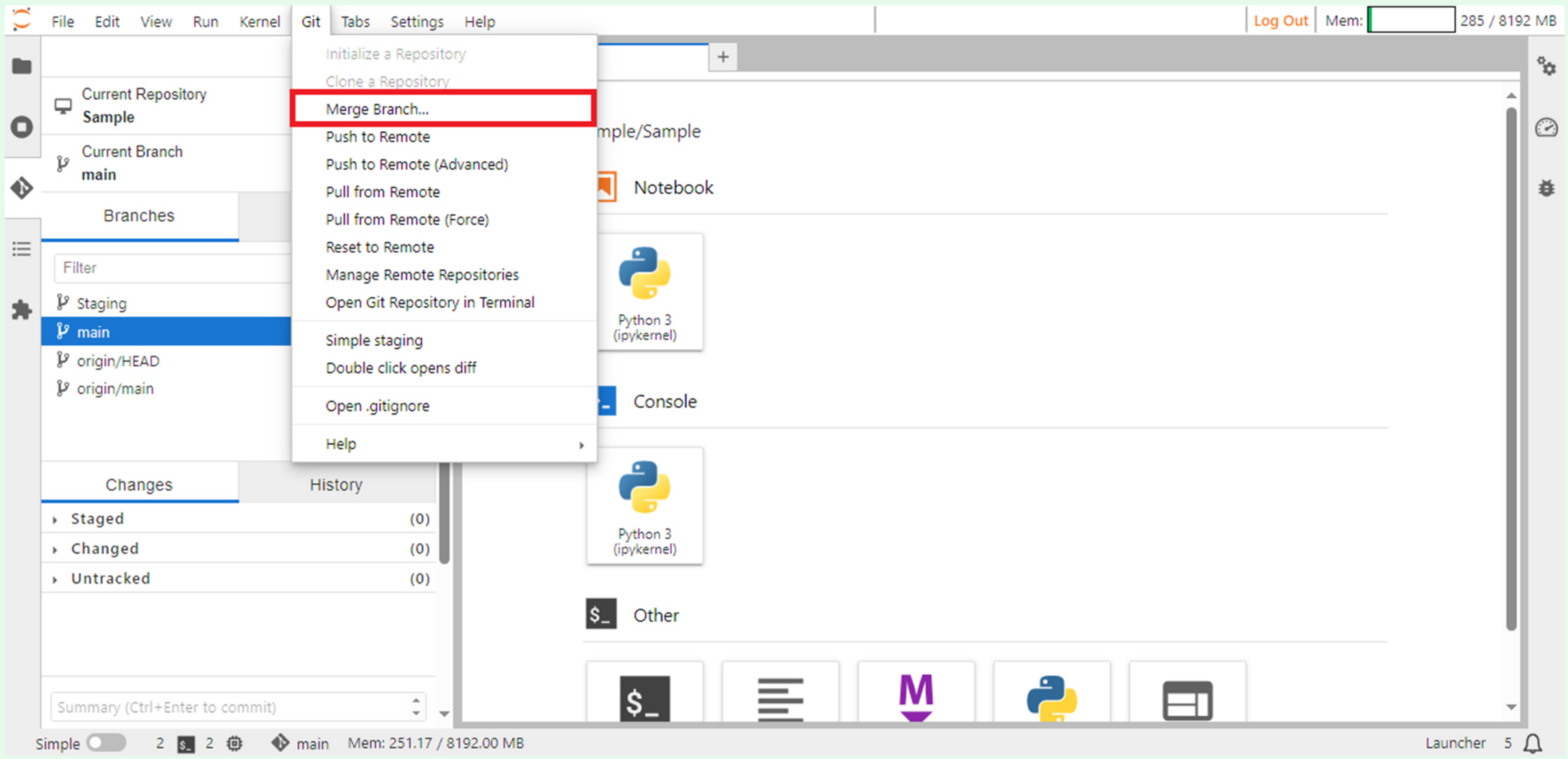
The **‘Merge Branch’** refers to the process of integrating the changes made in one branch of a Git repository with another branch. It serves to integrate work developed separately, often by multiple collaborators, into a unified codebase.

The following are the steps to merge branches:

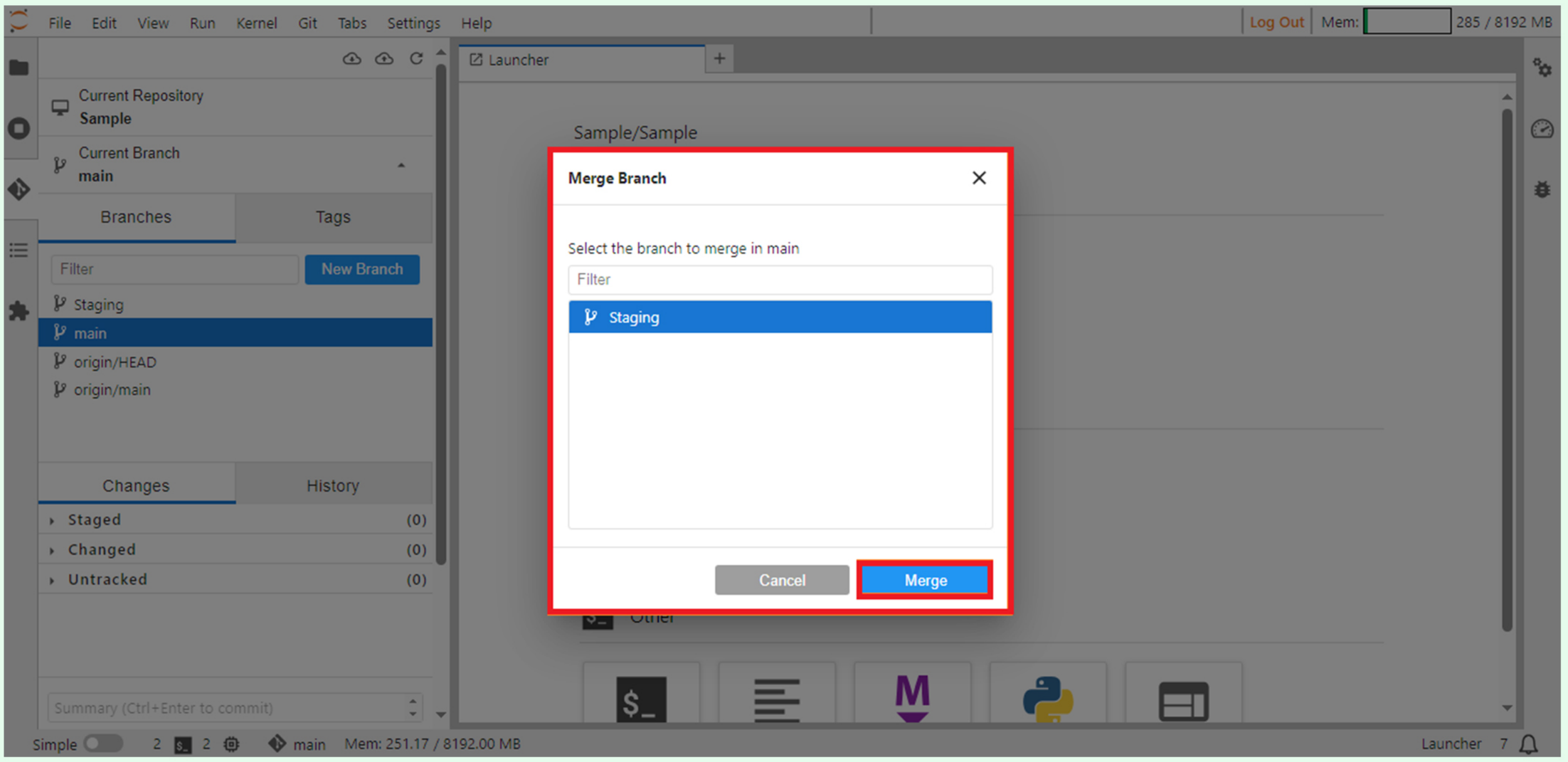
- Start by switching to the target branch.
- From the Git panel, select the branch where you want to merge the changes by clicking its name.



- Access the **‘Git’** menu in the menu bar and choose the **‘Merge Branch’** option.



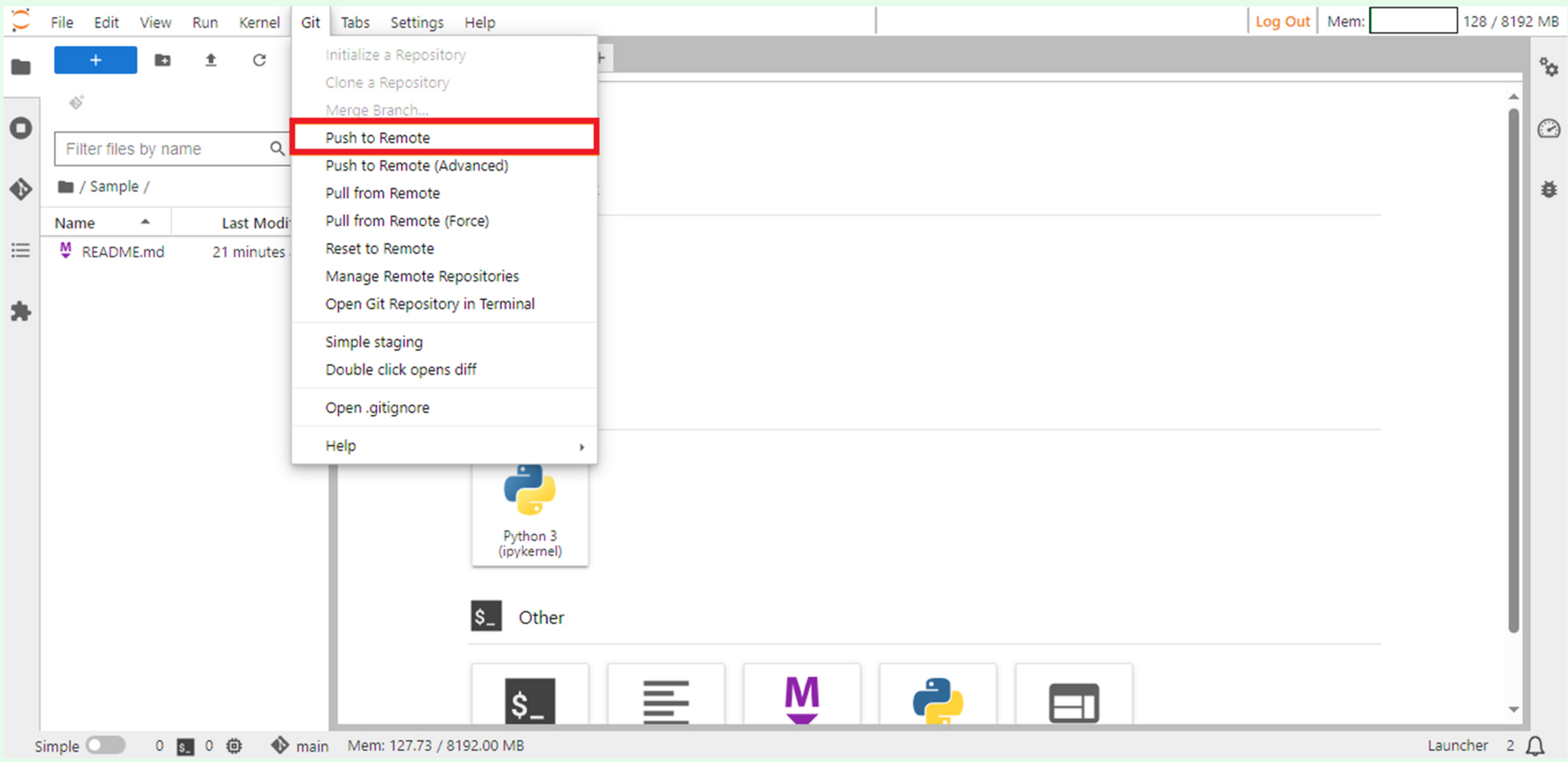
A dialog will appear, asking you to select the branch you want to merge with. Choose the branch containing the changes you wish to integrate and click the **'Merge'** button to initiate the process.



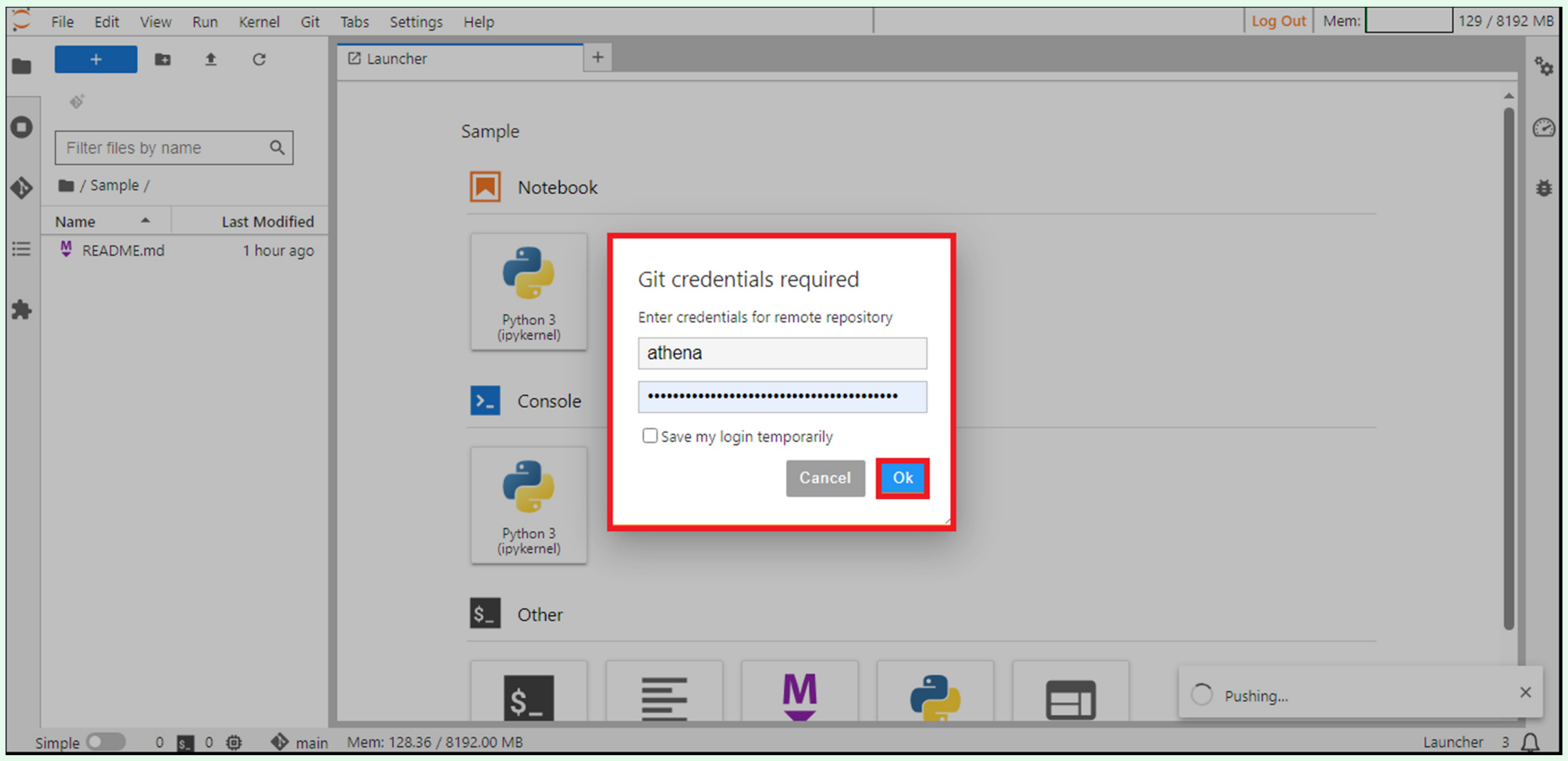
If conflicts arise between branches, ConverSight Notebooks will highlight them in the **'Conflicted Files'** section of the **'Git'** menu. Resolve conflicts manually by editing the affected files, then mark each conflict as resolved. Upon resolution (or if there are no conflicts), finalize the merge by clicking **'Merge'**. This streamlined process ensures smooth collaboration and efficient code integration within your projects.

4.2 Pushing Changes to Remote

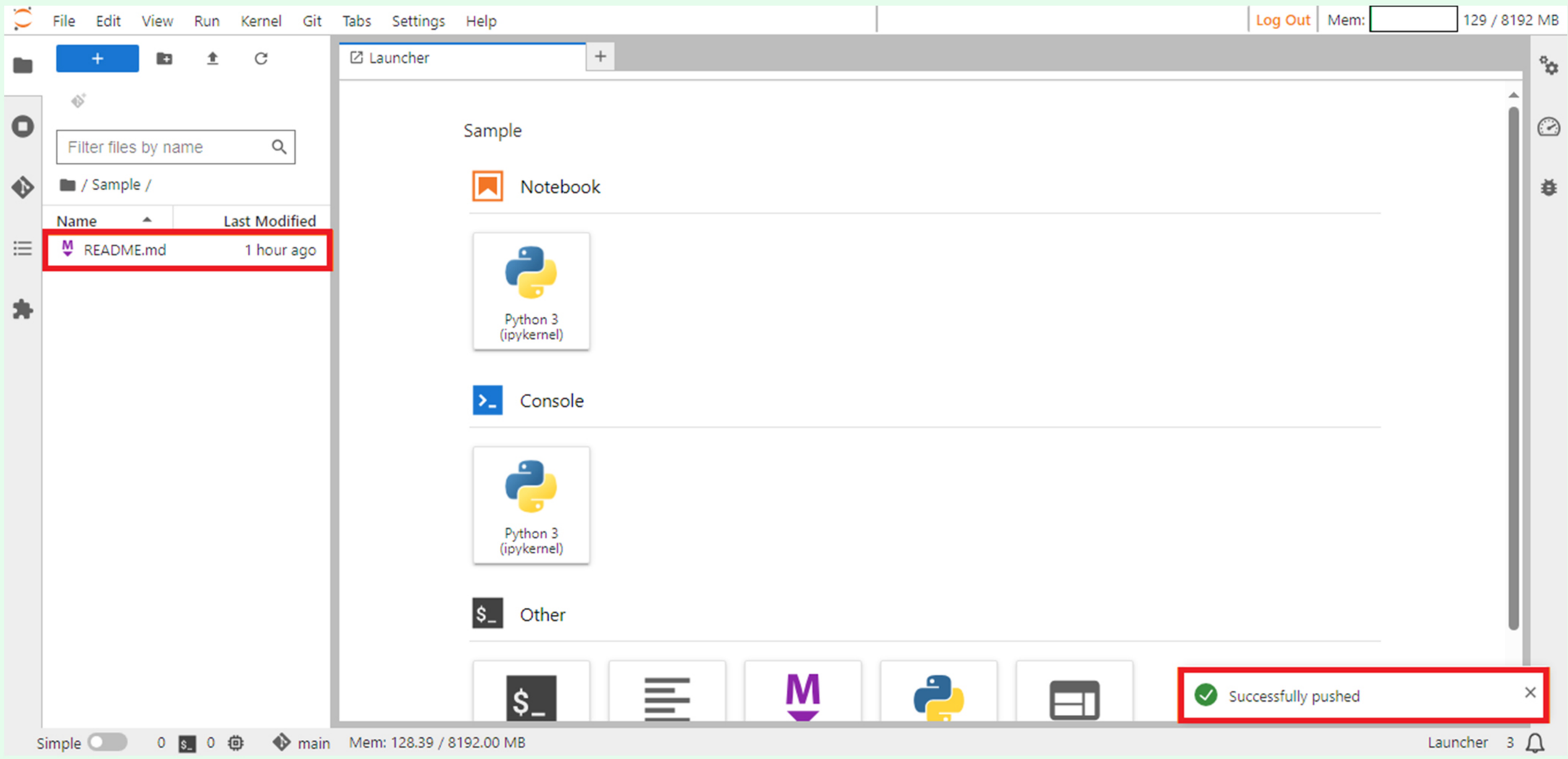
The term **‘Push to Remote’** denotes the process of transmitting the alterations made in a local Git repository to a remote repository. When modifications are made locally (such as creating new commits), it may be necessary to distribute these alterations to others or update the project's version stored on the remote server. To accomplish this, utilize the **'Push to Remote'** feature located within the **‘Git’** menu.



Select the **‘Push to Remote’** option and a popup will prompt you to provide your Git credentials. Complete all the required details and click **‘Ok’** to proceed with the operation.

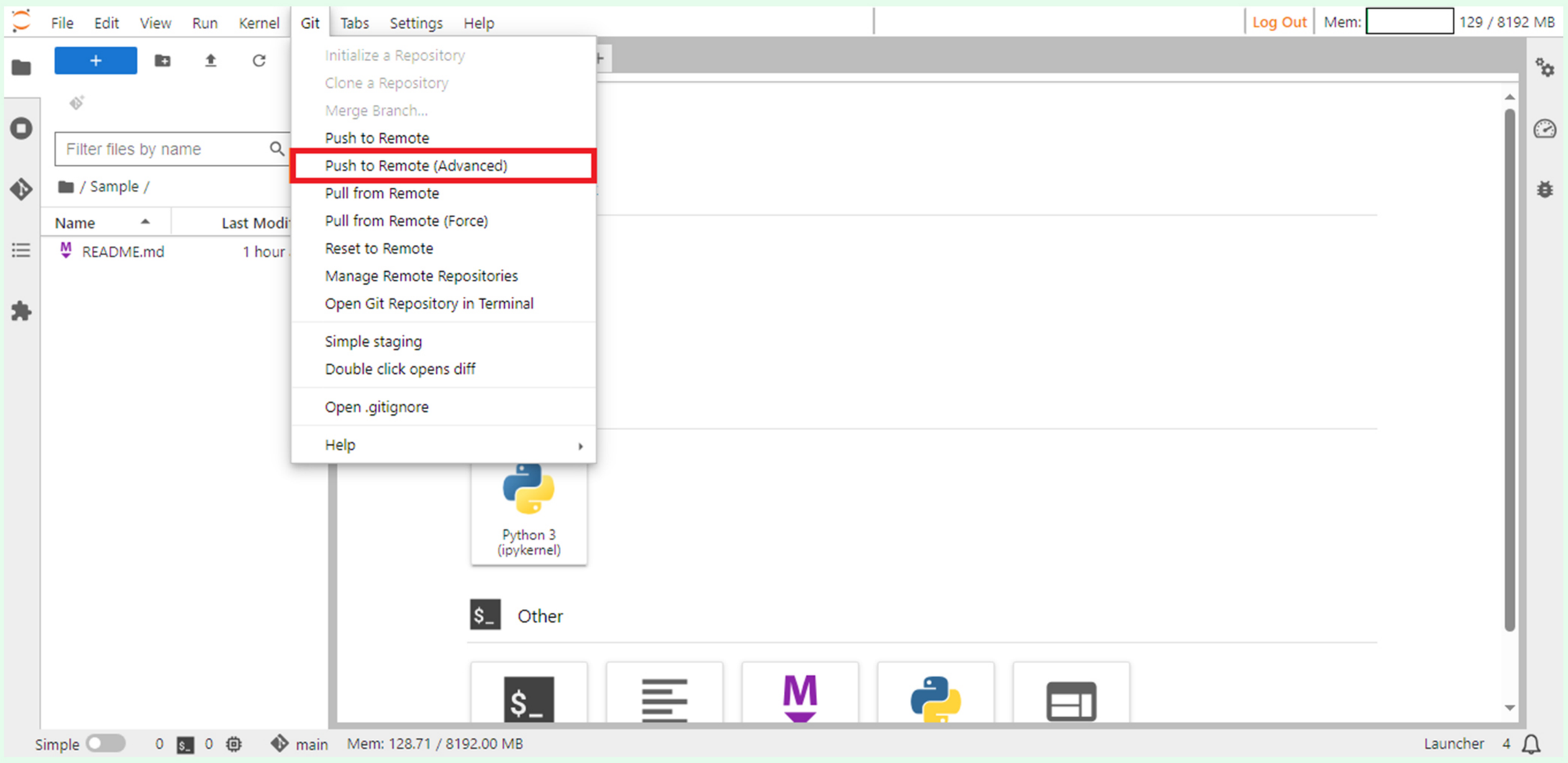


A popup notification will appear, indicating that the **‘Push to Remote’** was successful.

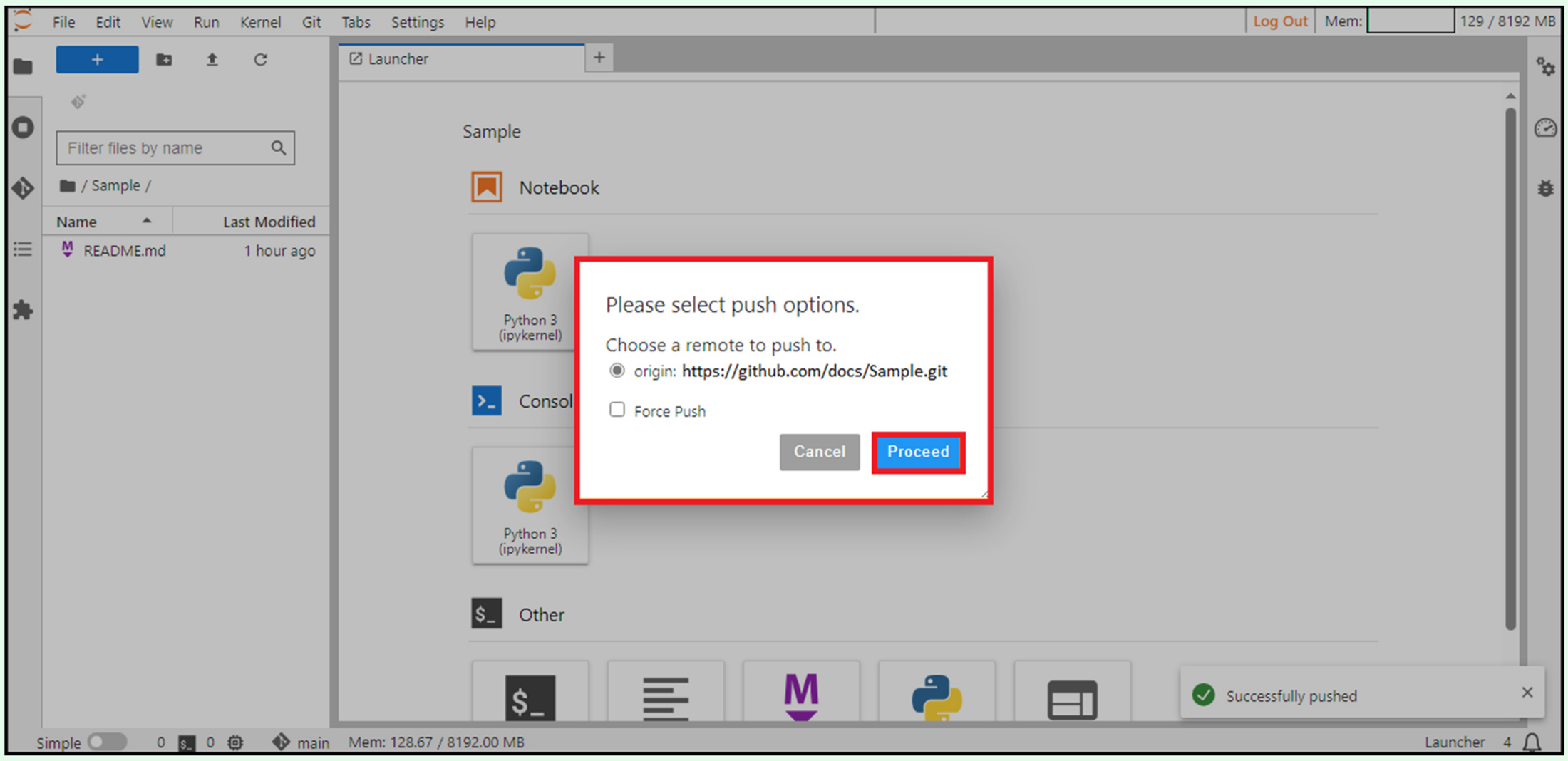


Push to Remote (Advanced)

Within the Git menu, the **‘Push to Remote (Advanced)’** feature allows you to precisely designate the remote repository where you wish to publish the changes.



The **‘Force Push’** option allows you to forcefully upload your changes to the online repository, even when conflicting changes are present. This option is useful when updating the online version with your local modifications, even if they are not identical. Once ready, proceed to push your changes.

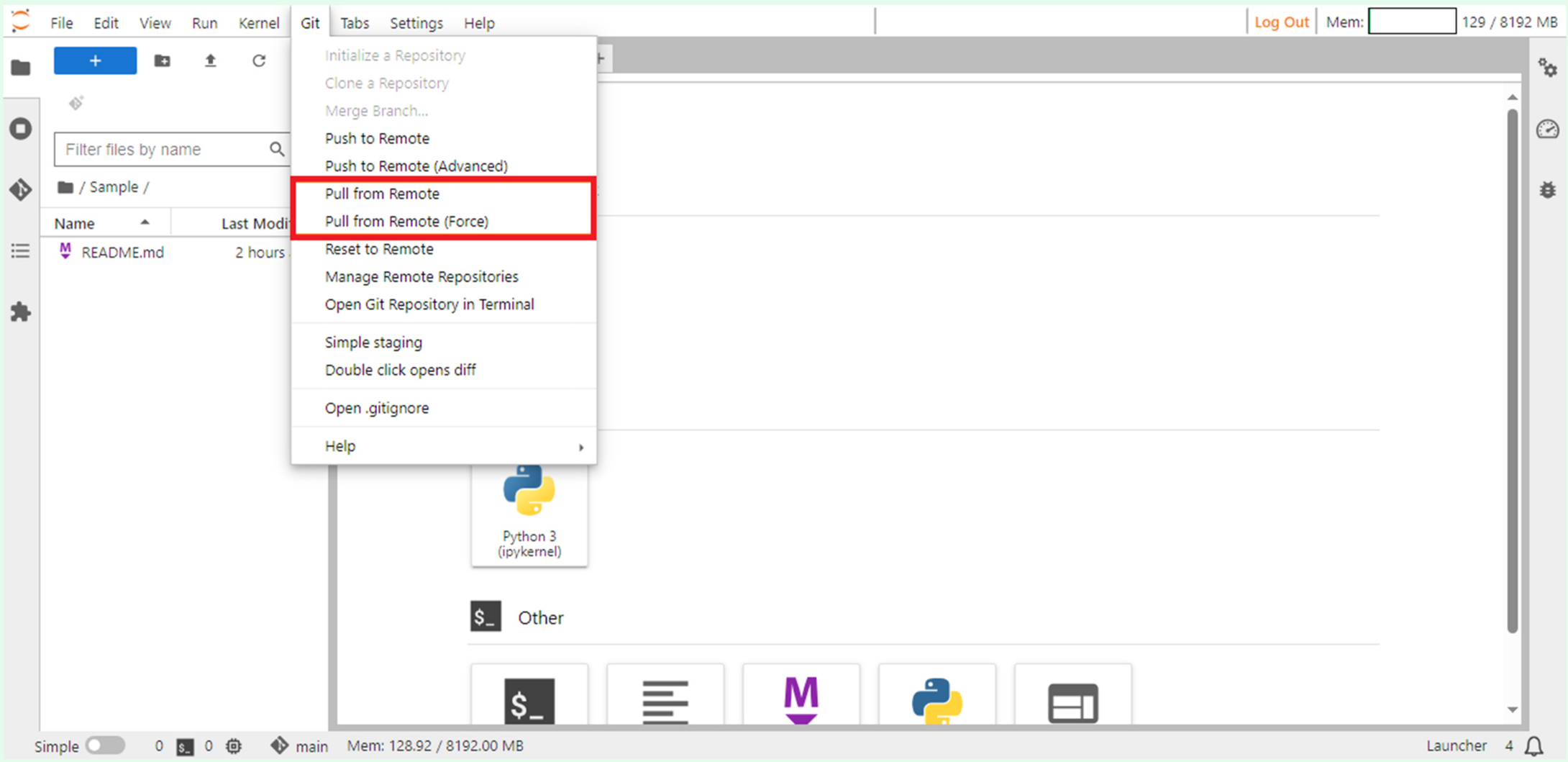


4.3 Pulling Changes from Remote

The **'Pull from Remote'** function in the Git menu enables you to incorporate changes from a remote repository into your local branch. This process comprises two main steps:

1. Fetching Changes: Initially, the **'Pull from Remote'** option retrieves updates from the remote repository, including any new commits or alterations made by others on the remote branch.

2. Merging Changes: Subsequently, Git automatically merges these fetched changes into your local branch. In instances of conflicts, manual resolution may be required.



Pull from Remote (Force)

The **'Pull from Remote (Force)'** option will reset your local branch to match the remote branch. This discards all your local changes.

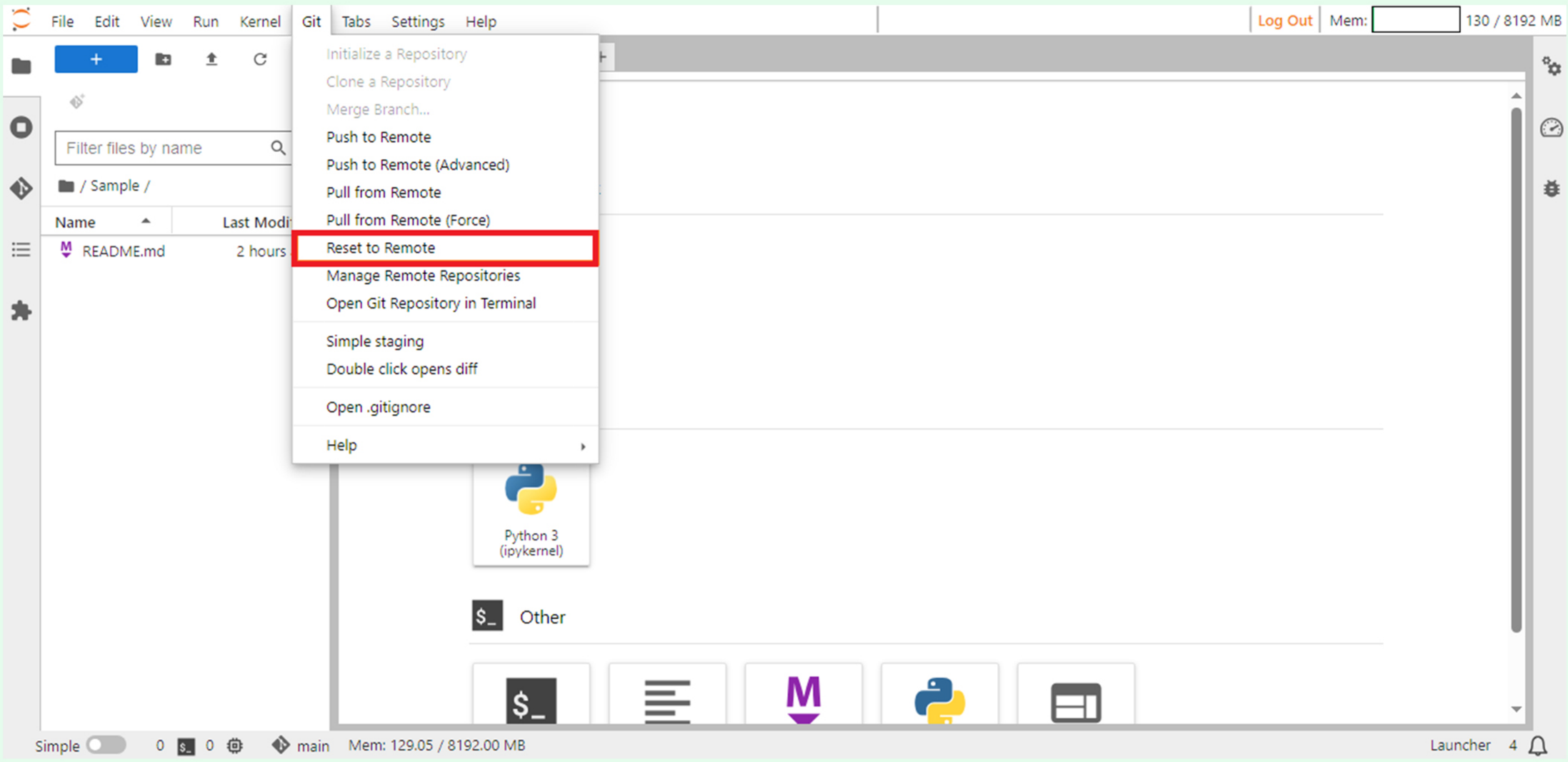
5. Advanced Git Operations

5.1 Resetting to Remote

Resetting to remote within Jupyter Notebook in ConverSight involves reverting the current state of a local branch to match the state of a remote branch. This action effectively discards any local changes that have not been pushed to the remote repository and brings the local branch in line with the remote branch.

It can be useful in scenarios where the local branch has diverged from the remote branch and you want to start fresh based on the remote state.

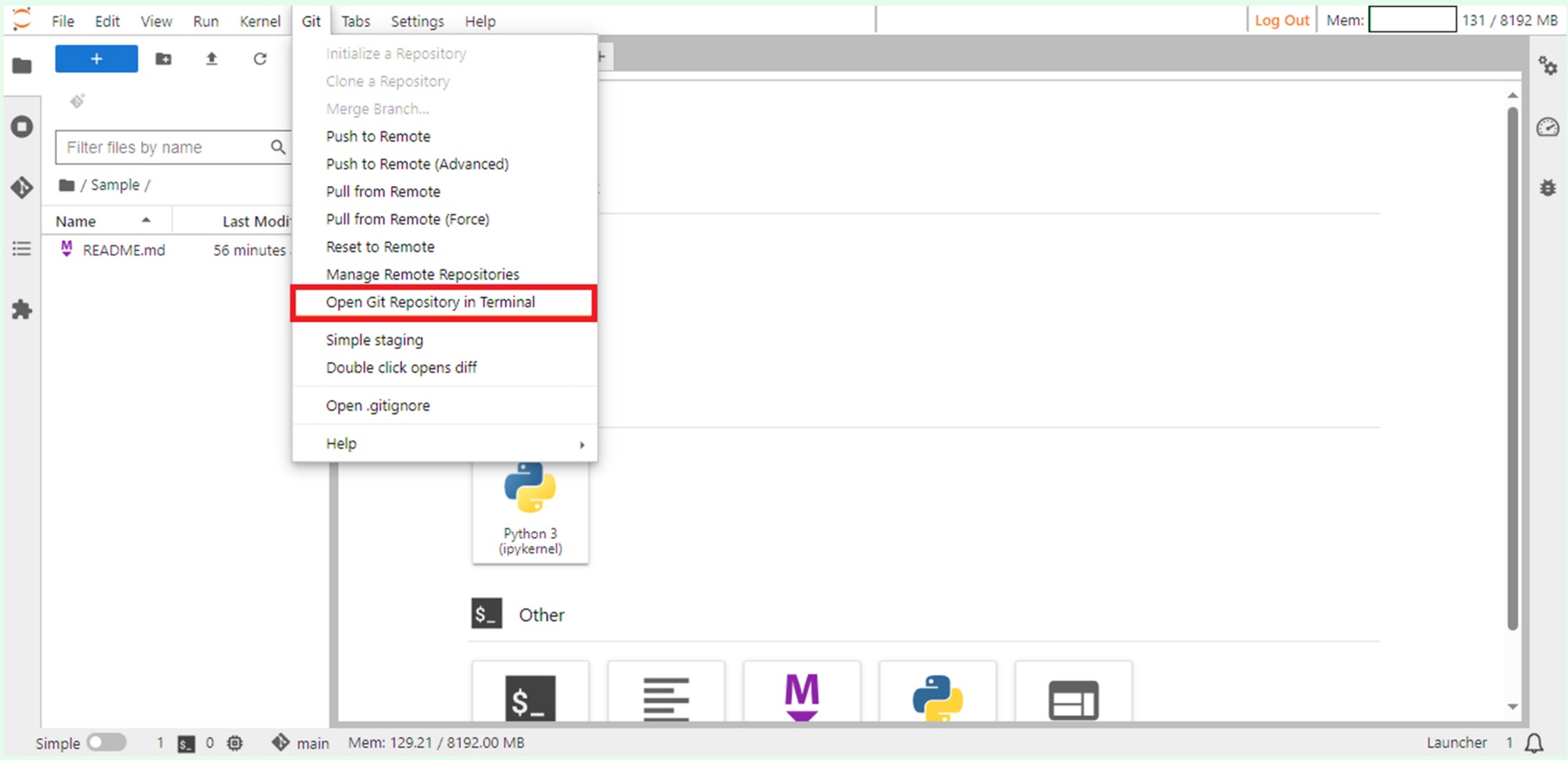
This process can be done by selecting the **'Reset to Remote'** option in the **'Git'** menu or by executing the appropriate Git commands in the terminal.



5.2 Opening Git Repository in Terminal

Opening a Git repository in the terminal involves accessing the command-line interface within Jupyter Notebook in ConverSight, with the working directory set to the location of the Git repository. This enables users to execute Git commands directly from the terminal window, facilitating tasks like committing changes, creating branches, merging branches and interacting with remote repositories. This functionality offers users enhanced flexibility and control over their Git workflow, particularly for tasks that are more efficiently performed via the command line.

To access the Git repository in a terminal, users should select the **'Open Git Repository in Terminal'** option from the **'Git'** menu.

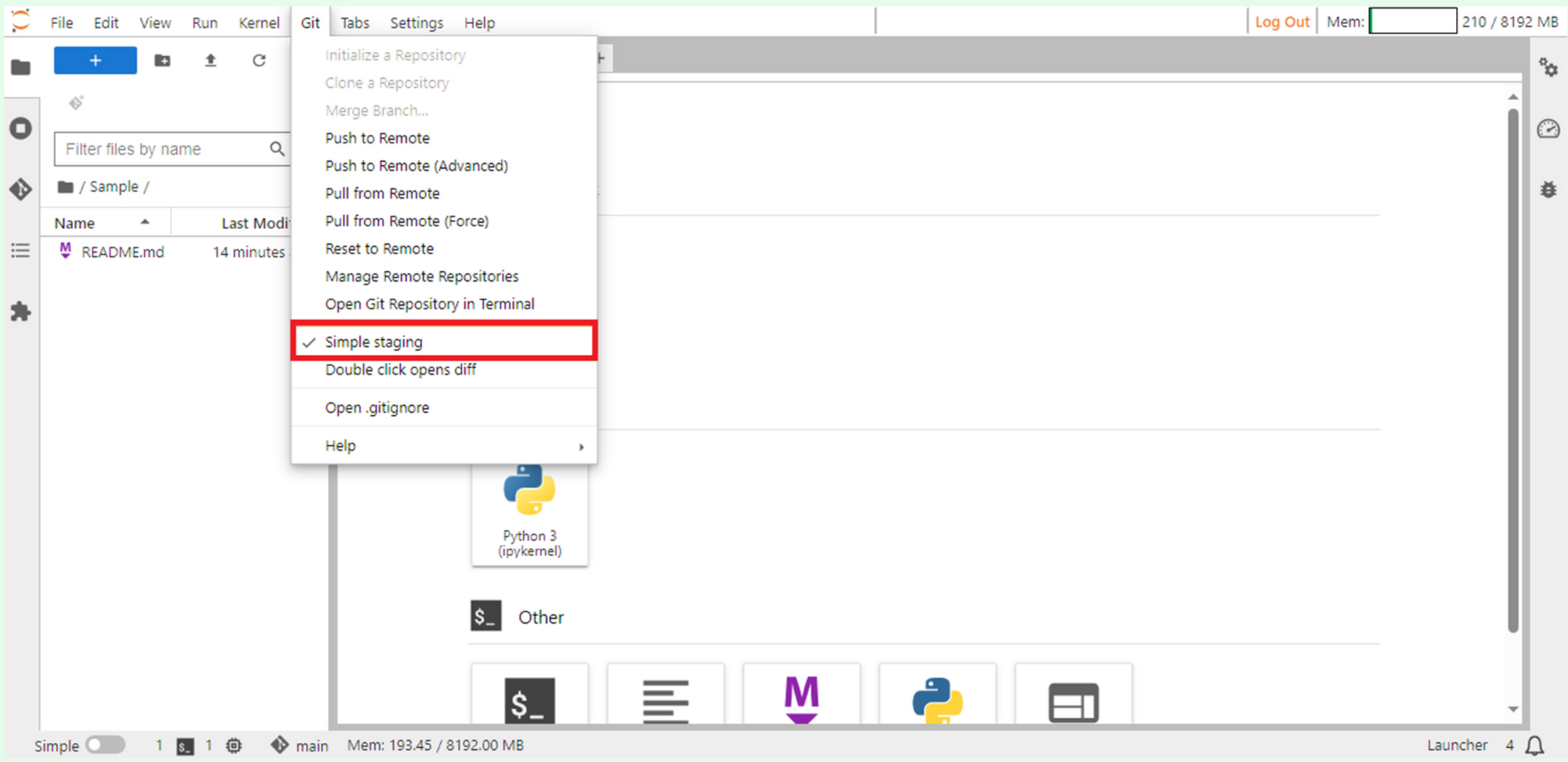


5.3 Simple Staging

'Simple Staging' is a feature designed to streamline the staging process within the graphical user interface. Within ConverSight, this feature automatically stages all files with changes, simplifying the process by eliminating the need to manually stage individual files using the '+' button. This streamlines the workflow, reducing clicks and manual overhead, particularly for users focused on committing all changes at once. By allowing users to selectively add changes to the staging area, it facilitates a clearer and more organized commit history, streamlining version control within the repository.

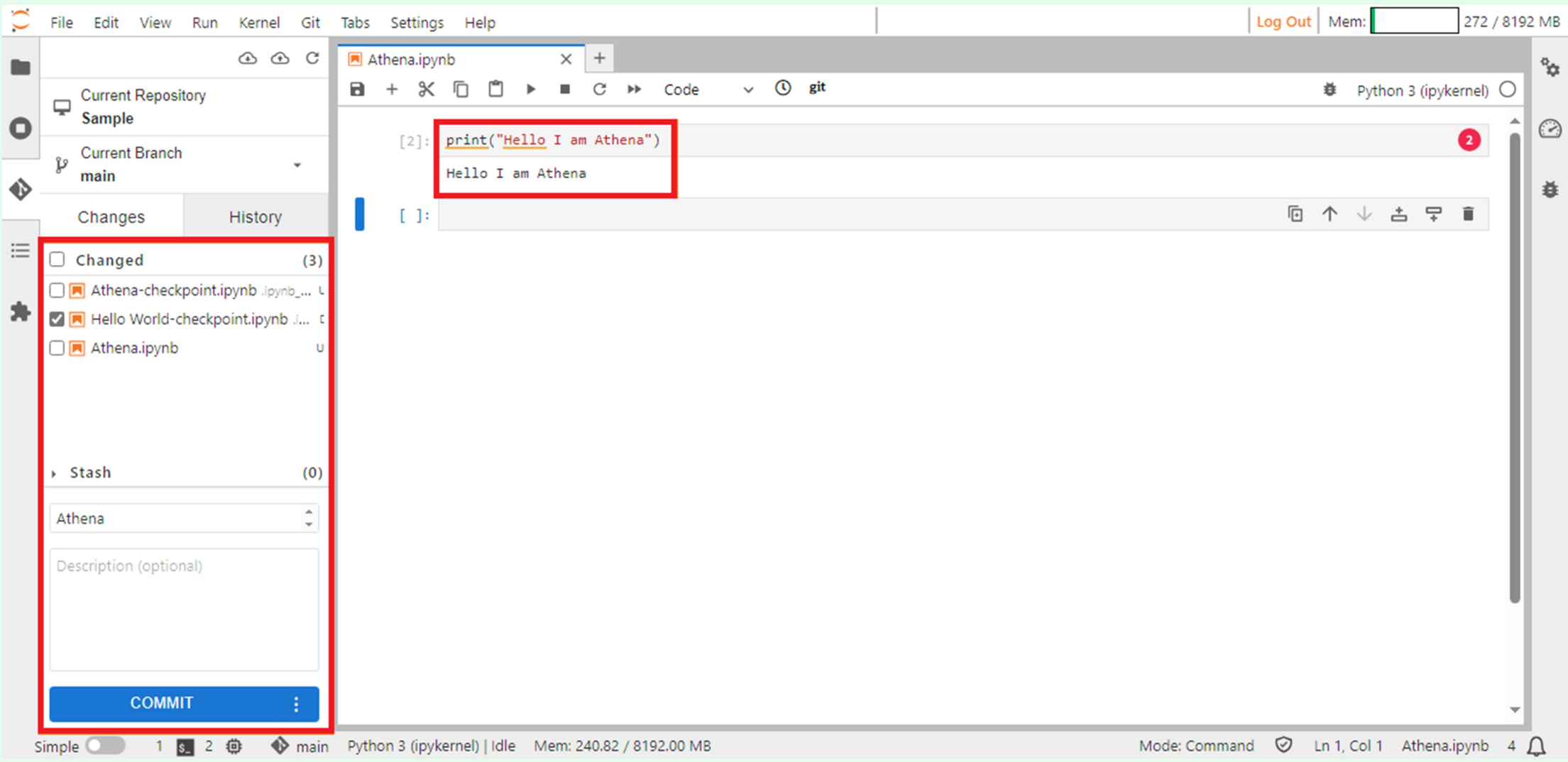
Here is how it works:

- Open the **'Git'** menu in JupyterLab.
- Enable the **'Simple Staging'** option by clicking on it.



- Make changes to files; the Git panel will automatically display all modified files as **'staged'**.

- Enter a commit message and click **'Commit'** to save the changes.

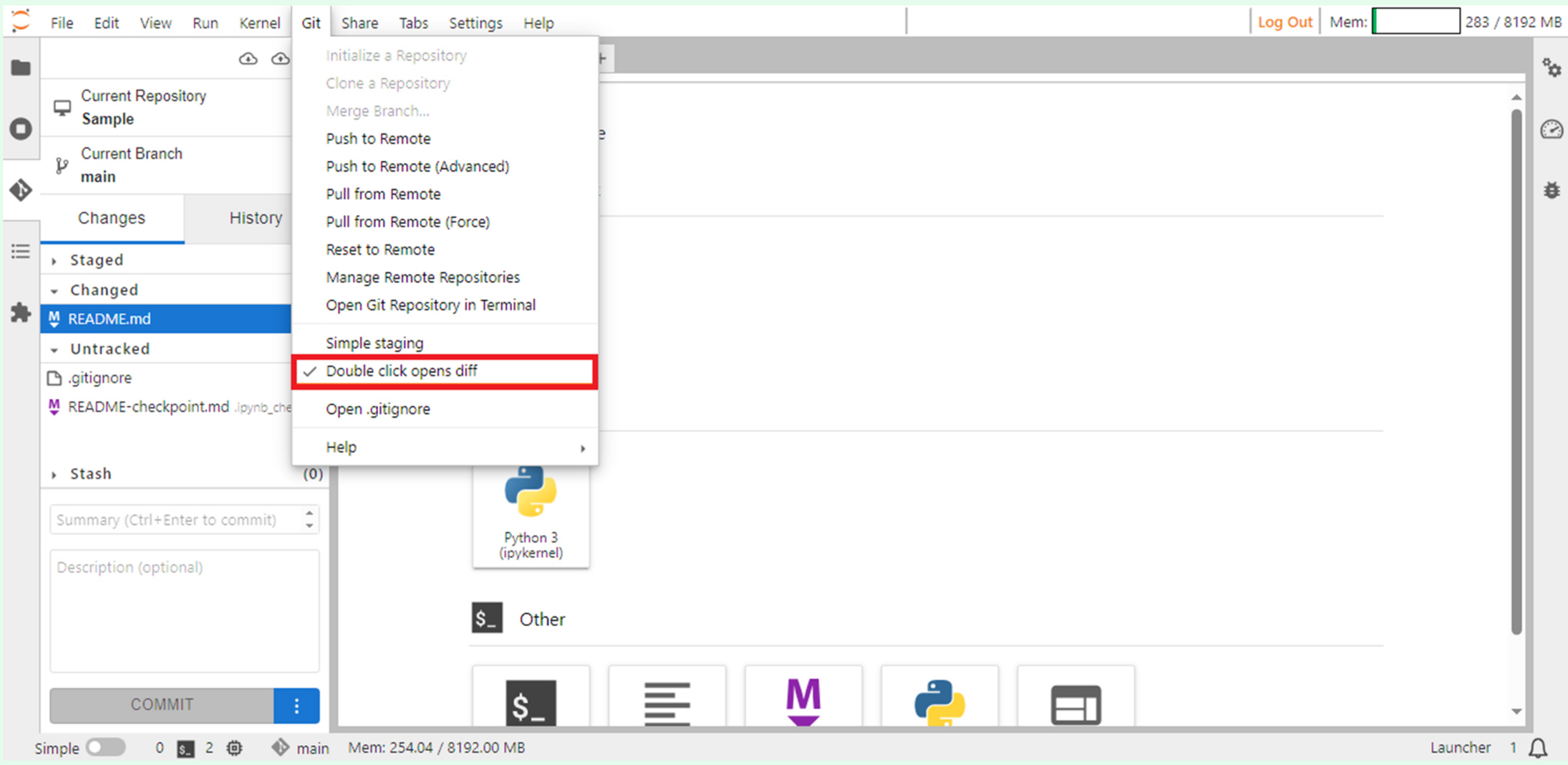


- All staged files will be included in the commit.

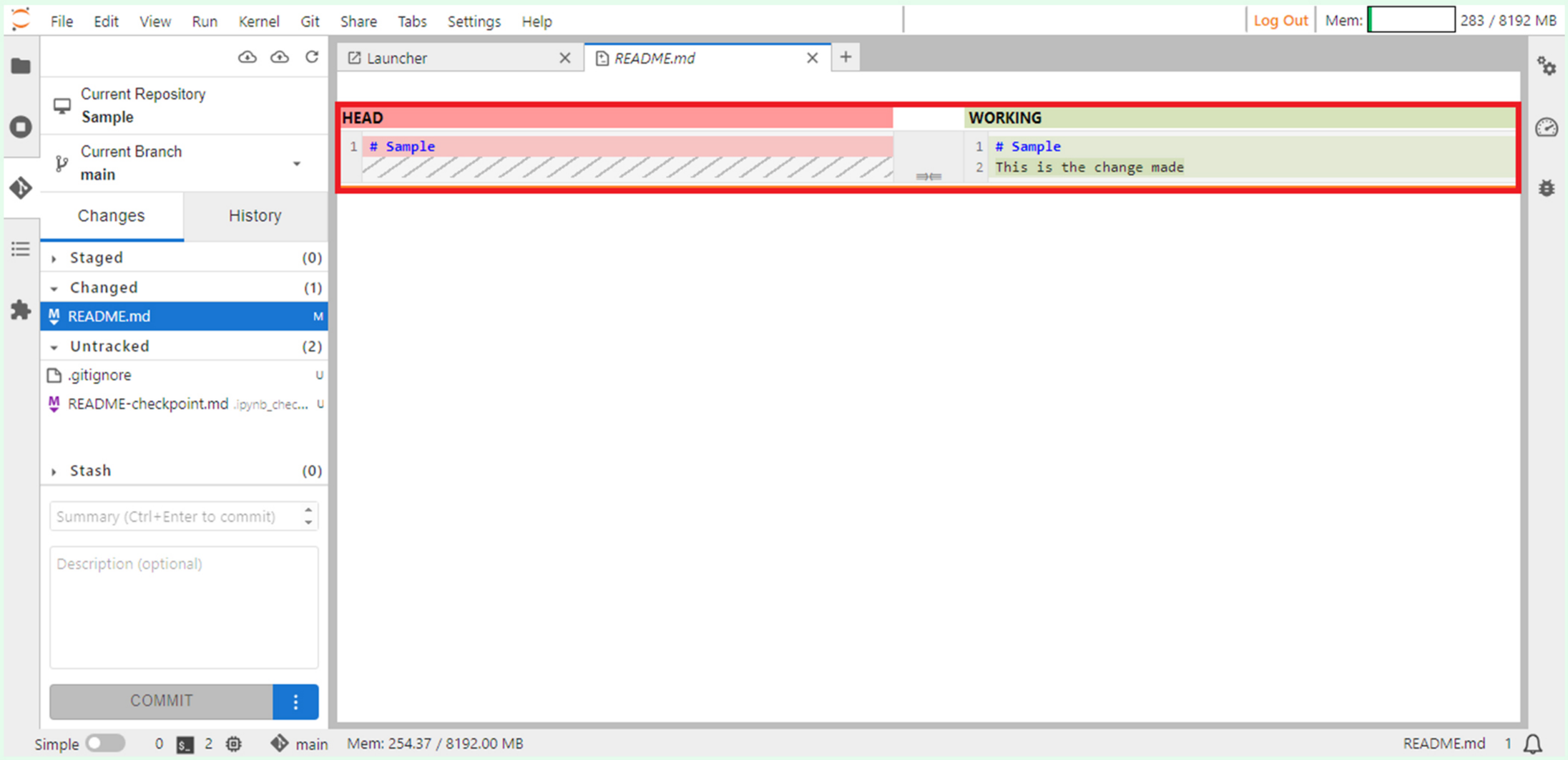
6. Visual Tools and Features

6.1 Double Click Opens Diff

The **‘Double click opens diff’** option under the **‘Git’** menu lets you check the differences between two commits or branches in the repository.



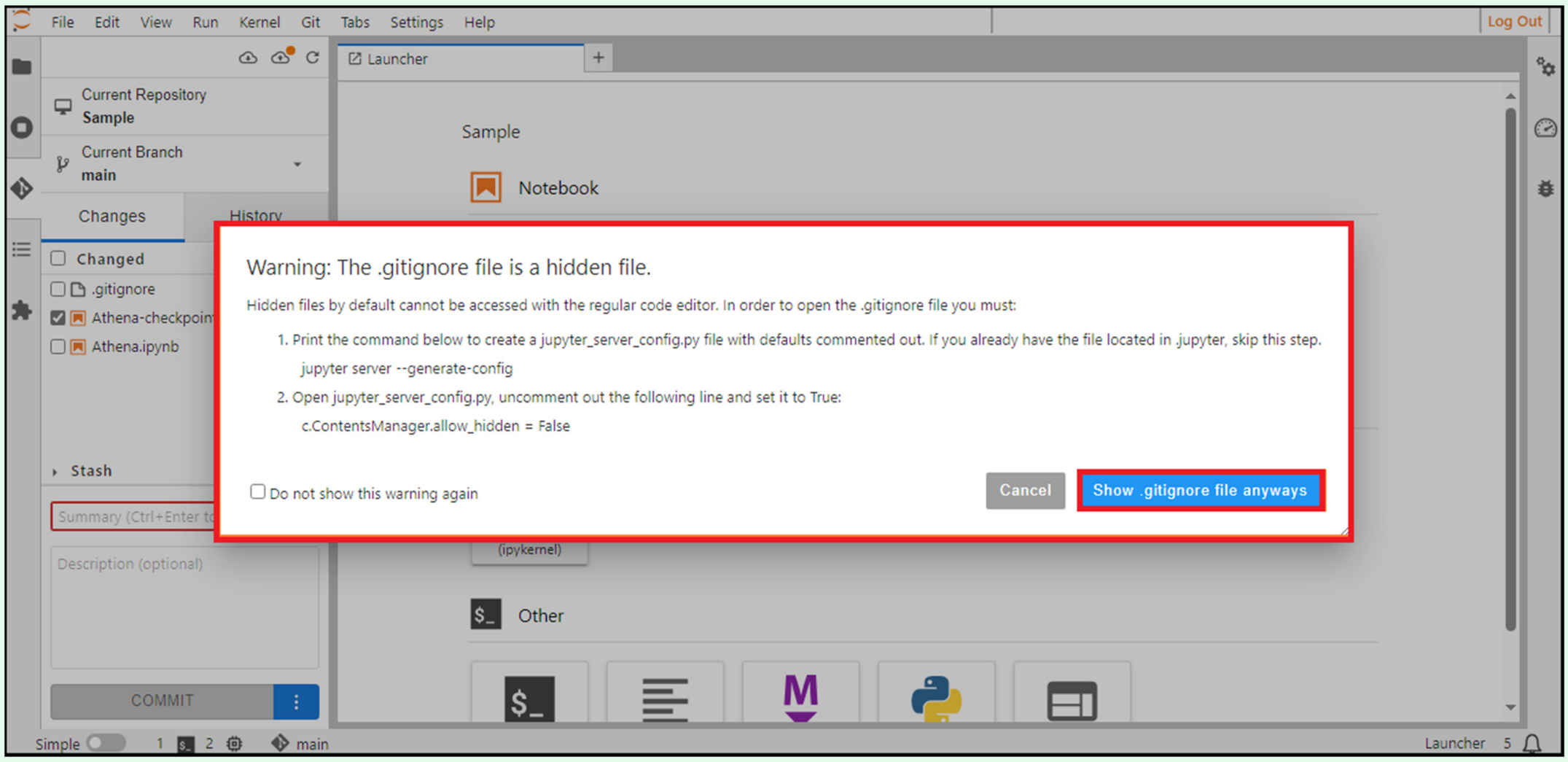
When you double-click on a file, it opens a tool that shows the changes made to that file. This makes it easier to compare different versions side by side. While the specific details of this feature may vary, it enhances the user experience by visually representing changes in the version control system.



6.2 Opening .gitignore

The **‘Open .gitignore’** option facilitates easy access to the **‘.gitignore’** file within the JupyterLab environment. This feature allows users to directly view and edit the **‘.gitignore’** contents, enabling control over which files and folders are intentionally excluded from Git version control.

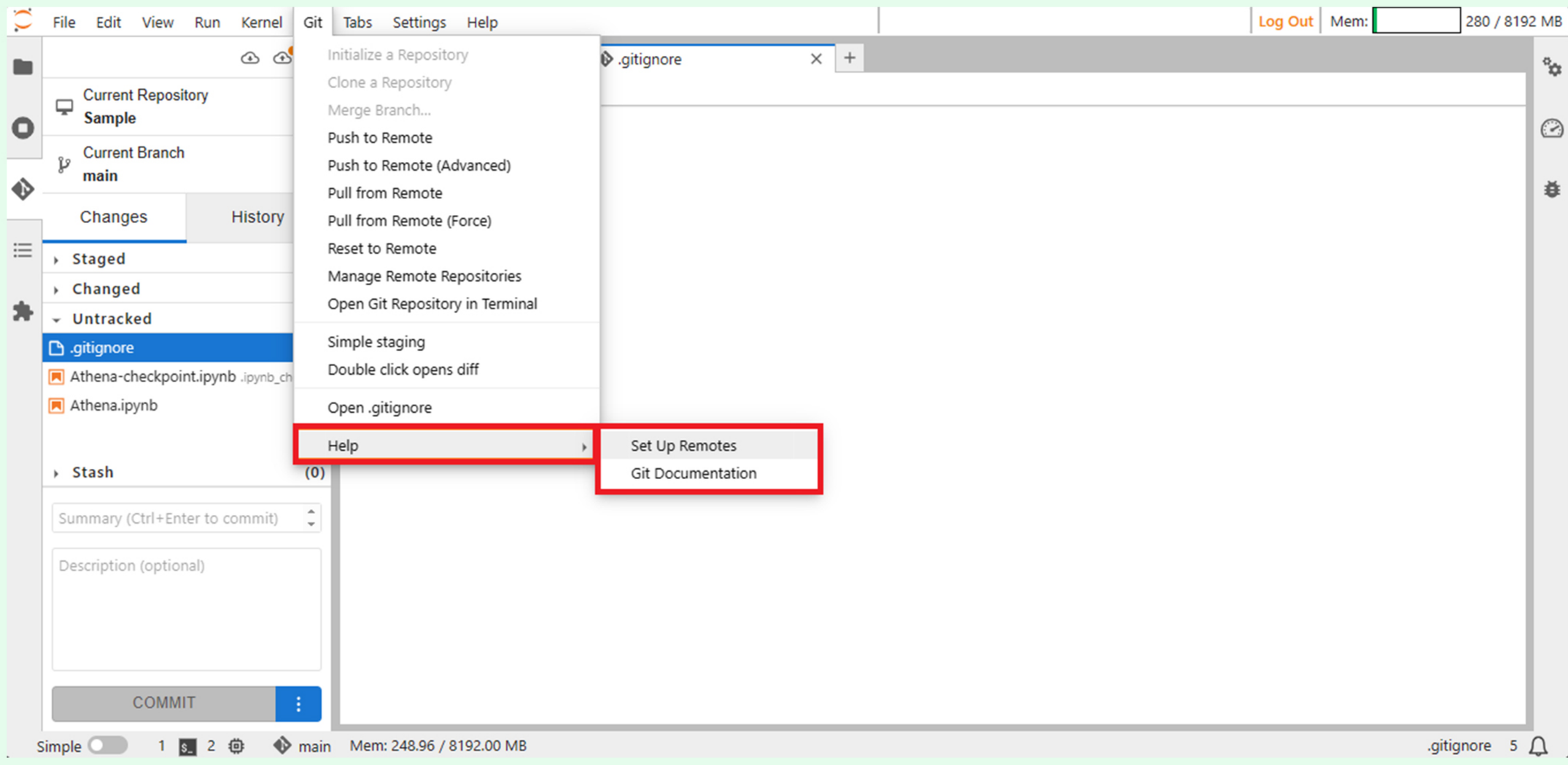
To utilize this option, access the **‘Git’** menu and click on **‘Open .gitignore.’** Despite a warning message, clicking **‘Show .gitignore file anyways’** will open the **‘.gitignore’** file in a text editor within JupyterLab, providing users with the ability to manage their repository's tracking behavior efficiently.



The **'.gitignore'** option prevents unnecessary files and folders from being tracked by Git, reducing clutter, potential conflicts and commonly ignored files.

6.3 Utilizing Help Functionality

Access comprehensive help resources and Git documentation directly within ConverSight Notebooks. Whether you need assistance with configuring remote repositories or understanding specific Git commands, the 'Help' feature provides valuable guidance to support your Git workflow and troubleshoot any issues that may arise.



7. Perks and Privileges

- **Streamlined Workflow Management:** With Git integration, ConverSight Notebooks offer users a streamlined approach to managing their workflows, allowing for efficient tracking of changes and seamless collaboration among team members.

- **Secure Cloud Storage:** By leveraging Git repositories, ConverSight Notebooks provide users with secure cloud storage, ensuring that their projects are safely stored and easily accessible from any location or device.

- **Effortless Collaboration:** Git integration facilitates effortless collaboration among team members, enabling them to work on projects simultaneously, track changes and merge contributions seamlessly.

- **Enhanced Version Control:** With Git integration, ConverSight Notebooks empower users with enhanced version control capabilities, allowing them to revert to previous versions, monitor changes and maintain a clear and organized history of their projects.

- **Increased Productivity:** Notebooks with Git integration ultimately contribute to increased productivity and efficiency within projects.

- **Flexibility and Control:** Gain greater flexibility and control over project workflows with the ability to execute Git commands directly from the terminal window within JupyterLab, enhancing productivity and efficiency.

- **Visual Representation of Changes:** Visual tools such as double-click to open diff enhance the user experience by providing clear visual representations of changes, facilitating easier comparison between different versions of files.

8. Conclusion

In conclusion, the integration of Git within ConverSight Notebooks revolutionizes the landscape of collaborative development and version control. By seamlessly merging these platforms, users benefit from enhanced workflow management, secure cloud storage and effortless collaboration. Whether it is monitoring changes, reverting to previous versions or facilitating efficient code integration, this integration fosters an environment conducive to productivity and innovation. With ConverSight Notebooks and Git integration, users embark on a journey towards streamlined workflow management and enhanced collaboration, ultimately driving efficiency and effectiveness within their projects.

Join our customers who have accelerated growth with ConverSight

DAIMLER | Koken USA | CEAT

Dr. Agarwals EYE HOSPITAL | gimme SEAWEED | EXTRACT

THERMO BOND BUILDINGS | ATD AMERICAN TIRE DISTRIBUTORS | ASHOK LEYLAND

Mahindra | [MAVPAK] | salesbricks

CHAMPIONX | motherson | ISTHARA

ZOOX | meditrend | LEADER PROMOS

About ConverSight

ConverSight's Adaptive Analytics platform uses conversational AI, Natural Language Processing and machine learning to converge the distance between humans and data through data stories, presenting the meaning of data in the most effective, personalized and efficient form possible. ConverSight's patented AI business assistant, Athena, connects distributed databases to answer questions and Augment the consumers through 4 key functions: Information on demand, Automated Story Telling, Proactive Insights, and Recommended Actions.

For more information, visit www.conversight.ai

Gartner peerinsights™ | Users Love Us