Article
José Pereira · Jun 5, 2021 8m read

Open Exchange

FHIRaaS overview

Introduction

This article aims to provide an overview of InterSystems IRIS FHIR Accelerator Service (FHIRaaS) driven by the implementation of application iris-on-fhir, available in OEX developed for the FHIRaaS contest.

A basic tutorial will guide you in configuring a function FHIRaaS deployment, including an API key and an OAuth 2.0 server.

A library to use FHIR resources through FHIRaaS also is briefly discussed.

Finally, some features of the iris-on-fhir application are shown in separted articles. You can check out the full code at the application ' s github repository.

This content will be presented in a serie of 3 artciles.

This first article seems to be a little big, but don't worry - this is beacuse I put a lot of images to help you in your configuration steps.

FHIRaaS

IRIS already provides a FHIR API environment built-in in IRIS for Health and IRIS Health Connect.

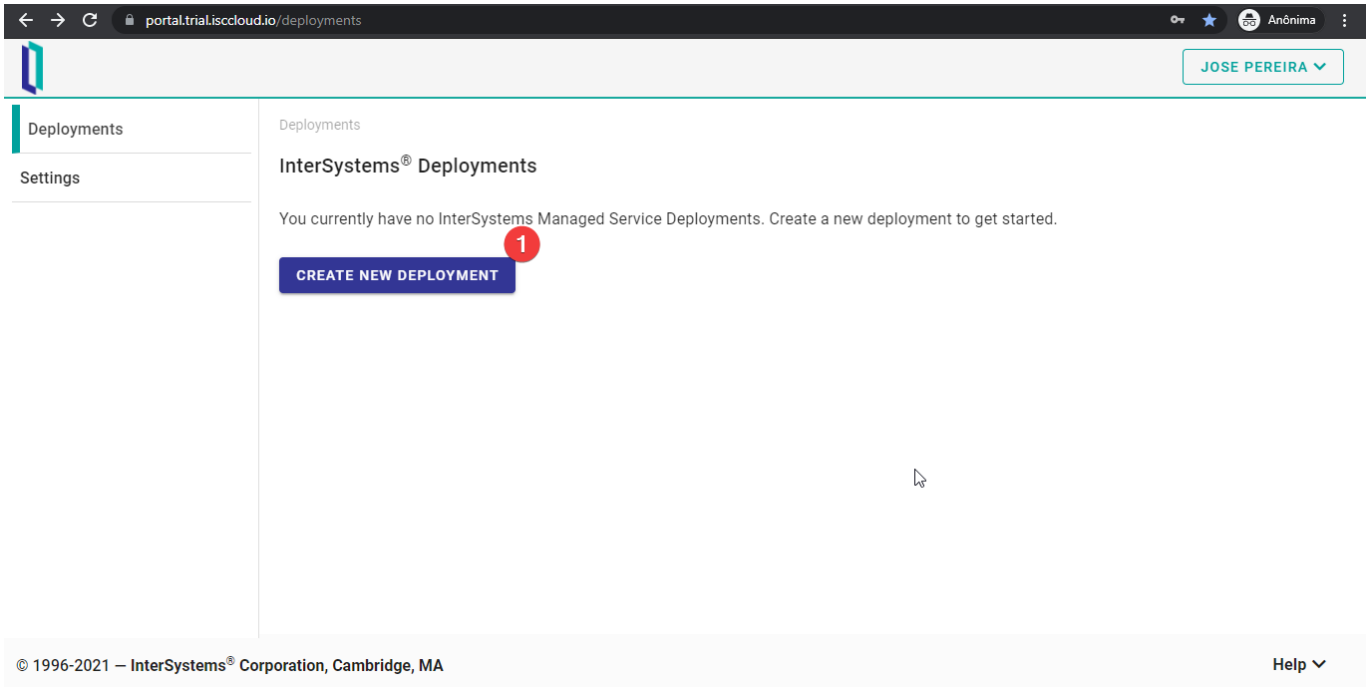
But, if you'd like to take advantage of the reliable, secure and low maintenance environment that cloud services can provide, now you can count on InterSystems IRIS FHIR Accelerator Service (FHIRaaS).

FHIRaaS is a ready-to-use FHIR infrastructure based on cloud services. You just need to set up a deployment and start to use the FHIR API in your applications wherever they are - JS client (SMART on FHIR), traditional backend or serverless applications.

To request your free trial of FHIRaaS, please contact InterSystems.

Configuring a deployment

After login, click on the " CREATE NEW DEPLOYMENT " button.



You will need to go through some steps. First one is to choose the deployment size. At the moment of writing this article, FHIRaaS just provides one option. So just hit the “ CONTINUE ” button.

Next step is to choose which cloud service provider will be used. Again, just one option was available when this article was written: AWS.

The last configuration is just the deployment name. There are some rules for this name, which FHIRaaS UI alerts you when an invalid name is provided. Also, note that you can ’ t change this name after this set up. Hit the “ CONTINUE ” button after selecting the name.

Finally, review your configuration and start your FHIRaaS deployment by hitting the button “ CREATE ”.

If everything goes OK, you ’ ll receive a nice message indicating that your FHIRaaS deployment is under construction. Wait a few minutes until this process ends.

After a few minutes, your FHIRaaS deployment is ready to use. Just hit the deployment button and start to use it.

After hitting the deployment button, the “ Overview ” tab is presented. Note that you have several tabs. In this article just tabs “ OAuth 2.0 ”, “ Credentials ” and “ API Development ” will be covered. But this is just due scope - the other ones aren ’ t complicated at all and you can easily explore them.

Access control

FHIRaaS supports two ways for control access: API Key and OAuth 2.0. Let ’ s walk through each of them.

API Key

API keys are tokens generated by FHIRaaS which allow your applications to interact with API without user interaction. This makes such a method ideal for communication between FHIRaaS server and authorized third party apps, like a chatbot API, for instance.

To create an API key, first access the “ Credentials ” tab, then hit the button “ CREATE API KEY ”.

Choose a name for your API key and hit the “ ADD ” button.

Nice! Your API key was created! Copy and save it in a secure location - you can ' t access this information anymore.

After creation you can only delete an API key.

To use this Api key, just add a header x-api-key to a HTTP request. For instance:

```
curl -X GET "https://fhir.lrwvcusn.static-test-account.isccloud.io/Patient" -H  "acce
pt: application/fhir+json" -H  "x-api-key: your-apy-key"
```

OAuth 2.0 - Creating a OAuth 2.0 server and adding users to it

As said before, API Key is suitable when you need to use API without user interaction. But, when you create an application for your users, OAuth 2.0 and OpenID Connect are nowadays the industry standard for authentication and authorization.

Just a side note: despite you can use OAuth 2.0 and OpenID Connect independently, it's very usual to see both running side by side. So when I say OAuth 2.0 here, I'm meaning OAuth 2.0 for authorization and OpenID connect for authentication.

So, let ' s configure an OAuth 2.0 server. First, select the " OAuth 2.0 " tab and then hit " CREATE AUTHENTICATION SERVER " button.

Next step is choose a name for your OAuth 2.0 server and select what Identity Provider (IdP) to use. At the time when this article was written, FHIRaaS supported only AWS Cognito as IdP. So, just hit the "CREATE" button.

If your request was successful, a message will be prompted to you, like in the image below. Now you can add users to your IdP by hitting the " ADD USER " button.

You will be redirected to the " Credentials " tab. To add an user, hit the button " CREATE USER " .

Input the user name and its password, then hit the "CREATE" button.

If everything goes OK, you can now see an user created in your IdP. This user can now sign in in applications authorized by this OAuth 2.0 server.

OAuth 2.0 - Adding applications to the OAuth 2.0 server

After creating an OAuth 2.0 server and adding users to it, such users can use applications authorized by this server. Now, we ' ll add an application to the server.

First, go to the " OAuth 2.0 " tab, select " Application " and then, hit the button " CREATE APPLICATION " .

Next, choose a name for your application in the server and the OAuth 2.0 server just created. The URLs must point to your application. The " Redirect URL " is the destination address when users successfully login. The " Logout URL " is the page where users are redirected to, when they use the IdP to logout.

You can redirect to localhost while you are developing, but, of course, for production you must provide an internet reachable URL.

The final steps are to choose the FHIR resources (scopes) which users must agree to share with the application. For this simple test, all resources are requested, but in real applications, you can control each FHIR resource, as if the application can just read, just write or both. If users don ' t agree with this authorization request, the OAuth 2.0 server will deny access for such resources.

After properly setting up scopes, hit the " CREATE " button.

If everything goes OK, you ' ll see a green message. Now, you can check out your new application parameters or delete it by hitting the application box. You can also create many applications you need.

API Development

A nice feature of FHIRaaS is the API Development tab. This provides you an OpenAPI Specification explorer of the FHIR API, letting you try all FHIR features easily.

To access it, click on the " API Development " tab. After it loads, you can select which FHIR resource you ' d like to explore. Note that FHIRaaS provides the R4 version for FHIR resources.

Now, let ' s authenticate to use the tool. You must create an API Key first.

Nice, now, let ' s get all patients in this instance of FHIRaaS:

As you can see in the animation above, you can perform all CRUD operations on Patient resource - the same for all other resources available.

The nice thing here is you don ' t need to know the whole structure of the resources in order to try operations over them. For instance, if you ' d like to try create a new patient, the tool provides you a template for such resource:

You have the same feature for other FHIR resources.

At the end of the page, the tool provides you a nice view of all related resources, as them schemas:

Conclusion

In this article we walk through some aspects of FHIRaaS and set up a functional deployment

Next article, we'll check some simple examples on how to use it in applications.

#InterSystems IRIS for Health
Check the related application on InterSystems Open Exchange