DHIS2-Go.Data Interoperability App

Implementation Guide

Last Updated: 28 Jan 2022

Table of Contents

Introduction

What is the main purpose of this app?

This Go.Data-DHIS2 Interoperability App enables metadata to be securely exchanged between Go.Data (an outbreak response tool developed by WHO) and DHIS2 (a widely used system for national health information management developed by the University of Oslo). This includes bi-directional exchange of reference medadata used across both platforms (i.e. location hierarchies, facility lists); and case/contact data (cases being registered and investigated; and their contacts who are being listed and traced). Such an integration enables field teams to make use of additional visualizations for chains of transmission and contact tracing follow up in Go.Data during an acute outbreak scenario while ensuring key field intel gets reflected back into overarching DHIS2 system.

Requirements

Which are the minimum IT requirements for the app to be installed / worked?

Kommentar [HS1]: Murod to advise.

This app requires installation of the DHIS2 software and the Go.Data software (see version requirements above).

What versions of Go.Data and DHIS2 is it compatible with?

Kommentar [HS2]: @Murod to advise

The current app is compatible with Go.Data V40.1 and above [testing to be performed for lower versions], and compatible with DHIS2 version 2.34 and above.

What DHIS2 modules is it compatible with?

Kommentar [HS3]: Confirm if it could work with Event Capture.

The current app is compatible with DHIS2 Tracker and Event Capture, across all types of Program Attributes and Data Elements.

Installation

How do I install the app?

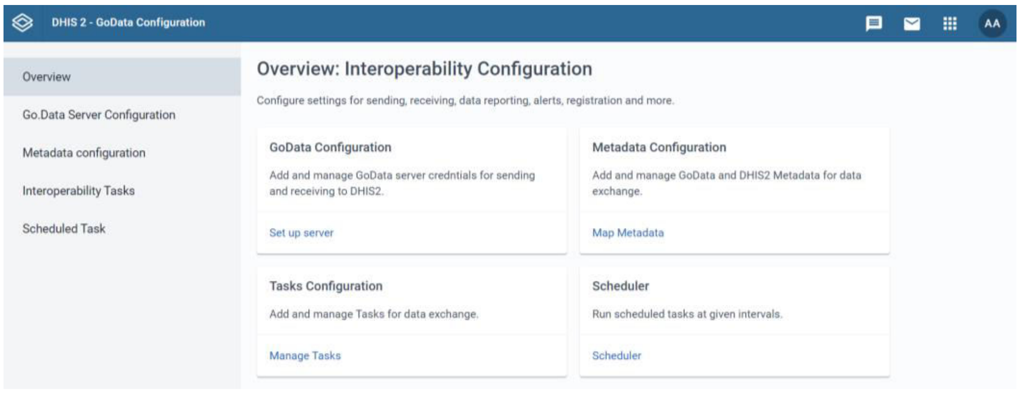
Until the app is made available on the DHIS2 App Hub, you can download the .zip file directly from the WHO Github here: https://github.com/WorldHealthOrganization/godata-dhis2-interop-app

Then, go to the App Management app in your DHIS2 instance and upload the .zip file.

Configuration

What are the main functionalities within the app?

The app module consists of four main menu options to navigate and a main pane for performing and viewing various actions within the module.


DHIS 2 - GoData Configuration

AA

Overview

Go.Data Server Configuration

Metadata configuration

Interoperability Tasks

Scheduled Task

Overview: Interoperability Configuration

Configure settings for sending, receiving, data reporting, alerts, registration and more.

GoData Configuration

Add and manage GoData server credntials for sending and receiving to DHIS2.

Set up server

Metadata Configuration

Add and manage GoData and DHIS2 Metadata for data exchange.

Map Metadata

Tasks Configuration

Add and manage Tasks for data exchange.

Manage Tasks

Scheduler

Run scheduled tasks at given intervals.

Scheduler

Figure 1. View of module menu system

1. **Go.Data Server Configuration** menu is used for configuring connection to Go.Data server of which you want to exchange data with.

2. **Metadata configuration** menu is used to create, modify, or delete metadata mappings. For all mappings and throughout subsequent system development, dot notation is used (a way of accessing JSON objects of any depth).

3. **Interoperability Tasks** menu is used to add and manage tasks for data exchange across platforms.

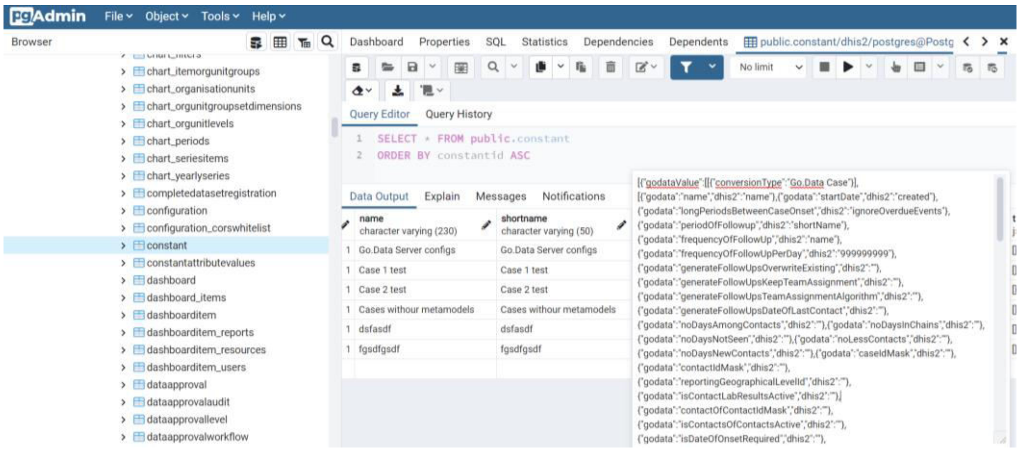
4. **Scheduled Task** menu is used to set up and run scheduled tasks at given intervals.

Does the app enable bi-directional data exchange?

No, at present, data exchange is moving only from DHIS2 to Go.Data. However, this additional functionality will be incorporated into subsequent versions soon.

What is the basic infrastructure of the app?

At present, the entirety of the app configuration is within the "constant" table within the DHIS2 PostgreSQL database. This "constant" table has few columns, and most are designed for storing numeric value. However, the description column is of type "text" and can hold large amount of textual data. Here, the all objects used by module are stored stringified js object notation (JSON) and arrays.



```
SELECT * FROM public.constant
ORDER BY constantid ASC
```

	name character varying (230)	shortname character varying (50)
1	Go.Data Server configs	Go.Data Server configs
1	Case 1 test	Case 1 test
1	Case 2 test	Case 2 test
1	Cases withour metamodels	Cases withour metamodels
1	dsfasdf	dsfasdf
1	fgsdfgsdf	fgsdfgsdf

```
[{"godataValue":[[{"conversionType":"Go.Data Case"}],
[{"godata":"name","dhis2":"name"},{"godata":"startDate","dhis2":"created"},
{"godata":"longPeriodsBetweenCaseOnset","dhis2":"ignoreOverdueEvents"},
{"godata":"periodOfFollowup","dhis2":"shortName"},
{"godata":"frequencyOfFollowUp","dhis2":"name"},
{"godata":"frequencyOfFollowUpPerDay","dhis2":"999999999"},
{"godata":"generateFollowUpsOverwriteExisting","dhis2":""},
{"godata":"generateFollowUpsKeepTeamAssignment","dhis2":""},
{"godata":"generateFollowUpsTeamAssignmentAlgorithm","dhis2":""},
{"godata":"generateFollowUpsDateOfLastContact","dhis2":""},
{"godata":"noDaysAmongContacts","dhis2":""},{"godata":"noDaysInChains","dhis2":""},
{"godata":"noDaysNotSeen","dhis2":""},{"godata":"noLessContacts","dhis2":""},
{"godata":"noDaysNewContacts","dhis2":""},{"godata":"caseIdMask","dhis2":""},
{"godata":"contactIdMask","dhis2":""},
{"godata":"reportingGeographicalLevelId","dhis2":""},
{"godata":"isContactLabResultsActive","dhis2":""},
{"godata":"contactOfContactIdMask","dhis2":""},
{"godata":"isContactsOfContactsActive","dhis2":""},
{"godata":"isDateOfOnsetRequired","dhis2":""},
```

Figure 2. "constant" table of DHIS2 showing "description" column

This approach provides significant flexibility while retaining a little overhead for validating objects on and from database read/writes. In this way, "description" column of table "constant" of DHIS2 becomes the "nano mongo DB".

In the same table, the column "value" was then used to group different object types. For example, value -1000000 is used for configuring Go.Data instance, -1000001 used for mapping of Go.Data metadata with DHIS2 metadata, -1000002 is used for creating tasks. Columns "name" and "shortName" are used for naming created entities.

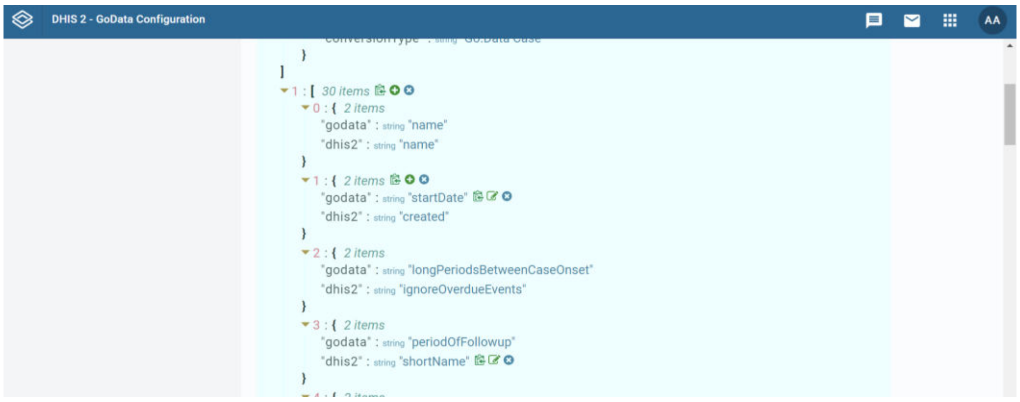
Of note, this infrastructure will be soon moved to Data Store, which was introduced by DHIS2 development team for storing of configuration of a single app and accessible by users, which may be more desirable from an ongoing maintenance perspective.

Kommentar [HS4]: Timeline on Data Store switch ?

How does the app configuration work?

The app configuration should only need to be done once, at the outset, by your DHIS2 administrator. The person configuring the mapping should be aware of use-cases and metadata collected by both DHIS2 and Go.Data.

The **Metadata configuration** section is built with slightly modified version of React JSON Viewer to enable flexible matting of metadata across the platforms regardless of metadata configuration.



DHIS 2 - GoData Configuration

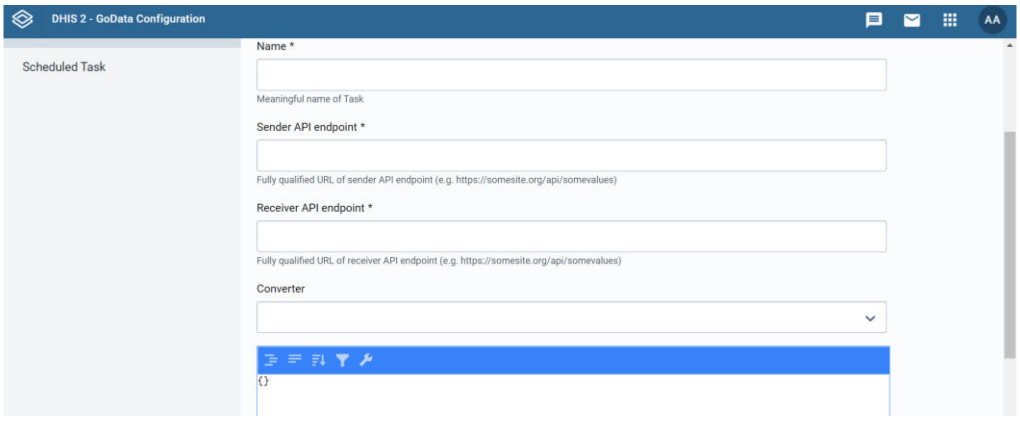
```
      }
    ]
  ▾ 1 : [ 30 items
    ▾ 0 : { 2 items
        "godata" : string "name"
        "dhis2" : string "name"
      }
    ▾ 1 : { 2 items
        "godata" : string "startDate"
        "dhis2" : string "created"
      }
    ▾ 2 : { 2 items
        "godata" : string "longPeriodsBetweenCaseOnset"
        "dhis2" : string "ignoreOverdueEvents"
      }
    ▾ 3 : { 2 items
        "godata" : string "periodOfFollowup"
        "dhis2" : string "shortName"
      }
```

Figure 3. Mapping screen

Pair values JSON are presented with commands. Each command in this editor has specific role of select/add/delete/edit. Select button opens modal with DHIS2 instance of the corresponding Go.Data object instance. While user selects corresponding DHIS2 attribute, it will be linked to pair elements of intermediary JSON object.

If there are cases of Go.Data metadata where no corresponding DHIS2 metadata available, the user can simply add the required value by typing it in. In the example of Outbreak, element “caseIdMask” does not have corresponding DHIS2 element. In this case, user adds “9999999” as value for the “caseIDMask”. There are two types of values: byValue and byReference. ByValue instances simple read DHIS2 “dhis2” element value and byReference reads dot notation from “dhis2” and reads the value from actual object passing dot notation. Result is placed for new instance element.

The **Scheduled tasks** section is dedicated for managing the integration tasks. Tasks are composed of “name”, “conversionType”, “senderAPIEndpoint”, “receiverAPIEndpoint”, “converter”, “referenceModel”, “senderAPIParams” and “sender” elements.



DHIS 2 - GoData Configuration

Scheduled Task

Name *

Meaningful name of Task

Sender API endpoint *

Fully qualified URL of sender API endpoint (e.g. https://somesite.org/api/somevalues)

Receiver API endpoint *

Fully qualified URL of receiver API endpoint (e.g. https://somesite.org/api/somevalues)

Converter

```
{}
```

Figure 4. Tasks screen

- “name” is the name of the task
- “conversionType” – type of objects in the conversion (Outbreak, Case, Contact, ContactsOfContact, Location)
- “senderAPIEndpoint” API endpoint of sender (http://godata.org/locations)

- “receiverAPIEndpoint” API endpoint of sender (http://dhis2.org/locations)
- “converter” – mapping created in mapping menu for conversion “godata” – “dhis2”
- “referenceModel” – JSON model accepted by the receiving endpoint
- “senderAPIParams” – parameters to be send to sender endpoint, filters=created.eq.”25/11/2021”&paging=false
- “sender” – determines which instance is sender (true – DHIS2, false – Go.Data)

Is it possible to add new variables to be migrated between DIHS2 and Go.Data?

…..

Do I need a highly IT trained team to implement and keep running the app?

….

Could this app be used for other platforms besides DHIS2 and Go.Data?

…..

Can a scheduler be configured to push data at certain intervals across systems?

Not yet implemented, but will be incorporated into subsequent versions.

Approximately what volume of case/contact records would this work well for?

More testing is needed, but only limitation would be on side of browser and server capacity side.

Documentation ideas:

- https://www.researchgate.net/publication/294091467_Interoperability_Guide_for_Indicator_Data_Reporting_Automation_of_indicator_data_reporting_from_IQCare_to_DHIS_2
- idea for documentation workflows: https://wiki.ohie.org/display/SUB/Use+Case+Summary:+Request+Community+Based+Follow-Up
- mobile app “preview” like commcare has…i.e. this video here: https://drive.google.com/file/d/1pN2Uxoukb-tX97BPffgHZVWtNOxfw2Cv/view