

## Lab Guide

### Hands-on-Lab

# DataStage DevOps with MettleCI

# Table of Contents

Introduction .....	3
Access the HOL VM environment .....	4
Lab-01 Check DataStage job compliance against out-of-box pre-defined job development rules .....	5
Lab-02 Use MettleCI to perform Unit Test DataStage jobs .....	14
Lab-03 Check in DataStage job into GitHub.....	26

## **Introduction**

**This Hands-On-Lab (HOL) session will cover on how to use DataStage and MettleCI to**

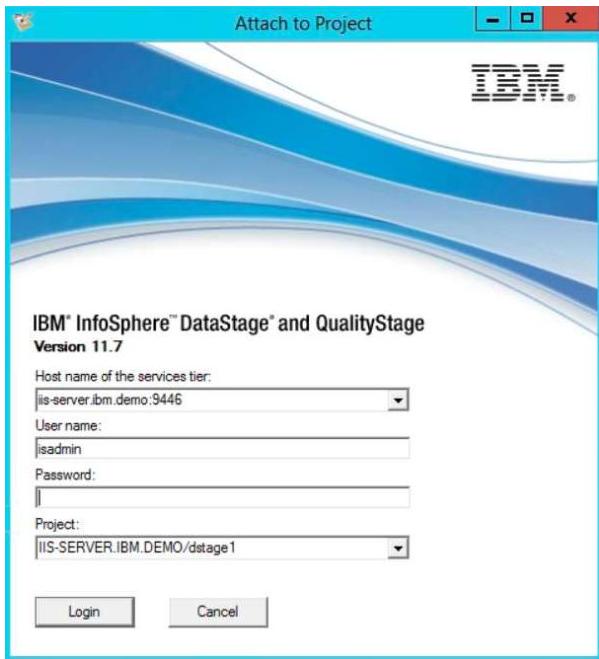
- 1. Check DataStage job compliance against out-of-box pre-defined job development rules,**
- 2. Unit test DataStage jobs, and**
- 3. Check in jobs into GitHub.**

## Access the HOL VM environment

Please refer to the Access Hands on Lab document from your instructor



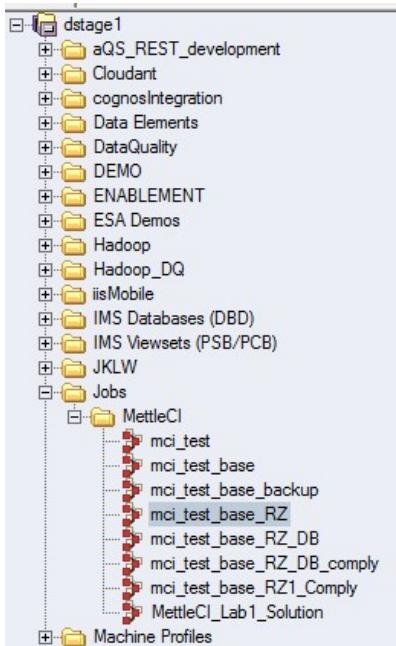
Click **DataStage Designer Client icon** on your desktop.  
Login with user ID/password "**isadmin/infoXserver**".



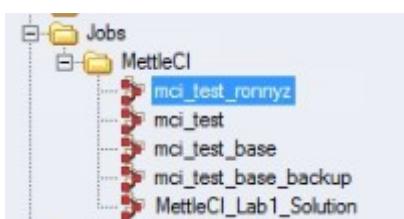
## Lab-01

# Check DataStage job compliance against out-of-box pre-defined job development rules

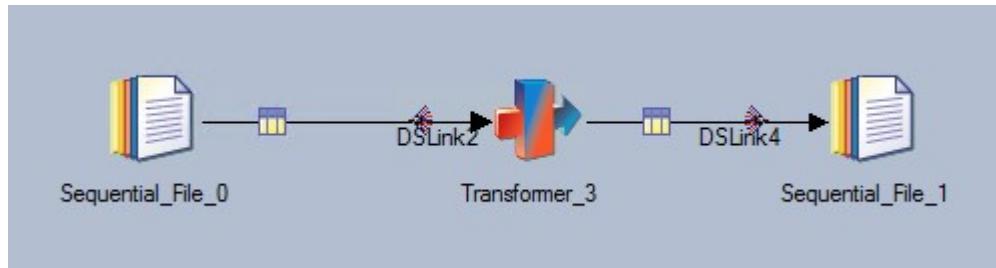
- With Designer's Repository view, open folder “**Jobs→ MettleCI**”.



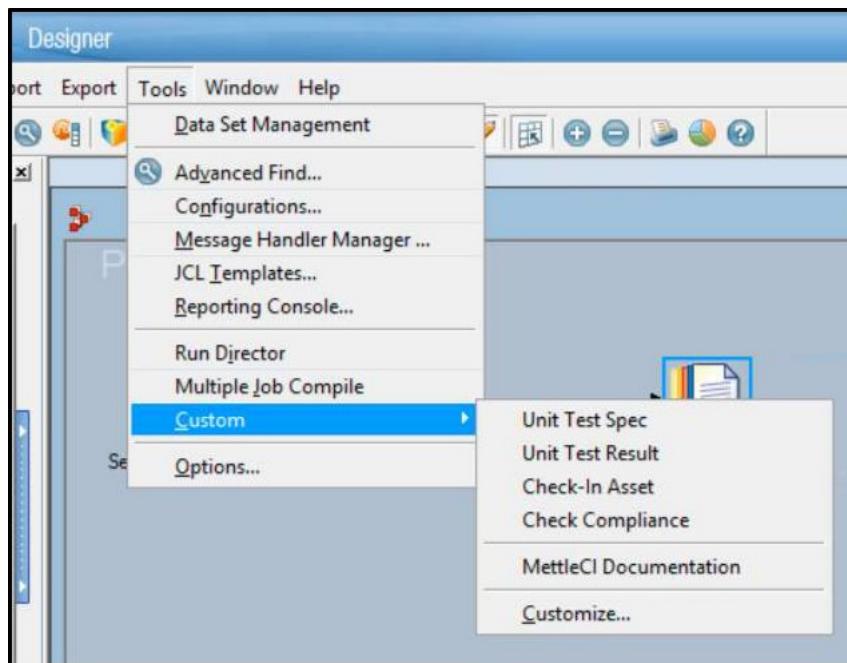
- Make a copy of job “**mci\_test\_base\_RZ**” and rename to “**mci\_test\_<your name>**” e.g. “**mci\_test\_ronnyz**”.



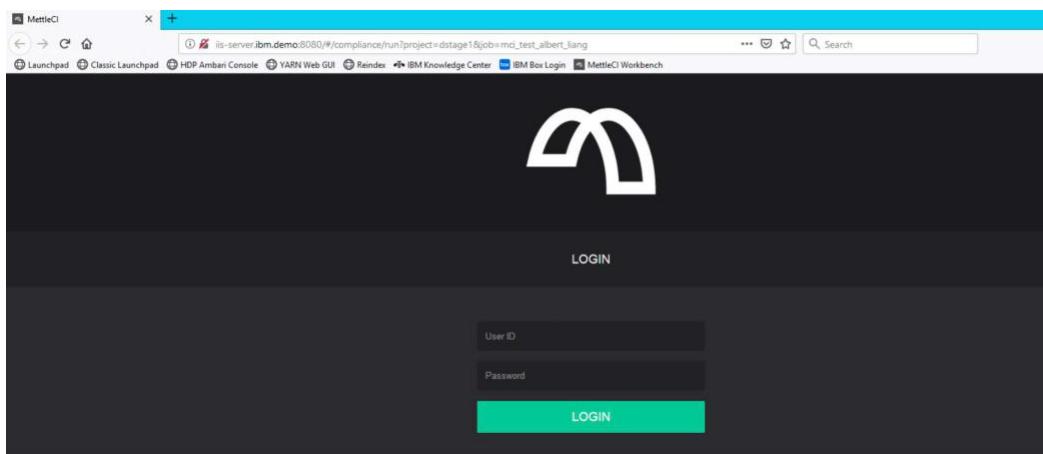
- **Double click job ““mci\_test\_<your name>” to open the job.**  
The job has a very simple design, reading from a file using Sequential File Stage, use transformer and writing into a file using a Sequential File Stage.



- **Click “Tools → Custom → Check Compliance”.**



- Your browser based **MettleCI workbench** home page will pop up.  
**Log in** with User ID/password “**isadmin/infoXserver**”.



- You will be presented with the page to run compliance against job “**mci\_test\_<your name>**”.  
Click “**Run compliance**”.

dstage1

DataStage Project

dstage1

DataStage Asset

mci\_test\_ronnyz

Run Compliance

- You will see two exceptions in the compliance report.

## 1. Default Naming

PxSequentialFile 'Sequential\_File\_0' uses DataStage's default naming.  
Please provide a meaningful name meeting naming standards.

CTransformerStage 'Transformer\_3' uses DataStage's default naming.  
Please provide a meaningful name meeting naming standards.

PxSequentialFile 'Sequential\_File\_1' uses DataStage's default naming.  
Please provide a meaningful name meeting naming standards. [Link](#)

DSLink2 uses DataStage's default naming. Please provide a meaningful name meeting naming standards.

Link DSLink4 uses DataStage's default naming. Please provide a meaningful name meeting naming standards.

## 2. Hardcoded File Paths

PxSequentialFile 'Sequential\_File\_0' uses hardcoded path '/opt/dm/mci/testbonus-in.csv'.

PxSequentialFile 'Sequential\_File\_1' uses hardcoded path '/opt/dm/mci/testbonus-out.csv'..

The screenshot shows a compliance report for a DataStage asset named 'mci\_test\_base\_RZ'. The top navigation bar includes 'dstage1', 'Administrator IIS', and a user icon. The main area displays the following summary:

Rules		Passed Rules		Failed Rules	
13	Rules	11	Passed Rules	2	Failed Rules

**Rule Details:**

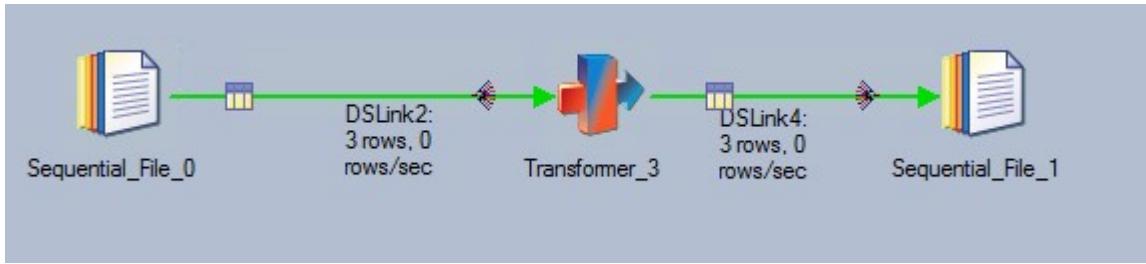
Rule	Duration	Status	Message
Adjacent Transformers	0.027	SUCCESS	
CCMigrateTool Stages	0.009	SUCCESS	
Database Row Limit	0.009	SUCCESS	
Debug Row Limit	0.005	SUCCESS	
Default Naming	0.018	FAILURE	PxSequentialFile 'Sequential_File_0' uses DataStage's default naming. Please provide a meaningful name meeting naming standards. CTransformerStage 'Transformer_3' uses DataStage's default naming. Please provide a meaningful name meeting naming standards. PxSequentialFile 'Sequential_File_1' uses DataStage's default naming. Please provide a meaningful name meeting naming standards. Link DSLink2 uses DataStage's default naming. Please provide a meaningful name meeting naming standards. Link DSLink4 uses DataStage's default naming. Please provide a meaningful name meeting naming standards.
Hardcoded File Paths	0.018	FAILURE	PxSequentialFile 'Sequential_File_0' uses hardcoded path '/opt/dm/mci/testbonus-in.csv'. PxSequentialFile 'Sequential_File_1' uses hardcoded path '/opt/dm/mci/testbonus-out.csv'.
Job Naming	0.004	SUCCESS	
Link Sort	0.008	SUCCESS	

**Activation Message:**

Activate Windows  
Go to System in Control Panel to activate Windows.

© 2018-2023 Data Migrators. Build 978

- You noticed that you can still compile and run the job successfully, but the job is not following the compliance rules.



- To fix the Default Naming exception, replace the stage names for the input and output Sequential File stage with some more meaningful names.

For example:

change “**Sequential\_File\_0**” to “**testbonus\_in**”,  
and change “**Sequential\_File\_1**” to “**testbonus\_out**”.

Replace the link named “**DSLink2**” with some meaningful name, e.g.  
**“to\_transformer”**.

Replace the link named “**DSLink4**” with some meaningful name, e.g.  
**“to\_output”**.

- To fix the Hardcoded File Paths exception, you need to define parameters for

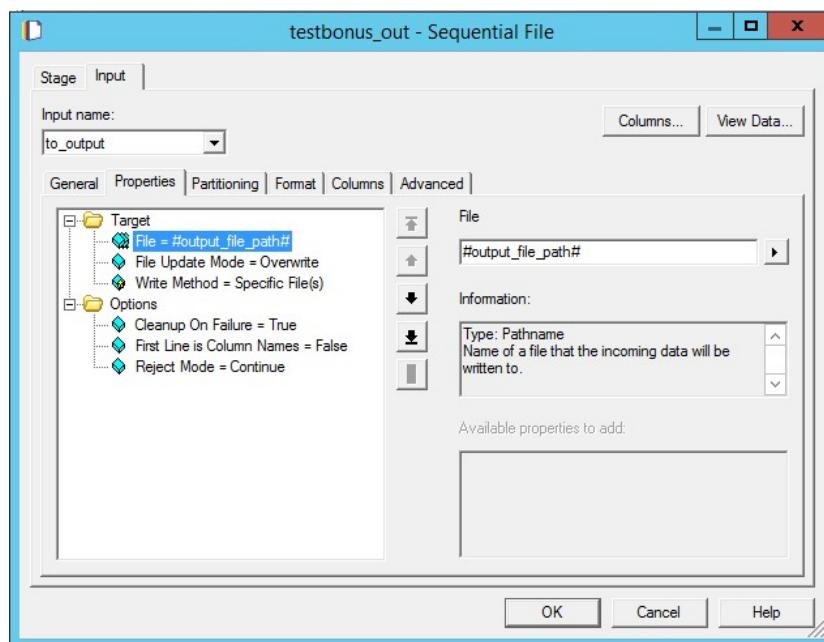
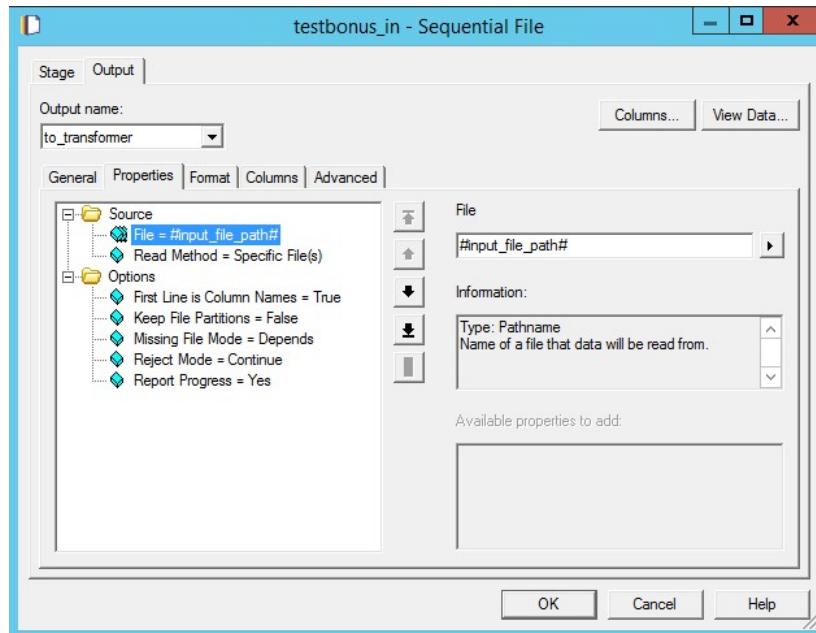
**Input\_file\_path** “/opt/dm/mci/testbonus-in.csv”

**output\_file\_path** “/opt/dm/mci/testbonus-out.csv”

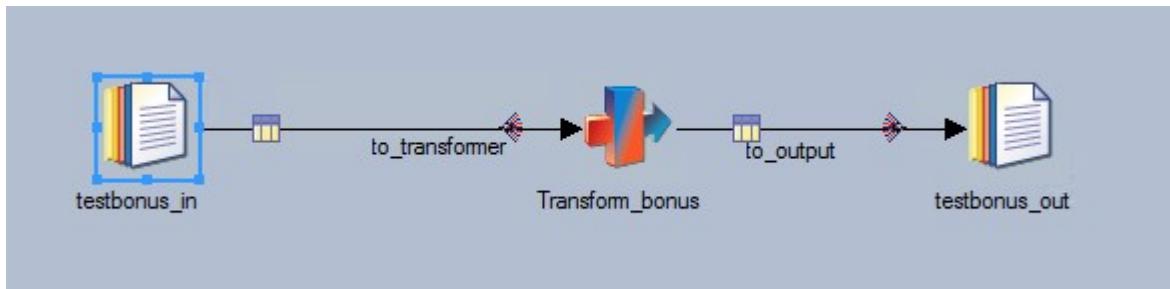
General   Parameters   Job control   Dependencies   NLS   Execution   Defaults					
	Parameter name	Prompt	Type	Default Value	Help Text
1	\$DM_ENABLE_UNIT	Unit Testing Configuration List	Disabled		
2	inout_file_path	input_file_path	String	/opt/dm/mci/testbonus-in.csv	
3	output_file_path	output_file_path	String	/opt/dm/mci/testbonus-out.csv	

And change sequential file stage to use parameter: like below

- The stages that have the parameters defined should look like below.



- Your modified Job “**mci\_test\_<your name>**” should look like below.



- To check the compliance against “mci\_test\_<your name>”, Open job “mci\_test\_<your name>”.

Click “Tools → Custom → Check compliance”.

If you modified your job properly, MettleCI workbench should show all 13 rules are satisfied by the compliance check and no exception reported

The screenshot shows the MettleCI Workbench interface. At the top, there's a navigation bar with various links like Launchpad, Information Governor, HDP Ambari Console, YARN Web GUI, Reindex, IBM Knowledge Center, IBM Box Login, MettleCI Workbench, Manta Keycloak Cons., Manta Admin UI, Manta Flow Viewer, Swagger UI, and MettleCI. Below the navigation bar, the main area displays a DataStage Project named 'dstage1' and a DataStage Asset named 'mci\_test\_ronnyz'. A prominent green banner at the top says 'SUCCESS'. Below this, a summary card shows '13 Rules' (purple background), '13 Passed Rules' (green checkmark icon), and '0 Failed Rules' (red exclamation mark icon). A detailed table follows, listing 13 rules with their duration and status:

Rule	Duration	Status	Message
Adjacent Transformers	0.001	SUCCESS	
CCMigrateTool Stages	0.001	SUCCESS	
Database Row Limit	0.002	SUCCESS	
Debug Row Limit	0.003	SUCCESS	
Default Naming	0.002	SUCCESS	
Hardcoded File Paths	0.003	SUCCESS	
Job Naming	0.002	SUCCESS	
Link Sort	0.002	SUCCESS	
Lookup Failure	0.002	SUCCESS	

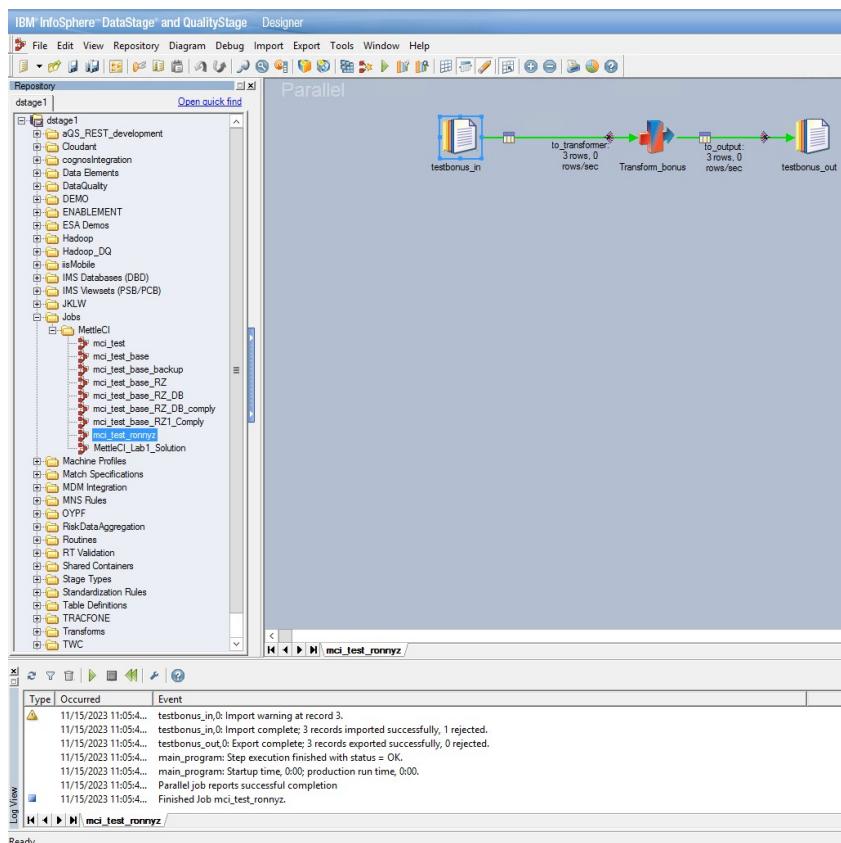
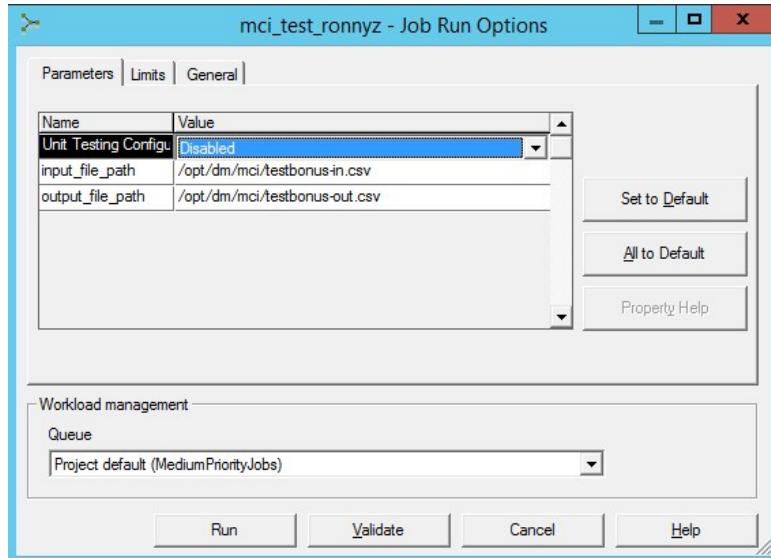
- You can review all the compliance checks you have done by clicking on the “Compliance” icon on the left pane.

The first screenshot shows the left sidebar expanded to reveal the 'Compliance Checks' section under the 'Compliance User' category. The second screenshot shows the main workspace displaying a table of compliance check results:

Executed	Job	Status
2022-Oct-17 06:35:50	MettleCI_Lab1_Solution	SUCCESS
2022-Oct-17 06:34:28	mci_test_ronnyz	SUCCESS
2022-Oct-17 06:33:41	mci_test_ronnyz	FAILURE

- After the compliance check passed successfully, you can compile and run the job again to verify the job can run successfully.

When you run job “mci\_test\_<your name>”, use the default values for the parameters (See below). The job should run successfully.

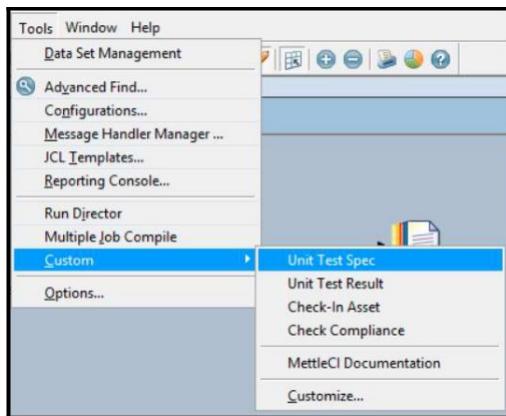


This concludes Lab 1

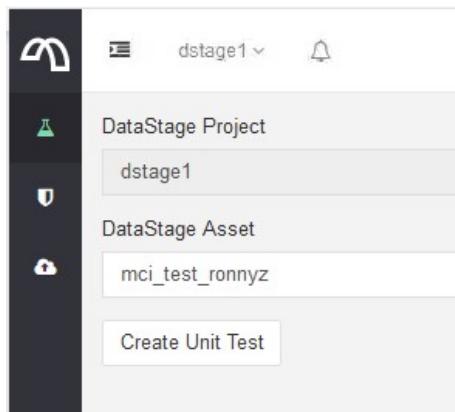
## Lab-02

### Use MettleCI to perform Unit Test DataStage jobs

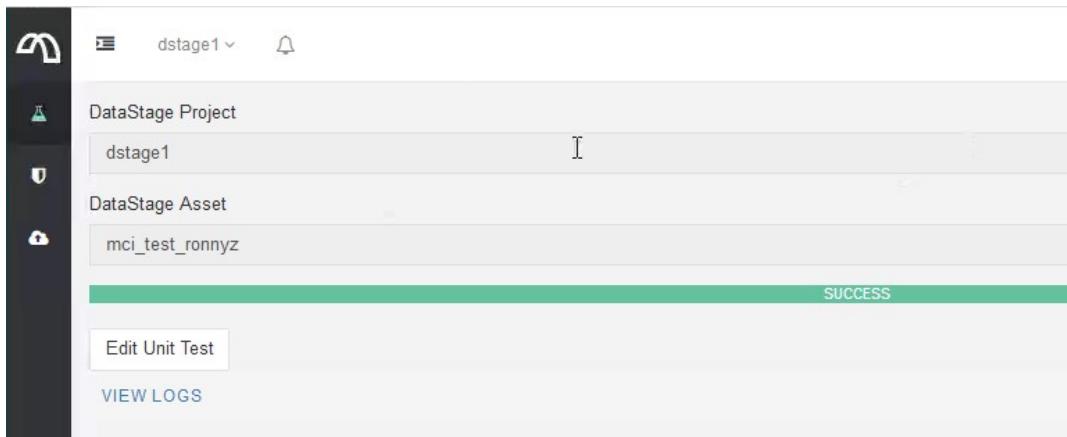
- Open job “**mci\_test\_<your\_name>**”
- Click “**Tools → Custom → Unit Test Spec**” to generate the Unit Test Specification.



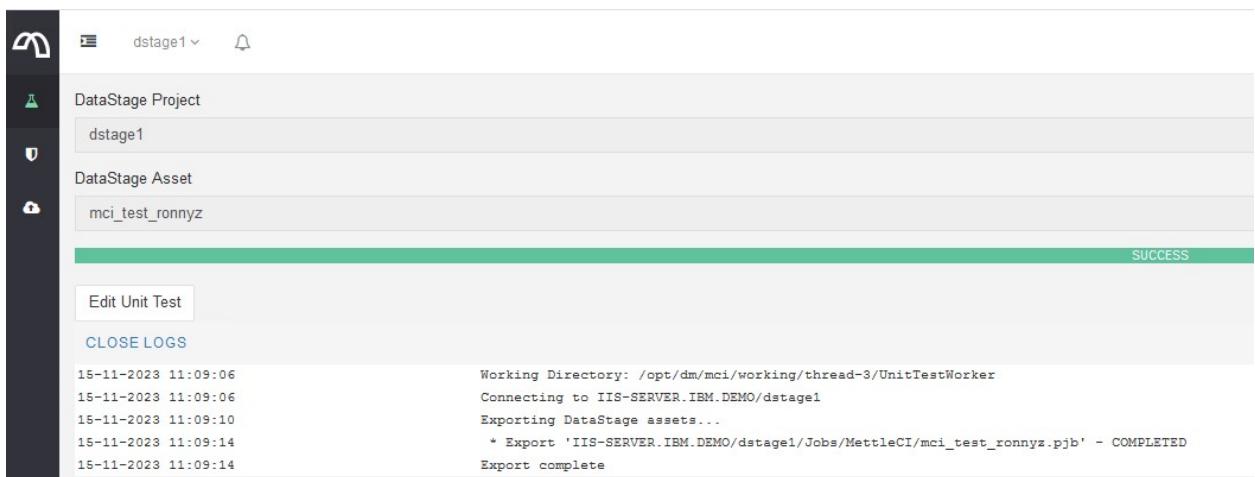
- If the Unit Test is not created, you should see below “**Create Unit Test**” screen on MettleCI Workbench popup window. Click “**Create Unit Test**” to create the unit test specification.



- You should see Unit Test Specification was created successfully.



**Click “VIEW LOGS” to view the “Create Unit Test” results. Click “CLOSE LOGS” to close the log.**



Click “Edit Unit Test” to view the unit test SPECIFICATION (on right pane below with black background) and unit test DATA.

The screenshot shows the DataStage interface with a project named "mci\_test\_ronnyz". On the left, there's a sidebar with icons for file operations. The main area has two sections: "SPECIFICATION" and "DATA". The "SPECIFICATION" section contains a single entry: "mci\_test\_ronnyz". The "DATA" section contains two entries: "testbonus\_in-to\_transformer" and "testbonus\_out-to\_output". To the right of the "DATA" section, there's a large text box displaying the JSON specification for the "testbonus\_in-to\_transformer" data file. The JSON content is as follows:

```
---  
given:  
- stage: "testbonus_in"  
  link: "to_transformer"  
  path: "testbonus_in-to_transformer.csv"  
when:  
  job: "mci_test_ronnyz"  
  controller: null  
  parameters: {}  
then:  
  - stage: "testbonus_out"  
    link: "to_output"  
    path: "testbonus_out-to_output.csv"  
    cluster: null  
    ignore: null
```

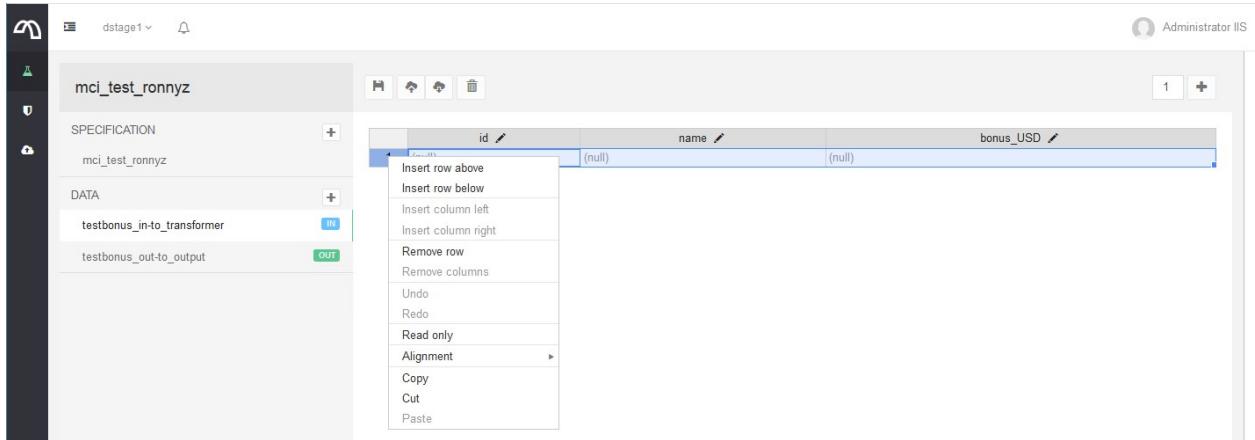
You can click on a specific data file (e.g. [testbonus\\_in-to\\_transformer](#) and [testbonus\\_out-to\\_output](#)) to review the contents of the file.  
To revise the contents, you can click a specific cell and add/review the cell contents.

The screenshot shows the DataStage interface with the same project "mci\_test\_ronnyz". The "DATA" section is expanded, showing the "testbonus\_in-to\_transformer" data file. This file is currently in edit mode, indicated by the blue "IN" button next to its name. A data grid is displayed, showing one row with the following values:

	id	name	bonus_USD
1	(null)	(null)	(null)

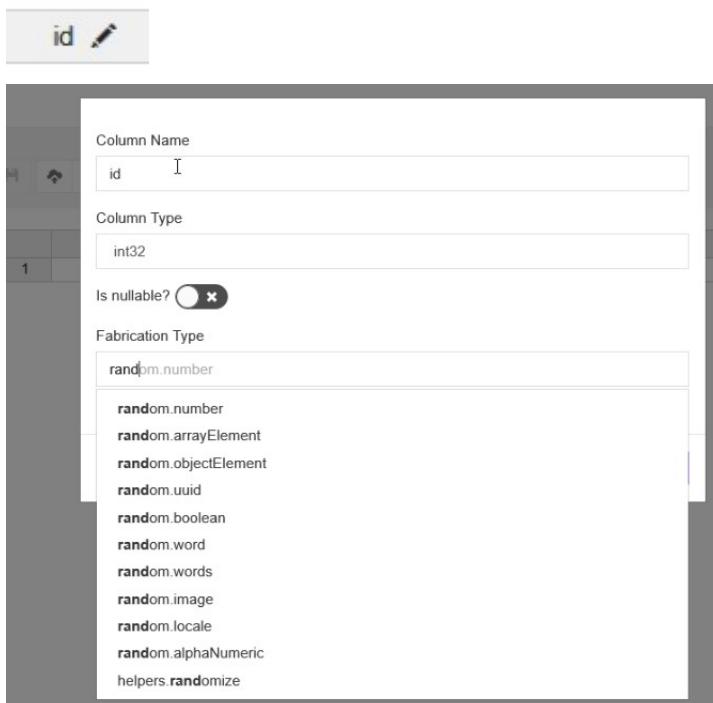
You can manipulate the row(s) of the data files by right clicking on the row number (e.g. row 1 below)

A popup window will display the functions you can use, e.g. insert/remove rows, copy/cut/paste rows.



Now we will provide input for the input test file  
click **testbonus\_in-to\_transformer** on the left pane under DATA section

MettleCI provide 200+ fabrication function for unit test data generation, you can click the pencil icon in the column header



Click the pencil icon beside **id** column ,  
then choose *random.number* Fabrication Type  
with min 1 and max 100  
then click **update**

Column Name

Column Type

Is nullable?  

Fabrication Type



min

max

precision



---

Close
Update

Now click pencil icon beside **name** column , choose *name.lastName* fabrication type, then click **update**

Column Name

Column Type

Is nullable?  

length

Fabrication Type



gender



---

Close
Update

Then go to top right, fill in 2 and click the plus sign,



notice that the **id** and **name** column, automatically populated based

on the fabrication type we choosed above

	id ↎	name ↎	bonus_USD ↎
1	46	Brakus	(null)
2	28	Kerluke	(null)
3	22	Bartell	(null)

now type in the **bonus\_USD** column: 100, 200 and 300 like below

	id ↎	name ↎	bonus_USD ↎
1	46	Brakus	100
2	28	Kerluke	200
3	22	Bartell	300

Then click “Save” icon



You can also “Upload” your csv file, “Download” the data you specified or “Delete” a specific data file, but we will not use it in this lab

highlight the content of **id** and **name** column, right click → **copy**

The screenshot shows the IBM Data and AI interface. On the left, there's a 'SPECIFICATION' pane with a '+' button and a section for 'mci\_test\_ronnyz'. Below it is a 'DATA' pane with sections for 'testbonus\_in-to\_transformer' (with an 'IN' button) and 'testbonus\_out-to\_output' (with an 'OUT' button). On the right is a data grid with three rows:

	id ↎	name ↎	bonus_USD ↎
1	46	Brakus	
2	28	Kerluke	
3	22	Bartell	

A context menu is open over the data grid, listing options such as Insert row above, Insert row below, Insert column left, Insert column right, Remove rows, Remove columns, Undo, Redo, Read only, Alignment, Copy, Cut, and Paste.

Now we will specify the expected test output file

click **testbonus\_out-to\_output** on the left pane as below.

The screenshot shows the mci\_test\_ronnyz interface. On the left, there's a sidebar with 'SPECIFICATION' and 'DATA' sections. The 'DATA' section contains 'testbonus\_in-to\_transformer' (IN) and 'testbonus\_out-to\_output' (OUT). On the right is a table with three columns: 'id', 'name', and 'bonus\_IDR'. A single row is present with values '(null)', '(null)', and '(null)' respectively. In the top right corner, there are two buttons: a blue square with the number '2' and a white square with a plus sign.

	<b>id</b>	<b>name</b>	<b>bonus_IDR</b>
1	(null)	(null)	(null)

Type in 2 on the top right, then click plus sign ,

2 new rows will be added, right click on cell id 1, then **paste**

This screenshot shows the same interface as above, but with a context menu open over the first row of the table. The menu options include: Insert row above, Insert row below, Insert column left, Insert column right, Remove row, Remove column, Undo, Redo, Read only, Alignment, Copy, Cut, and Paste. The first row has '1' in the 'id' column.

	<b>id</b>	<b>name</b>	<b>bonus_IDR</b>
1	(null)	(null)	(null)
2	(null)	Insert row above	(null)
3	(null)	Insert row below	(null)

The value from previous input test file will be pasted like below

This screenshot shows the interface after pasting values. The table now contains three rows with data: Row 1 has 'id' 46, 'name' Brakus, and 'bonus\_IDR' (null). Row 2 has 'id' 28, 'name' Kerluke, and 'bonus\_IDR' (null). Row 3 has 'id' 22, 'name' Bartell, and 'bonus\_IDR' (null). The 'id' column is highlighted in blue.

	<b>id</b>	<b>name</b>	<b>bonus_IDR</b>
1	46	Brakus	(null)
2	28	Kerluke	(null)
3	22	Bartell	(null)

Now fill in the **bonus\_IDR** column with:

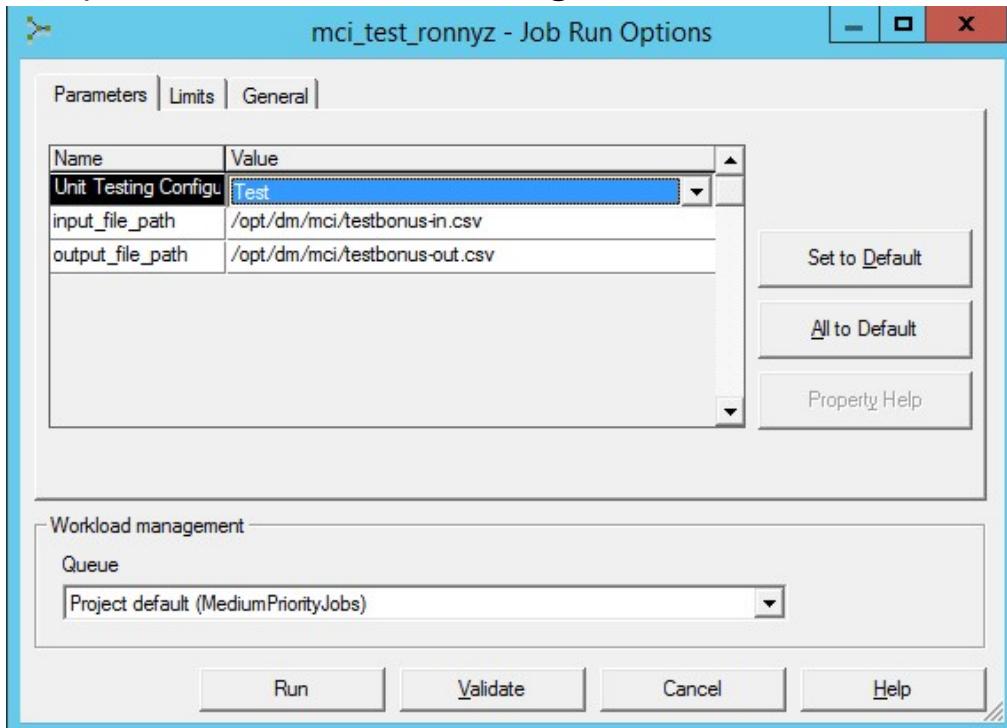
1500000 ; 3000000 ; 4500000 , the expected test output file become

This screenshot shows the table with the 'bonus\_IDR' column filled in. The values are 1500000 for row 1, 3000000 for row 2, and 4500000 for row 3. The 'id' column is highlighted in blue.

	<b>id</b>	<b>name</b>	<b>bonus_IDR</b>
1	46	Brakus	1500000
2	28	Kerluke	3000000
3	22	Bartell	4500000

after you finished input 3 rows above, then click save icon

Go back to DS Designer, click “**Compile**”, then “**Run**” to run your job with parameter “**Unit Test Configuration**” set to “**Test**”.

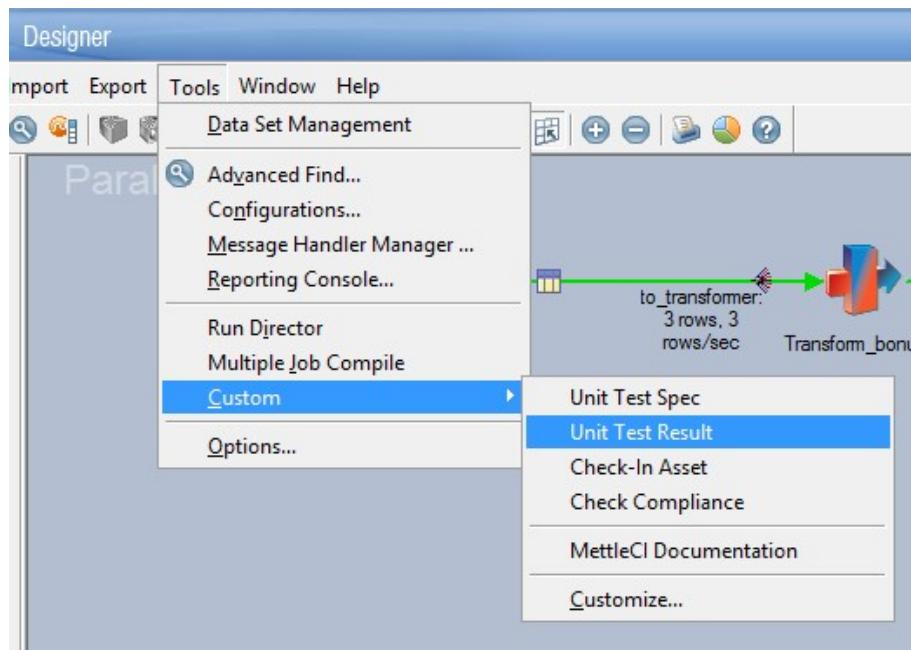


You should see the log displayed as below with below key messages.

**main\_program: MettleCI Unit Testing - Enabled (build 1.0-276)**  
**testbonus\_out,0: Unit Test Failed**  
**Testing.Report,0: Unit Test report(s) available:**

Type	Occurred	Event
	11/15/2023 12:20:1...	Environment variable settings: ...
	11/15/2023 12:20:1...	Parallel job initiated
⚠	11/15/2023 12:20:1...	Parallel job default NLS map UTF-8, default locale OFF
⚠	11/15/2023 12:20:1...	main_program: MettleCI Unit Testing - Enabled (build 1.0-276)
	11/15/2023 12:20:1...	main_program: Loading OSH
	11/15/2023 12:20:1...	main_program: Loading Job Parameters
	11/15/2023 12:20:1...	main_program: Loading Test Specs from : /opt/dm/mci/specs/dstage1
	11/15/2023 12:20:1...	main_program: HV000001: Hibernate Validator null
	11/15/2023 12:20:1...	main_program: IBM InfoSphere DataStage Enterprise Edition 11.7.1.9779 ...
	11/15/2023 12:20:1...	main_program: The timezone environment variable TZ is currently not set in your environment which can lead to significant performance degradati...
	11/15/2023 12:20:1...	main_program: The open files limit is 1024; raising to 4096.
	11/15/2023 12:20:1...	main_program: conductor uname: -s=Linux; -r=3.10.0-957.el7.x86_64; -v=#1 SMP Thu Oct 4 20:48:51 UTC 2018; -n=iis-server.ibm.demo; -m=x86_64
⚠	11/15/2023 12:20:1...	main_program: orchgeneral: loaded ...
⚠	11/15/2023 12:20:1...	main_program: APT configuration file: /opt/IBM/InformationServer/Server/Configurations/default.apt ...
⚠	11/15/2023 12:20:1...	testbonus_out,0: Unit Test Failed
⚠	11/15/2023 12:20:1...	Testing.Report,0: Unit Test report(s) available: ...
	11/15/2023 12:20:1...	main_program: Step execution finished with status = OK.
	11/15/2023 12:20:1...	main_program: Startup time, 0:00; production run time, 0:00.
■	11/15/2023 12:20:1...	Parallel job reports successful completion
■	11/15/2023 12:20:1...	Finished Job mci_test_ronnyz.

On DS Designer, click Tools → Custom → Unit Test Result”



You will see the message Test 'mci\_test\_<your name>' failed with red background. The reason for the unit test failure is indicated as The job didn't produce the correct expected output value as specified in the **testbonus\_out.to\_output**, it expect '1500000' while it received '100' value it expect '3000000' while it received '200' value, it expect '4500000' while it received '300' value in real world case this can be wrong calculation in the job logic, can also be a wrong lookup logic etc.

A screenshot of the 'mci\_test\_ronnyz' test results. The top bar shows 'Test 'mci\_test\_ronnyz' failed' and the date '12:20 15-Nov-2023'. The left sidebar has sections for 'SPECIFICATION' and 'mci\_test\_ronnyz'. The main area shows a failure for 'testbonus\_out.to\_output': '1 output(s) failed to matched expected results while running 'mci\_test\_ronnyz' test.' Below this, it says '3 row(s) changed from expected output' and shows a table with three rows. The table has columns 'id', 'name', and 'bonus\_IDR'. The data is as follows:

	id	name	bonus_IDR
->	46	Brakus	1500000->100
->	22	Bartell	4500000->300
->	28	Kerluke	3000000->200

Now go back to DS Designer, open your job, and double click the transformer stage to modify the transformation logic



edit the bonus\_IDR logic, add \* 15000

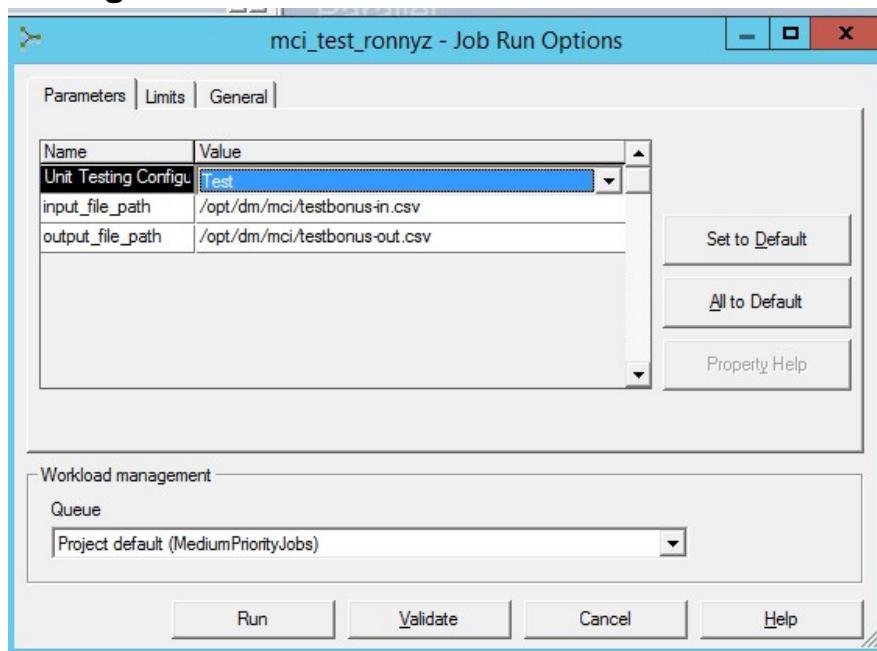
(this represents conversion rate from USD to IDR)

A screenshot of the DS Designer interface showing the 'to\_output' constraint editor. It displays a table with three rows: 'Derivation' and 'Column Name'. The third row contains the value 'to\_transformer.bonus\_USD \* 15000', which is highlighted with a red box. There is an 'OK' button at the bottom right of the editor.

Then click ok

**OK**

click “Compile”, then “Run” to run the job with parameter “Unit Test Configuration” set to “Test”.



You should see the log displayed as below with below key messages.

**main\_program: MettleCI Unit Testing - Enabled (build 1.0-276)**  
**Testing.Report,0: Unit Test report(s) available:**

Type	Occurred	Event
▶	11/15/2023 12:41:5...	Starting Job mci_test_ronnyz. ...
	11/15/2023 12:41:5...	Environment variable settings: ...
	11/15/2023 12:41:5...	Parallel job initiated
	11/15/2023 12:41:5...	Parallel job default NLS map UTF-8, default locale OFF
⚠	11/15/2023 12:41:5...	main_program: MettleCI Unit Testing - Enabled (build 1.0-276)
	11/15/2023 12:41:5...	main_program: Loading OSH
	11/15/2023 12:41:5...	main_program: Loading Job Parameters
	11/15/2023 12:41:5...	main_program: Loading Test Specs from : /opt/dm/mci/specs/dstage1
	11/15/2023 12:41:5...	main_program: HV000001: Hibernate Validator null
	11/15/2023 12:41:5...	main_program: IBM InfoSphere DataStage Enterprise Edition 11.7.1.9779 ...
	11/15/2023 12:41:5...	main_program: The timezone environment variable TZ is currently not set in your environment which can lead to significant performance degradati...
	11/15/2023 12:41:5...	main_program: The open files limit is 1024; raising to 4096.
	11/15/2023 12:41:5...	main_program: conductor uname: -s=Linux; -r=3.10.0-957.el7.x86_64; -v=#1 SMP Thu Oct 4 20:48:51 UTC 2018; -n=iis-server.ibm.demo; -m=x86_64
	11/15/2023 12:41:5...	main_program: orchgeneral: loaded ...
	11/15/2023 12:41:5...	main_program: APT configuration file: /opt/IBM/InformationServer/Server/Configurations/default.apt ...
	11/15/2023 12:41:5...	Testing.Report,0: Unit Test report(s) available: ...
	11/15/2023 12:41:5...	main_program: Step execution finished with status = OK.
	11/15/2023 12:41:5...	main_program: Startup time, 0:00; production run time, 0:00.
	11/15/2023 12:41:5...	Parallel job reports successful completion
■	11/15/2023 12:41:5...	Finished Job mci_test_ronnyz.

**Failed Message has disappeared.**

On **MettleCI workbench**, click “**Unit Testing → Test Reports**” You will see a list of test reports for various jobs.

The status for job “**mci\_test\_<your name>**” is “**Passed**”.

Click “**Passed**” icon for job “**mci\_test\_<your name>**”.

Job Name	Test Case	Last Executed	Result
mci_test_ronnyz	mci_test_ronnyz	12:41 15-Nov-2023	Passed
MettleCI_Lab1_Solution	MettleCI_Lab1_Solution	13:35 23-Sep-2022	Passed
MettleCI_2	MettleCI_2	15:24 31-Dec-2020	Passed
CopyOfCopyOfmci_test2	CopyOfCopyOfmci_test2	14:59 31-Dec-2020	Passed
mci_test	mci_test	18:13 22-Dec-2020	Failed
mci_test_base_backup	mci_test_base_backup		Not Available
mci_test_base	mci_test_base		Not Available

You will see the message **Test 'mci\_test\_<your name>' successful** with **green** background, showing No difference detected

Test 'mci\_test\_ronnyz' successful  
12:41 15-Nov-2023

SPECIFICATION  
mci\_test\_ronnyz

testbonus\_out.to\_output  
All output matched expected results while running 'mci\_test\_ronnyz' test.  
No differences detected

This concludes Lab 2

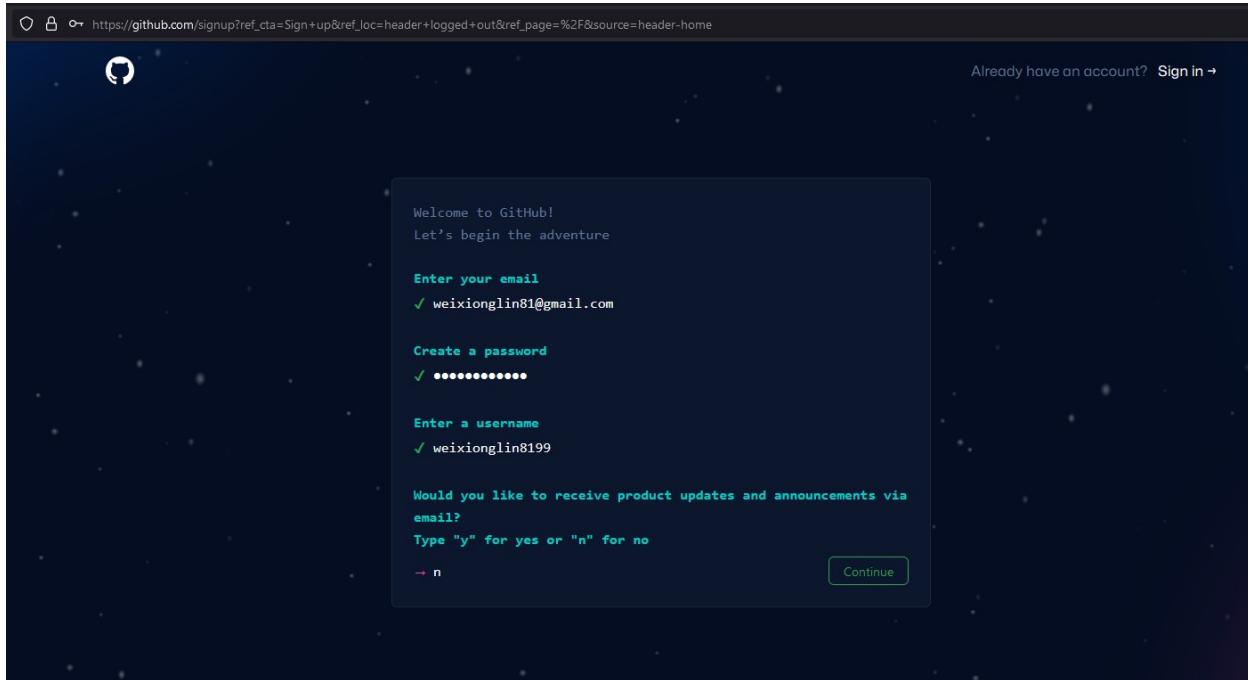
## Lab-03

### Check in DataStage job into GitHub

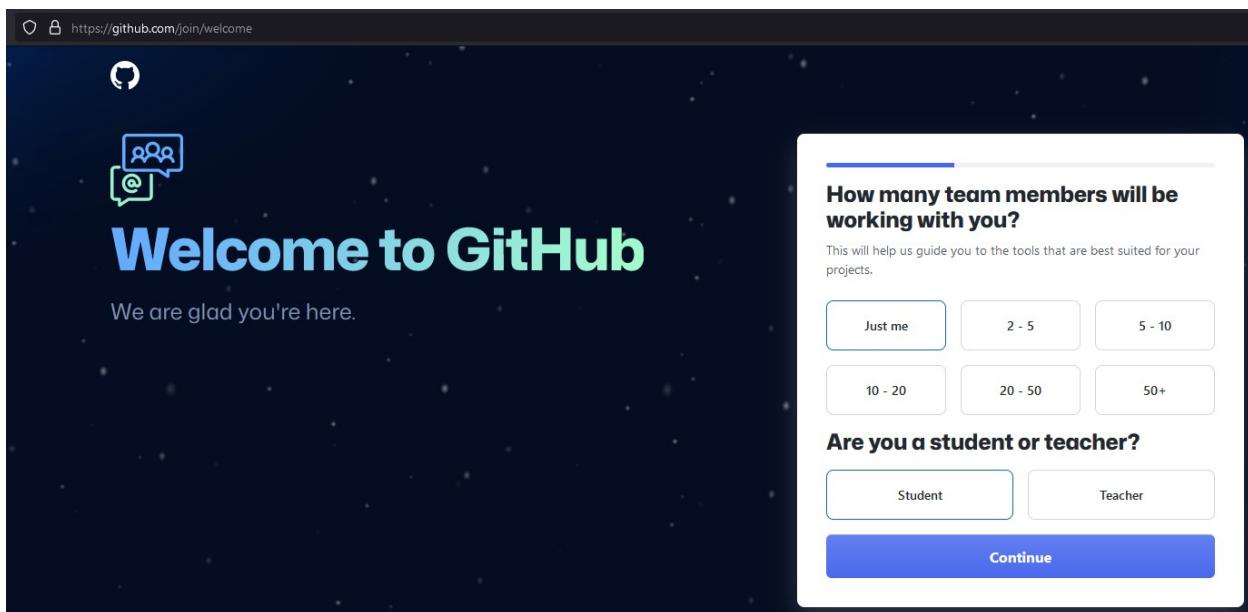
This lab will require you to have a Github account and Github repository

If you don't have one, **Sign up** a Github account from [github.com](https://github.com)

Use your email address to sign up and follow the steps from your computer



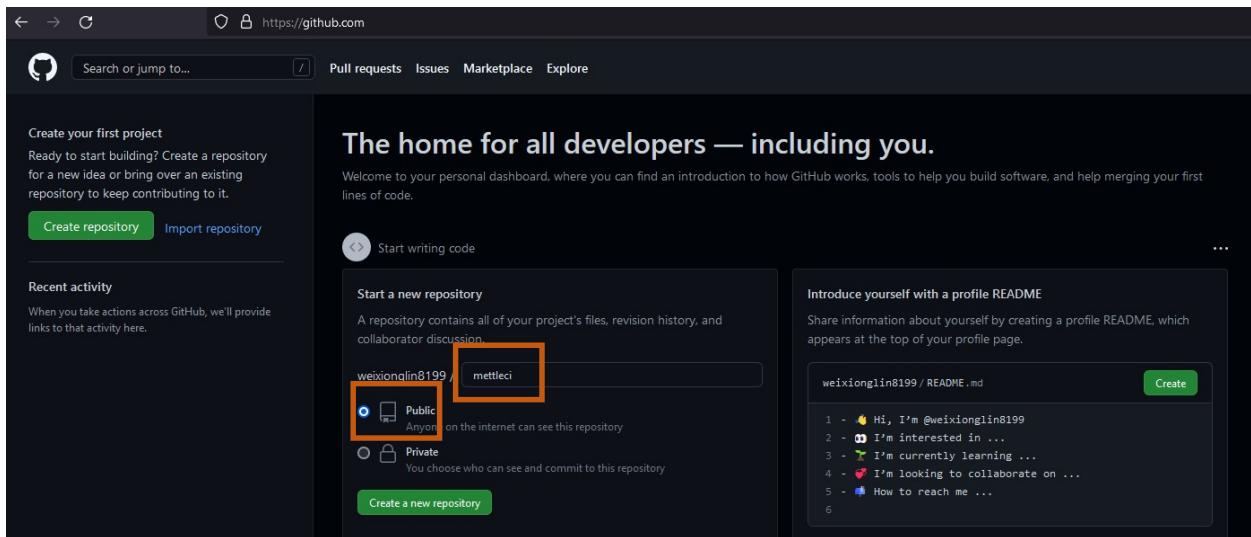
Then follow to start the puzzle and key in the code sent to your email



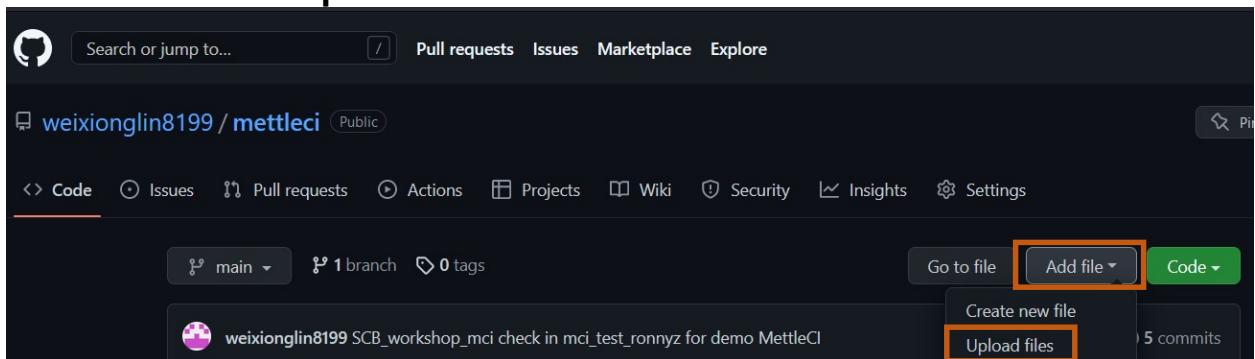
You can also click **Skip personalization** at the bottom

Skip personalization

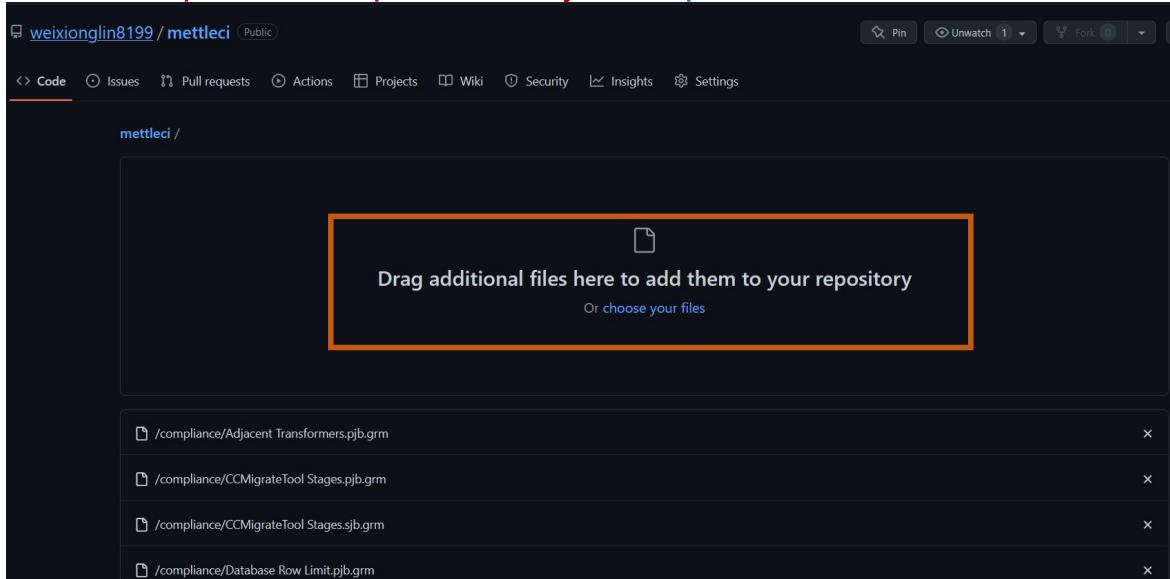
After you created your github account, you need to [Create a public repository](#), give it a name **mettlencl**



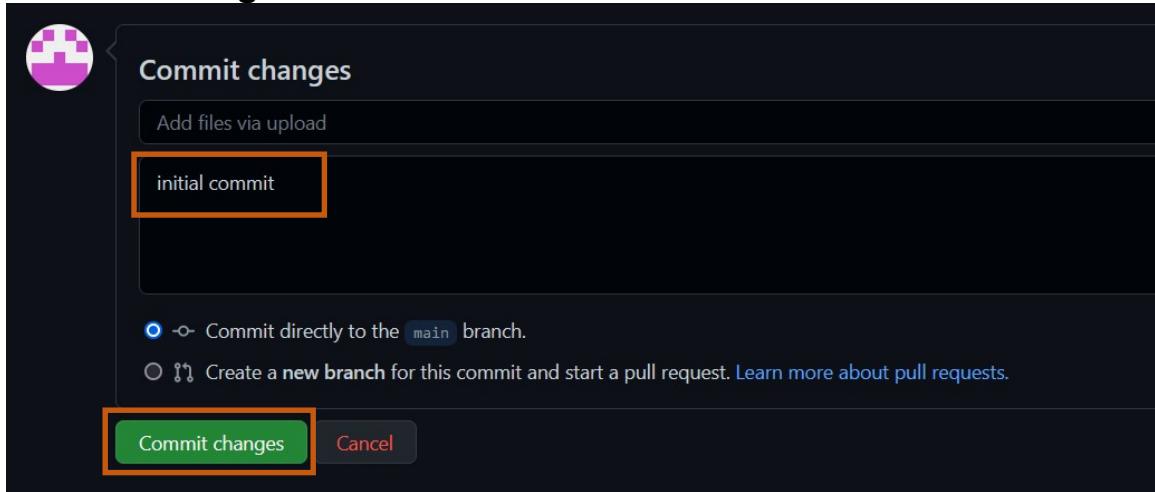
- Now from your Techzone Image computer, go to:  
<https://ibm.ent.box.com/folder/234744591722>  
download the zip repo files **proj-techzone-dastage-mettlencl.zip**
- Then on your mettlencl repository you created above make a commit and push to origin, this can be done by uploading the content of zip folder like below (**make sure you extract the zip file first and upload the content, DO NOT upload the zip file directly**  
click **Add File → Upload files**)



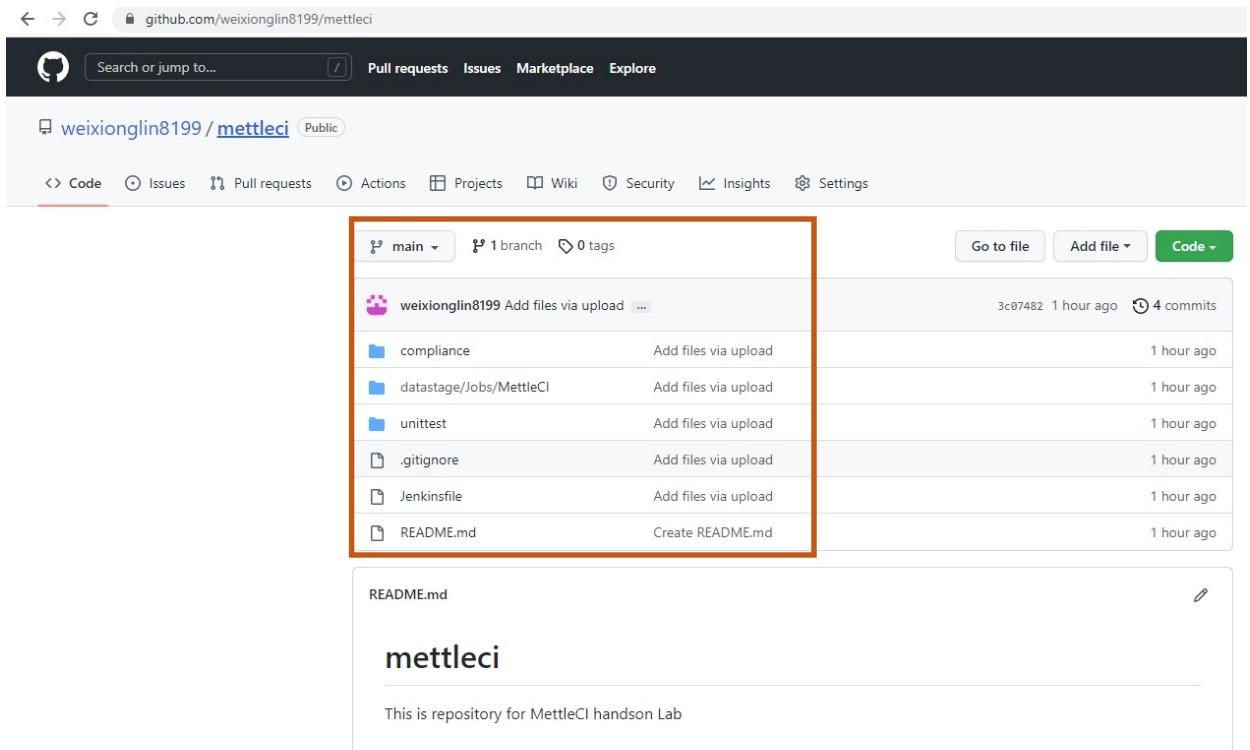
Drag you're the content of zip folder to upload it  
**DO NOT upload the zip file directly, but upload the extracted content**



After the file is uploaded, then add **comment 'initial commit'** click **commit changes** at the bottom of the screen



After the Upload your mettlencl repository should be like below structure



If you already have github account,

- Login to GitHub page at <https://github.com/>
- And create your own **MettleCI repository** example:  
(Please note that the link will be different, depends on your git username)  
<https://github.com/weixionglin8199/mettleci>
- Then perform the upload file step in page 30 above to push to origin

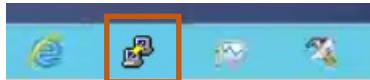
## Set up the GitHub SSH key between your GitHub account and MettleCI:

This step required because the workbench.key in the iis-server environment is created using rsa and Github has dropped support for *rsa* keytype.

The next version of MettleCI workbench will be enhanced to create ***ecdsa*** keytype for the workbench.key which will be created during the setup wizard.

- We will now generate a key pair on iis-server.  
The public key will be copied into GitHub, the private key will stay on iis-server.

- Open **putty** from iis-client windows taskbar



Log into iis-server with User: **root** Pass: **inf0Xserver**  
and type the following commands:

```
cd /opt/dm/mci
mv workbench.key old.workbench.key
```

```
root@iis-server:/opt/dm/mci
login as: root
root@iis-server's password:
Last login: Mon Oct 17 08:35:27 2022
[root@iis-server ~]# cd /opt/dm/mci/
[root@iis-server mci]# mv workbench.key old.workbench.key
```

```
ssh-keygen -t ecdsa -b 521 -f workbench.key
```

Press **ENTER/RETURN** when prompted for a passphrase.

```
[root@iis-server mci]# ssh-keygen -t ecdsa -b 521 -f workbench.key
Generating public/private ecdsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in workbench.key.
Your public key has been saved in workbench.key.pub.
The key fingerprint is:
SHA256:Bw9tVmo+NqF6RLe+sy6LeG/bPFJpllZfiAuGD6xjC8 root@iis-server.ibm.demo
The key's randomart image is:
+---[ECDSA 521]---+
| . .
| o o .
| .o+ o.+o
| +o=S.o=
| . Ooo++* . .
| + =+X... .
| E *O=.o
| .=X@.
+---[SHA256]----+
[root@iis-server mci]#
```

Then type the following commands to change the owner of the file:  
*chown mciworkb workbench.key*

Then restart the mettleni workbench service by executing 2 commands below

*sudo service dm-mettleci-workbench stop*

*sudo service dm-mettleci-workbench start*

```
[root@iis-server mci]# chown mciworkb workbench.key
[root@iis-server mci]# sudo service dm-mettleci-workbench stop
Stopping MettleCI Workbench ...
MettleCI Workbench has been stopped
[root@iis-server mci]# sudo service dm-mettleci-workbench start
MettleCI Workbench is not running
Starting MettleCI Workbench ...
MettleCI Workbench: Java Executable location /usr/local/sbin/jdk8u275-b01-jre/bin/java
MettleCI Workbench: Java Vendor is AdoptOpenJDK
MettleCI Workbench: Java Version is 1.8
MettleCI Workbench has been started
```

- The private key is set up, we now need to get your public key “**workbench.key.pub**” copied into your GitHub profile.
- Log into iis-client and open FileZilla Client from the Desktop icon



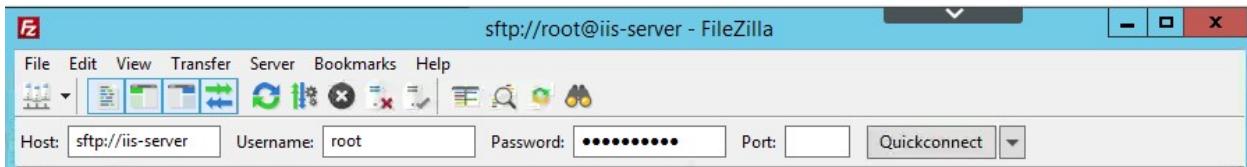
and enter the following credentials

Host: **sftp://iis-server**

Username: **root**

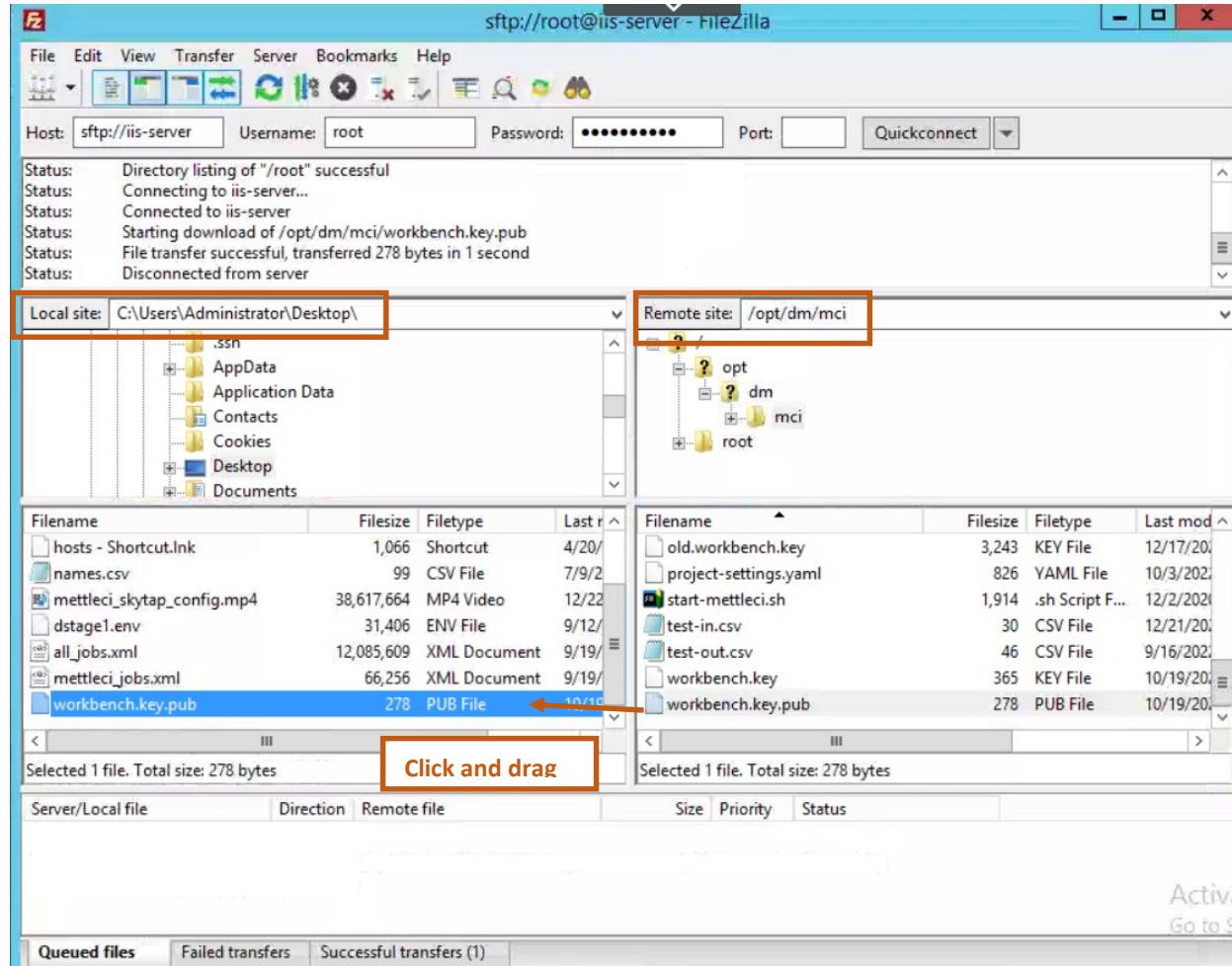
Password: **inf0XerVer**

and click “**Quickconnect**”.

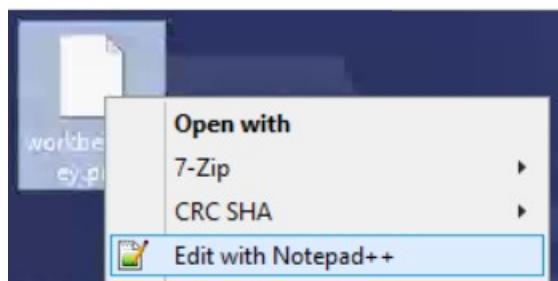


- Now, type in the “Remote site” textbox: “**/opt/dm/mci**” and press ENTER/RETURN.  
On the left side, change the “Local site” to your Desktop, or type “**C:\Users\Administrator\Desktop\**” Like the screenshot below.

- Click and drag the file “**workbench.key.pub**” from the right side of the screen over to the left side, which will transfer the file to your Desktop.

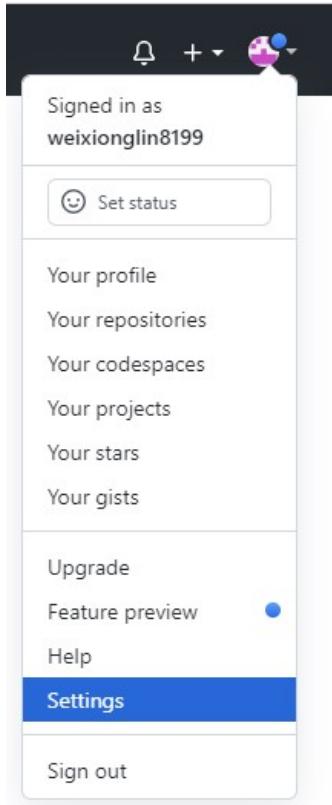


- Open “**workbench.key.pub**” using **notepad++ text editor**



- Then copy the entire text.

- Open the GitHub page from iis-client **Chrome browser**, click your profile picture on the top right and click “**Settings**”.



- Then click “**SSH and GPG keys**” then the green “**New SSH key**”.

The screenshot shows the GitHub Settings page with the following sections:

- SSH keys**: Shows “There are no SSH keys associated with your account.” and a “New SSH key” button (highlighted with a red box).
- GPG keys**: Shows “There are no GPG keys associated with your account.” and a “New GPG key” button.
- Vigilant mode**: Shows “Vigilant mode”.

On the left sidebar, the “SSH and GPG keys” link is highlighted with a red box. Other links in the sidebar include: Public profile, Account, Appearance, Accessibility, Notifications, Access, Billing and plans, Emails, Password and authentication, and Organizations.

- Paste the text into the “Key” section on the github page open. Give any title, example ‘mciworkbench’ and click “Add SSH key”

SSH keys / Add new

Title: mciworkbench

Key type: Authentication Key

Key:

```
ecdsa-sha2-nistp521
AAAAE2VjZHNhLXNoYTItbmlzdHA1MjEAAAECBAD98gDjamr3jQQmt6BrT1QKlcZ8wnY0wirremOofd
woMhBr3WTkfYNZSDYwrDXcpMJGoNtq7FomZAHSku8oVfk0GQC9uMBx9gwsXQqh43DyWSfAhGn14J0y2O+LHeV8sFe/
b6C4NsCNS+5n6bP31veyqdDOoDsn12rq6TM+xzwOehlqg== root@iis-server.ibm.demo|
```

Activate | Go to System

- Now go to your Github **mettleci repository**,

Click **Code**, then **SSH**

**Copy the SSH URL**

e.g: <git@github.com:weixionglin8199/mettleci.git>

Search or jump to... Pull requests Issues Marketplace Explore

weixionglin8199 / mettleci Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file Code

weixionglin8199 SCB\_workshop\_mci check in mci\_test\_ronnyz for demo Met

compliance Add files via upload

datastage/Jobs/MettleCI SCB\_workshop\_mci check in mci\_te

Clone HTTPS SSH GitHub CLI

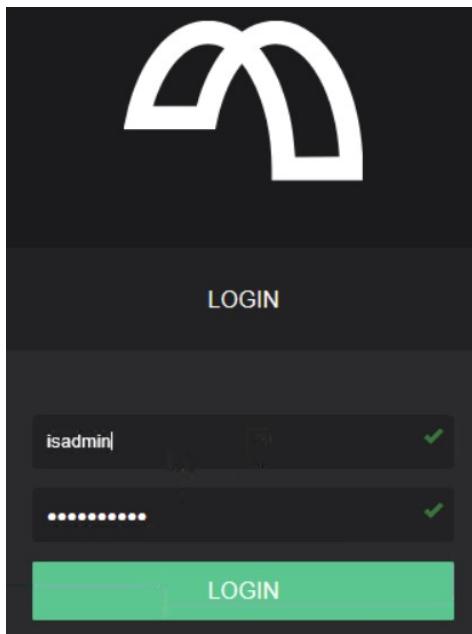
git@github.com:weixionglin8199/mettleci.git

- From your Windows Remote Desktop, open mozilla firefox browser, click the **MettleCI Workbench** bookmark toolbar

MettleCI iis-server.ibm.demo:3087/#/?

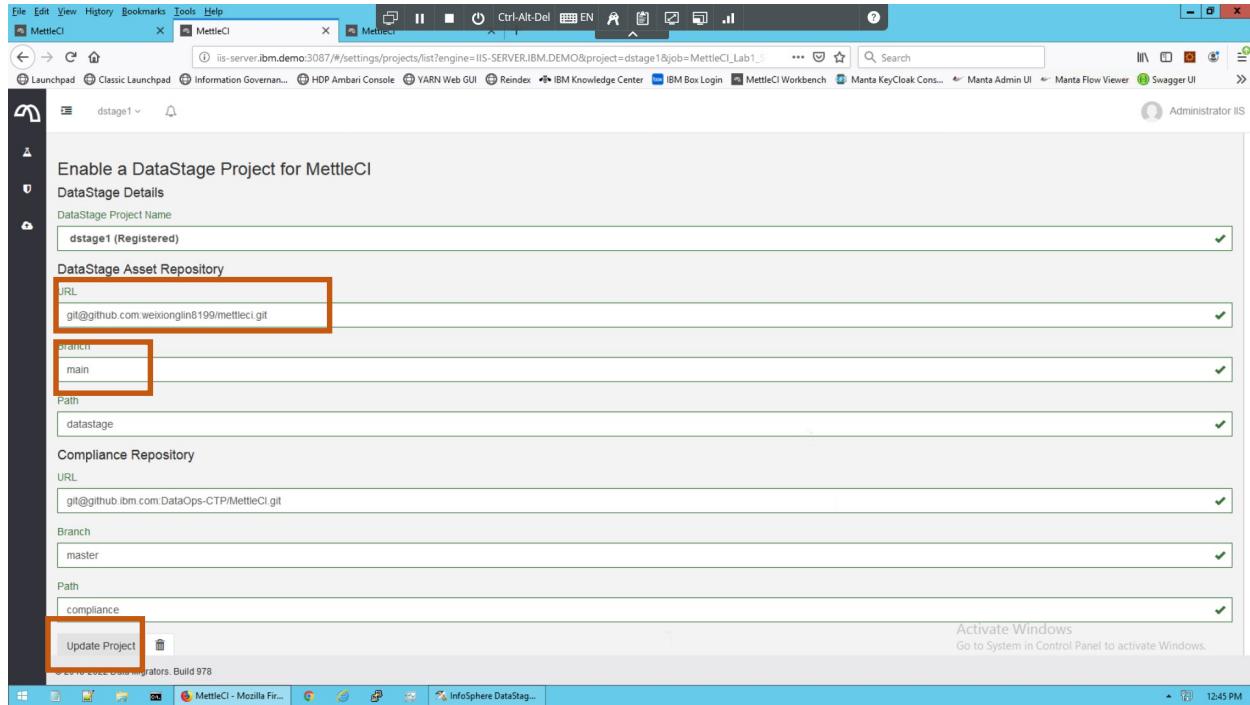
Launchpad Classic Launchpad Information Governor... HDP Ambari Console YARN Web GUI Reindex IBM Knowledge Center IBM Box Login MettleCI Workbench

- Login using **isadmin/inf0Xerver**

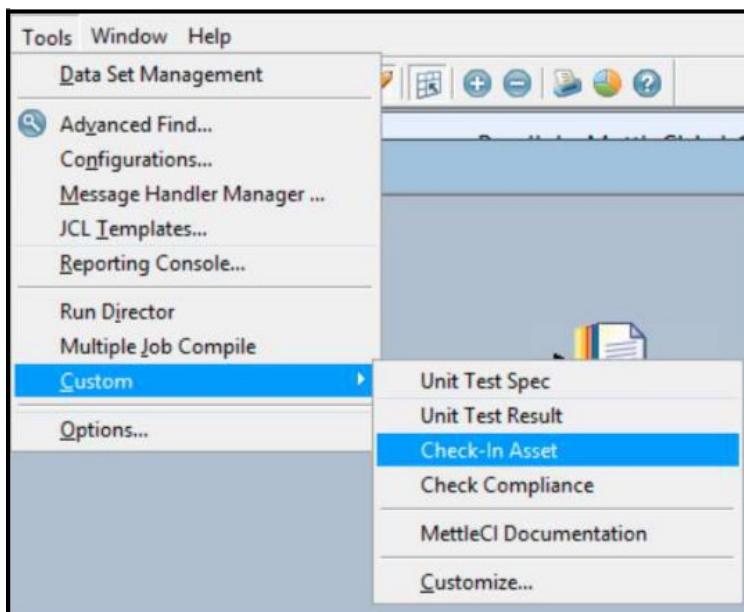


- Click on **Administrator IIS → Manage Projects** on the top right

- Change the **DataStage Asset Repository** with your GitHub repository **SSH URL** that you copy above e.g: <git@github.com:weixionglin8199/mettleci.git> and change the Branch value: **main**
- Then click **Update Project**



- With Designer, open the job with your name “**mci\_test\_<your name>**” (e.g. **mci\_test\_ronnyz**”).
- Click “**Tools → Custom → Check-in Asset**”.



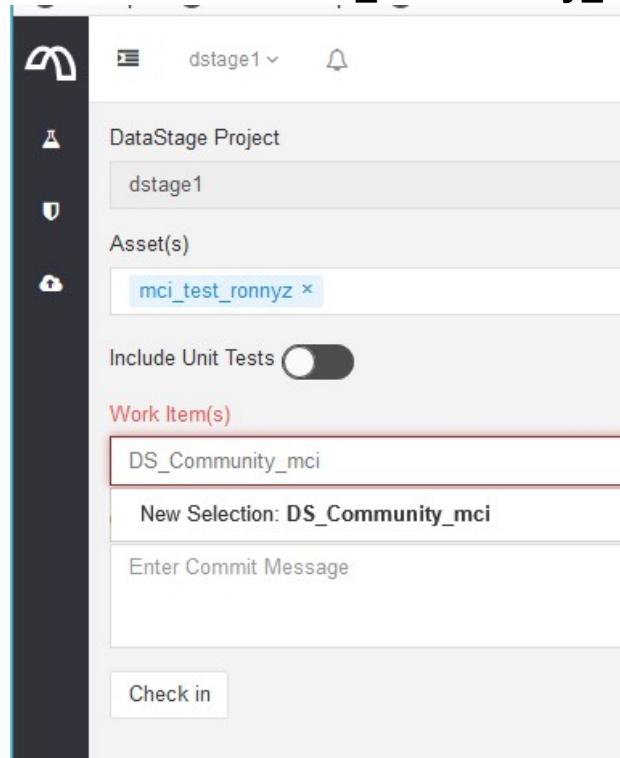
- The check in screen will popup. You need to select/fill in below information.
  - Include Unit Tests**  
Decide to include unit test assets or not
  - Work items**  
Provide the work items name
  - Commit Messages**  
Provide messages you want to associate with the check-in asset on GitHub
- Select/Fill in below values.

### Include Unit Tests - YES

#### Work items

– “DS\_Community\_mci”

**NOTE:** You will get a red warning message, just click  
**“New Selection: DS\_Community\_mci”**



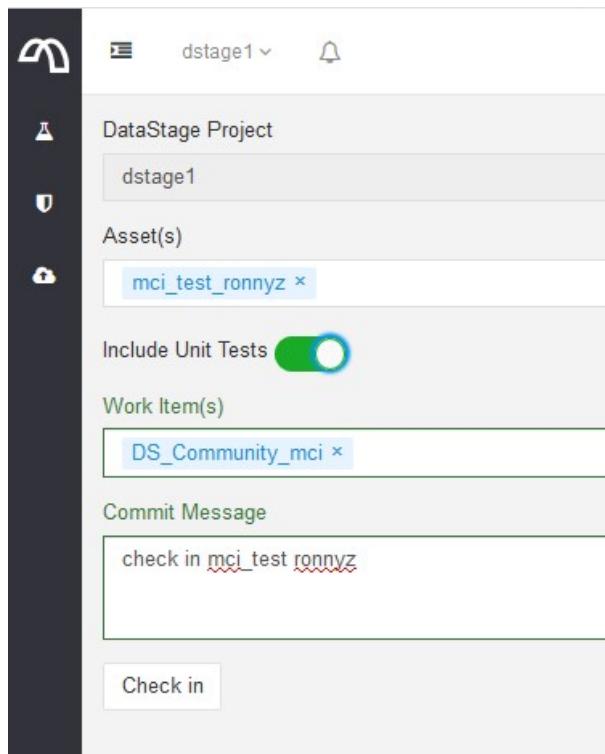
*MettleCI can integrate with many work item solutions such as: Jira, Azure DevOps, ServiceNow, etc, but in this lab we don't integrate with any work item solutions.*

## Commit Message

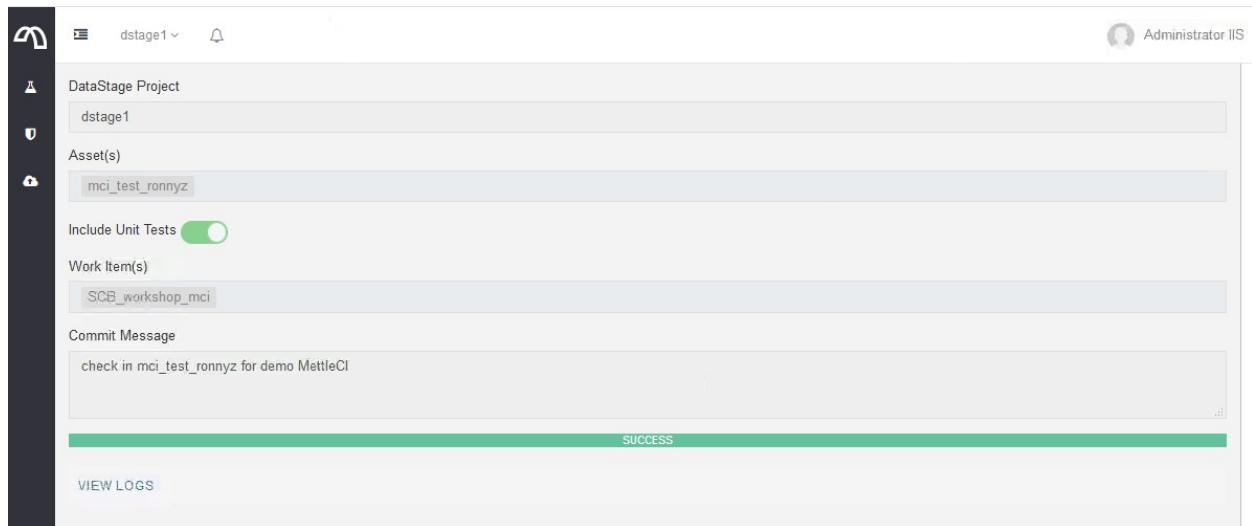
– “Check in mci\_test\_<your name>”

When ready, click “**Check In**”,

please note that check in for the 1<sup>st</sup> time will take a bit longer



- You will see the check-in process running then with a SUCCESS message.



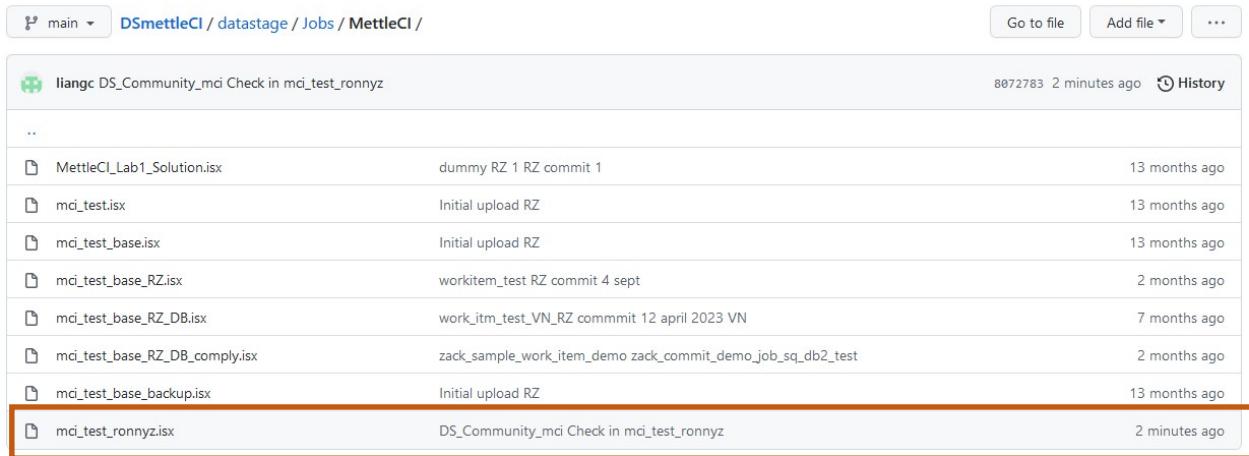
- Refresh your GitHub MettleCI main page. You will see the assets you just check in under “**datastage/Jobs/MettleCI**” and “**unittest/mci\_test\_<your name>**”.

The screenshot shows a GitHub repository for 'liangc DS\_Community\_mci'. It lists several files and folders:
 

- DS\_Community\_mci Check in mci\_test\_ronnyz (8072783, 28 seconds ago)
- compliance (Add files via upload, 2 months ago)
- datastage/Jobs/MettleCI** (DS\_Community\_mci Check in mci\_test\_ronnyz, 29 seconds ago) - This folder is highlighted with a red box.
- unittest (zack\_sample\_work\_item\_demo zack\_commit\_demo\_job\_sq\_db2\_test, 2 months ago)
- Jenkinsfile (Initial upload RZ, 13 months ago)
- README.md (Create README.md, 13 months ago)

 Below the file list, there is a 'README.md' file with an edit icon. The title 'DSmettleCI' is visible at the top of the repository page.

- Click “**datastage/Jobs/MettleCI**” link.  
You will see your job “**mci\_test\_<your name>**” is checked in to the repository with the isx file format of your job.

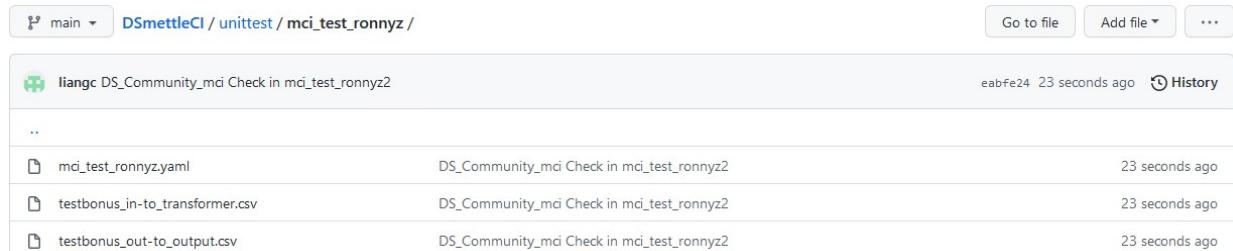


The screenshot shows a GitHub repository interface. At the top, there's a navigation bar with 'main' and a dropdown, followed by the path 'DSmettleCI / datastage / Jobs / MettleCI /'. On the right, there are buttons for 'Go to file', 'Add file', and three dots. Below the path, a commit message from 'liangc' is shown: 'DS\_Community\_mci Check in mci\_test\_ronnyz'. The commit ID is 8072783 and it was made 2 minutes ago. The commit history shows several other files checked in, including 'MettleCI\_Lab1\_Solution.isx', 'mci\_test.isx', 'mci\_test\_base.isx', 'mci\_test\_base\_RZ.isx', 'mci\_test\_base\_RZ\_DB.isx', 'mci\_test\_base\_RZ\_DB\_comply.isx', 'mci\_test\_base\_backup.isx', and 'mci\_test\_ronnyz.isx'. The last file, 'mci\_test\_ronnyz.isx', is highlighted with a red border.

 liangc	DS_Community_mci Check in mci_test_ronnyz	8072783	2 minutes ago	
..				
 mci_test_ronnyz.isx	DS_Community_mci Check in mci_test_ronnyz	2 minutes ago		
 mci_test.isx	Initial upload RZ	13 months ago		
 mci_test_base.isx	Initial upload RZ	13 months ago		
 mci_test_base_RZ.isx	workitem_test RZ commit 4 sept	2 months ago		
 mci_test_base_RZ_DB.isx	work_itm_test_VN_RZ commit 12 april 2023 VN	7 months ago		
 mci_test_base_RZ_DB_comply.isx	zack_sample_work_item_demo zack_commit_demo_job_sq_db2_test	2 months ago		
 mci_test_base_backup.isx	Initial upload RZ	13 months ago		
 mci_test_ronnyz.isx	DS_Community_mci Check in mci_test_ronnyz	2 minutes ago		

- Click “**<your Github username>/mettleci**” link on the top.  
Click “**unittest/mci\_test\_<your name>**” link to review the unit test assets you checked in. You will see below assets are checked in.

- 1. Your unit test input file,**
- 2. Your unit test output file,**
- 3. Your unit test specification in yaml**



The screenshot shows a GitHub repository interface. At the top, there's a navigation bar with 'main' and a dropdown, followed by the path 'DSmettleCI / unittest / mci\_test\_ronnyz /'. On the right, there are buttons for 'Go to file', 'Add file', and three dots. Below the path, a commit message from 'liangc' is shown: 'DS\_Community\_mci Check in mci\_test\_ronnyz2'. The commit ID is eabfe24 and it was made 23 seconds ago. The commit history shows three files checked in: 'mci\_test\_ronnyz.yaml', 'testbonus\_in-to\_transformer.csv', and 'testbonus\_out-to\_output.csv'.

 liangc	DS_Community_mci Check in mci_test_ronnyz2	eabfe24	23 seconds ago	
..				
 mci_test_ronnyz.yaml	DS_Community_mci Check in mci_test_ronnyz2		23 seconds ago	
 testbonus_in-to_transformer.csv	DS_Community_mci Check in mci_test_ronnyz2		23 seconds ago	
 testbonus_out-to_output.csv	DS_Community_mci Check in mci_test_ronnyz2		23 seconds ago	

**THIS IS THE END OF THE  
HANDS ON LAB**