

---

# **Facility Reconciliation Tool documentation**

***Release 0.9***

**IntraHealth International**

**Jun 26, 2019**



---

## Contents

---

<b>1</b>	<b>Quick Starts</b>	<b>3</b>
1.1	Example data . . . . .	3
1.2	Match data sources . . . . .	6
1.3	Add DHIS2 source . . . . .	10
<b>2</b>	<b>User Guide</b>	<b>15</b>
2.1	Data Sources . . . . .	15
2.2	Match . . . . .	16
2.3	Users and sharing . . . . .	18
<b>3</b>	<b>Developer Guide</b>	<b>21</b>
3.1	DHIS2 app installation . . . . .	21
3.2	DHIS2 users and sharing . . . . .	28
3.3	Quickstart with Docker . . . . .	33
3.4	Install Locally . . . . .	34
3.5	Vagrant . . . . .	35
3.6	Production considerations . . . . .	35
3.7	Ansible . . . . .	36
3.8	Terraform . . . . .	38
3.9	Contribute . . . . .	38
3.10	Update and Build Documentation . . . . .	39
<b>4</b>	<b>FAQ</b>	<b>43</b>
4.1	Is there an API? . . . . .	43
4.2	Can this tool be used in education or agriculture? . . . . .	43
4.3	Does the tool clean the source data? . . . . .	43
4.4	Can I run the tool on my own PC? . . . . .	43
<b>5</b>	<b>Roadmap</b>	<b>45</b>
5.1	Version: 0.1.0 . . . . .	45
5.2	Version: 0.2.0 . . . . .	45
5.3	Version: 0.3.0 . . . . .	46
5.4	Version: 0.4.0 . . . . .	46
5.5	Version: 0.5.0 . . . . .	46
5.6	Version: 0.6.0 . . . . .	46
5.7	Version: 0.7.0 . . . . .	47



The Facility Reconciliation Tool is an open source and open standards-based product used to compare lists of facilities from different data sources. The tool supports uploading CSV, and connecting to FHIR servers and DHIS2.

The tool can be used as a standalone application with its own authentication or as an easily installable DHIS2 app that uses DHIS2 for authentication that runs the tool in the background.

If you are new to the tools, please try the quick starts!

For support, please have a look through this guide, including the FAQ.

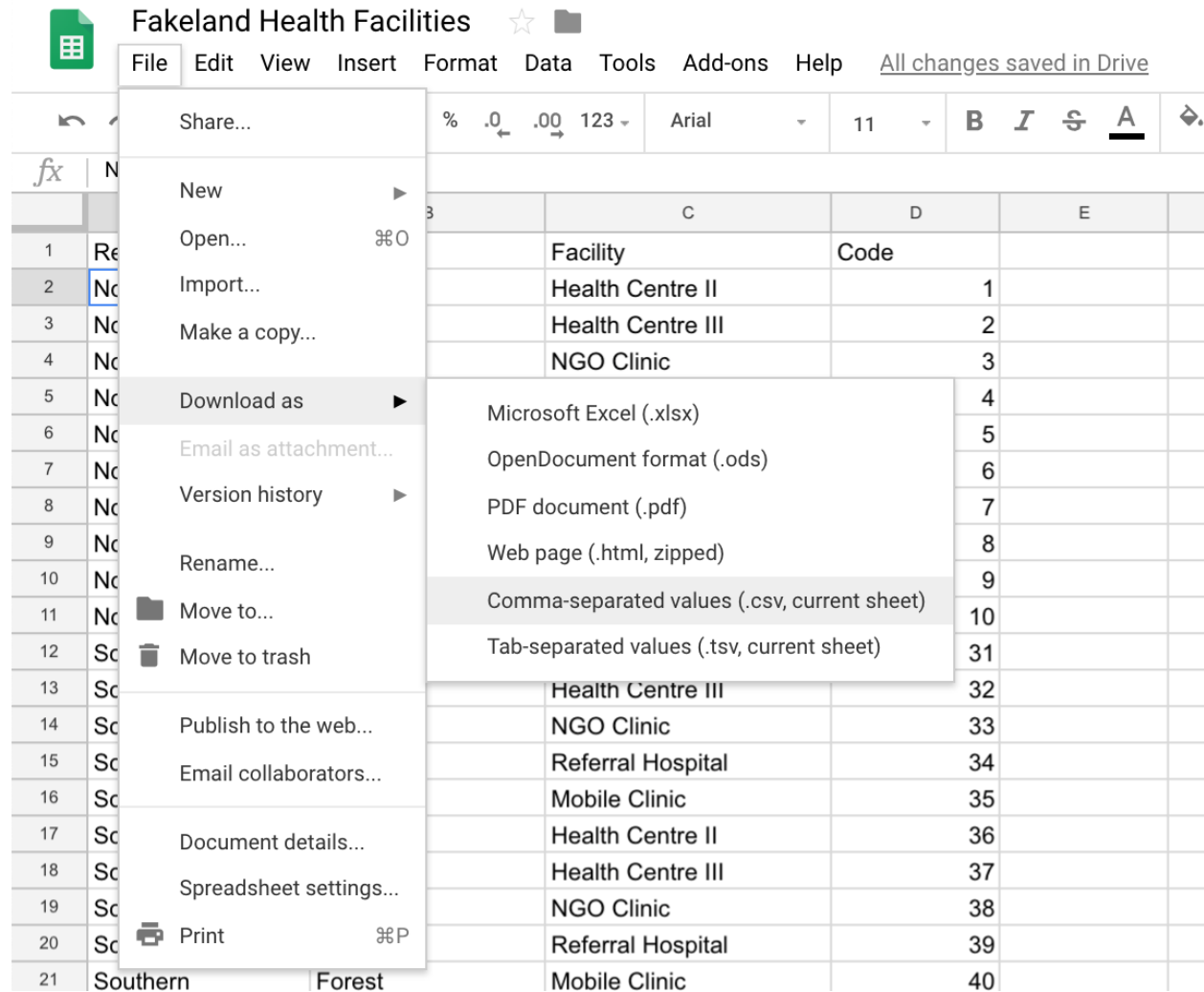
- For announcements and discussions, join the [Facility Registry Google Group](#).
- For monthly discussions about facility registries, join the [OpenHIE Facility Registry Community](#).
- Create an issue in the [GitHub Repository](#).



## 1.1 Example data

### 1.1.1 Download the example data

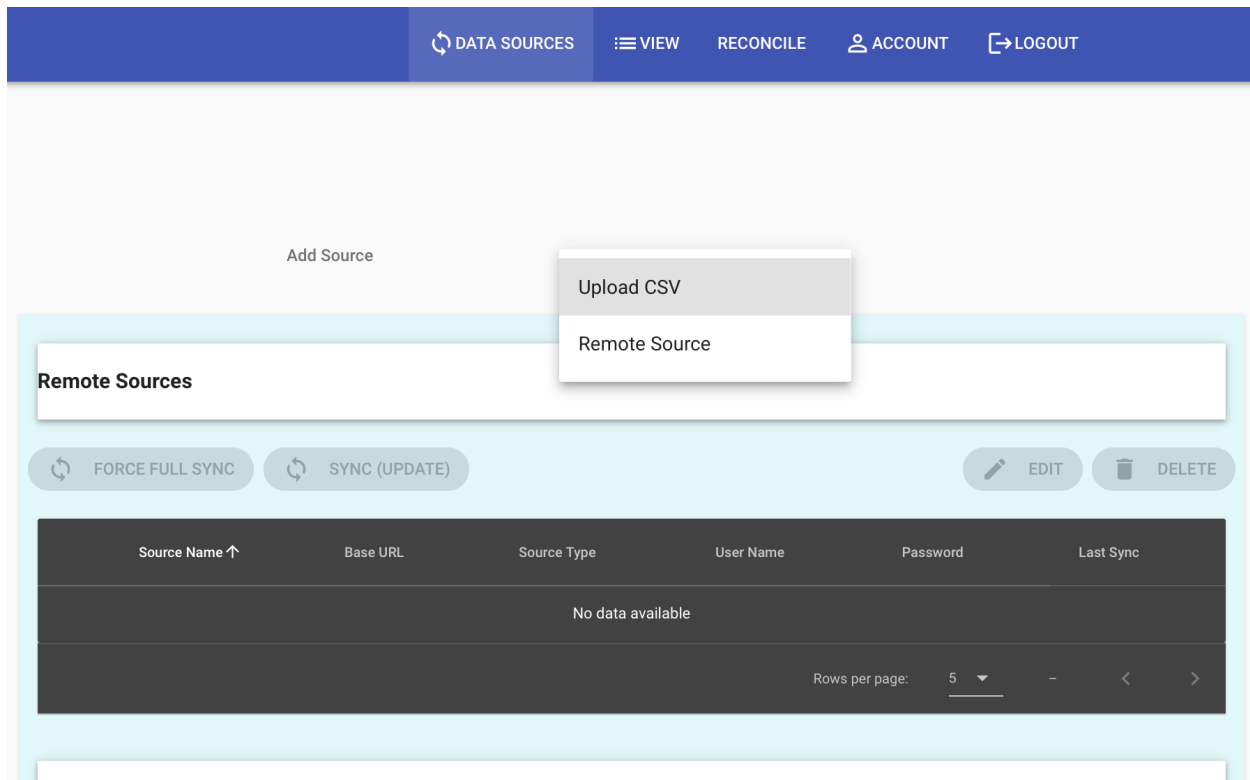
- Go to the example workbook for [Fakeland Health Facilities data](#).
- There are two worksheets inside (see the tabs at the bottom), one for Source 1 NGO and one for Source 2 DHIS2. These are fake data for use in the tool.
- Download each tab as a separate CSV files to your computer by clicking under the File menu and choosing Download as: Comma-separated values
- Save the files to your computer in an easily accessible folder.



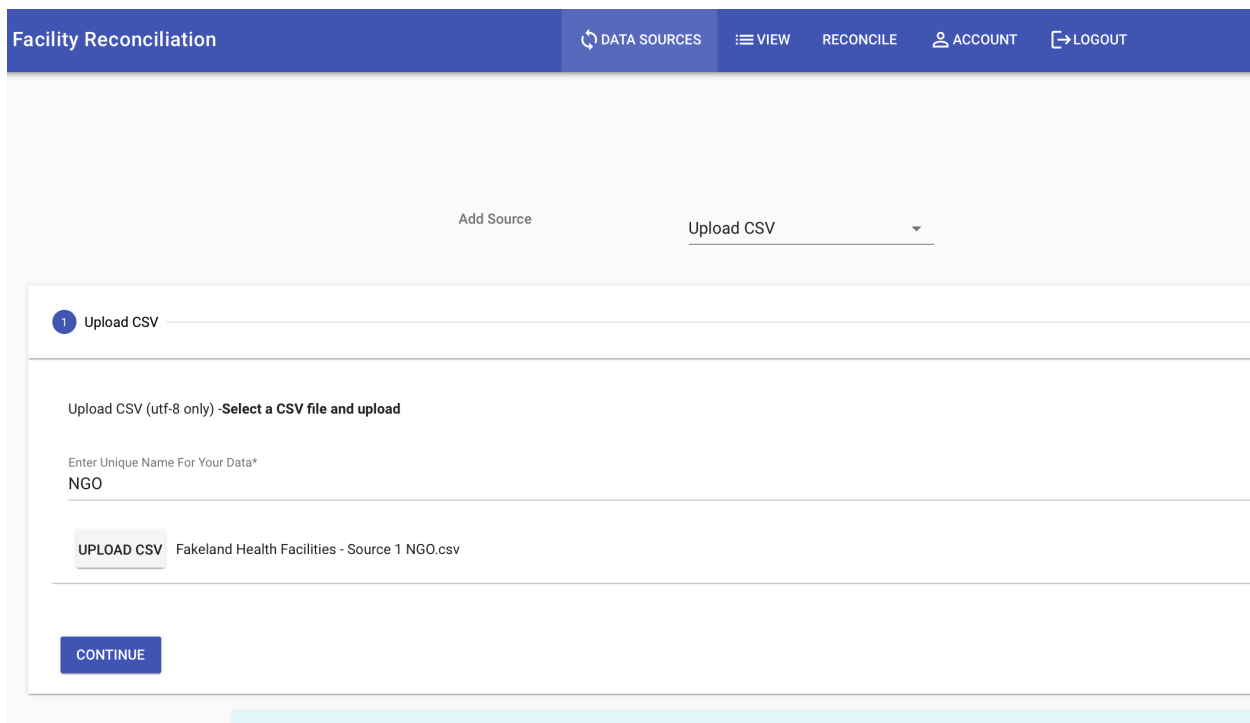
## 1.1.2 Upload Data

- Go to the reconciliation tool from the site: <https://facilitymatch.org> and click DEMO. You will be shown a disclaimer. If you agree, then click to continue.
- On the hosted version, login using demo:demo for user:password.
- On the site, choose Add Source in the top center of the page and select Upload CSV.





- Name the upload as desired and click Upload CSV. Select the Fakeland Health Facilities - Source 1 NGO file from where you saved it on your computer.
- Click Continue. The data will be loaded into the system.



- You will be in a new dialog that asks you to assign headers for facility name, facility ID code, and administrative

levels. Please select the correct headers from the dropdown menu as indicated in the screenshot below.

Map an appropriate CSV header against those on the app.

CSV Header	App Header
Facility*	Facility
Code	Code
Latitude	Select
Longitude	Select
Level 1*	Region
Level 2*	District

GO BACK UPLOAD

- Click the blue Upload button.
- Repeat the above steps for uploading data but this time call the upload DHIS2 and choose the file Fakeland Health Facilities - Source 2 DHIS2 file on your computer.
- You should now have two data sources.

CSV Name	Source
DHIS2	
NGO	

Rows per page: 5 1-2 of 2

## 1.2 Match data sources

To match facility lists the steps are to first create a pair of data sources and then to do the reconciliation starting at the highest level and going down the hierarchy to the last level.

### 1.2.1 Pair data sources

- Go to the Reconcile tab and choose Create and Manage Data Source Pair. Select both data sources. The data source you select under Source 1 source of truth. The data source you select under Source 2 is the one that you

want to fix.

- Choose DHIS2 as Source 1 and NGO as Source 2.

- Click Save and you will be taken to the Reconcile page.

Note. After saving the pair should be active. If not, then below the pair choose Active under Existing Data Source Pairs. Then click Activate Pair.

## 1.2.2 Reconciliation

- There are only two regions (the top administrative level in the fakeland data). They are automatically matched. The status wheels labeled Matched indicate that 2 of 2 regions have been matched. In the bottom portion of the screenshot below, you can also see that for Level 1 there is 100% match, as the two locations in Source 1 match the two locations in Source 2.
- Near the bottom left click Proceed to District to continue.

CSV EXPORT

FHIR EXPORT

Reconciling Region

Region

RECALCULATE SCORES ?

Source 1 Reconciliation Status

Matched

2/2

100%

Flagged

0/2

0%

Unmatched

0/2

0%

No Match

0/2

0%

Source 1 Unmatched

Search

Location ↑

No data available

Rows per page: 5

Source 2 Unmatched

Search

Location ↑

No data available

Rows per page: 5

Source 2 Reconciliation Status

Matched

2/2

100%

Flagged

0/2

0%

Unmatched

0/2

0%

Not in Source 1

0

0%

MATCHED (2)

NO MATCH (0)

FLAGGED (0)

Search

Source1 Location ↑	Source1 ID	Source2 Location	Source2 ID	
Northern	fe87b7c5-9a68-5515-b85c-44b8693aa308	Northern	fe87b7c5-9a68-5515-b85c-44b8693aa308	Break Match
Southern	a2649efe-5f56-5928-a3fb-4bd2f1d4f29d	Southern	a2649efe-5f56-5928-a3fb-4bd2f1d4f29d	Break Match

Rows per page: 5 1-2 of 2

PROCEED TO DISTRICT

- At Level 2, the districts will automatically be matched. Click Proceed to Facility to continue.

Source 1 Reconciliation Status

Matched

4/4

100%

Flagged

0/4

0%

Unmatched

0/4

0%

No Match

0/4

0%

Source 1 Unmatched

Search

Location ↑

No data available

Rows per page: 5

Source 2 Unmatched

Search

Location ↑

No data available

Rows per page: 5

Source 2 Reconciliation Status

Matched

4/4

100%

Flagged

0/4

0%

Unmatched

0/4

0%

Not in Source 1

0

0%

MATCHED (4)

NO MATCH (0)

FLAGGED (0)

Search

Source1 Location ↑	Source1 ID	Source2 Location	Source2 ID	
Beach	b222f85c-ee6f-5114-9cc6-ac8c31955587	Beach	b222f85c-ee6f-5114-9cc6-ac8c31955587	Break Match
Forest	cc1612e2-6061-5a95-858d-d9c17a7058e8	Forest	cc1612e2-6061-5a95-858d-d9c17a7058e8	Break Match
Lake	ee6f3a2dc-6a85-5f3f-8265-92d4427f6c92	Lake	ee6f3a2dc-6a85-5f3f-8265-92d4427f6c92	Break Match
Mountain	fbdbf129-5ab9-5e3f-80c4-efbad16c9bb1	Mountain	fbdbf129-5ab9-5e3f-80c4-efbad16c9bb1	Break Match

Rows per page: 5 1-4 of 4

PROCEED TO FACILITY

- Click Proceed to Level 3.
- At Level 3, some facilities will not match.
- There is a list of unmatched facilities in the light green box for Source 1.

The screenshot shows the 'Reconciling Facility' interface. At the top, there are buttons for 'CSV EXPORT' and 'FHIR EXPORT'. The main area is divided into three panels. The left panel shows 'Source 1 Reconciliation Status' with a 'Matched' status of 15/20 (75%) and an 'Unmatched' status of 5/20 (25%). The middle panel shows 'Source 1 Unmatched' facilities, including 'General Referral Hospital', 'Lake Health Center II', 'Mtnn Health Center II', 'New NGO Clinic', and 'Old Mobile Clinic'. The right panel shows 'Source 2 Reconciliation Status' with a 'Matched' status of 15/20 (75%) and an 'Unmatched' status of 5/20 (25%). Below these panels is a table of 'MATCHED (15)' facilities. The table has columns for 'Source1 Location', 'Source1 ID', 'Source2 Location', 'Source2 ID', and a 'Break Match' button. The table lists five rows of matched facilities, all with a 'Break Match' button.

- Select one of the facilities, like General Referral Hospital.
- A pop-up dialog will let you choose which facility to match it to.

The screenshot shows a pop-up dialog titled 'Matching General Referral Hospital'. It has a search bar and a 'Parents: Mountain->Northern' filter. The dialog displays a table of potential matches. The table has columns for 'Source 2 Location', 'ID', 'Parent', 'Geo Dist (Miles)', and 'Score'. There are two rows of matches: 'Referral Hospital' with a score of 8 and 'Health Centre II' with a score of 19. Each row has a 'FLAG' button and a 'SAVE MATCH' button. At the bottom of the dialog, there is a 'NO MATCH' button, a 'SHOW ALL SUGGESTIONS' button, and a 'BACK' button.

- Select the most appropriate match, in this case it is Referral Hospital, and click Save Match.
- Go through the unmatched facilities in Source 1, select and save the best matches. One you are done, you will see:

The screenshot displays the Facility Reconciliation Tool interface. At the top, there are two status boxes for Source 1 and Source 2. Source 1 shows 100% Matched (20/20) and 0% Unmatched (0/20). Source 2 shows 100% Matched (20/20) and 0% Unmatched (0/20). Below these are search filters for Source 1 Unmatched and Source 2 Unmatched. The Source 1 Unmatched filter shows 'Northern' and 'Southern' locations. The Source 2 Unmatched filter shows 'No data available'. Below the filters is a table with columns: Source1 Location, Source1 ID, Source2 Location, Source2 ID, and a 'Break Match' button. The table contains five rows of data, all with 'Break Match' buttons.

Source1 Location	Source1 ID	Source2 Location	Source2 ID	Break Match
General Referral Hospital	59	Referral Hospital	9	Break Match
Health Centre II	81	Health Centre II	31	Break Match
Health Centre II	86	Health Centre II	36	Break Match
Health Centre III	52	Health Centre III	2	Break Match
Health Centre III	57	Health Centre III	7	Break Match

- The Source 1 Reconciliation Status, in the top left-hand status box, should be 100%.

## 1.2.3 Export a reconciliation report

- On the top left of the reconciliation tab there is an option to output either a FHIR-based report of the final reconciled dataset or a CSV of what matched and did not match in either dataset. Choose CSV Export. You will be able to choose three files of matches and unmatches.

## 1.2.4 View reconciliation status

- Select Reconciliation Status and view the overall status.

The screenshot displays the Facility Reconciliation Tool interface in the 'Reconciliation Status' view. At the top, there are two status boxes for Source 1 and Source 2. Source 1 shows 100% Matched (26/26) and 0% Unmatched (0/26). Source 2 shows 100% Matched (26/26) and 0% Unmatched (0/26). Below these are search filters for Source 1 Unmatched and Source 2 Unmatched. The Source 1 Unmatched filter shows 'Northern' and 'Southern' locations. The Source 2 Unmatched filter shows 'No data available'. Below the filters is a table with columns: Source1 Location, Source1 ID, Source2 Location, Source2 ID, and a 'Break Match' button. The table contains two rows of data, both with 'Break Match' buttons.

Source1 Location	Source1 ID	Source2 Location	Source2 ID	Break Match
Northern	fe87b7c5-9a68-5515-b85c-44b8693aa308	Northern	fe87b7c5-9a68-5515-b85c-44b8693aa308	Break Match
Southern	a2649efe-5f56-5928-a3fb-4bd2f1d4f29d	Southern	a2649efe-5f56-5928-a3fb-4bd2f1d4f29d	Break Match

## 1.3 Add DHIS2 source

This example uses the DHIS2 Playground which is already loaded with fake but realistic facility data.

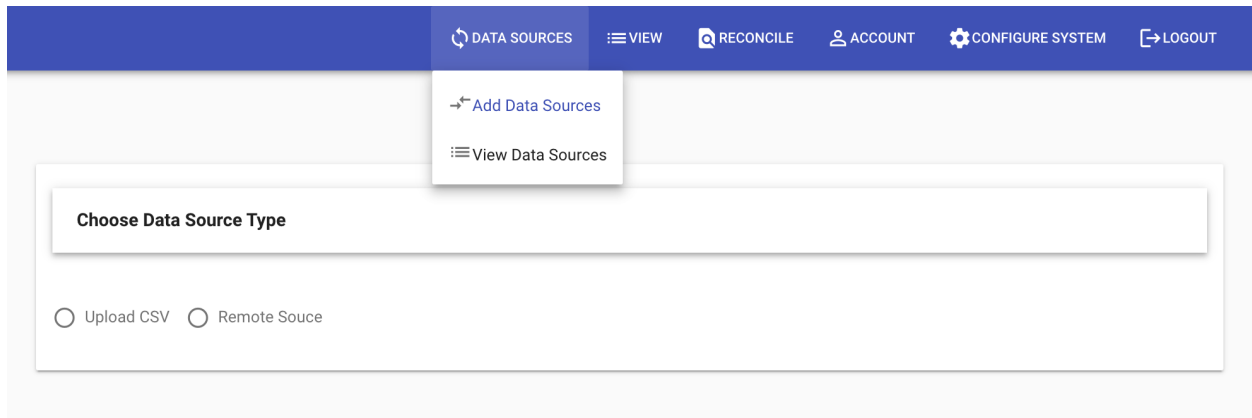
### 1.3.1 Configure DHIS2 Data Source

To connect to any DHIS2, you need a username and password and the URL (website address) of the DHIS2. This example uses the demo playground provided username and password.

Any DHIS2 server can be used as a source for which the user can authenticate and is authorized to get the facility and hierarchy.

The Base URL is the website for the DHIS2. The example URL for the playground may be a bit confusing because it has '/2.31.3' but that is because the playground is hosting many DHIS2 versions at the same address like '/2.32' etc. In other DHIS2, there will be no '/version'.

- Open the Facility Reconciliation Tool. Select 'Data Sources'



The screenshot shows the Facility Reconciliation Tool interface. At the top is a blue navigation bar with the following items: 'DATA SOURCES' (active), 'VIEW', 'RECONCILE', 'ACCOUNT', 'CONFIGURE SYSTEM', and 'LOGOUT'. Below the navigation bar, a dropdown menu is open under 'DATA SOURCES', showing 'Add Data Sources' and 'View Data Sources'. The main content area is titled 'Choose Data Source Type' and contains two radio buttons: 'Upload CSV' and 'Remote Source'.

- Select Remote Source
- In the Add New Remote Source form, put in the following:

Add New Remote Source

Source Type

DHIS2

Source Name

DHIS2 play

Base URL

https://play.dhis2.org/2.31.3/

Username

admin

Password

.....

CLEAR

ADD

- Click Add.

### 1.3.2 Sync

Select View Data Sources from the Data Sources tab.

The data will not be pulled from DHIS2 unless you Force Sync or Sync Update.

- Click next to the DHIS source you added to select it.
- Click Sync Update.



Remote Sources

FORCE FULL SYNC

SYNC (UPDATE)

EDIT

DELETE

	Source Name ↑	Base URL	Source Type	User Name	Password	Last Sync	Owner	Shared To	Created Time	
	DHIS2 play	https://play.dhis2.org/2.31.3/	DHIS2	admin	*****		admin		1st May 2019 5:03:08 pm	SHARE

Rows per page: 5

1-1 of 1

The Facility Reconciliation Tool will now add the data from the DHIS2 source. To continue, add another another data source or if there is already one you wish to match, then create a data pair.



## 2.1 Data Sources

*Note that there is a quick start in this documentation on how to upload CSV.*

The tool supports CSV as a file source, as well as remote connections to DHIS2 and FHIR servers.

### 2.1.1 Upload CSV

The CSV file should have columns names in the first row of the file. Empty lines should be removed. The CSV file should be encoded as Unicode (utf-8) as that is what is used internally in the backend. If some entities are encoded in another format then matches that appear to be the same may not match as expected.

Latitude and longitude are optional columns. If they are included they will be used to facilitate manual matching but they are not used or required for automatic matching.

Once uploaded, in the View tab, CSV entries can be edited. Any edits do not modify the original data source but the edits will be exported after reconciliation.

### 2.1.2 Select Levels

The user may choose any levels in their hierarchy to include but they must be ordered with the top most level first.

It is also possible to select no levels to match on a flat list with no hierarchy. To do so, don't select levels.

### 2.1.3 Remote Servers – DHIS2 or FHIR

The tool supports remote sources. Any DHIS2 or FHIR server can be used as a source if the user has credentials to access it.

Extensive compatibility testing has not been performed but DHIS2 versions  $\geq 2.22$  should be supported. Please contact the maintainers if there is an issue.

FHIR is supported for STU3 and R4 support is anticipated. Other FHIR servers may be added in future versions of the tool, such as the HAPI FHIR server.

## 2.2 Match

*Note that there is a video tutorial on how to match on the [OpenHIE YouTube channel](#) and a quick start in this documentation.*

Reconciling data sources involves choosing a pair of sources to work with and then running the automatic or manual matching tool. Any kind of match can be undone.

### 2.2.1 Pair sources

Under the Reconcile tab is the Create and Manage Data Source page. On the pair tab select one source on the left and one on the right. The source on the left (source 1) is the leader – the source of truth. The source on the right is the follower, the source that is meant to be cleaned.

Create/Choose a pair of data sources to use for reconciliation. Source 1 is the source while source 2 is the target

### Choose Data Source Pair

Source 1	Source 2
<input checked="" type="radio"/> dhis2 test	<input type="radio"/> dhis2 test
<input type="radio"/> ngo test	<input checked="" type="radio"/> ngo test

1-2 of 2
<
>

Rows per page: 5

### Existing Data Source Pairs

Search

Pair ↑	Owner	Active	Shared To
--------	-------	--------	-----------

In the pairing process it is possible to share the pair with another user who may join in helping to match, for example where they are familiar with a specific area.

### 2.2.2 Automatic matching

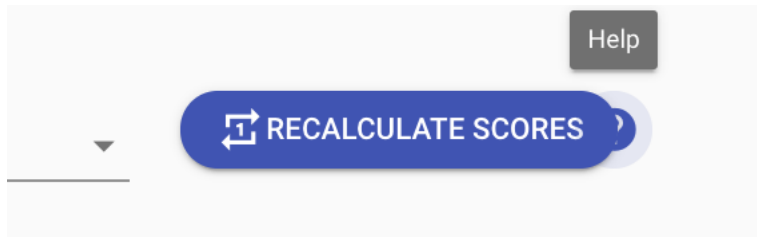
When the reconciliation process starts it uses automatic matching. Matching proceeds like this:

- The first level matches the highest administrative area names (termed region in the tool) using the [Levenshtein distance](#).

- The second level matches based on the first level and also the Levenshtein distance for the second level names, termed district in the tool.
- The final level matches based on the second level (which was already matched according to the level above it) as well as the Levenshtein distance.

### 2.2.3 Recalculate scores

During the matching process at any level, it is possible to ask the tool to match unmatched entities using the Recalculate Scores button. This process does not remove matches.



One common use for this is after manual matching of any entities, to rerun the matching process and incorporate the results. This can also be used when an entity is freed for matching after having been previously flagged.

### 2.2.4 Manual matching

Manual matching brings up a dialog box to choose options. If latitude and longitude coordinates were provided in the data sources, it additionally scores matches based on the [haversine formula](#) for shortest path across a sphere (geodesic distance) between the points. This is not used in the automatic matching but is provided for manual matching.

Any administrative area or facility match may be broken. If this is desired, click Recalculate Scores to rebuild the scoring index and manually match or flag as desired.

### Parent constraints

The default is to match facilities between sources based on hierarchies only. This means that if a facility is in the wrong nested administrative level, it will not be matched.

Under the Configure System tab the parent constraint can be disabled to allow for matching of all facilities across the sources.

### Reconcile without levels

It is possible to match facilities from a flat list with no hierarchies.

### Notifications

When admins share a set of facilities for matching to a data manager, the data manager is notified by email when the matching is completed.

## Flagging

Flagging allows for an export of facilities that require further examination and research. When a flag is set, the user may also include a comment.

Flags may be set on any entity. For example, an organizational unit or a facility.

In order to see flagged items and their comments click the FLAGGED tab on the bottom of the RECONCILE menu for the level of interest. In this example, an org unit is flagged.

<div> <div> </div> <div> </div> <div> </div> </div>				
<div> <div>MATCHED (0)</div> <div>NO MATCH (0)</div> <div>IGNORED (0)</div> <div>FLAGGED (1)</div> </div>				
Search				
Source 1 Location ↑	Source 1 ID	Source 2 Location	Source 2 ID	Comment
Sierra Leone	lmsPTQPwCqd	Sierra Leone	lmsPTQPwCqd	This facility is no longer in use.
				<div> <div>  Confirm Match </div> <div>  Release </div> </div>
				<div> <div>Rows per page: 5</div> <div>1-1 of 1</div> </div>

Also in this example, once flagged, a facility or administrative unit may be released back for matching to another entity, or the match that was flagged may be confirmed.

## 2.3 Users and sharing

There are two classes of users, admin and data managers. The initial server configuration includes an administrator role. This account and password should be immediately changed after installation. See the Developer Guide for more information.

### 2.3.1 Roles

- **Admin** accounts can add users, configure the system, and all tasks of data managers.
- **Data manager** accounts can manage data only. They can share data sources and do other matching tasks.
- **Custom** accounts with specific roles may be created in the future. Please let the community know if you have use cases for custom account roles.

### 2.3.2 Self-registration

Administrators may configure the system to allow for anyone to self-register. This is under the Configure System tab. This option is off by default.

### 2.3.3 Share a data source

Data sources may be shared by location to others or entirely.

On the Data Sources tab, select View Data Sources.

Select the Share button on the far right.

Uploaded Sources

EXPORT

DELETE

	Source Name ↑	Owner	Shared To	Created Time	
<input type="radio"/>	dhis2 test	root@gofr.org		4th Jun 2019 7:36:30 pm	<a href="#">SHARE</a>
<input type="radio"/>	ngo test	root@gofr.org		4th Jun 2019 7:37:00 pm	<a href="#">SHARE</a>

Rows per page: 5 1-2 of 2

This brings up a dialog to share data sources by user and location.

Sharing dhis2 test

▼ Select location to limit sharing

> Northern

> Southern

No file chosen

🔒 Limiting Sharing to: **No limit**

Search

Username ↑	Firstname	Surname
1-1 of 1		

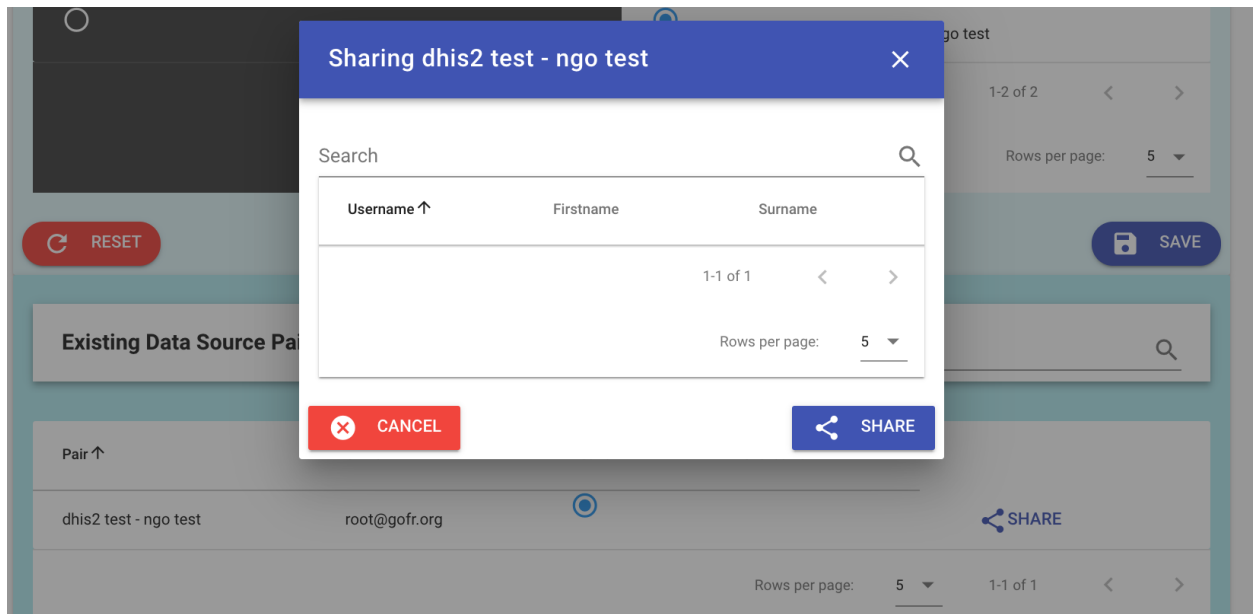
Rows per page: 5

CANCEL

SHARE

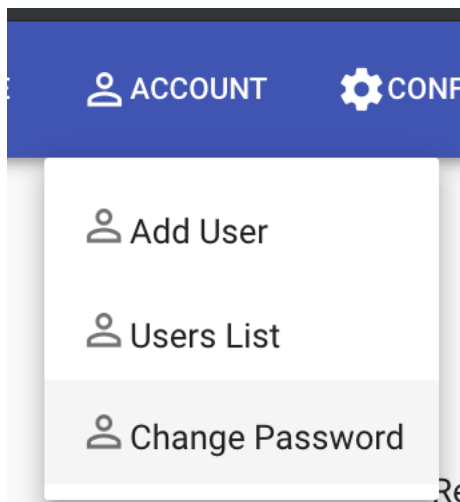
### 2.3.4 Add to a Pair

Data managers and administrators are able to add other users to help with reconciliation. To do so, the data source pairing has an option to share the pairing with another user.



### 2.3.5 Password reset

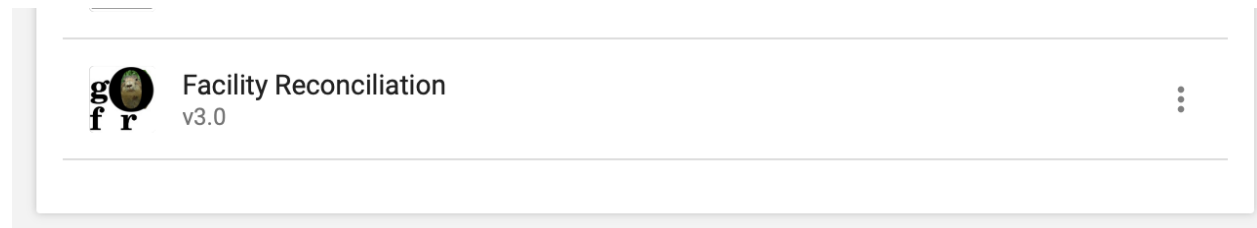
Under the Accounts tab a user password can be reset. The reset password is the user's surname exactly as it was entered including case.





### 3.1 DHIS2 app installation

DHIS2 is a popular open source and open standards-based HMIS in global health. The Facility Reconciliation Tool can be installed as a DHIS2 app and configured as required to restrict users to the organization units they need to access.



#### 3.1.1 Features

The Facility Reconciliation DHIS2 App has the ability to:

- Authenticate only using DHIS2 users and roles.
- Add DHIS2 users to data sources and data pairs.
- Restrict DHIS2 users to specific data pairs or one pair.
- Restrict users to specific organizational units in DHIS2 and have this reflected in access in the Facility Reconciliation Tool.

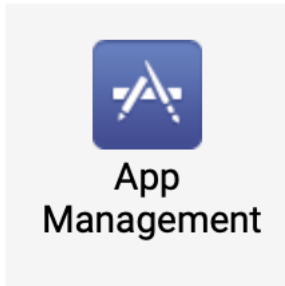
#### 3.1.2 Install DHIS2 app on the same server running DHIS2

For the Facility Reconciliation Tool to be used as a DHIS2 app, the standalone tool itself must be installed. It is expected that the tool is installed on the same server as DHIS2.

Clone the GitHub repository.

```
git clone git@github.com:openhie/facility-recon.git
cd facility-recon
```

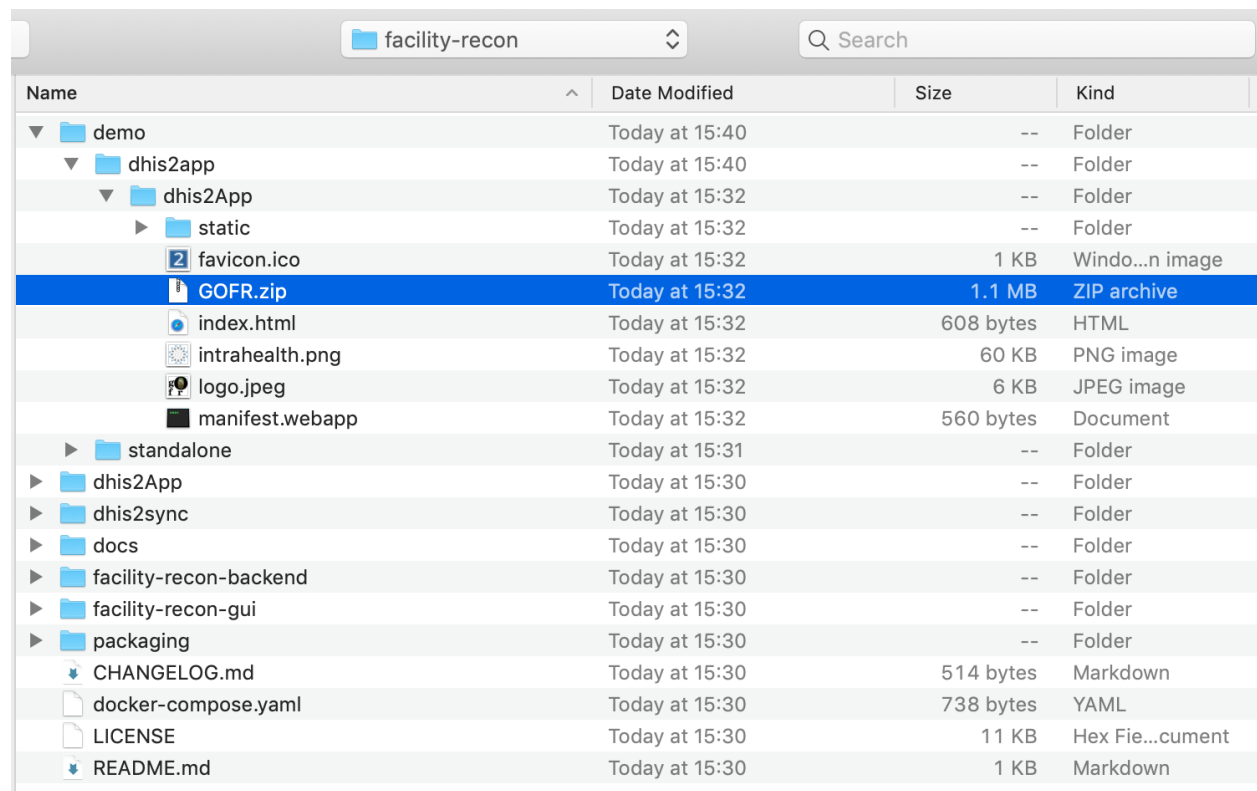
Log in as an admin in DHIS2 with permission to install apps. Open the app called App Management.



Choose to install an app using the button on the far right.



A file chooser will open. Select the already packaged .zip file in the GitHub repository folder dhis2App.



To install the app on the DHIS2 demo playground you need a different package, see modifications below.

To point the app at another server for the backend you need a different package, see modifications below.



Facility Reconciliation  
v3.0

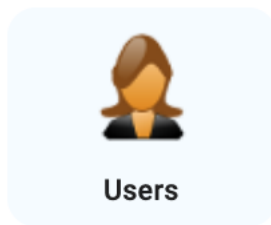
Now the DHIS2 app is installed. Confirm:

**After installing the app but before setting up DHIS authentication, set the user role in DHIS2 to have permissions.**

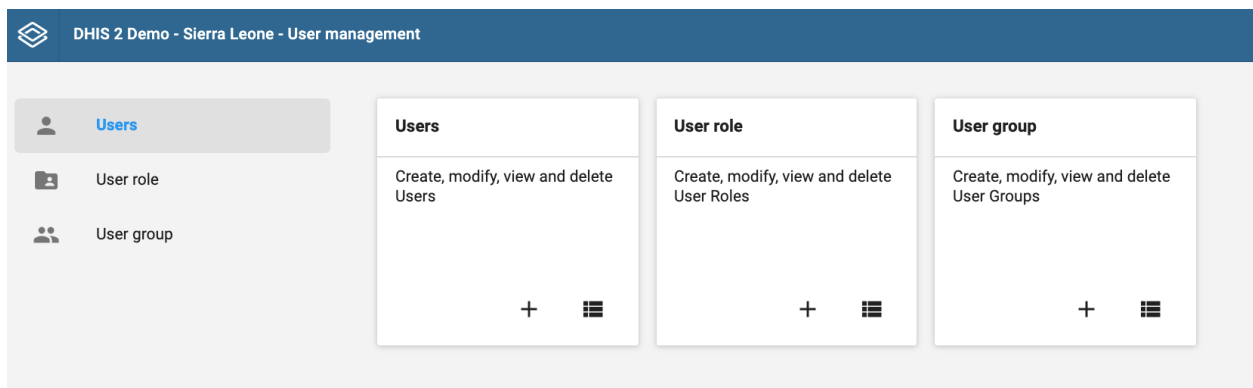
### 3.1.3 Add app permission to User Role

External authentication works by associating a DHIS2 user role. The Facility Reconciliation Tool must know the user role. The user role that is selected to manage the tool must have permissions added otherwise if it is not then the user will be locked out.

Click on the Users app in the app menu dropdown.



In the users app menu, select User Roles.



Select the name of the user role you will use to manage the Facility Reconciliation App. This will take you to a detailed view of the authorities granted to that user role.

Under Apps check the box next to Facility Reconciliation app.

## Apps

Name
<input checked="" type="checkbox"/> Data Visualizer app
<input type="checkbox"/> Datastore app
<input type="checkbox"/> Event Reports app
<input type="checkbox"/> Event Visualizer app
<input checked="" type="checkbox"/> Facility Reconciliation app
<input type="checkbox"/> Function Maintenance app
<input checked="" type="checkbox"/> GIS app
<input type="checkbox"/> Immunization analysis app
<input type="checkbox"/> Interactive Dashboard 2 app
<input checked="" type="checkbox"/> Interpretation app
<input type="checkbox"/> Maintenance app
<input checked="" type="checkbox"/> Maps app

Click Save at the bottom on the page to submit the change.

SAVE

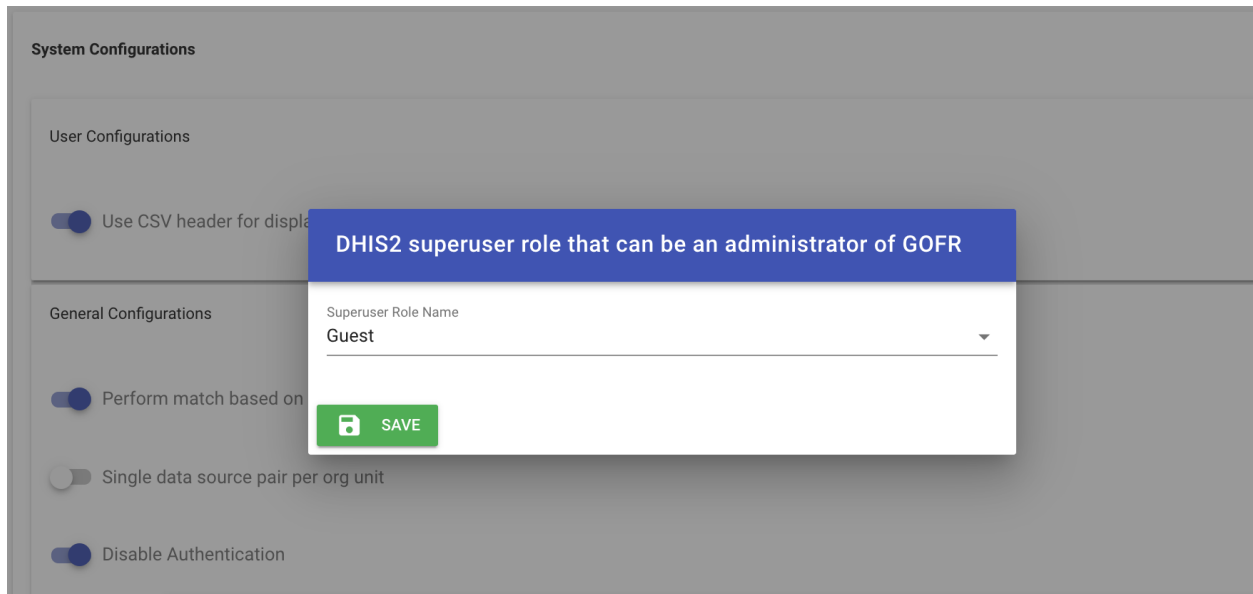
CANCEL

### 3.1.4 Setup DHIS2 authentication

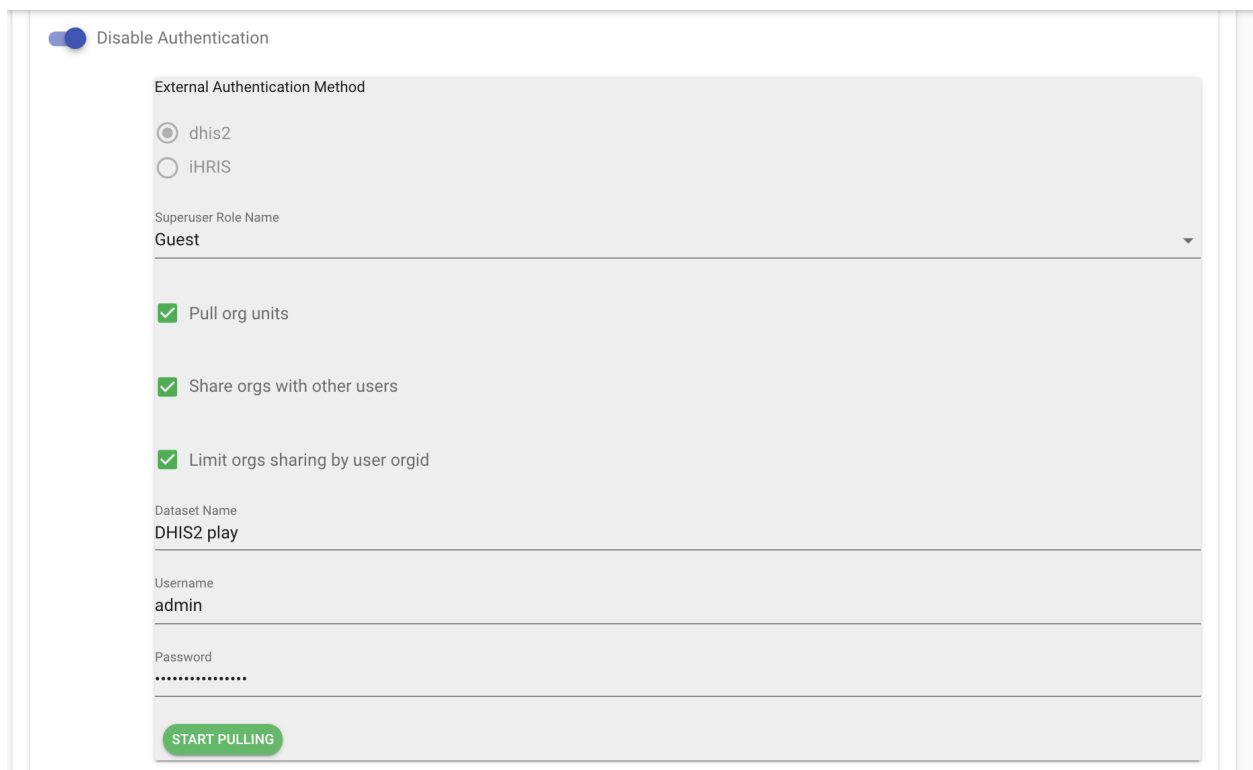
Back on the apps dropdown, click on the now-installed Facility Reconciliation app icon to finish configuration and setup authentication.

Under Configure Settings turn on external authentication. Select DHIS2 as the authentication point.

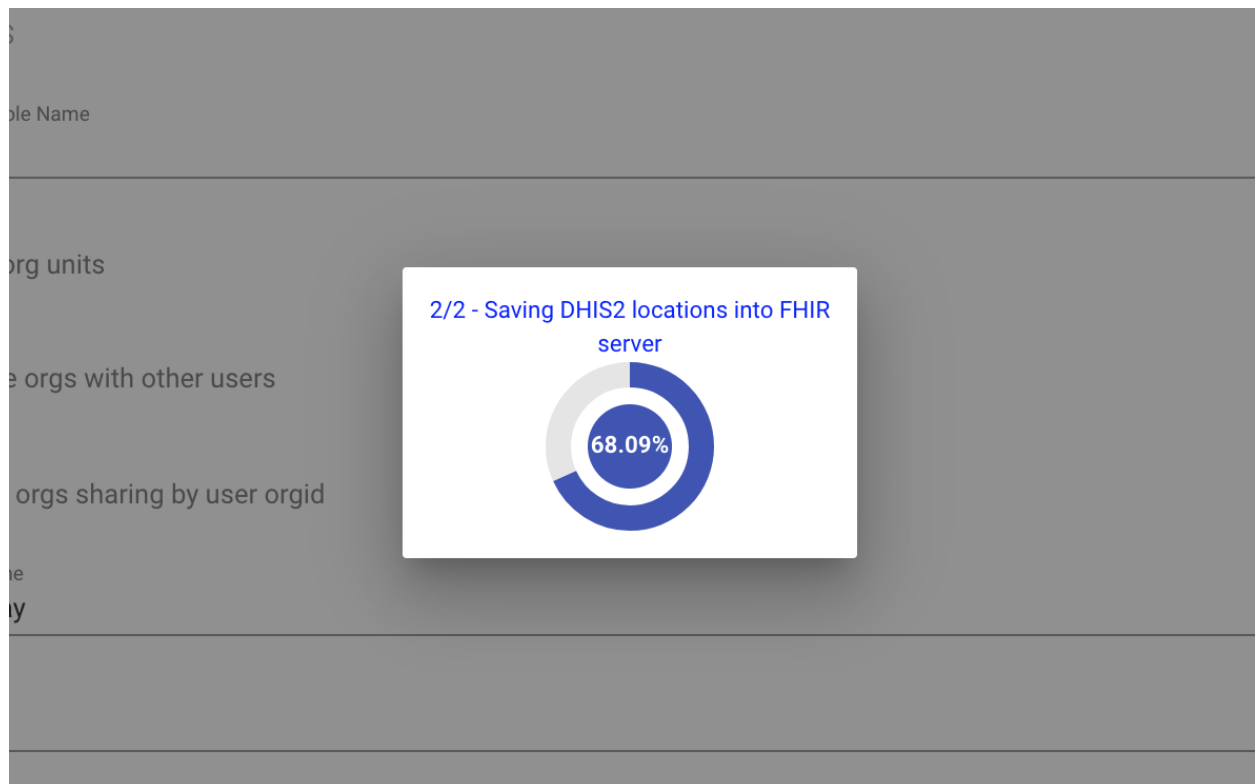
Select the superuser role in DHIS2 that will be used to administer facility reconciliation.



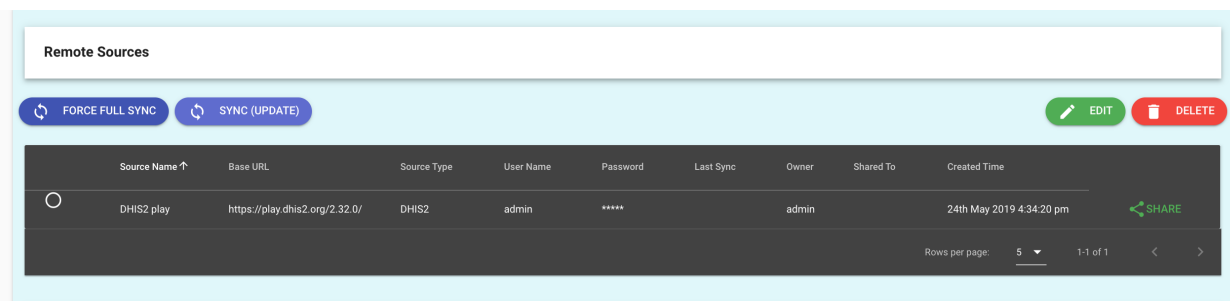
Then, enter a username and password that the backend will use to pull data from DHIS2.



Select Pull to get data into the app. This is necessary for two-way communication to work between the app and the reconciliation backend.



Once the app is done it will redirect to the data sources view to confirm the DHIS2 data source is now there.



### 3.1.5 Install DHIS2 app on DHIS playground

The Facility Reconciliation Tool should be running. A quick way to do this for a demo is to use Docker.

Visit <https://play.dhis2.org> and select a DHIS2 demo version.

Name	Date Modified	Size	Kind
▶ dhis2App	Today at 12:27	--	Folder
▶ dhis2sync	Today at 12:27	--	Folder
▶ docs	Today at 12:27	--	Folder
▼ examples	Today at 12:33	--	Folder
▼ dhis2	Today at 12:27	--	Folder
dhis2-play-demo.zip	Today at 12:27	1.1 MB	ZIP archive
▶ standalone	Today at 12:27	--	Folder
▶ facility-recon-backend	Today at 12:27	--	Folder
▶ facility-recon-gui	Today at 12:27	--	Folder
▶ packaging	Today at 12:27	--	Folder
CHANGELOG.md	Today at 12:27	514 bytes	Markdown
docker-compose.yaml	Today at 12:27	738 bytes	YAML
LICENSE	Today at 12:27	11 KB	Hex Fie...cument
README.md	Today at 12:27	1 KB	Markdown

Follow the above recipe but use the prebuilt version designed for DHIS2 playground.

Name	Date Modified	Size	Kind
▼ examples	Today at 19:12	--	Folder
▶ standalone	Today at 19:13	--	Folder
▼ dhis2	Today at 19:12	--	Folder
dhis2-play-demo.zip	Today at 19:06	1.1 MB	ZIP archive
▶ packaging	Today at 19:11	--	Folder
▶ facility-recon-gui	Today at 19:11	--	Folder
▶ facility-recon-backend	Today at 19:11	--	Folder
▶ docs	Today at 19:11	--	Folder
docker-compose.yaml	Today at 19:11	738 bytes	YAML
▶ dhis2sync	Today at 19:11	--	Folder
▶ dhis2App	Today at 19:11	--	Folder
README.md	Today at 19:11	1 KB	Markdown
LICENSE	Today at 19:11	11 KB	Hex Fie...cum
CHANGELOG.md	Today at 19:11	514 bytes	Markdown

Then follow the rest of the recipe above.

### 3.1.6 Setup a different backend server

If the DHIS2 app must talk to a backend that is on another server, then the prebuilt package is not suitable and must be rebuilt with the correct address.

The file `/facility-recon-gui/config/prod.env.js` must be modified.

```
'use strict'
module.exports = {
  NODE_ENV: '"production"',
```

(continues on next page)

(continued from previous page)

```
    BACKEND_SERVER: '"/'
  }
```

Change the BACKEND\_SERVER variable to the server. For example, for localhost:

```
'use strict'
module.exports = {
  NODE_ENV: '"production"',
  BACKEND_SERVER: '"http://localhost:3000"'
}
```

Once the file is edited and saved, build the gui.

```
cd facility-recon-gui
npm install
# there may be build warnings.
npm run build
```

The new app is installed in the root as /dhis2App/GOFR.zip. This is now built with the server chosen in the above step and can be installed into DHIS2.

## 3.2 DHIS2 users and sharing

With DHIS2 set as the authentication layer, users from DHIS2 can be given access to particular data sources and pairs, be restricted to pairs, and have their organizational unit restrictions also be reflected in the Facility Reconciliation Tool.

### 3.2.1 Prerequisites

- The Facility Reconciliation Tool application must be running.
- Then, the app should be installed. See the developer guide page for DHIS2 app installation.
- Once DHIS2 is the authentication layer, users must exist in DHIS2 for them to be able to use the Facility Reconciliation App.

### 3.2.2 About DHIS2 users

The building blocks of user management in DHIS2 are authorities, users, user roles, and user groups. See the [latest DHIS2 documentation](#) for an overview which is partly reproduced here.

### 3.2.3 Configuration options

When the Facility Reconciliation DHIS2 App is installed and DHIS2 is selected as an authentication source, a DHIS2 superuser role must be selected. There are several additional options.



☐ Single data source pair per org unit

☒ Disable Authentication

External Authentication Method

☒ dhis2

☐ iHRIS

Superuser Role Name

Guest

☒ Pull org units

☒ Share orgs with other users

☒ Limit orgs sharing by user orgid

Dataset Name

DHIS2 play

Username

admin

Password

.....

START PULLING

In the default settings:

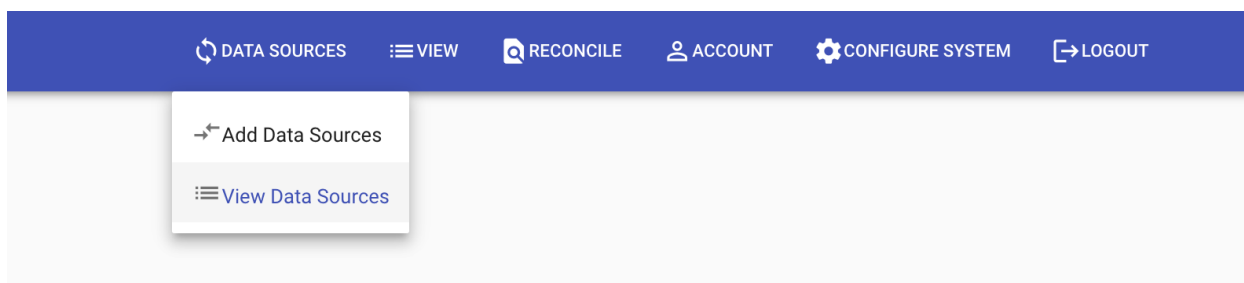
- Users will be able to only see the org units they have permission to see in DHIS2.
- Org units can be pulled into the Facility Reconciliation Tool into a dataset named as desired.
- Org units can be shared to users and not just the user who configured the system.

The default configurations may be switched on or off as desired.

### 3.2.4 Share data sources

Any existing data source can be shared to a DHIS2 user that has permission to use the app. One may share a dataset to particular users or all users or to limit the dataset based on the organization unit attached to DHIS2.

Click on the Data Sources tab and choose to View Data Sources.



Sharing status is listed for each data source and if not being shared it is blank.

Remote Sources

FORCE FULL SYNC

SYNC (UPDATE)

EDIT

DELETE

	Source Name ↑	Base URL	Source Type	User Name	Password	Last Sync	Owner	Shared To	Created Time	
<input type="radio"/>	dhis2test	https://play.dhis2.org/dev/	DHIS2	admin	*****		admin		31st May 2019 8:54:07 pm	<a href="#">SHARE</a>
<input type="radio"/>	DHIS2test2	https://play.dhis2.org/dev	DHIS2	admin	*****		admin		31st May 2019 8:56:51 pm	<a href="#">SHARE</a>

Rows per page:

5

1-2 of 2

<

>

Click on the share button to bring up the menu.

EDIT

DELETE

Shared To	Created Time	
	31st May 2019 8:26:51 pm	<a href="#">SHARE</a>

Rows per page:

5

1-1 of 1

<

>

Sharing can be limited by either or both the user or location (DHIS2 org unit).

Sharing DHIS2test

▼ Select location to limit sharing

▼ Sierra Leone

> Bo

> Bombali

> Bonthe

> Kailahun

> Kambia

> Kenema

> Koinadugu

> Kono

> Moyamba

> Port Loko

> Pujehun

> Tonkolili

> Western Area

Limiting Sharing to: No limit

Search

Username ↑

Firstname

Surname

☐

root@gofr.org

Root

Root

1-2 of 2

<

>

Rows per page:

5

▼

×

CANCEL

SHARE

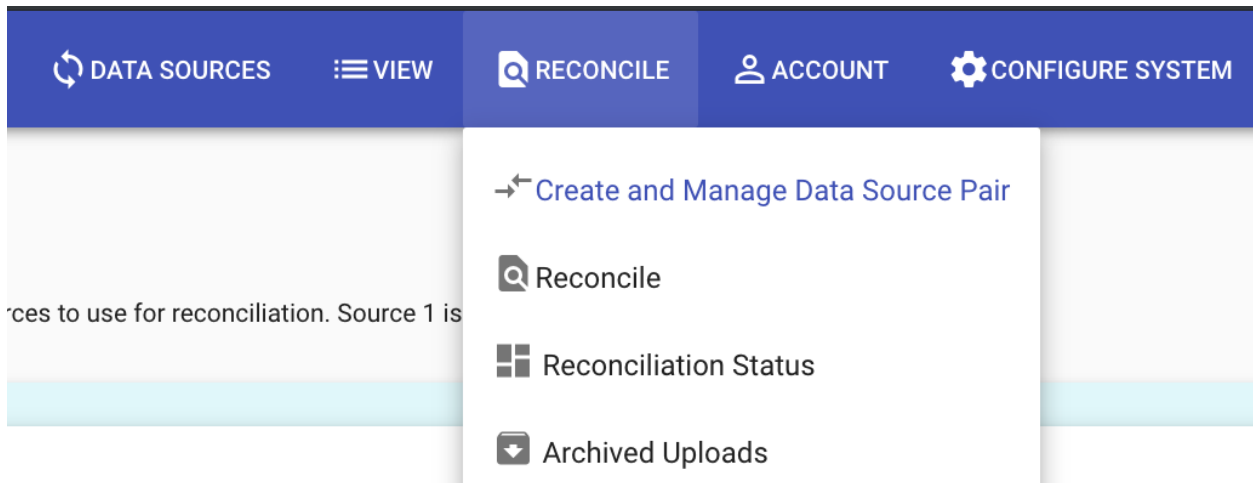
3.2. DHIS2 users and sharing

31

### 3.2.5 Share data pairs

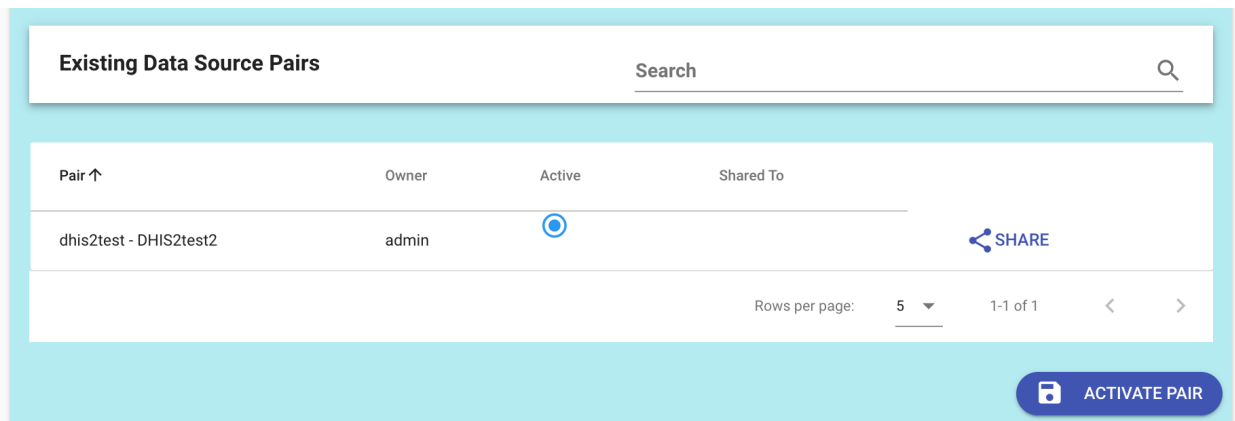
Data sharing of pairs is similar to that of data sources.

Go to the Reconcile tab and select Create and Manage Data Source Pair.

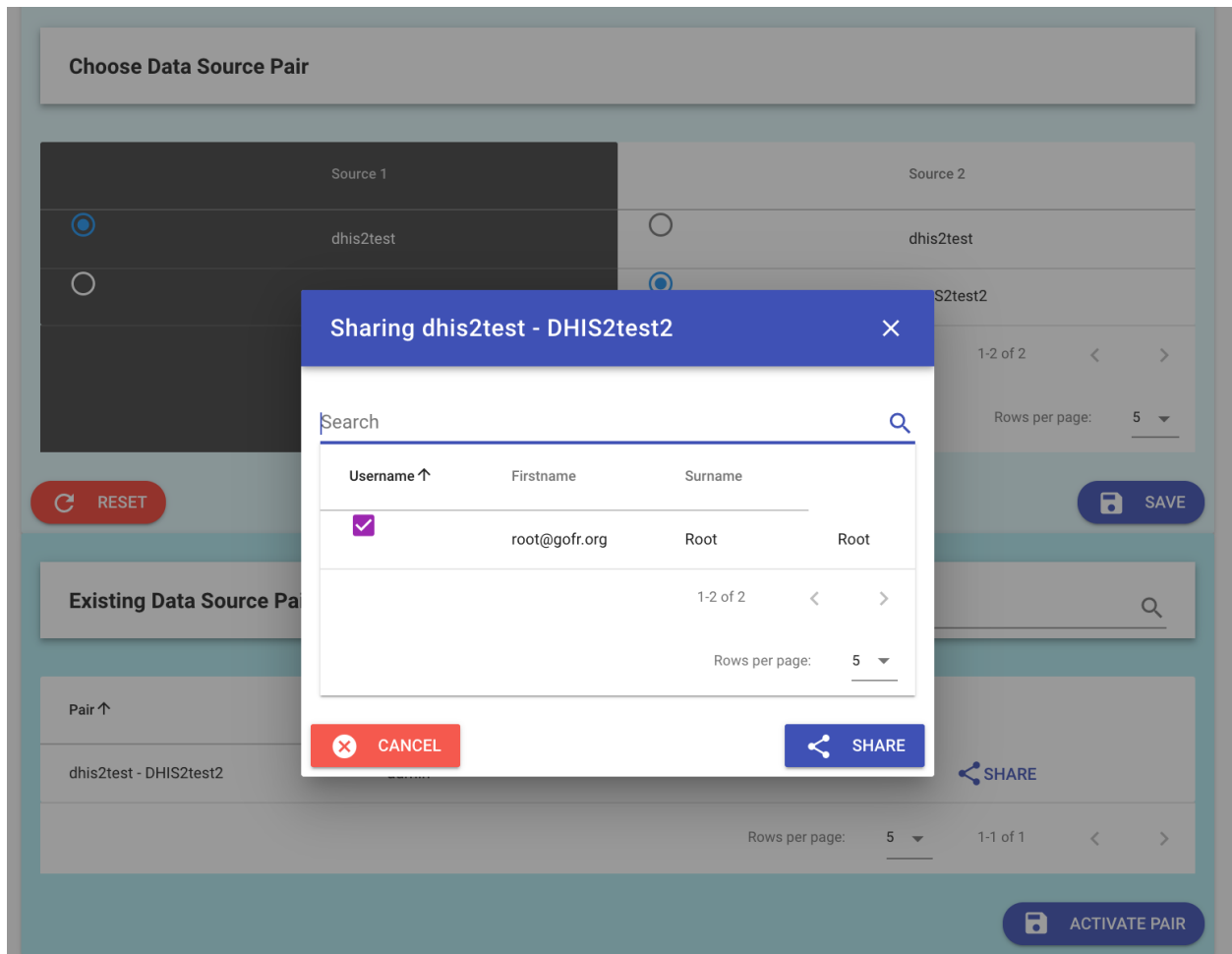


This page lists the pairs and is where to create pairs. In this example, we have two data sources from the DHIS2 currently being used and one for a remote DHIS2.

The sharing status can be seen in the list data pairs.



Click the Share button on the right hand side of the data source in the list to expand the pair sharing menu.



### 3.3 Quickstart with Docker

The fastest way to get started is to use Docker Compose which runs all components: app, Redis, MongoDB, and the Hearth FHIR server.

Clone the repo and start the docker apps.

```
git clone https://github.com/openhie/facility-recon.git
cd facility-recon
docker-compose up
```

Visit: <http://localhost:3000>

The default admin user is `root@gofr.org` and pass is `gofr`. A different admin users should immediately be created, the default deleted, and ordinary users added as well.

## 3.4 Install Locally

### 3.4.1 Hearth (FHIR Server) Installation

Make sure mongo 3.6 or above is installed and running before proceeding with below instructions

```
git clone https://github.com/intrahealth/hearth.git
cd hearth
npm install
```

Open the config file located under config/default.json and disable authentication by setting authentication type to disabled i.e “authentication”: { “type”: “disabled”}

Start hearth

```
npm run dev:start
```

### 3.4.2 Backend

Download the repo.

```
git clone https://github.com/openhie/facility-recon.git
```

```
cd facility-recon/facility-recon-backend
npm install
```

Install and run Redis

```
wget http://download.redis.io/redis-stable.tar.gz
tar xvzf redis-stable.tar.gz
cd redis-stable
make
redis-server
```

Run the backend

```
cd ~/facility-recon/facility-recon-backend
node lib/index.js
```

### 3.4.3 Frontend

```
cd facility-recon/facility-recon-gui
npm install
npm run build
```

### 3.4.4 DHIS2 App (Optional)

Note that the DHIS2 app is not required for running the tool as it can be run standalone from the GUI.

- Copy the frontend build contents from facility-recon/facility-recon-gui/dist into facility-recon/dhis2App/ and then zip the content of of dhis2App

```
cp -r facility-recon/facility-recon-gui/dist/* facility-reco/dhis2App/  
cd facility-recon/dhis2App  
rm GOFr.zip  
zip GOFr.zip ./*
```

Login to DHIS2 and install the zipped file (GOFr.zip)

## 3.5 Vagrant

### 3.5.1 Install

This an example using Vagrant for an Ubuntu or CentOS VM for end-to-end testing of facility recon.

To use just change directory into the OS (CentOS or Ubuntu) you want to use and fire it up.

```
cd ubuntu  
vagrant up
```

### 3.5.2 Troubleshooting

- For cleaning up, note that Vagrant boxes are stored in `~/ .vagrant .d/boxes`.
- [macOS] If there is a port conflict error for ssh, then clear out port forwarding entry for 2222 in `/Library/Preferences/VMware\ Fusion/networking` and `/Library/Preferences/VMware\ Fusion/vmnet8/nat.conf`

## 3.6 Production considerations

### 3.6.1 System Resources

Systems planners should test locally with the largest data sources that would represent their use case. With no user interaction, the demo server uses ~740MB of RAM (mongo: 200MB, redis: 10MB, hearth: 250MB, backend: 250MB).

The demo version uses 2 virtual CPUs, 4GB RAM, and 20GB SSD in a cloud-hosted VPS. For use cases with large data sources, such as 10k facilities, expect to increase RAM allocation.

### 3.6.2 DNS

DNS can be configured a myriad of ways for hosting. One method is to create a DNS record for the domain and then add an A record under the root domain for the application server.

### 3.6.3 Reverse Proxy and HTTPS

To host the tool behind a reverse proxy like nginx, configure it to pass requests to the nodejs server running the application on port 3000.

A full recipe for reverse proxy and SSL is here: <https://www.linode.com/docs/web-servers/nginx/use-nginx-reverse-proxy/>

If using Ubuntu with Nginx, you may have to disable Apache.

```
sudo update-rc.d apache2 disable
```

## 3.7 Ansible

These steps are for installing on a server OS directly and require experience with remote configuration.

To use Ansible, your SSH public key should be in `.ssh/authorized_keys` on the remote host and you must also create an `/etc/ansible/hosts` or similar with the IP address or hostname of the remote host. An `ansible/hosts` file that has an entry for localhost and one server would be:

```
[local]
localhost ansible_connection=local

[servers]
172.16.174.137
```

### 3.7.1 SSH setup

An example playbook is provided to show how to create a `fr` user with sudo permissions using Ansible to be used with VM. See `/packaging` for Terraform (Digital Ocean) and Vagrant (CentOS 7 and Ubuntu 18) for working examples.

Create a VM. Make sure to include a public ssh key for the user who will install prerequisites.

Create the `fr` user and give it sudo access:

```
ansible-playbook -i /usr/local/etc/ansible/hosts user.yaml
```

As necessary, add additional ssh keys to the user `fr`. (Ensure that the user's public key is available on github, ie. <https://github.com/citizenrich.keys>):

```
ansible-playbook -i /usr/local/etc/ansible/hosts keys.yaml
```

### 3.7.2 Installation

Prerequisites: git, redis, mongo, nodejs, native build pkgs for node:

```
# for centos
ansible-playbook -i /usr/local/etc/ansible/hosts prep_centos.yaml
# for ubuntu
ansible-playbook -i /usr/local/etc/ansible/hosts prep_ubuntu.yaml
```

Install the services and load and start them in systemd:

```
# prepare hearth and the app
ansible-playbook -i /usr/local/etc/ansible/hosts install.yaml
# install into systemd and begin the hearth and backend services
ansible-playbook -i /usr/local/etc/ansible/hosts services.yaml
```

### 3.7.3 Troubleshooting

Check that all processes are running and see the latest status for `hearth` and the backend:



```
ansible-playbook -i /usr/local/etc/ansible/hosts troubleshoot.yaml
```

### 3.7.4 Upgrades

Rerunning the `install` playbook updates `intrahealth/hearth` and `app` repos on the remote server. Rerunning the `services.yaml` playbook updates services. Services are restarted (not just reloaded).

The `install.yaml` playbook uses:

- `git pull` to get the latest updates to the master branch.
- `npm install` to update packages.

#### Basic status

```
# on centos, use `mongod`
systemctl status mongod.service
# on ubuntu, use `mongodb`
systemctl status mongod.service
systemctl status redis.service
systemctl status facility-recon.service
systemctl status hearth.service
```

#### Logs

```
# on centos, use `mongod`
journalctl -u mongod.service -b
# on ubuntu, use `mongodb`
journalctl -u mongod.service -b
journalctl -u facility-recon.service -b
journalctl -u hearth.service -b
journalctl -u redis.service -b
```

#### Restart services

```
sudo systemctl restart facility-recon.service
sudo systemctl restart hearth.service
```

#### Restart databases

```
# on centos, use `mongod`
sudo systemctl restart mongod.service
# on ubuntu, use `mongodb`
sudo systemctl restart mongod.service
sudo systemctl restart redis.service
```

## Networking

Ensure processes are listening on the correct ports: See <https://serverfault.com/questions/725262/what-causes-the-connection-refused-message>

```
# gui: 8080, backend: 3000, hearth: 3447, mongo: 27017, redis: 6379
sudo netstat -tnlp | grep :8080
sudo netstat -tnlp | grep :3000
sudo netstat -tnlp | grep :3447
sudo netstat -tnlp | grep :27017
sudo netstat -tnlp | grep :6379
```

Check for firewall blocks. Rerun the gui and:

```
sudo tcpdump -n icmp
```

## 3.8 Terraform

This is an example **Terraform** remote installation on a VM provided by Digital Ocean. See the `/packaging/terraform` directory for the example configuration.

- Create an account on DO and upload a public SSH key and note your API token.
- Export a `DIGITALOCEAN_TOKEN` variable with your API token.
- Get the hash fingerprint that DO generated for your SSH key (this is a DO thing).

```
curl -X GET -H "Content-Type: application/json" -H "Authorization: Bearer $DO_TOKEN"
↪ "https://api.digitalocean.com/v2/account/keys"
```

- Export a `TF_VAR_fingerprint` variable with the hash.
- Get the latest droplet instances and sizes, you'll need the awesome `jq` parser for JSON installed.

```
curl -X GET --silent "https://api.digitalocean.com/v2/images?per_page=999" -H
↪ "Authorization: Bearer $DO_TOKEN" > droplets.json
curl --silent -X GET "https://api.digitalocean.com/v2/sizes" -H "Authorization:
↪ Bearer $DO_TOKEN" | jq '.sizes[].slug' > sizes.json
```

- Adjust droplet config as you wish using info from the above.
- Profit!

```
terraform init
terraform apply
terraform state pull | jq --raw-output
```

You should be able to ssh as root if the public key was set correctly (this is the default of DO).

Clean up.

```
terraform destroy
```

## 3.9 Contribute

The Reconciliation Tool is distributed under the Apache 2.0 license. The community welcomes contributions!

- For substantial features please consider engaging with the community first and be prepared to be asked to create a design document. A design document can simply be a brief Google Doc that community members can provide feedback towards.
- For smaller features please engage in a GitHub issue or approach the community and ask early on about the approach.

Please join the OpenHIE Facility Registry Community for open monthly discussions and the Facility Registry Google Group for announcements and discussions.

## 3.10 Update and Build Documentation

These docs are hosted at <https://readthedocs.io>. The documentation is built using Sphinx with Markdown support enabled. In order to translate, the documentation must first be in the base (English) language. Upon a successful build, the translatable files (.pot) are then pushed to [Transifex](#). The pot files must be synchronized with Transifex for editors to use them.

### 3.10.1 Simple Build

Install prerequisites (see below). Then to do a build:

```
make html
```

### 3.10.2 Sphinx Extensions

Python extensions are required to build the docs, including `sphinx_markdown_tables` for tables to render. The `.readthedocs.yaml` file in the root of the repo sets which version of Python (3.7) and the location of the requirements file in `docs/requirements.txt`.

For more on how markdown tables can render correctly see: <https://pypi.org/project/sphinx-markdown-tables/>

### 3.10.3 Update Sources for Translation

This is from the [rtd docs](#) with minor changes for folder names.

This should be done if content changes in the base language documentation.

Create the .pot files.

```
cd docs
sphinx-build -b gettext source build/gettext -l fr
sphinx-intl update -p build/gettext -l fr
```

If a page is created it needs to be added to `.tx/config` by using `tx init` mapping and then pushed to Transifex. For example

```
# try
tx config mapping --resource facility-recon.docs_build_gettext_dev_dhis2users \
--source-lang en --type PO --source-file docs/build/gettext/dev/dhis2users.pot \
--expression 'docs/source/locales/<lang>/LC_MESSAGES/dev/dhis2users.po'
# add --execute to do it if all looks well
tx config mapping --resource facility-recon.docs_build_gettext_dev_dhis2users \
```

(continues on next page)

(continued from previous page)

```
--source-lang en --type PO --source-file docs/build/gettext/dev/dhis2users.pot \  
--expression 'docs/source/locales/<lang>/LC_MESSAGES/dev/dhis2users.po' --execute
```

Push files to Transifex.

```
tx push -tl fr
```

### 3.10.4 Build Documentation from New Translations

```
tx pull -l fr
```

Next git add, commit, push the changes to the repository master branch. Then rebuild the English and French projects on [readthedocs.org](https://readthedocs.org).

### 3.10.5 Update Base (English) Documentation

Note: Python 3.x is expect in the Makefile.

Install prerequisites:

```
pip install sphinx  
pip install recommonmark  
pip install sphinx_rtd_theme  
pip install sphinx-intl  
pip install transifex-client  
pip install sphinx_markdown_tables
```

Clone the respository, create a branch, and enter the docs directory. (If you are not a maintainer, please fork, branch, then send a pull request.)

```
git clone https://github.com/openhie/facility-recon.git  
cd facility-recon/docs
```

Build the documentation. There is no need to commit documentation builds. The `.gitignore` includes the built documentation so built docs will not be committed by default. [readthedocs](https://readthedocs.org) builds the hosted documentation itself.

Build docs in default (English) language.

```
make html
```

Built docs will go in the `/docs/build` directory. Open `/docs/build/index.html` in a browser to see the built docs.

### 3.10.6 Transifex Setup

Transifex provides an accessible interface for contributors to enter and correct translations. The workflow is explained at [readthedocs](https://readthedocs.org).

Setup an account on [transifex.com](https://transifex.com). Be added to the OpenHIE organization for facility-recon project. Obtain a token in user settings and initialize transifex command line client. You do not need to `tx init` the directly, as the `.tx/` config should be version control. Setup a `$TFX_TOKEN` environment variable for yourself.

### 3.10.7 Create Translation Intermediate Files for New Languages

In the /docs directory, create the locale files. Use the appropriate [locale](#). Note that Sphinx supports less locales than Transifex.

```
sphinx-intl update -p build/gettext -l es
```

### 3.10.8 How Localization was done (Do Not Reproduce)

Do not replicate this process. It is here to document how it was done in the first place. Doing it again would delete or write over translations or remove them.

- Setup Sphinx for the project. Enable Markdown support. Setup readthedocs.org. Enable the webhook to trigger builds.
- Create the source language docs.
- Create projects in readthedocs.org for both the source language (facility-recon - no prefix/suffix) and the new one (facility-recon-fr)
- Set languages correctly on both projects.
- Assign the -fr project as a translation of the source. You must have two projects.

Create gettext dir. `source` is required as the `conf.py` file is not in the docs directory.

```
sphinx-build -b gettext source build/gettext
```

Create `.pot` files for French. This puts a `locales` directory in `/docs/source/` and creates files for every `.md` and `.rst` file.

```
cd docs
sphinx-intl update -p build/gettext -l fr
```

Init Transifex support for the repo. Do it from the repo root, not docs. Add a Transifex API token env var prior to doing this.

```
# must be in repo root
cd ../
tx init --token $TFX_TOKEN --no-interactive
```

Bulk-map all of the documents. This must be done in the root of the repo.

```
# must be in repo root
tx config mapping-bulk --project facility-recon --file-extension '.pot' --source-file-
→dir docs/build/gettext \
  --source-lang en --type PO --expression 'docs/source/locales/<lang>/LC_MESSAGES/
→{filepath}/{filename}.po' --execute
```

Do the initial push of the source language documents to Transifex.

```
tx push --source
```

Test by translating and editing a few files on Transifex. Pull from Transifex.

```
tx pull -l fr
```

Manually check the `.po` files translated to confirm the changes.

Build locally to check. (FYI, this puts `.mo` files into the repo which could potentially be gitignored and removed.)

```
cd docs
sphinx-build -b html -D language=fr ./source build/html/fr
```

... then open the `index.html` in the built files for French to confirm.

readthedocs.io should rebuild based on the existing webhook.

#### 4.1 Is there an API?

The tool uses a FHIR server, [Hearth](#) for persistent storage. Hearth uses MongoDB for its storage layer.

To obtain data sources and reconciled sources a user with access to the underlying stack can make queries to MongoDB or to the FHIR-based REST API for Hearth. There is no authentication for the raw access in this manner.

#### 4.2 Can this tool be used in education or agriculture?

Yes! The tool was first created for the health sector and uses the [FHIR](#) standard based on the [mCSD](#) IHE profile.

Any hierarchical location list can potentially be supported. The backend representation can be exported in CSV and others may be added as use cases dictate.

#### 4.3 Does the tool clean the source data?

No. The tool takes a pair of sources and uses automatic and manual methods to identify matches. If there are corrections needed, the tool can export a CSV or FHIR representation of what was and was not matched. The data sources have to be cleaned outside of the tool.

If this is a feature for your use case, please get in touch.

#### 4.4 Can I run the tool on my own PC?

The tool is comprised of four components: the app itself, Redis as an in-memory database for performance, the [Hearth](#) FHIR server, and MongoDB which is used by both [Hearth](#) for storing resources and the app for managing state.

The tool is designed to be hosted on a server, either locally or in the cloud. The tool supports user management and is meant to be available for many users to collaborate on matching facility lists. Server installation is the recommended way to deploy the tool.

Docker is not recommended as by default it will not persist the data, meaning that when you stop Docker you lose your data. This can be changed by mounting a volume to store data. But, the tool is meant as a platform for multiple users to collaborate on matching. It can be deployed on a server using Docker but the administrator should be careful to mount a volume to ensure data persistence.

Developers may install the stack directly but this is not for production.



Proposed features are managed in a publicly-viewable [Google Doc](#) and managed in [GitHub issues](#).

### 5.1 Version: 0.1.0

Timeline: February 2018 to July 2018, Status: Completed

#### 5.1.1 Features:

- Matching based on one uploaded data source.
- Other data source is one customized DHIS2 instance (GeoAlign).
- Preliminary DHIS2 authentication.
- Preliminary exports of matched and unmatched facilities.

### 5.2 Version: 0.2.0

Timeline: August 2018 to October 2018, Status: Completed

#### 5.2.1 Features:

- User-configurable to any data pair.
- Any DHIS2 instance for which the user has access.

**Note: Version 0.2.0 was extensively production-tested on ~80K facilities.**

## 5.3 Version: 0.3.0

Timeline: November 2018 to December 2018, Status: Completed

### 5.3.1 Features:

- User interface overhaul.

## 5.4 Version: 0.4.0

Timeline: January 2019 to February 2019, Status: Completed

### 5.4.1 Features:

- Containerization.
- Hosting scripts and utilities.

**Note: Version 0.3.0 was tested in user-centered design workshops using a cloud-hosted instance.**

## 5.5 Version: 0.5.0

Timeline: March 2019 to April 2019, Status: Completed

### 5.5.1 Features:

- Documentation on [facility-recon.readthedocs.org](http://facility-recon.readthedocs.org).
- Enhanced export to show all matched and unmatched records.
- Matching enabled on Facility IDs and Geo Coordinates.

## 5.6 Version: 0.6.0

Timeline: May 2019 to June 2019, Status: Completed

### 5.6.1 Features:

- Options for external authentication (DHIS2, iHRIS)
- DHIS2 data source sharing.
- DHIS2 pair sharing.
- Easily-installable DHIS2 app.
- Configurable matching - matching based on parent constraint can be turned off

## 5.7 Version: 0.7.0

Timeline: TBD, Status: In-progress

### 5.7.1 Features:

- Preparation for 1.0 release.
- Extensive end-to-end testing.
- Bugs resolved.
- Updated FHIR version.
- Consider moving to HAPI FHIR.



## CHAPTER 6

---

### About

---

The tool was built by the international development community for the initial use case of supporting a monitoring and evaluation platform for multiple countries. The tool was first created in the .NET framework.

This version of the tool was created in an open framework in order to create a larger community around it and ensure long-term contributions from the community as well as to add new features.

The initial use cases have been centered on the health sector. It is built using the emerging [FHIR](#) standard based on the [mCSD](#) IHE profile. The backend is a FHIR server which means that a FHIR-compliant API is available from server for operations. All data sources are converted to FHIR [Location Resources](#). This means that ordinary FHIR REST API queries can be pursued against the backend server.

In the future, the tool may easily support any hierarchical or flat data, both in and outside of the health sector. Matching algorithms, new data sources, and export formats can be created to meet a variety of use cases as the community requires.