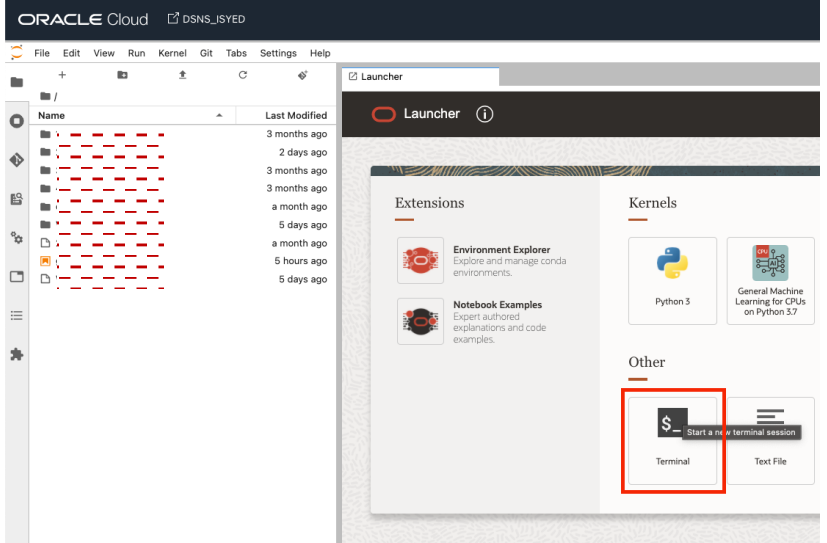


Integrate OCI Data Science Service with GitHub

Description: In this guide we will take a look at how to connect your OCI Data Science Notebook Session with your GitHub Repository.

High Level Steps:

- Authenticate between OCI Data Science and Git using SSH Keys
 - Generate SSH Key Pair
 - Store Private Key within OCI Data Science
 - Adding your SSH key to the ssh-agent
 - Copy Contents of Public Key to GitHub
 - Add GitHub as a Known Host within OCI Data Science
 - Add Username to Git Config
 - Add Email to Git Config
- Utilise the Jupyter Lab Git Extension or Terminal

Step	Screenshot
Login to your OCI Data Science Notebook Session and open a Terminal from the Launcher.	

[illegible]

<p>Next, we will add the SSH Key to the ssh-agent to be managed.</p> <p>First, start the ssh-agent</p> <p>(eval "\$(ssh-agent -s)")</p>	<pre>(base) bash-4.2\$ (base) bash-4.2\$ eval "\$(ssh-agent -s)" Agent pid 5312 (base) bash-4.2\$ █</pre>
<p>Add your SSH Private Key to the ssh-agent.</p> <p>(ssh-add ~/.ssh/id_ed25519)</p>	<pre>(base) bash-4.2\$ (base) bash-4.2\$ ssh-add ~/.ssh/id_ed25519 Identity added: /home/datascience/.ssh/id_ed25519 (-----co.uk) (base) bash-4.2\$ (base) bash-4.2\$ (base) bash-4.2\$</pre>
<p>Once done, lets add GitHub as a 'Known Host', so the OCI Data Science Platform can authenticate with the GitHub Server.</p> <p>(ssh-keyscan -t rsa github.com >> ~/.ssh/known_hosts)</p> <p>You can list the files within the .ssh directory (ls -la) to see the known_hosts file and display its contents. (cat known_hosts)</p>	<pre>(base) bash-4.2\$ (base) bash-4.2\$ ssh-keyscan -t rsa github.com >> ~/.ssh/known_hosts # ----- (base) bash-4.2\$ (base) bash-4.2\$ (base) bash-4.2\$ ls -la total 20 drwxr-xr-x. 2 datascience users 4096 Mar 30 13:57 . drwxr-xr-x. 21 datascience users 4096 Mar 30 10:57 .. -rw----- 1 datascience users 419 Mar 30 13:38 id_ed25519 -rw-r--r-- 1 datascience users 104 Mar 30 13:38 id_ed25519.pub -rw-r--r-- 1 datascience users 392 Mar 30 13:57 known_hosts (base) bash-4.2\$ (base) bash-4.2\$ cat known_hosts ----- (base) bash-4.2\$ (base) bash-4.2\$ (base) bash-4.2\$ █</pre>
<p>We will add our GitHub Username and Email to the Git Config.</p> <p>(git config --global user.name "username")</p> <p>(git config --global user.email "email")</p>	<pre>(base) bash-4.2\$ (base) bash-4.2\$ git config --global user.name "-----" (base) bash-4.2\$ (base) bash-4.2\$ git config --global user.email "-----co.uk" (base) bash-4.2\$ (base) bash-4.2\$ █</pre>



We now need to add our Public Key to our GitHub Profile.

First let's display the contents of our public key and copy the contents to the clipboard.

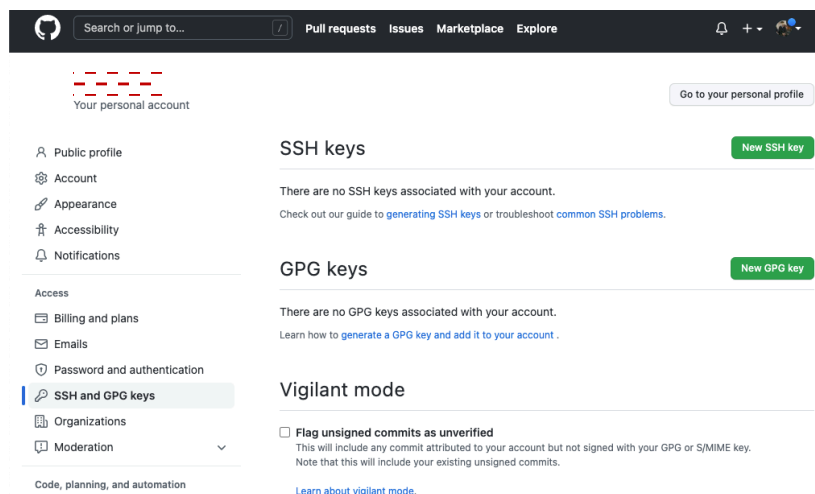
(cat id_ed25519.pub)

Copy the contents displayed in the Terminal.

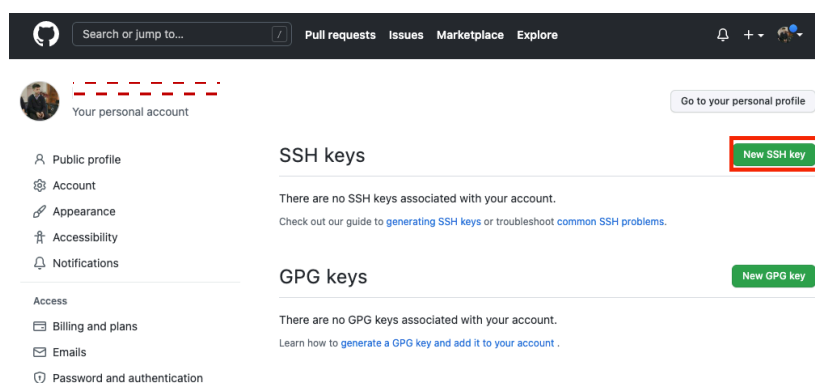
```
(base) bash-4.2$ cat id_ed25519.pub  
-----  
(base) bash-4.2$
```

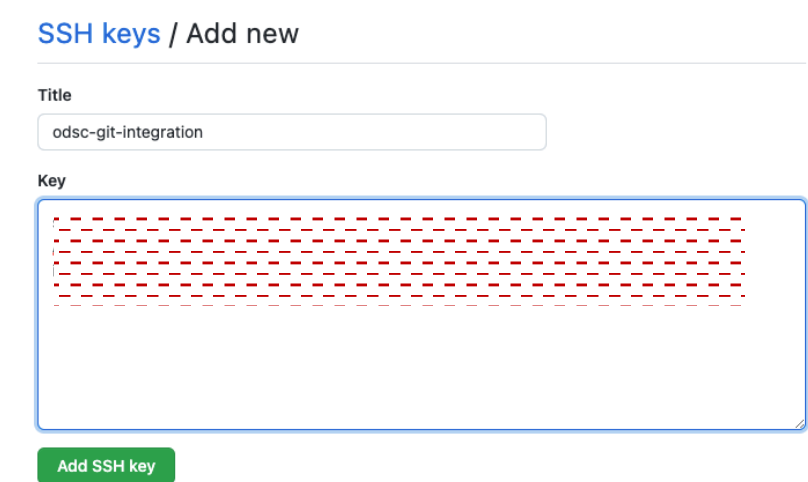
Visit your GitHub account online and navigate to:

(Profile -> Settings -> SSH and GPG keys)



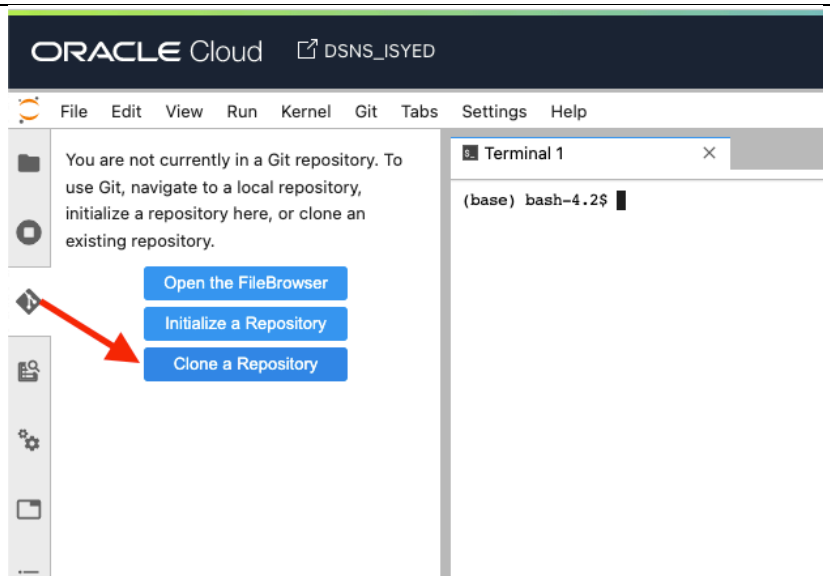
Click on **New SSH Key**



<p>Enter a Title for the new SSH Key.</p> <p>Paste in the contents of the Public SSH Key that you copied over from the Public Key within the OCI Data Science Service.</p> <p>Click Add SSH Key</p>	
<p>We have now successfully integrated a GitHub Account with the OCI Data Science Service.</p>	
<p>We will now try to clone an existing GitHub Repository.</p> <p>Visit an existing repository > click on the Code dropdown > select the SSH Tab and copy the command displayed.</p>	

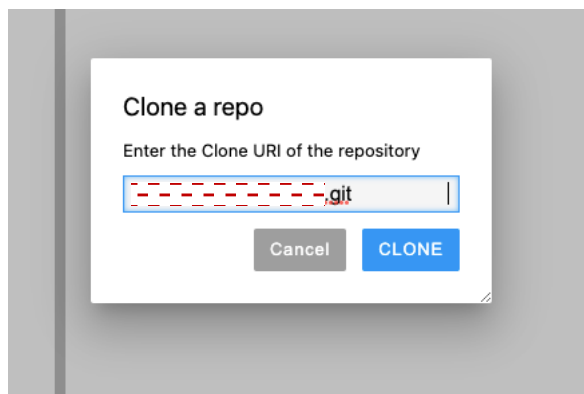
We can use the Jupyter Lab Git Extension to clone a repository.

Click on the Git Extension icon in the tool bar and select Clone a Repository.

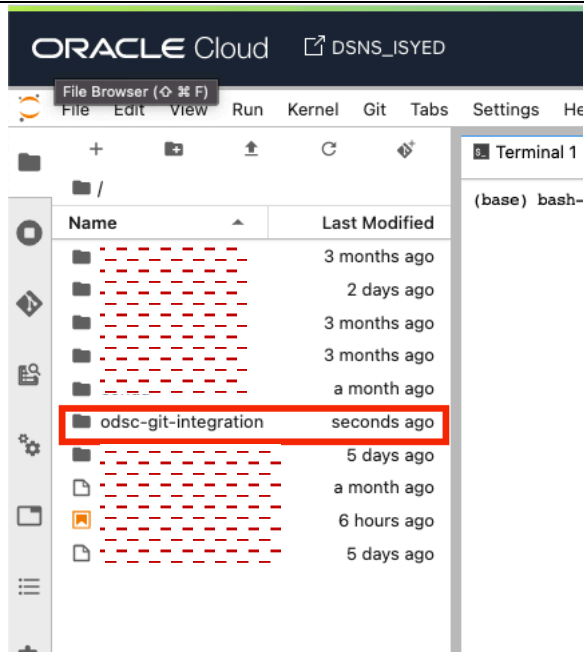


Paste in the Copied Command from the GitHub Repository Page.

Click CLONE.

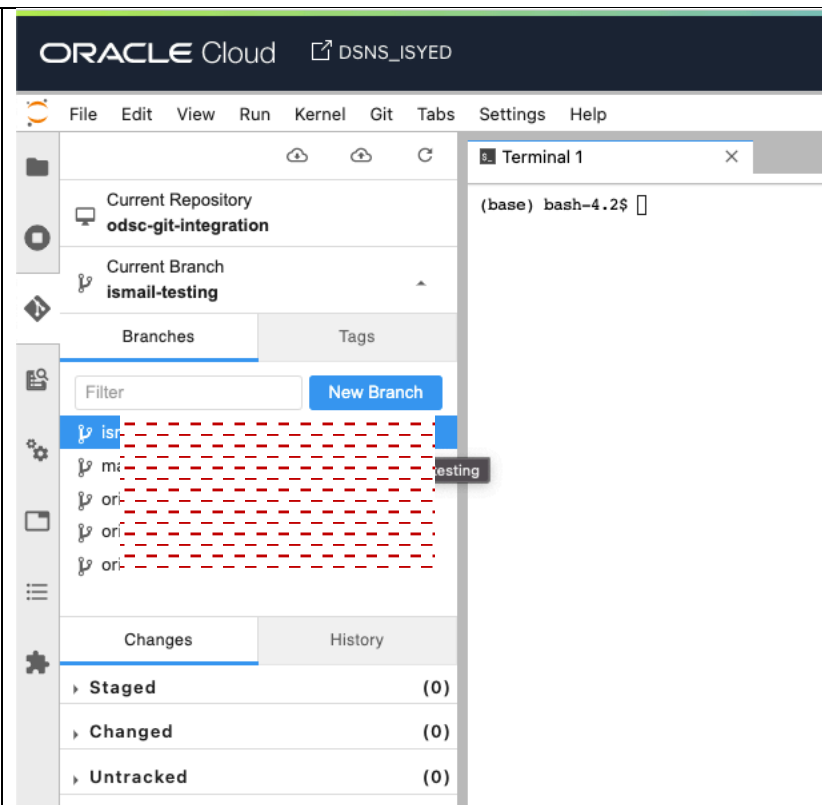


You can now see the Cloned Repository in your directory.

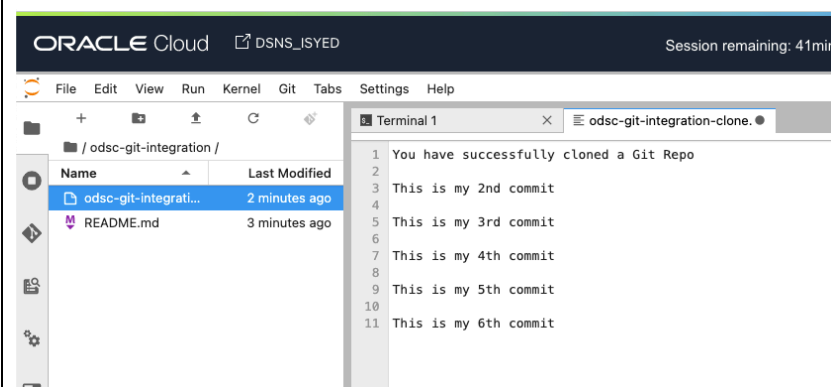


We can use the Git Extension to view our different branches and switch between them.

In this instance, I have selected my **ismail-testing** branch.



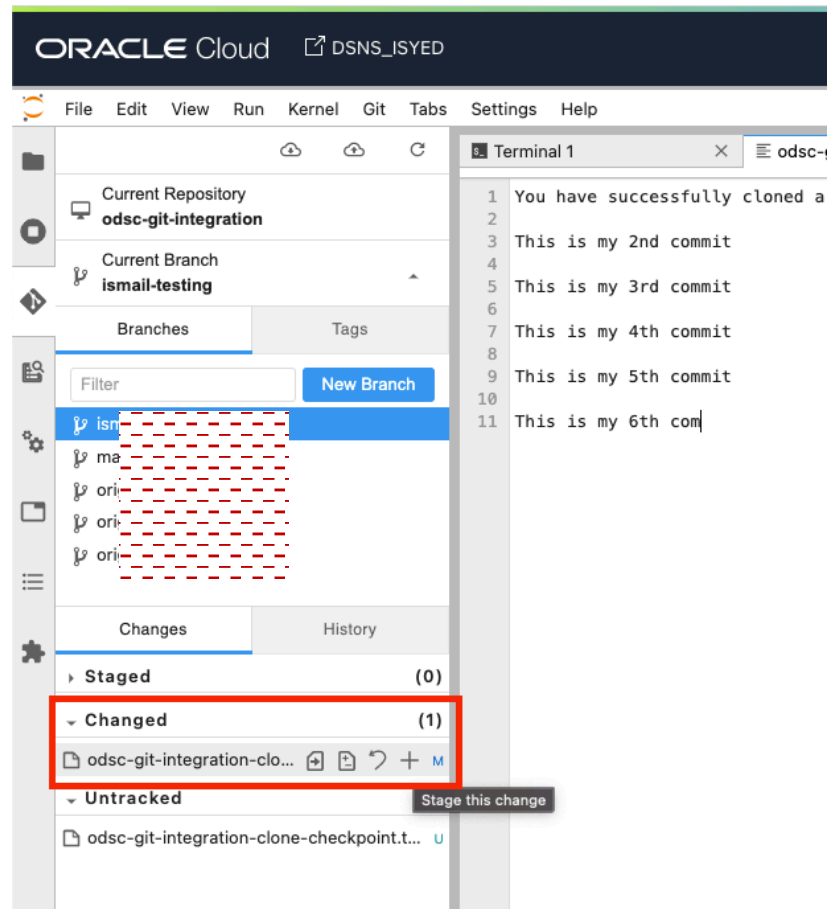
I will now open one of my files and add a new line.



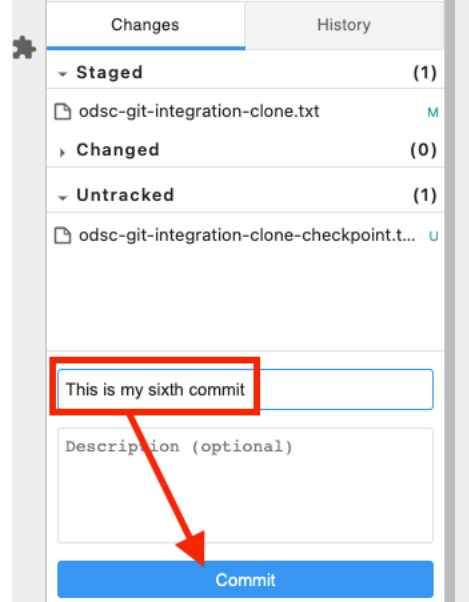
As we start to make changes to our file, we can see this within our Git Extension and the file appears under the Changed section.

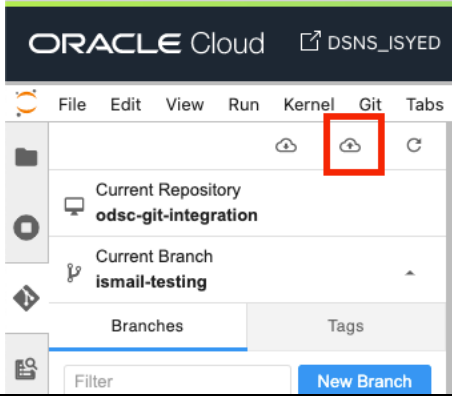
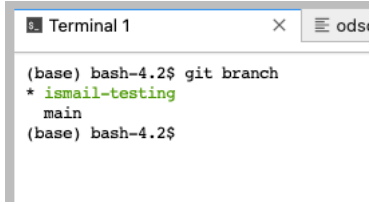
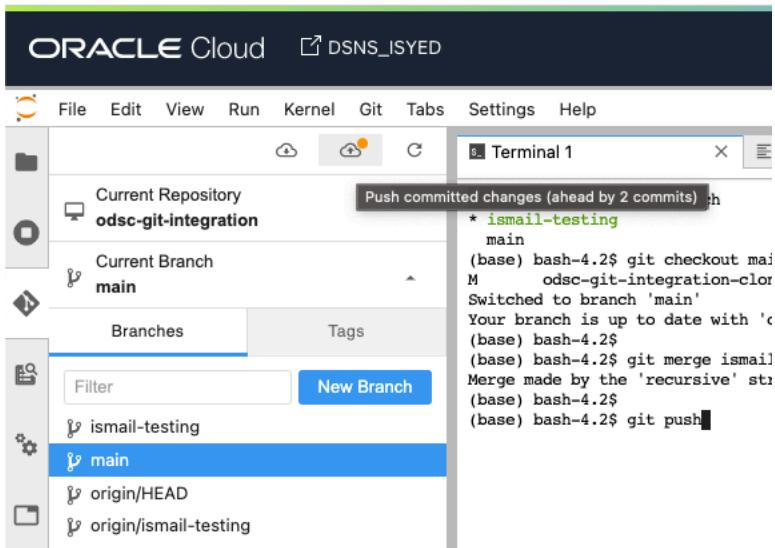
We have the option to stage the changes, revert changes, view differences etc...

Once reviewed we can click the plus icon (+) to stage the changes.



Once staged, we can add a commit message and commit the changes.



<p>We can use the UI to push the branch changes to the Git Repository.</p>	
<p>Alternative to the Jupyter Lab Git Extension we can use the bash terminal to interact with Git</p>	 <pre>(base) bash-4.2\$ git branch * ismail-testing main (base) bash-4.2\$</pre>
<p>I can use git checkout to switch to the main branch.</p> <p>git checkout main</p>	<pre>(base) bash-4.2\$ git checkout main M odsc-git-integration-clone.txt Switched to branch 'main' Your branch is up to date with 'origin/main'. (base) bash-4.2\$</pre>
<p>I can then merge in my branch into my main.</p> <p>git merge ismail-testing</p>	<pre>(base) bash-4.2\$ (base) bash-4.2\$ git merge ismail-testing Merge made by the 'recursive' strategy. (base) bash-4.2\$ (base) bash-4.2\$</pre>
<p>I can then use git push to push my changes to the Git Repository or alternatively I can use the Jupyter Lab Git Extension to do this.</p>	

We can then visit our Git Repository to see the changes pushed.

