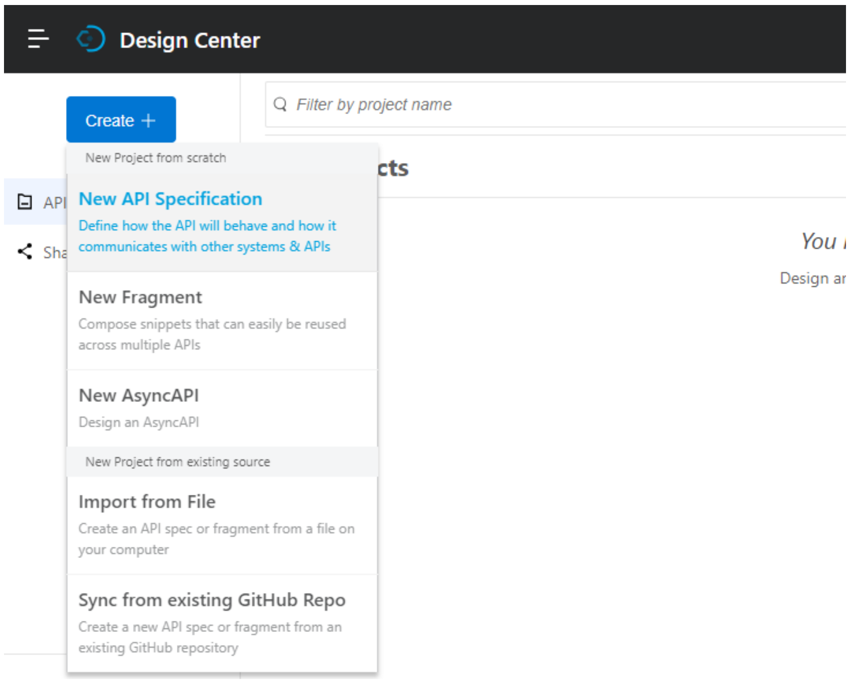
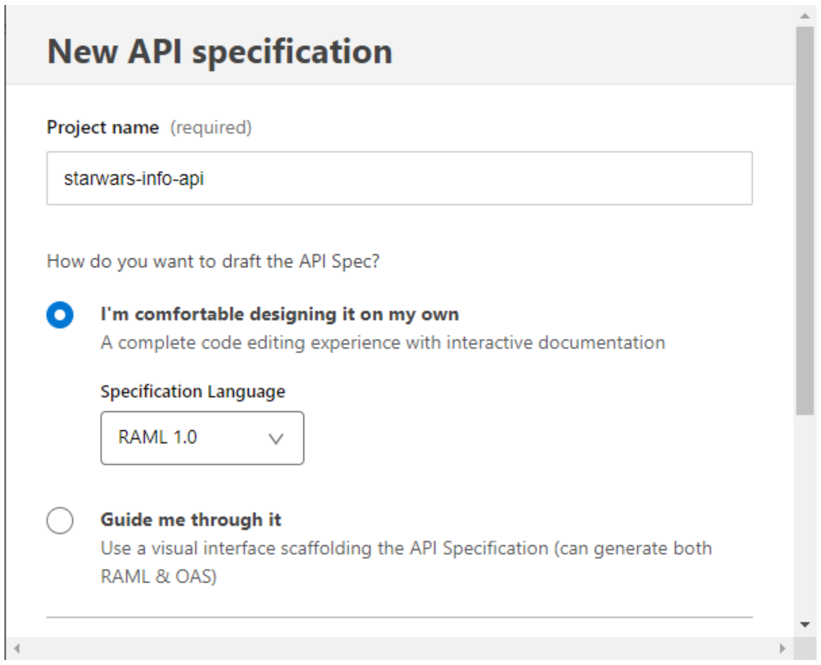
1. Creation of the **API Specification (RAML)** in the Anypoint Platform, in the Design Center we select the option **"Create +"**, followed by **"New API Specification",** we place the name of the project, we verify that it is in RAML 1.0 and we click on it in **"Create API"**


Design Center
Create +
Filter by project name
New Project from scratch
New API Specification
Define how the API will behave and how it communicates with other systems & APIs
New Fragment
Compose snippets that can easily be reused across multiple APIs
New AsyncAPI
Design an AsyncAPI
New Project from existing source
Import from File
Create an API spec or fragment from a file on your computer
Sync from existing GitHub Repo
Create a new API spec or fragment from an existing GitHub repository


New API specification

Project name (required)

starwars-info-api

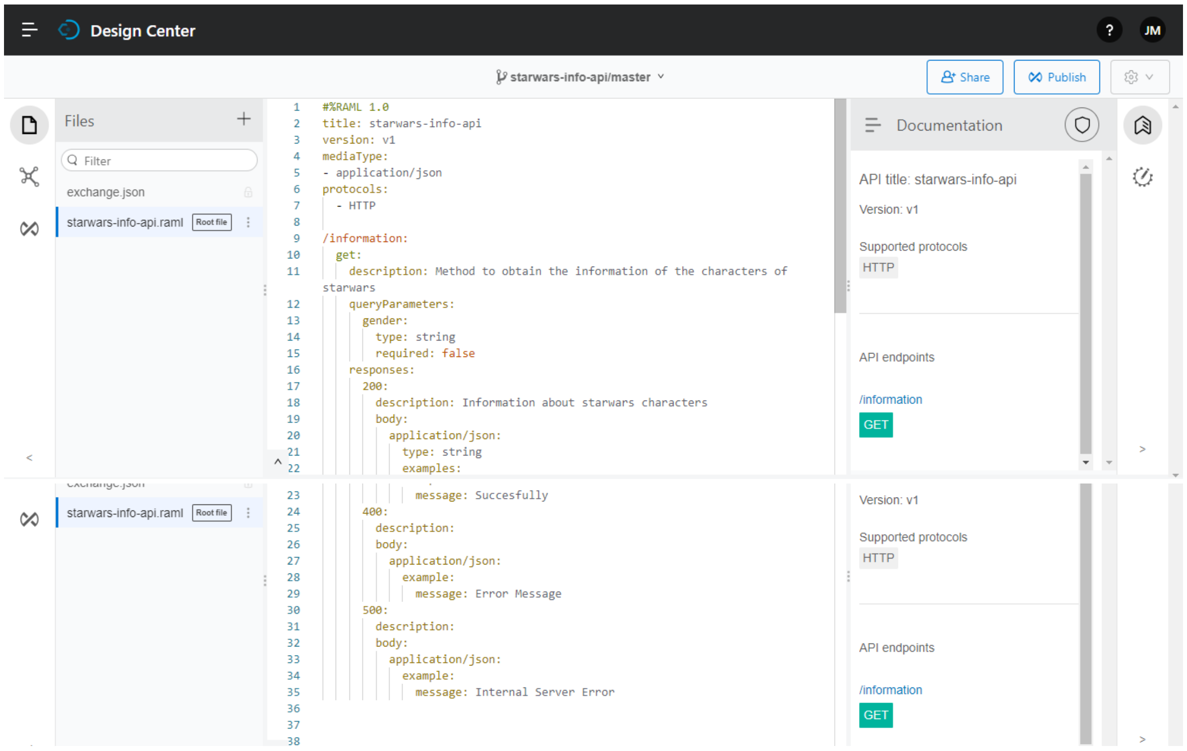
How do you want to draft the API Spec?

[✓ I'm comfortable designing it on my own]
A complete code editing experience with interactive documentation

Specification Language: RAML 1.0

Guide me through it
Use a visual interface scaffolding the API Specification (can generate both RAML & OAS)

2. Once the **API specification** is created, we begin to configure the version, media type, and protocol data that we will use. We define the **path** with which we will call our **GET method**, we also define the **query Parameter** to be able to filter the data, as well as their respective responses for response and error handling.



```
#%RAML 1.0
title: starwars-info-api
version: v1
mediaType:
- application/json
protocols:
  - HTTP

/information:
  get:
    description: Method to obtain the information of the characters of
starwars
    queryParameters:
      gender:
        type: string
        required: false
    responses:
      200:
        description: Information about starwars characters
        body:
          application/json:
            type: string
            examples:
              message: Succesfully
      400:
        description:
        body:
          application/json:
            example:
              message: Error Message
      500:
        description:
        body:
          application/json:
            example:
              message: Internal Server Error
```

API title: starwars-info-api

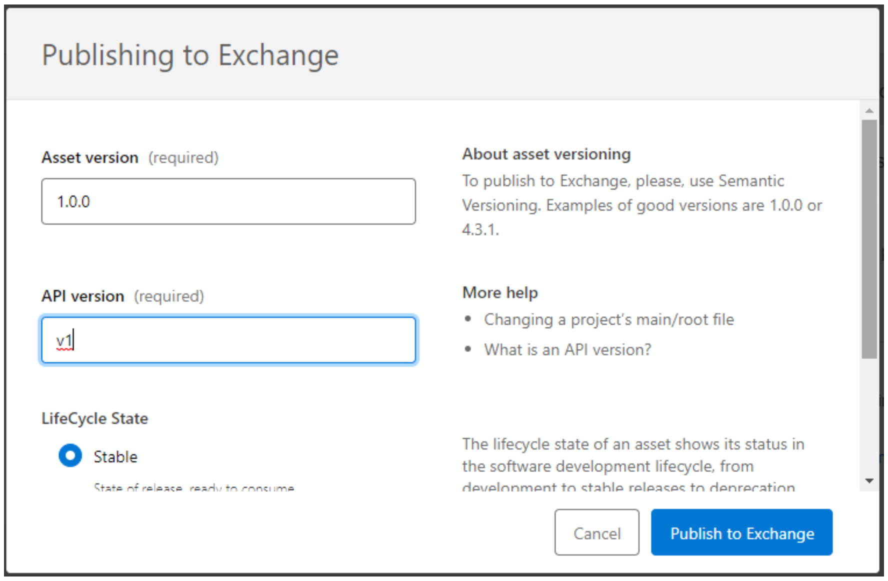
Version: v1

Supported protocols
HTTP

API endpoints

/information
GET

3. Once the design is finished, we publish it in **Exchange**, defining the version of the **Asset version** and **API version** to continue with the implementation.



Publishing to Exchange

Asset version (required)
1.0.0

API version (required)
v1

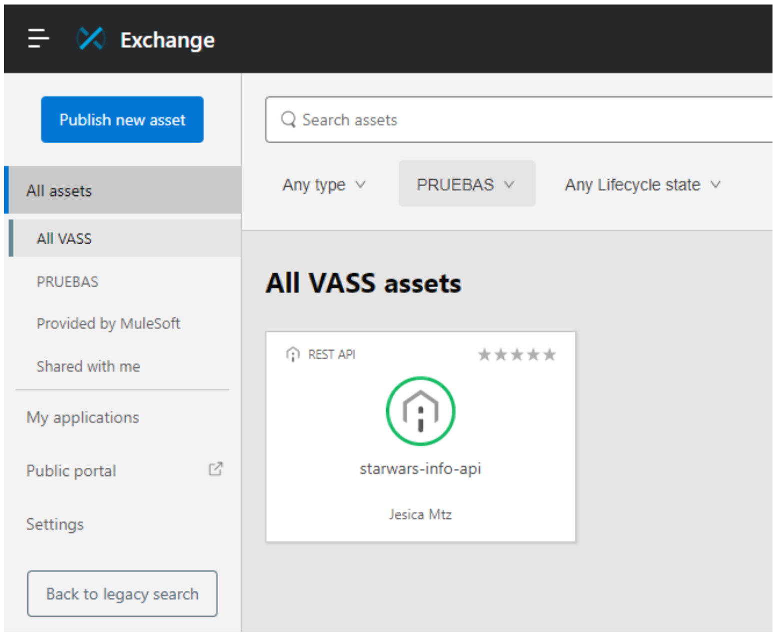
LifeCycle State
● Stable
State of release, ready to consume

About asset versioning
To publish to Exchange, please, use Semantic Versioning. Examples of good versions are 1.0.0 or 4.3.1.

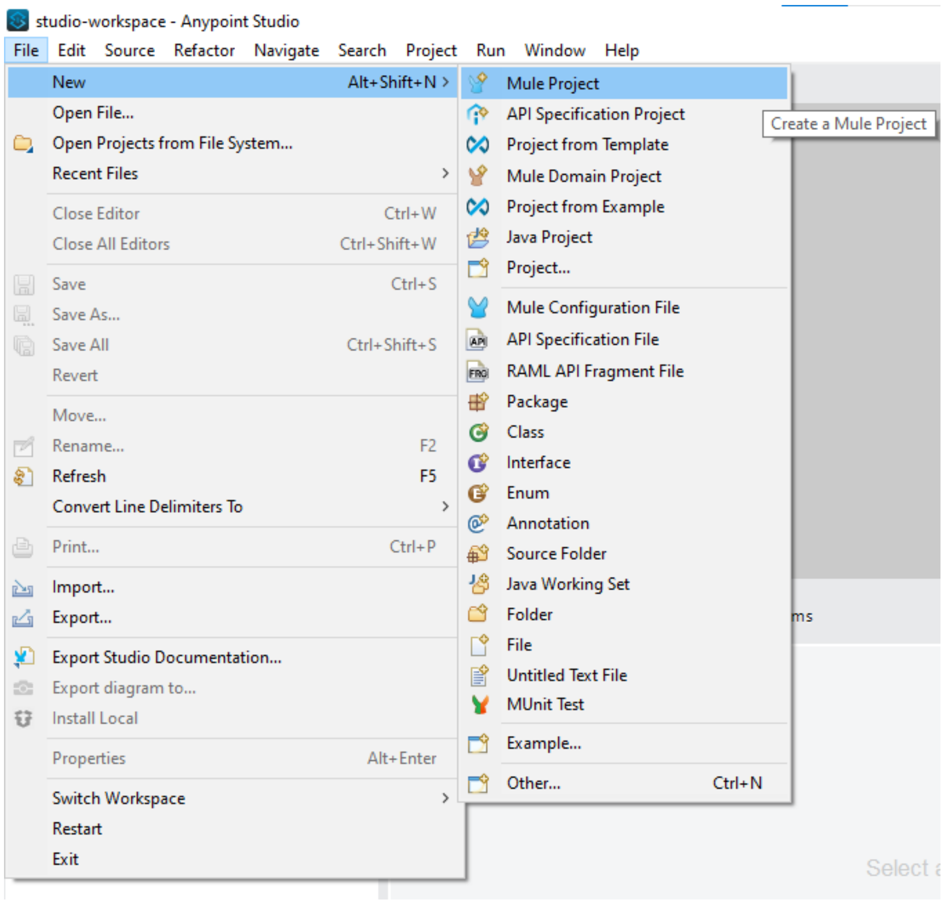
More help
- Changing a project's main/root file
- What is an API version?

The lifecycle state of an asset shows its status in the software development lifecycle, from development to stable releases to deprecation

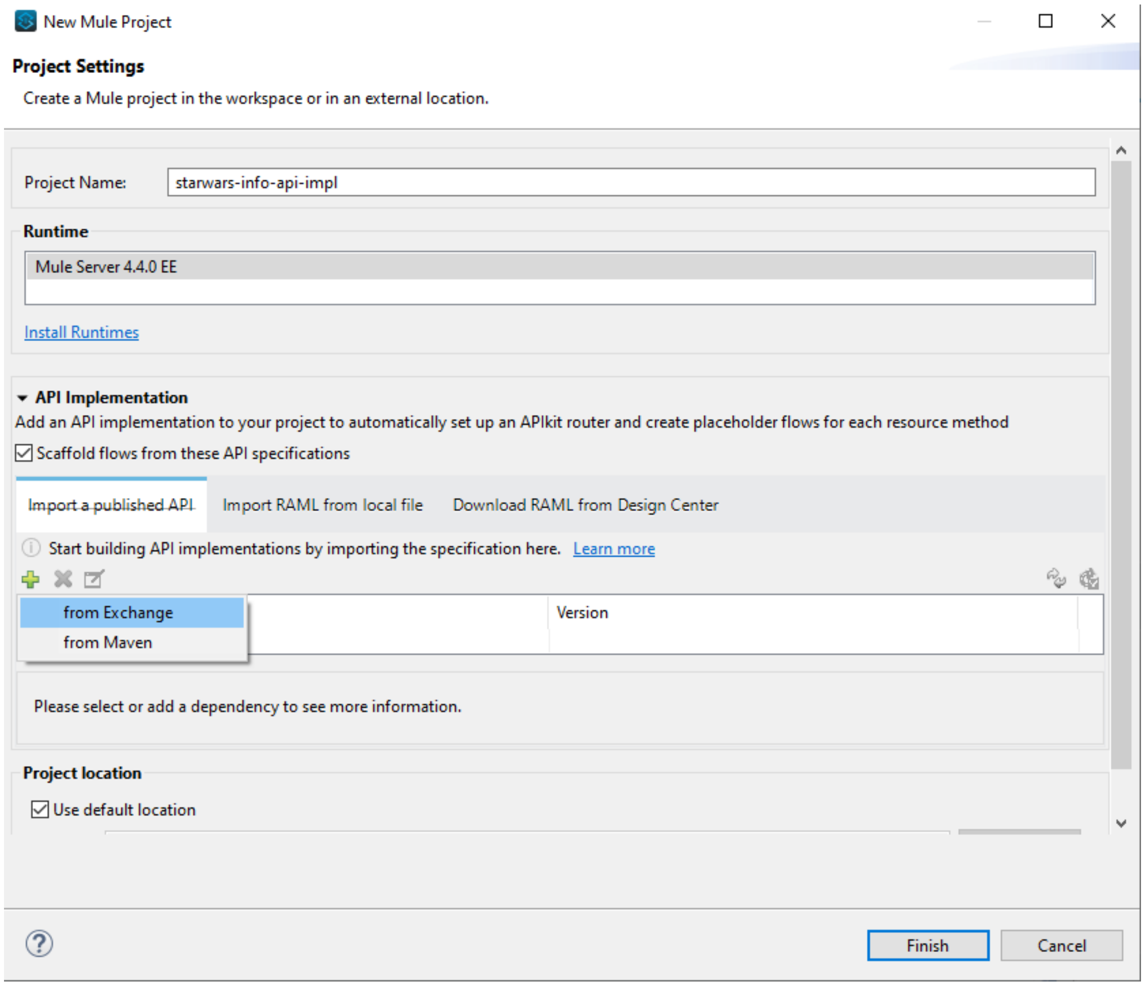
If we go to the **Exchange** window, we can see it.



4. To continue with the implementation, in Anypoint Studio we click **File > New > Mule Project**



5. We define the name of the project and in the **API Implementation** section we select the symbol **"+" > Exchange** to import the specification that we created previously, and we finish with the **OK** button


New Mule Project

Project Settings

Create a Mule project in the workspace or in an external location.

Project Name: starwars-info-api-impl

Runtime

Mule Server 4.4.0 EE

Install Runtimes

API Implementation

Add an API implementation to your project to automatically set up an APIkit router and create placeholder flows for each resource method

☑ Scaffold flows from these API specifications

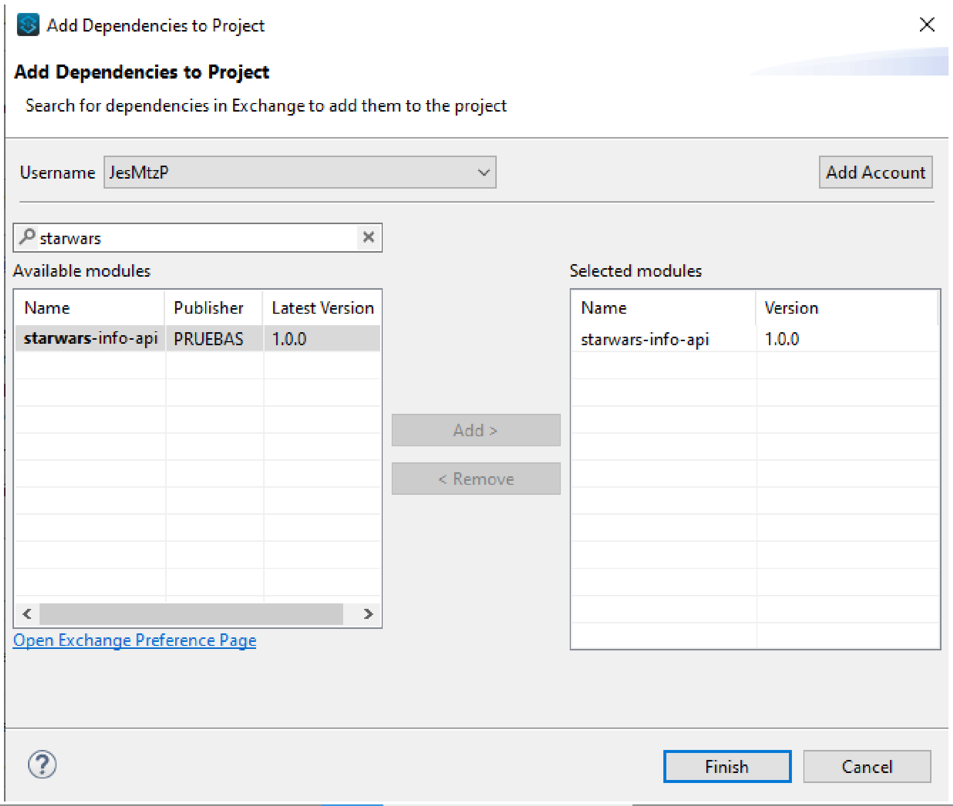
Import a published API | Import RAML from local file | Download RAML from Design Center

Start building API implementations by importing the specification here. Learn more

Please select or add a dependency to see more information.

Project location

☑ Use default location



Add Dependencies to Project

Add Dependencies to Project

Search for dependencies in Exchange to add them to the project

Username: JesMtzP

Add Account

starwars

Available modules

Name	Publisher	Latest Version
starwars-info-api	PRUEBAS	1.0.0

Add >

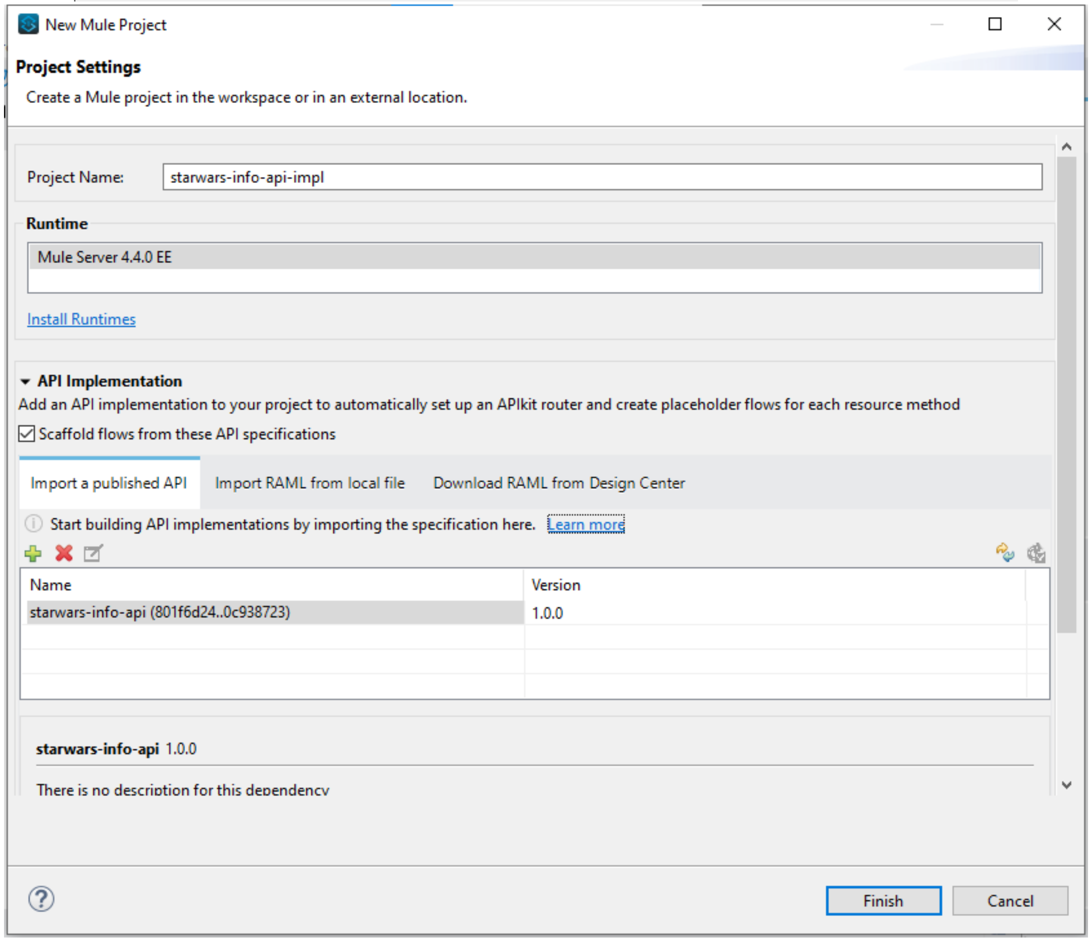
< Remove

Selected modules

Name	Version
starwars-info-api	1.0.0

Open Exchange Preference Page

Finish | Cancel



New Mule Project

Project Settings

Create a Mule project in the workspace or in an external location.

Project Name: starwars-info-api-impl

Runtime

Mule Server 4.4.0 EE

Install Runtimes

API Implementation

Add an API implementation to your project to automatically set up an APIkit router and create placeholder flows for each resource method

☑ Scaffold flows from these API specifications

Import a published API | Import RAML from local file | Download RAML from Design Center

Start building API implementations by importing the specification here. Learn more

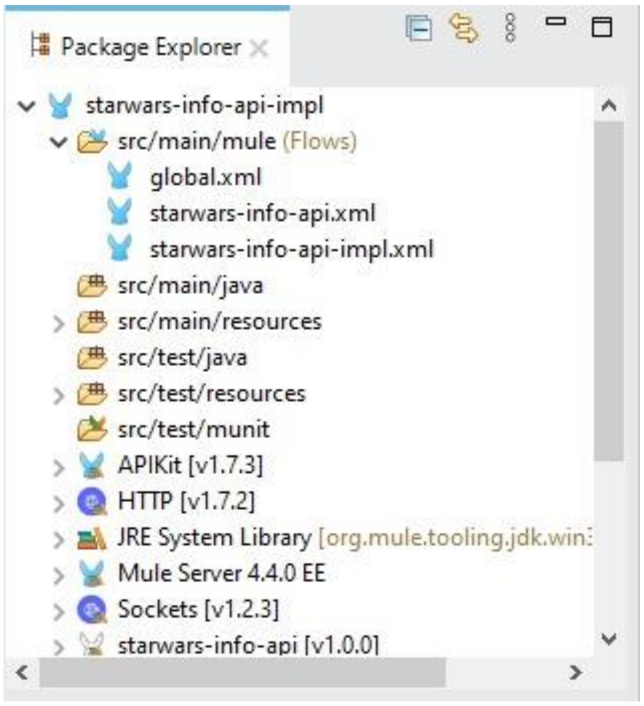
Name	Version
starwars-info-api (801f6d24..0c938723)	1.0.0

starwars-info-api 1.0.0

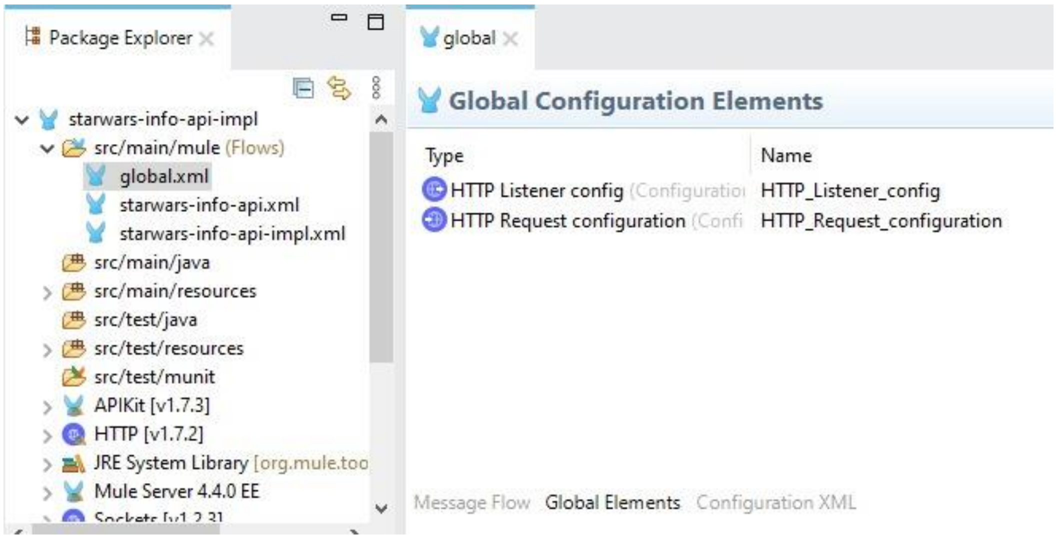
There is no description for this dependency

Finish | Cancel

6. Once the project is created it will come with two files in this case, **starwars-info-api.xml** (the API specification) and **starwars-info-api-impl.xml** (file created by default), we create a third one called **global. xml**

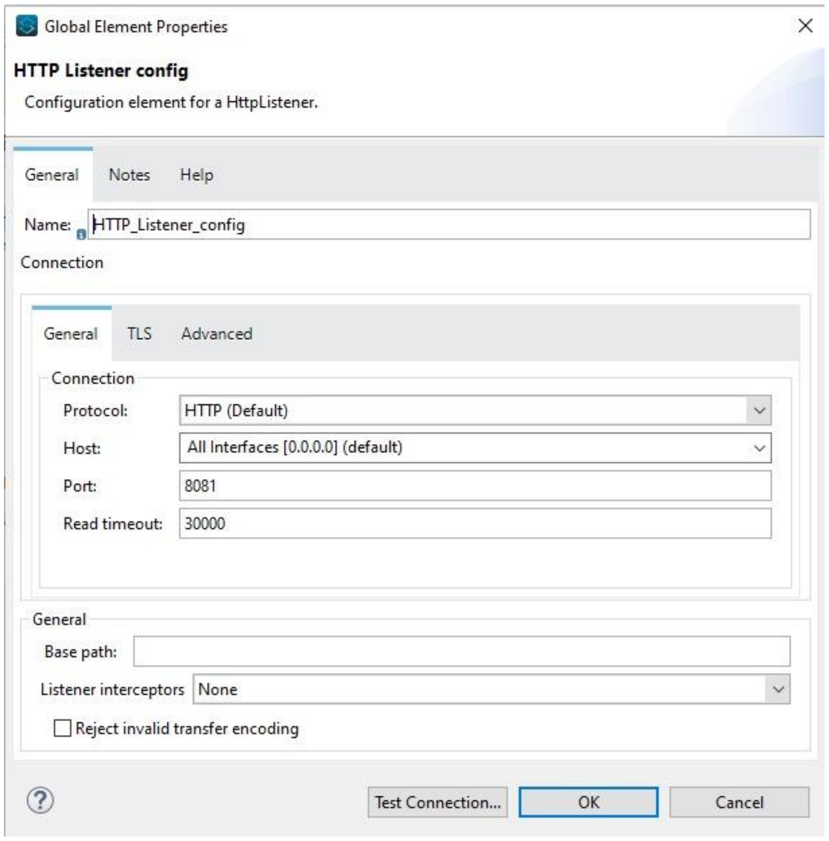


7. In the global file we create **two connectors**, one for the **Listener** and the other for the **Request** of **Swapi** consumption.



Global Configuration Elements

Type	Name
HTTP Listener config (Configuratio	HTTP_Listener_config
HTTP Request configuration (Confi	HTTP_Request_configuration



Global Element Properties

HTTP Listener config

Configuration element for a HttpListener.

Name: HTTP_Listener_config

Connection

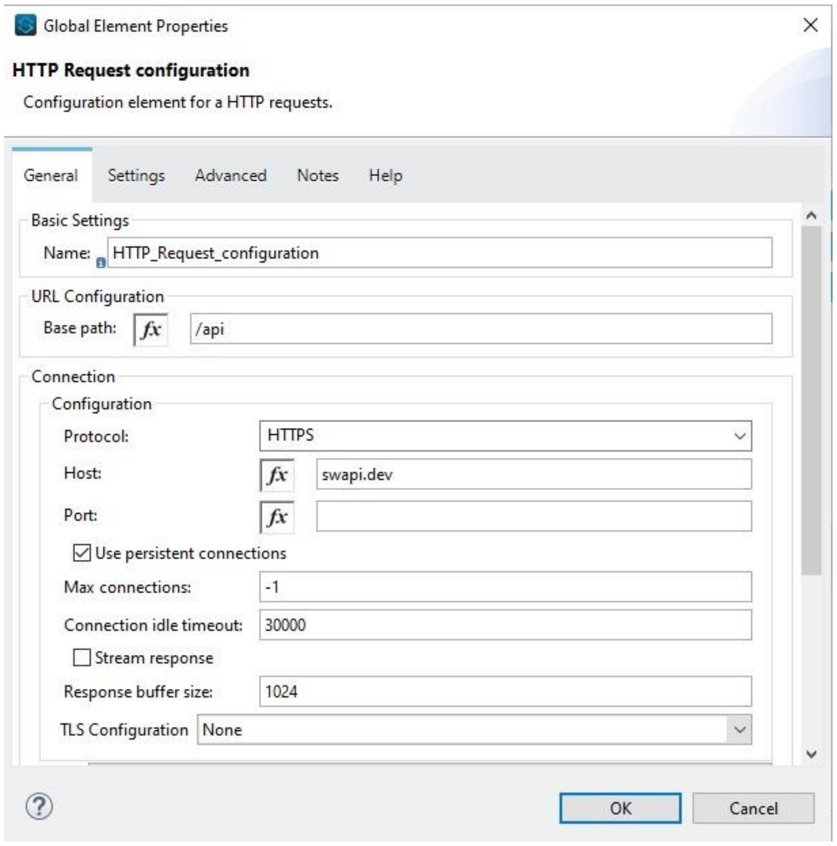
Connection

Protocol:	HTTP (Default)
Host:	All Interfaces [0.0.0.0] (default)
Port:	8081
Read timeout:	30000

General

Base path:	
Listener interceptors	None

☐ Reject invalid transfer encoding



Global Element Properties

HTTP Request configuration

Configuration element for a HTTP requests.

Basic Settings

Name: HTTP_Request_configuration

URL Configuration

Base path: /api

Connection

Configuration

Protocol:	HTTPS
Host:	swapi.dev
Port:	

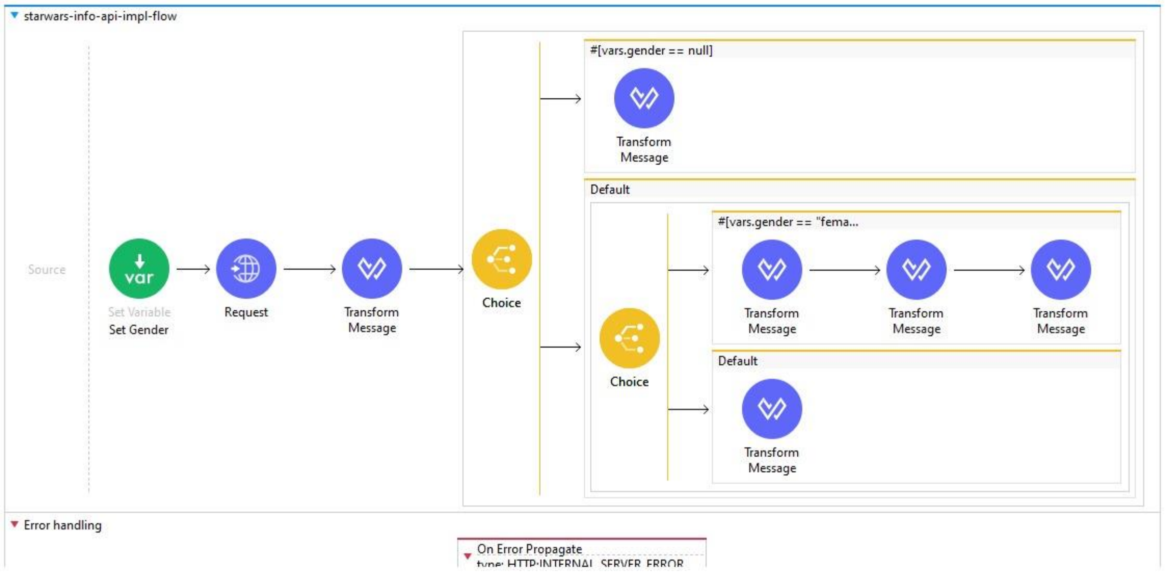
☑ Use persistent connections

Max connections:	-1
Connection idle timeout:	30000

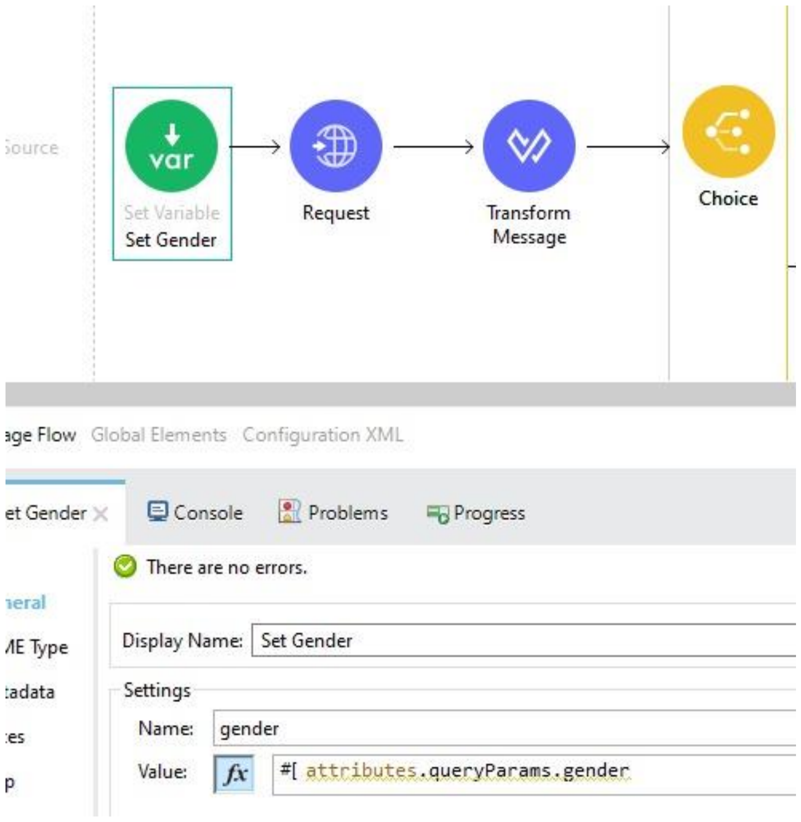
☐ Stream response

Response buffer size:	1024
TLS Configuration	None

8. Once the connectors are configured, the main flow is carried out, which is made up of:



~ A **variable** to store the value of the **query parameter**



There are no errors.

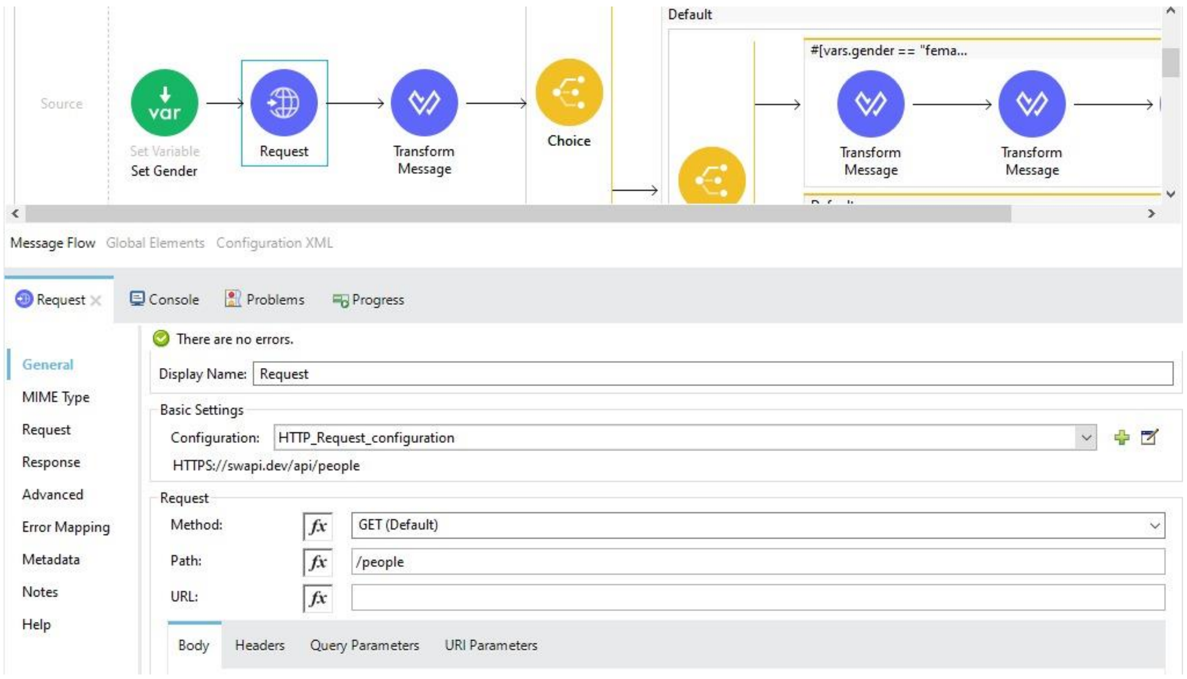
Display Name: Set Gender

Settings

Name: gender

Value: #[attributes.queryParams.gender

~ A **request** from the **Swapi** call where we will get the data from



There are no errors.

Display Name: Request

Basic Settings

Configuration: HTTP_Request_configuration

HTTPS://swapi.dev/api/people

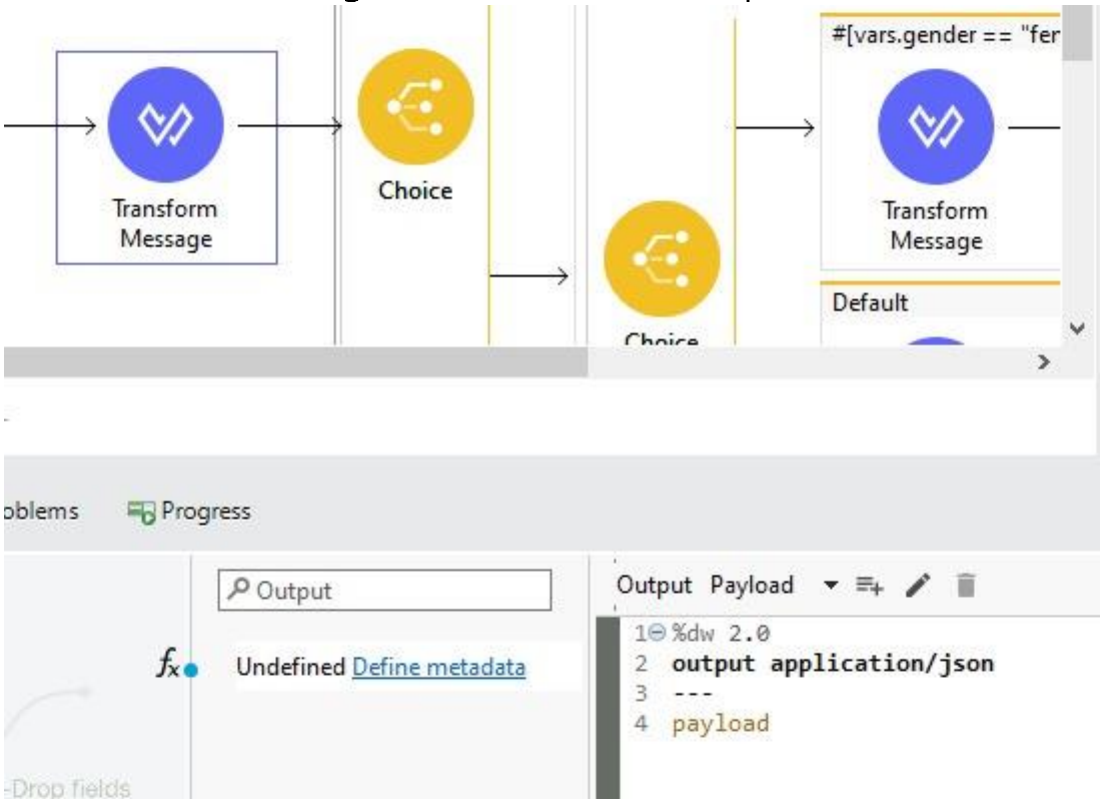
Request

Method: GET (Default)

Path: /people

URL:

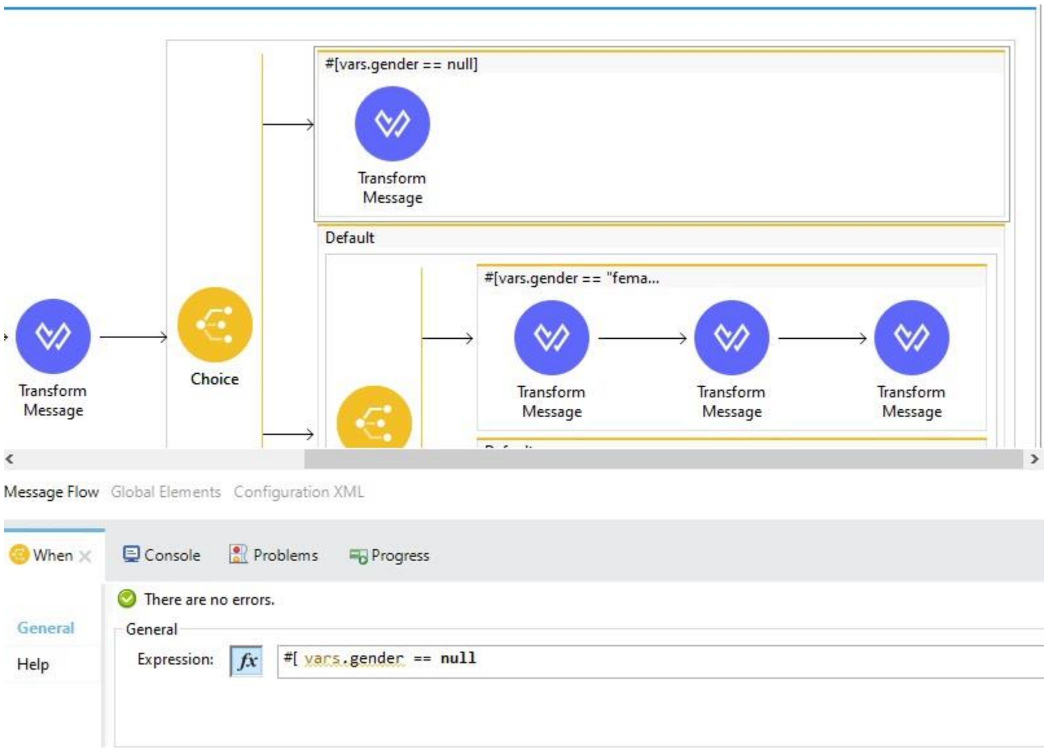
~ A **transform message** to save what the Swapi returns us



```
%dw 2.0
output application/json
---
payload
```

~

~ The main **Choice Router** to check if the **query parameter** was supplied, we compare it with the information returned by the variable we declared at the beginning.

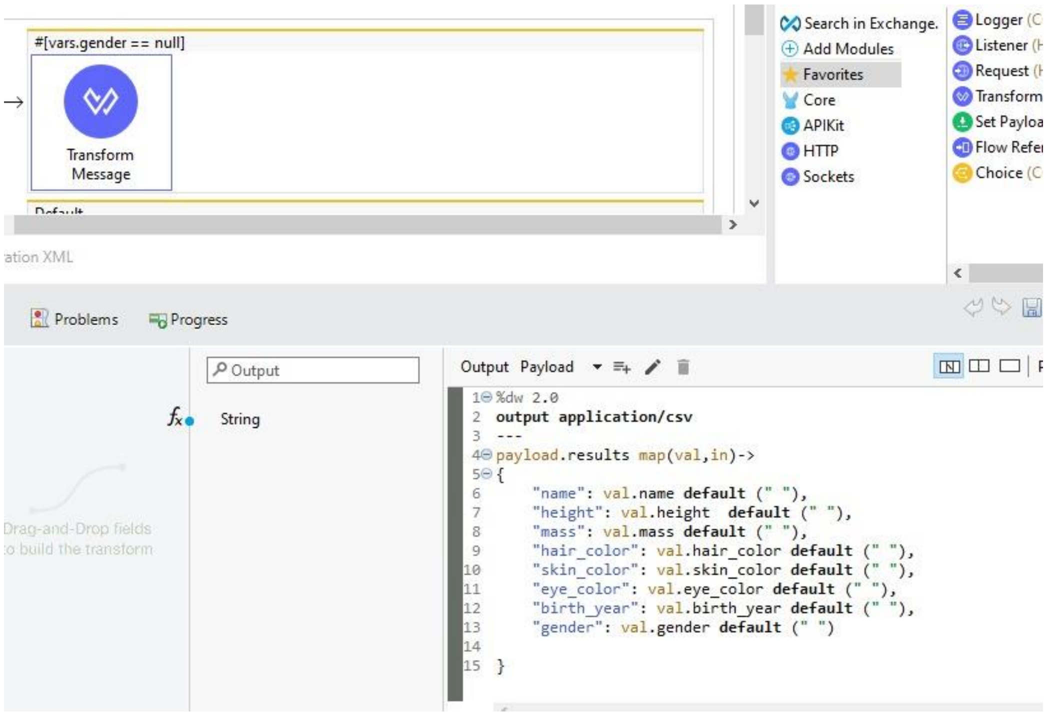


There are no errors.

General

Expression: `#[ vars.gender == null`

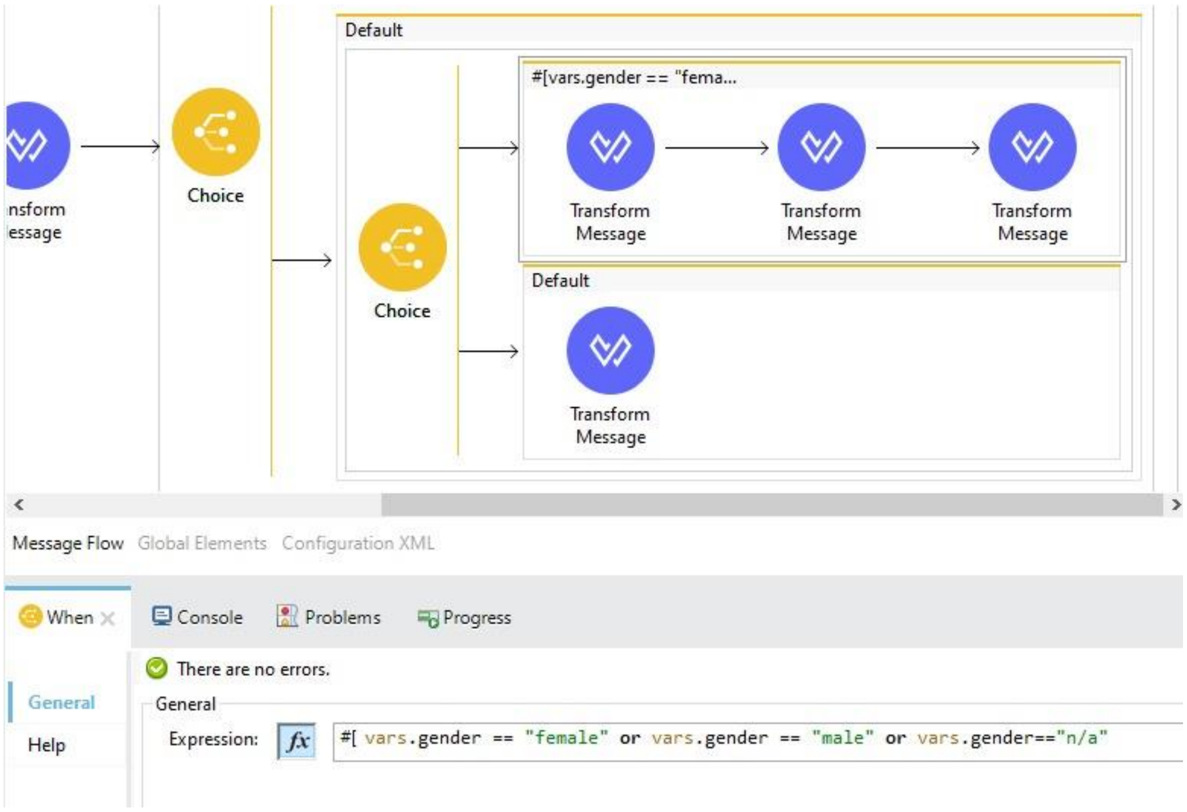
If empty or null, all data is sent with its proper transformation from **json to cvs**



```
%dw 2.0
output application/csv
---
payload.results map(val,in)->
{
     "name": val.name default (" "),
     "height": val.height  default (" "),
     "mass": val.mass default (" "),
     "hair_color": val.hair_color default (" "),
     "skin_color": val.skin_color default (" "),
     "eye_color": val.eye_color default (" "),
     "birth_year": val.birth_year default (" "),
     "gender": val.gender default (" ")

}
```

If it is not empty, we use another **Choice Router** to verify that the **query Parameter** data is what we need to filter **(female, male or n/a)**

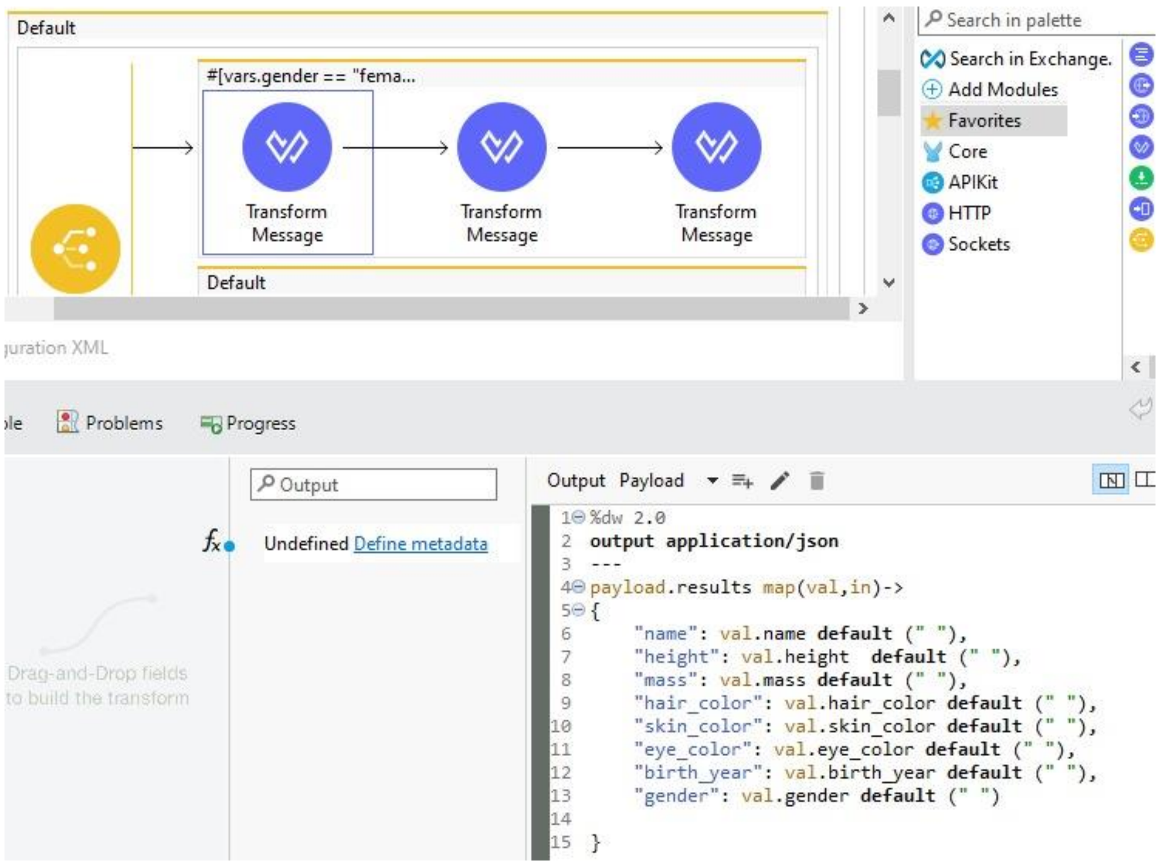


There are no errors.

General

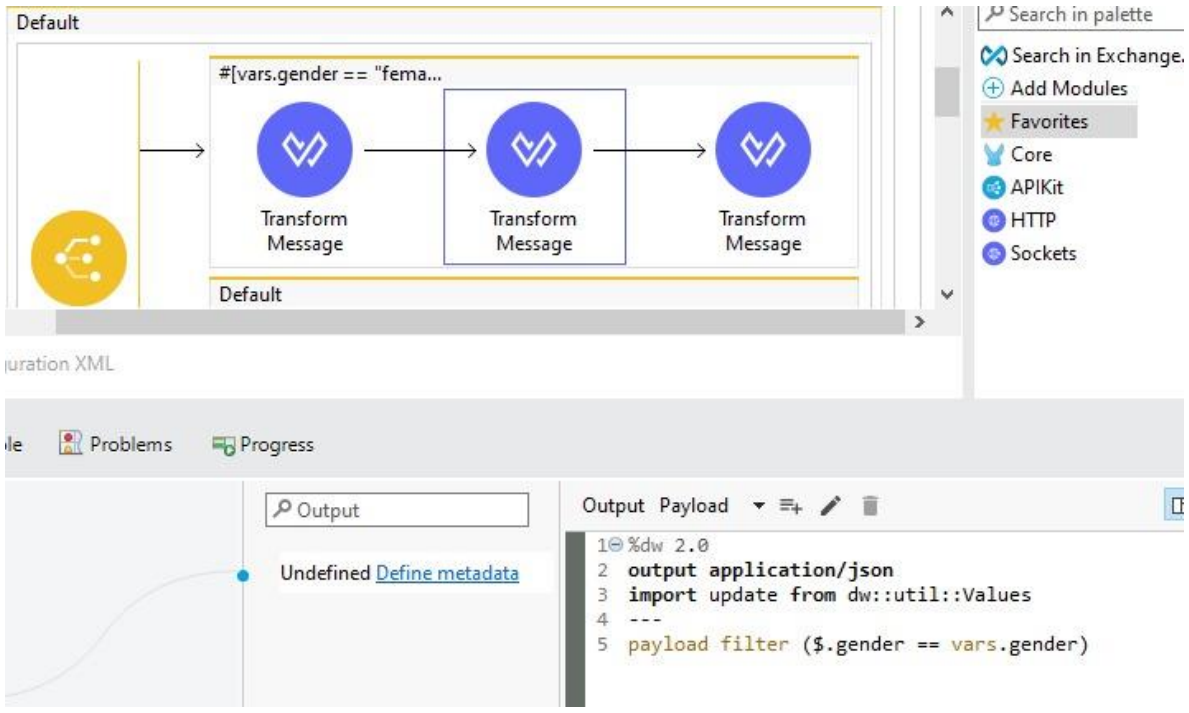
Expression: `#[ vars.gender == "female" or vars.gender == "male" or vars.gender=="n/a"`

If it is one of these options to filter, it will get the necessary data, and this will filter based on the data that has been entered and saved in the variable.



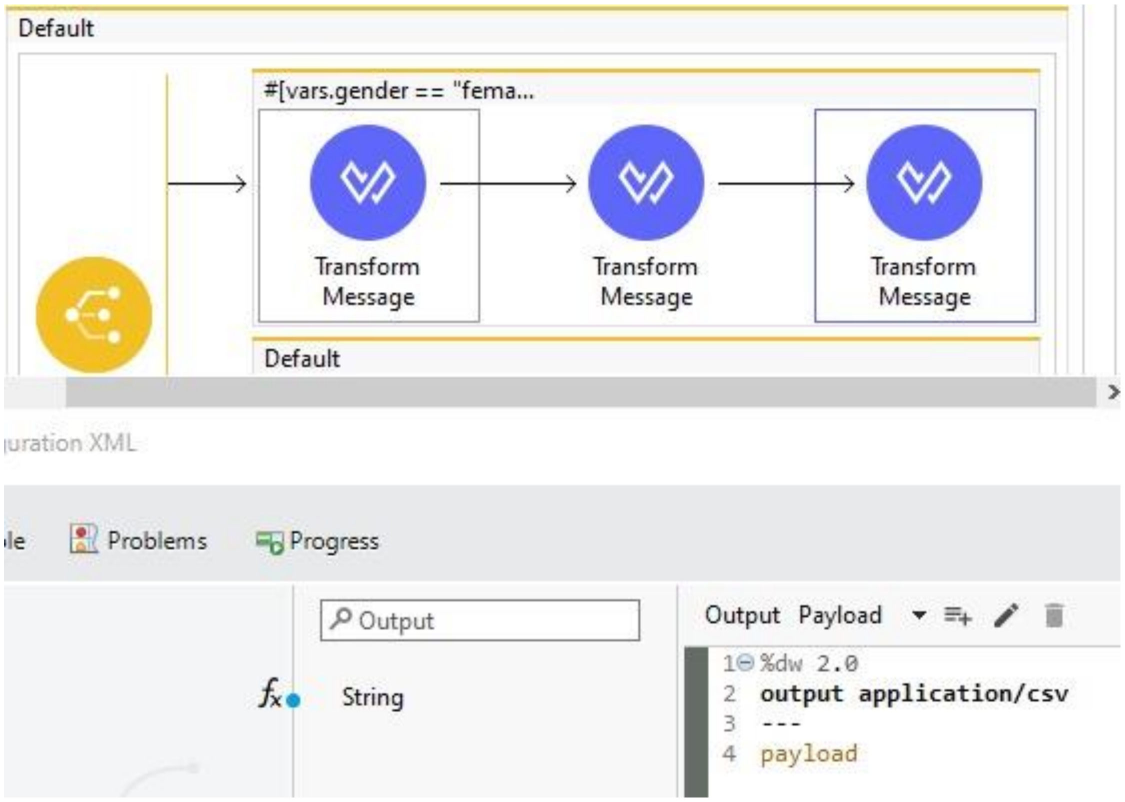
```
%dw 2.0
output application/json
---
payload.results map(val,in)->
{
     "name": val.name default (" "),
     "height": val.height  default (" "),
     "mass": val.mass default (" "),
     "hair_color": val.hair_color default (" "),
     "skin_color": val.skin_color default (" "),
     "eye_color": val.eye_color default (" "),
     "birth_year": val.birth_year default (" "),
     "gender": val.gender default (" ")

}
```



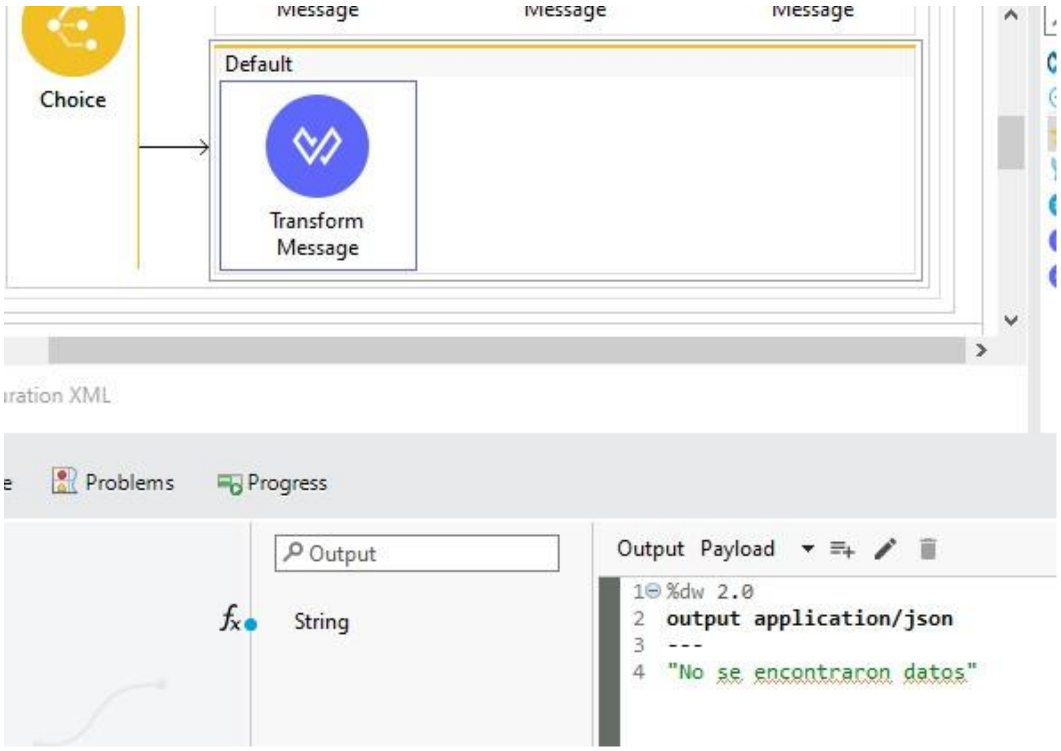
```
%dw 2.0
output application/json
import update from dw::util::Values
---
payload filter ($.gender == vars.gender)
```

And it will be displayed by means of a **transform message**



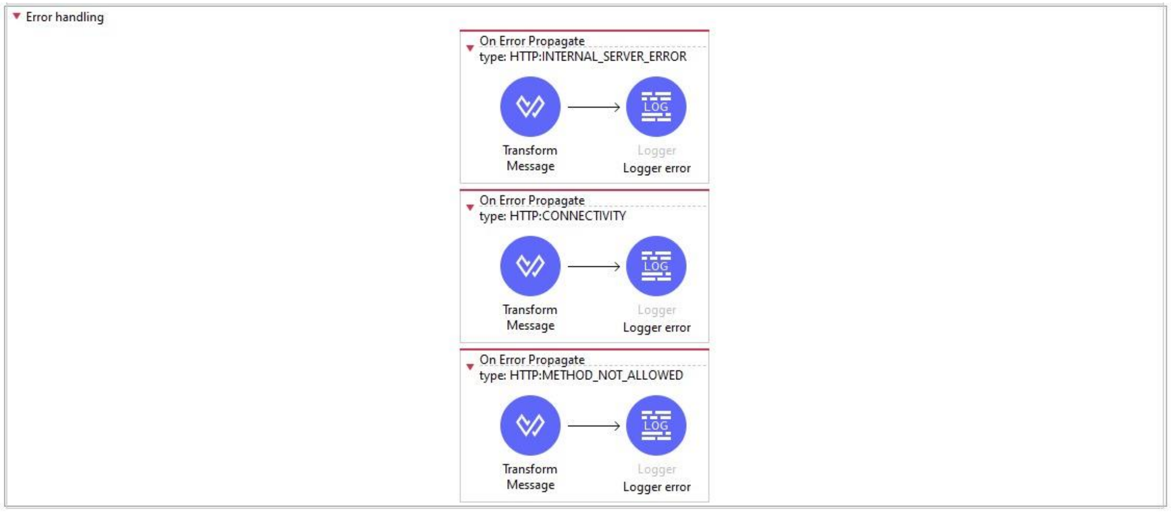
```
%dw 2.0
output application/csv
---
payload
```

And if there are no such filter options, a message that no data was found will be displayed.

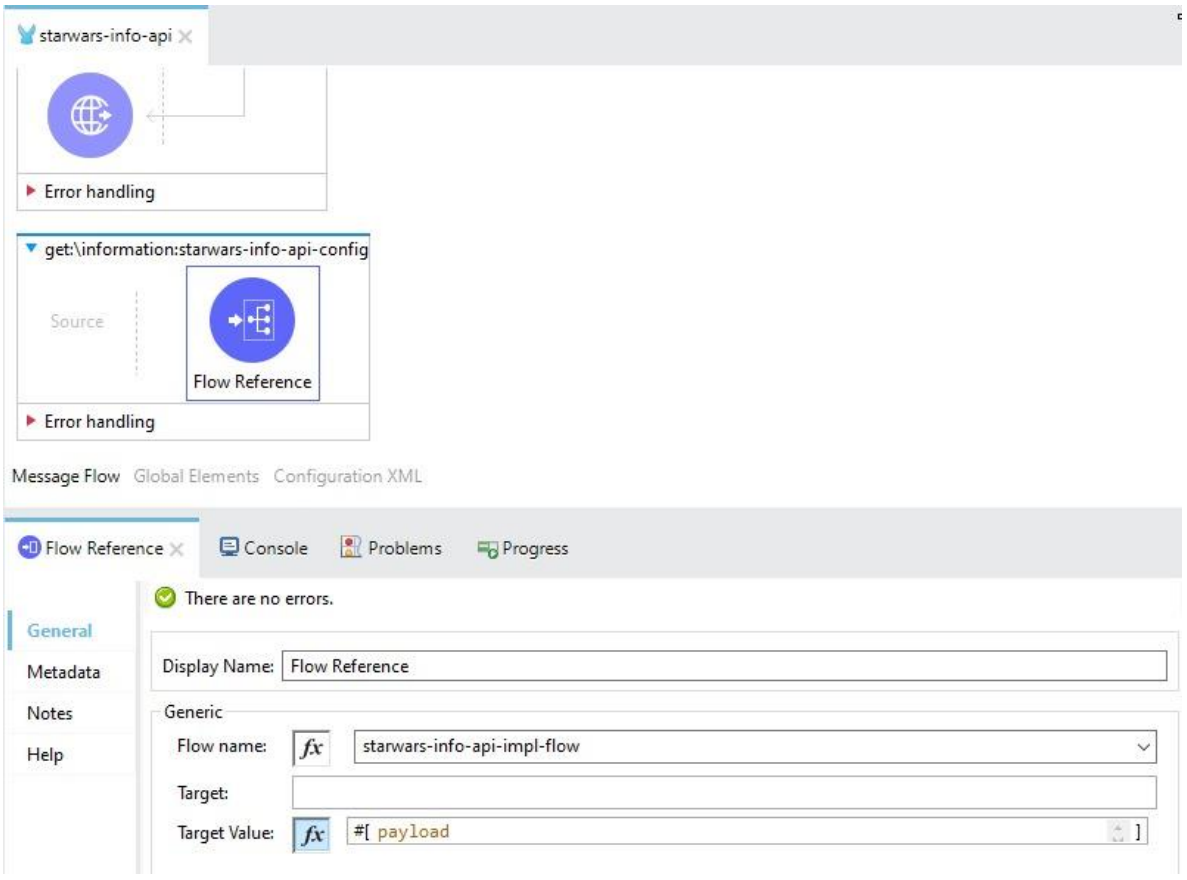


```
%dw 2.0
output application/json
---
"No se encontraron datos"
```

9. Added **On Error Propagate** to handle possible errors



10. Having completed the main flow, it remains to associate it with the **starwars-info-api.xml** file that was generated by the **API specification**.



There are no errors.

Display Name: Flow Reference

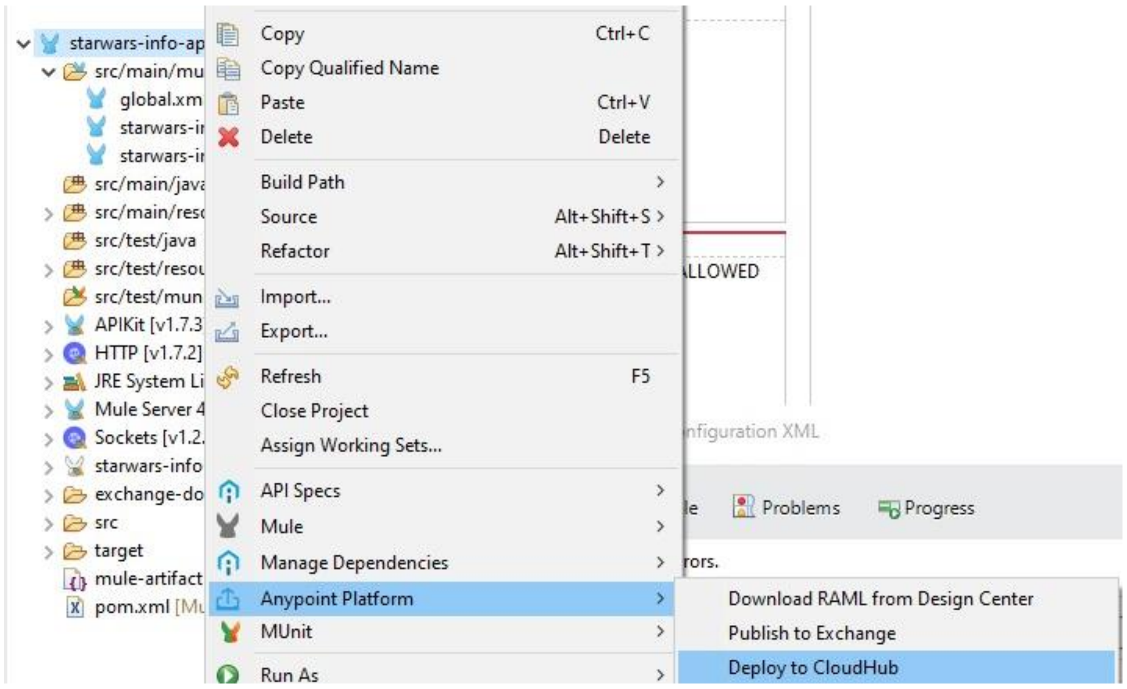
Generic

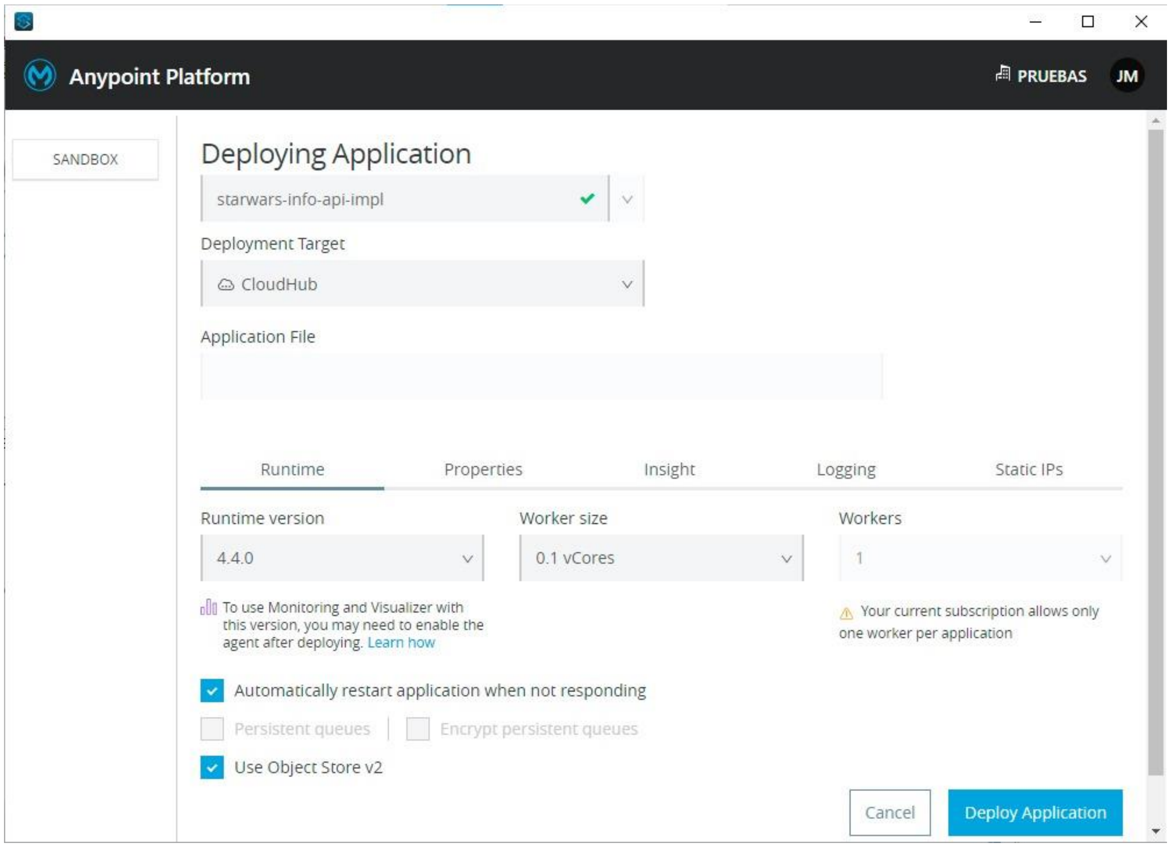
Flow name: starwars-info-api-impl-flow

Target:

Target Value: #[payload]

11. Once the implementation is finished, we deploy it to CloudHub to start consuming it and testing, for this we right click on the **project folder > Anypoint Platform > Deploy to CloudHub**




Anypoint Platform

PRUEBAS JM

SANDBOX

Deploying Application

starwars-info-api-impl

Deployment Target: CloudHub

Application File

Runtime | Properties | Insight | Logging | Static IPs

Runtime version: 4.4.0

Worker size: 0.1 vCores

Workers: 1

To use Monitoring and Visualizer with this version, you may need to enable the agent after deploying. Learn how

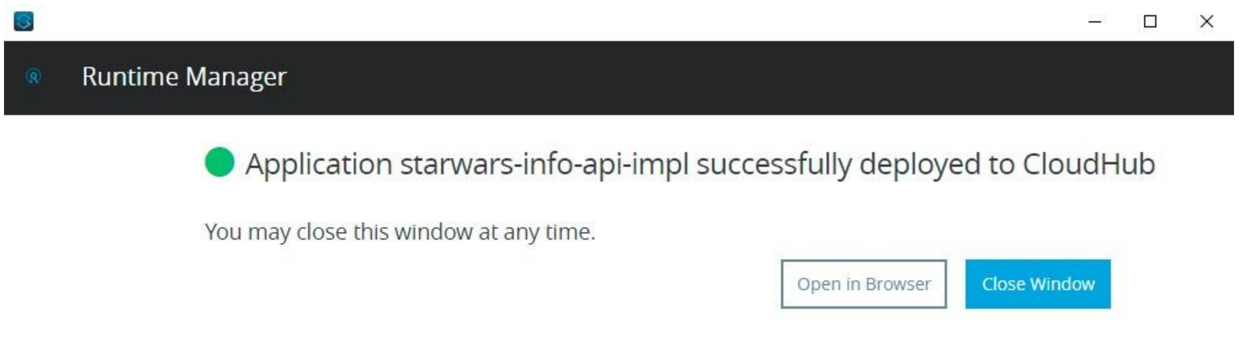
Your current subscription allows only one worker per application

☑ Automatically restart application when not responding

☐ Persistent queues | ☐ Encrypt persistent queues

☑ Use Object Store v2

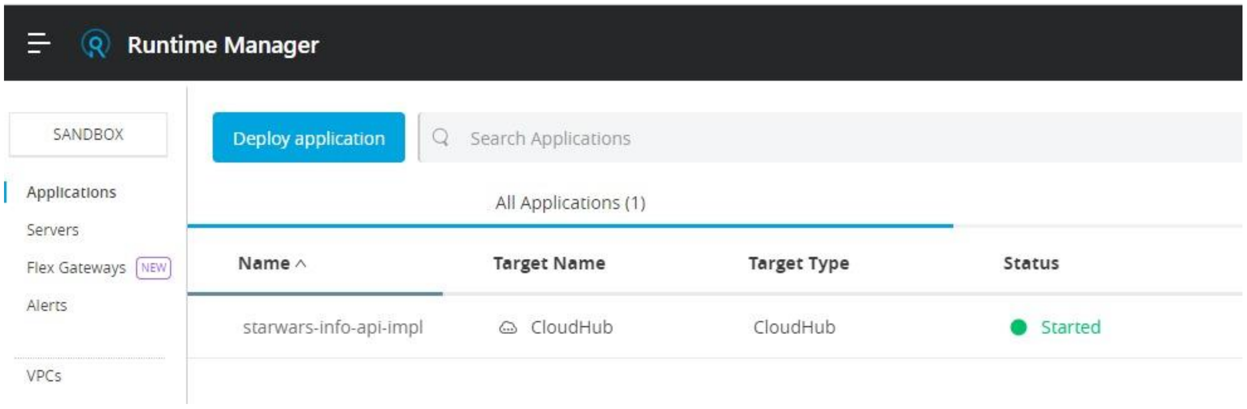
Cancel | Deploy Application


Runtime Manager

Application starwars-info-api-impl successfully deployed to CloudHub

You may close this window at any time.

12. In the **Anypoint Platform > Runtime Manager** you can see the api already deployed and from there we obtain the url to be able to carry out the tests http://starwars-info-api-impl.us-e2.cloudhub.io/api/information

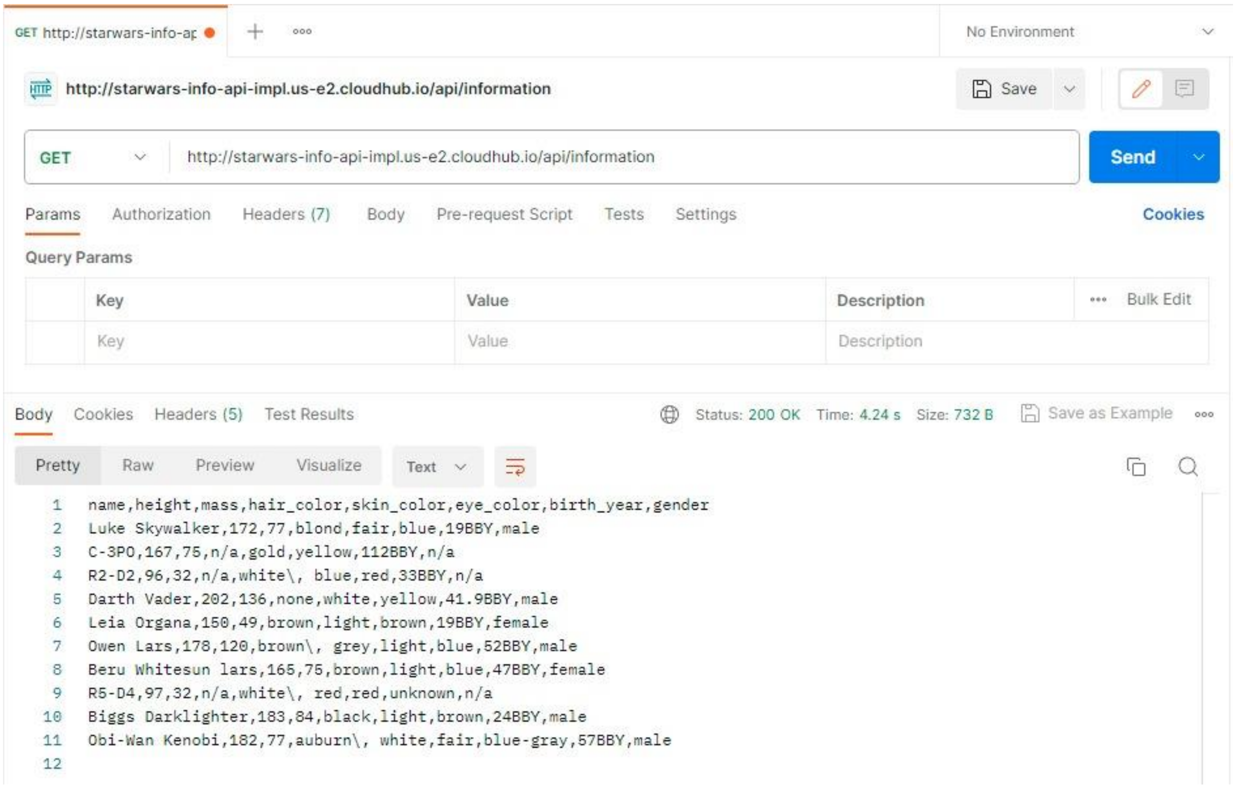


Name ^	Target Name	Target Type	Status
starwars-info-api-impl	CloudHub	CloudHub	● Started

Jesica Martínez Palacios

TEST 1

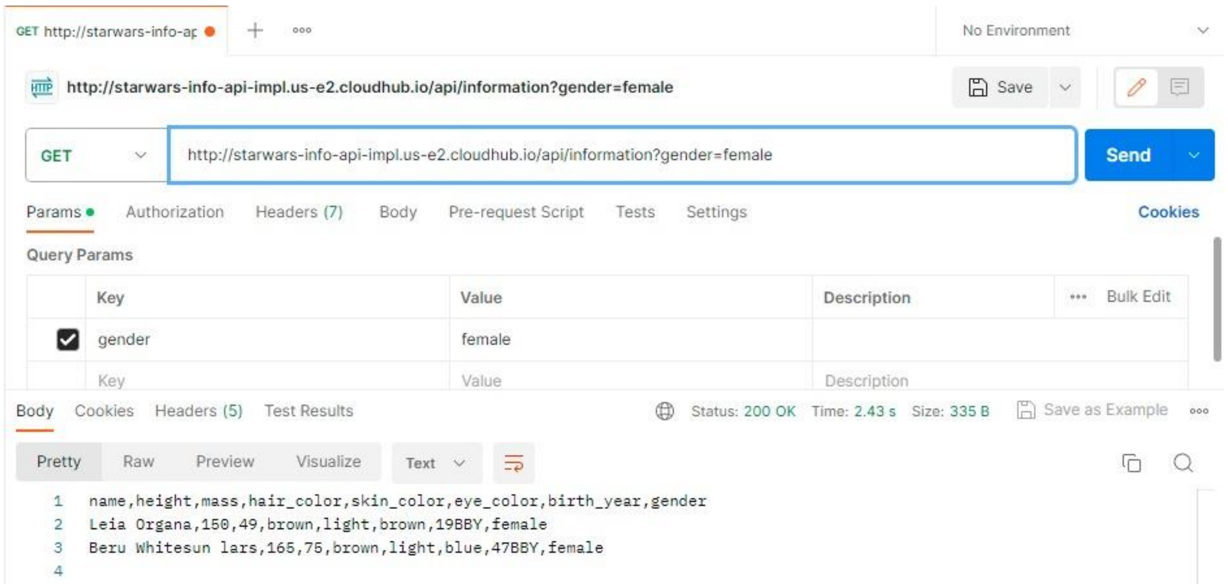
http://starwars-info-api-impl.us-e2.cloudhub.io/api/information



```
name,height,mass,hair_color,skin_color,eye_color,birth_year,gender
Luke Skywalker,172,77,blond,fair,blue,19BBY,male
C-3PO,167,75,n/a,gold,yellow,112BBY,n/a
R2-D2,96,32,n/a,white\, blue,red,33BBY,n/a
Darth Vader,202,136,none,white,yellow,41.9BBY,male
Leia Organa,150,49,brown,light,brown,19BBY,female
Owen Lars,178,120,brown\, grey,light,blue,52BBY,male
Beru Whitesun lars,165,75,brown,light,blue,47BBY,female
R5-D4,97,32,n/a,white\, red,red,unknown,n/a
Biggs Darklighter,183,84,black,light,brown,24BBY,male
Obi-Wan Kenobi,182,77,auburn\, white,fair,blue-gray,57BBY,male
```

TEST 2

http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=female

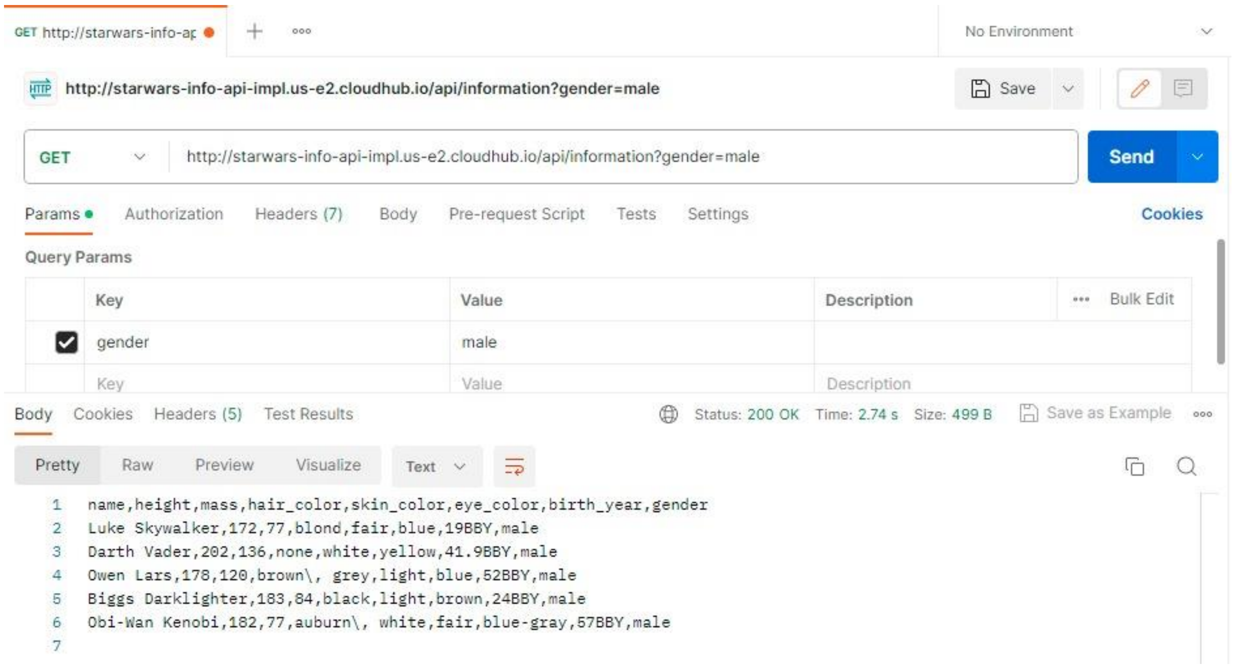


Key	Value	Description
✓ gender	female	

```
name,height,mass,hair_color,skin_color,eye_color,birth_year,gender
Leia Organa,150,49,brown,light,brown,19BBY,female
Beru Whitesun lars,165,75,brown,light,blue,47BBY,female
```

TEST 3

http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=male



http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=male

GET http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=male

Query Params

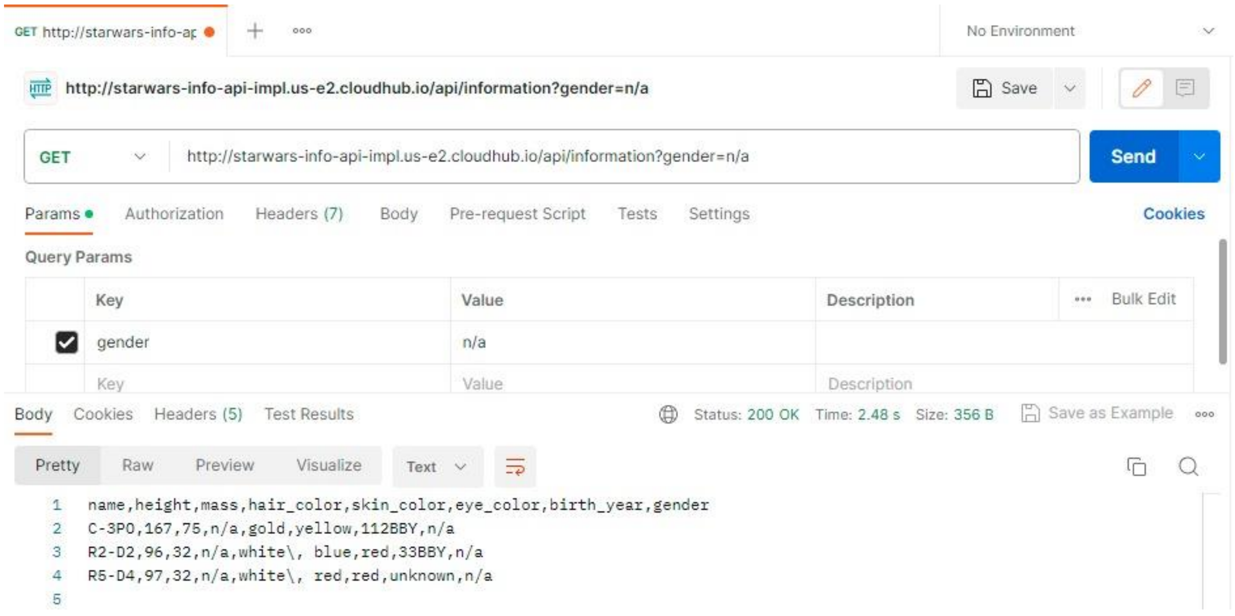
	Key	Value	Description
✓	gender	male	
	Key	Value	Description

Status: 200 OK Time: 2.74 s Size: 499 B

```
name,height,mass,hair_color,skin_color,eye_color,birth_year,gender
Luke Skywalker,172,77,blond,fair,blue,19BBY,male
Darth Vader,202,136,none,white,yellow,41.9BBY,male
Owen Lars,178,120,brown\, grey,light,blue,52BBY,male
Biggs Darklighter,183,84,black,light,brown,24BBY,male
Obi-Wan Kenobi,182,77,auburn\, white,fair,blue-gray,57BBY,male
```

TEST 4

http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=n/a



http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=n/a

GET http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=n/a

Query Params

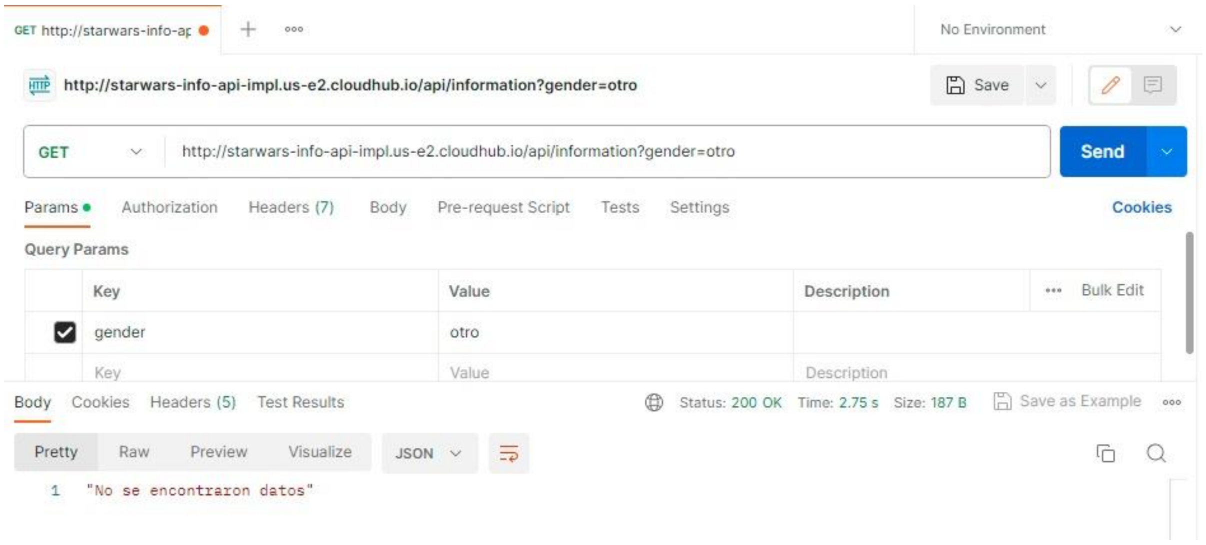
	Key	Value	Description
✓	gender	n/a	
	Key	Value	Description

Status: 200 OK Time: 2.48 s Size: 356 B

```
name,height,mass,hair_color,skin_color,eye_color,birth_year,gender
C-3PO,167,75,n/a,gold,yellow,112BBY,n/a
R2-D2,96,32,n/a,white\, blue,red,33BBY,n/a
R5-D4,97,32,n/a,white\, red,red,unknown,n/a
```

TEST 5

http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=otro



http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=otro

GET http://starwars-info-api-impl.us-e2.cloudhub.io/api/information?gender=otro

Query Params

	Key	Value	Description
✓	gender	otro	
	Key	Value	Description

Body — Status: 200 OK Time: 2.75 s Size: 187 B

```
"No se encontraron datos"
```