Une image contenant texte, signe

Description générée automatiquement

**How to start WRAPI Example Middleware**

**Author**

Quentin WENZINGER – Writer

**DOCUMENT TRACKING**

|  |  |  |  |
| --- | --- | --- | --- |
| Mises à jour | | | |
| *Version* | *Date* | *Authors* | *Purpose of the update* |
| 1.0 | 04/01/2022 | Quentin WENZINGER | Initial version |
| 2.0 | 24/01/2022 | Raunak Lodha | Update for Beta Testing |
| 3.0 | 01/02/2022 | Quentin WENZINGER | Adding AWS exemple |
| 4.0 | 06/11/2022 | Quentin WENZINGER | Adding environment variables |
| 5.0 | 18/05/2023 | Quentin WENZINGER | Updating git source url |

|  |  |  |  |
| --- | --- | --- | --- |
| Mailing list | | | |
| Radioplayer France |  | Radioplayer WorldWide |  |
| Yann Legarson | *I* | Michael Hill | *I* |
| Victor Perrot | *I* | Leo Andrews | *O* |
|  |  | Raunak Lodha | *O* |
|  |  | Laurence Harrison | *I* |
|  |  | Lawrence Galkoff | *I* |

***A*** *= Application,* ***O*** *= Observations,* ***I*** *= Information,* ***V****= Validation*

**SUMMARY**

[1. Prerequisites 4](#_Toc93944149)

[2. Clone/Download Middleware gitlab 4](#_Toc93944150)

[3. Install Dependencies 4](#_Toc93944151)

[4. HTTPS private key and certification deposit 5](#_Toc93944152)

1. [HTTPS\_PRIVATE\_KEY=privkey 5](#_Toc93944153)
2. [HTTPS\_CERTIFICATION=cert 5](#_Toc93944154)

[5. WRAPI private key deposit 5](#_Toc93944155)

[6. Middleware Environment Settings 6](#_Toc93944156)

[7. Start Middleware 6](#_Toc93944157)

[8. Endpoints – Queries, Example & Responses 6](#_Toc93944158)

1. [Queries & SoapUI project setup 6](#_Toc93944159)
2. [Example 7](#_Toc93944160)
3. [Responses 8](#_Toc93944161)

[9. WRAPI Error Handling 8](#_Toc93944162)

[10. Common Errors 9](#_Toc93944163)

# Prerequisites

IDE – *e.g. Visual Studio Code, etc.*

*Node.js*

Command Line Tool – *e.g. Windows Power Shell, etc.*

API Testing Tool – *SoapUI, Postman, etc.*

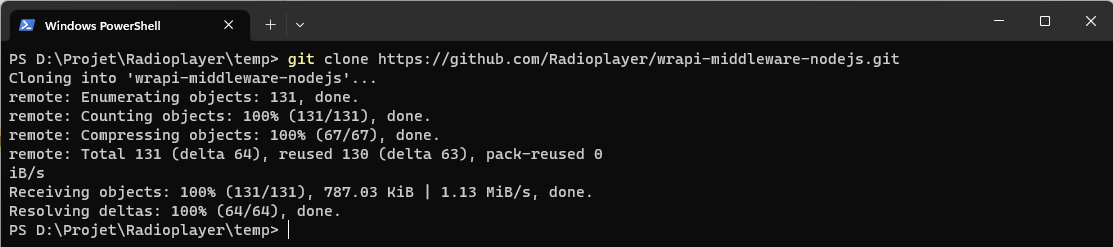
RSA Key Generator (2048 bit) – e.g. *Win64 OpenSSL Command Prompt* or other tools where *OpenSSL* commands can be run.

# Clone/Download Middleware gitlab

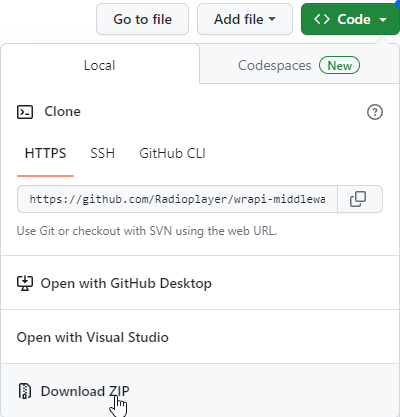
Using Windows Powershell or another SW tool of your choice navigate to the preferred project folder/directory e.g. cd projectfolder

Cloning by git command :

git clone <https://github.com/Radioplayer/wrapi-middleware-nodejs.git>



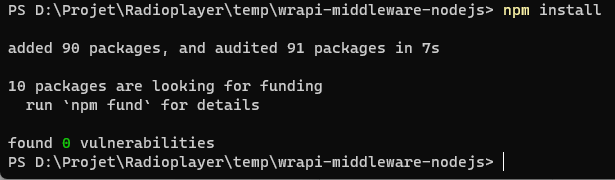
Or download it and extract zip into the project folder of your choice.



# Install Dependencies

The installation of dependencies allowing the creation of a Middleware is done by the following command and works on Mac OS/Windows/Linux OS:

npm install



# HTTPS private key and certification deposit

Please deposit your HTTPS private key and certification that you wish to use in the root folder of the middleware sources.

# HTTPS\_PRIVATE\_KEY=privkey

This file can be created using Win64 OpenSSL Command Prompt or another tool using the command - openssl req -newkey rsa:2048 -new -nodes -x509 -days 3650 -keyout key.pem -out cert.pem

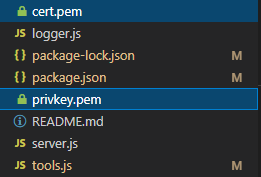
Once generated, put this file in the middleware root folder *wrapi-example-middleware.*

Note – You may need to rename this file to match with the .env variable name *privkey* (Please check - 6. Middleware Environment Settings).

# HTTPS\_CERTIFICATION=cert

This file can be created using Win64 OpenSSL Command Prompt or another tool using the command - openssl req -newkey rsa:2048 -new -nodes -x509 -days 3650 -keyout key.pem -out cert.pem

Once generated, put this file in the middleware root folder *wrapi-example-middleware.*



# WRAPI private key deposit

Please deposit your WRAPI private key that you wish to use in the root folder of the middleware sources e.g. wrapi-example-middleware

Une image contenant texte

Description générée automatiquement

Note – this is the WRAPI Key.pem file provided to you by Radioplayer.

Please do not change the original name as its name also represents your Key ID.

# Middleware Environment Settings

Please open the .env file located in the root folder of the middleware sources.

There you will find all the environment variables necessary for the middleware settings.

Une image contenant texte

Description générée automatiquement

Please fill in the different fields in accordance with the following standards:

* HTTPS\_PRIVATE\_KEY : must be the name of the file without its .pem extension
* HTTPS\_CERTIFICATION : must be the name of the file without its .pem extension
* WRAPI\_KEY\_ID : must be the name of the file without its .pem extension
* CUSTOM\_CACHING : must be a Boolean (true, false)
* ONAIR\_CACHING : must be a number in milliseconds
* SCHEDULE\_CACHING : must be a number in milliseconds
* ONDEMAND\_CACHING : must be a number in milliseconds
* STATIONS\_CACHING : must be a number in milliseconds
* CATEGORIES\_CACHING : must be a number in milliseconds
* RECOMMENDATIONS\_CACHING : must be a number in milliseconds
* frApiUrl : French API URL
* frApiAuth : French API credential

# Start Middleware

The middleware is launched by the following command:

npm start

Une image contenant texte

Description générée automatiquement

# Endpoints – Queries, Example & Responses

# Queries & SoapUI project setup

Download SoapUI Open Source (https://www.soapui.org/)

Download SoapUI test project (WRAPI Example Middleware -soapui-project.xml) you received, put it into your Middleware root folder and import the project into SoapUI.

You can also use other API tools to test e.g. POSTMAN.

A picture containing text

Description automatically generated

# Example

Let’s choose one endpoint

Text

Description automatically generated

Set parameters as required

Graphical user interface, application, table

Description automatically generated

Run and check results.

Graphical user interface, text, application

Description automatically generated

# Responses

Middleware responses are rebuilt like WRAPI responses

Files concerned are meta.js and response.js

Graphical user interface, application

Description automatically generated

# WRAPI Error Handling

* Every WRAPI call is covered by error handling and tries to catch errors to prevent the middleware from crashing.
* For every error that is detected when running the Middleware, a log is created and:
  + Sent to a file (middleware\_error.txt) and to console. This log file includes information about the errors.
  + Sends Wrapi Error Code to user if the error response comes from WRAPI
  + Sends a HTML 500 error otherwise
* When Query Parameters are not valid then HTML 400 error is sent
* When a query is received by middleware at it start and if it not fully loaded then a HTML 425 is sent

Text

Description automatically generated with medium confidence

# Common Errors

Error messages appear on missing mandatory values/files needed to run the Middleware. Middleware stops on these for debugging.

Text

Description automatically generated

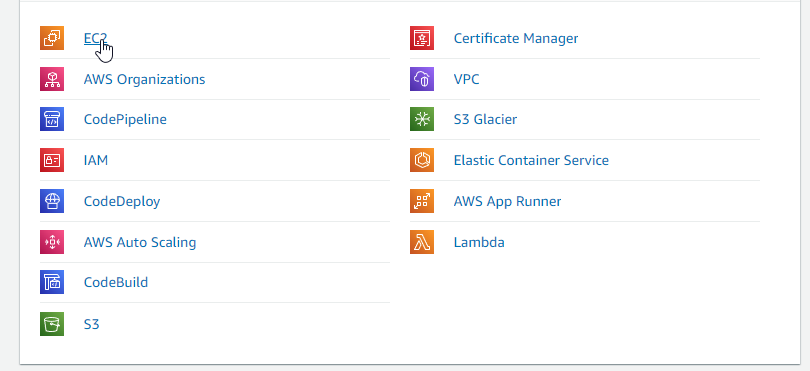
Read the error message to find out more about the error and resolve accordingly.

# AWS deploy example

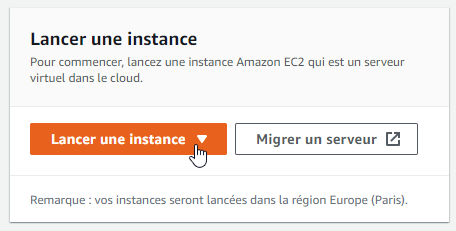
Here an example step by step to deploy middleware on AWS service.

First, log to you AWS account.

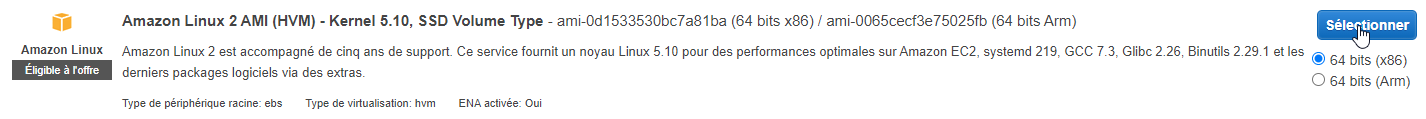
Then go on EC2



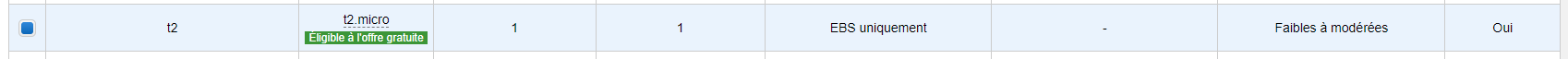
Start an EC2 instance



Choose an OS (Amazon Linux will be chosen for this example)



Choose a structure (t2.micro will be chosen for this example)



Don’t click on “Start” button, choose to configurate instance.

Une image contenant texte

Description générée automatiquement

Create configuration you need (nothing is necessary until 6. Security)

Add a rule, HTTPS for protocole TCP at port 443

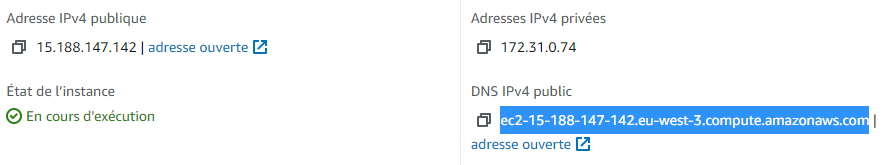
Une image contenant texte

Description générée automatiquement

Then next and start instance. (and key for ssh configuration if you need)

After few minutes instance will start then connect to it.

You can find IP to connect if you click the instance ID.



Connect to it through FilleZilla (<https://filezilla-project.org/>)

Une image contenant texte

Description générée automatiquement

Send middleware source code, WRAPI key, HTTPS credentials.

Une image contenant table

Description générée automatiquement

Connect to it through PowerShell

Une image contenant texte

Description générée automatiquement

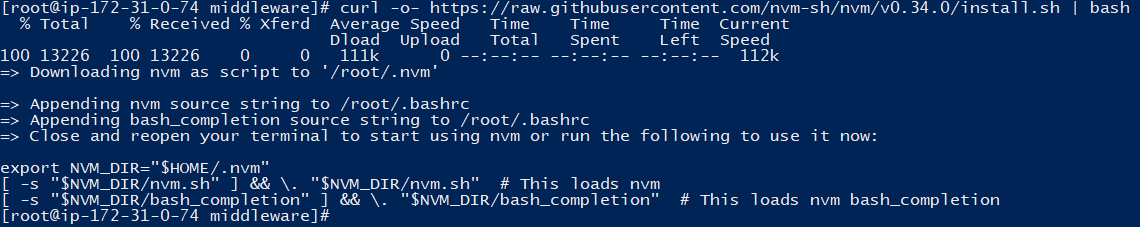
Change to root user



Follow step to install NodeJS on EC2 (<https://docs.aws.amazon.com/fr_fr/sdk-for-javascript/v2/developer-guide/setting-up-node-on-ec2-instance.html>)

With command :

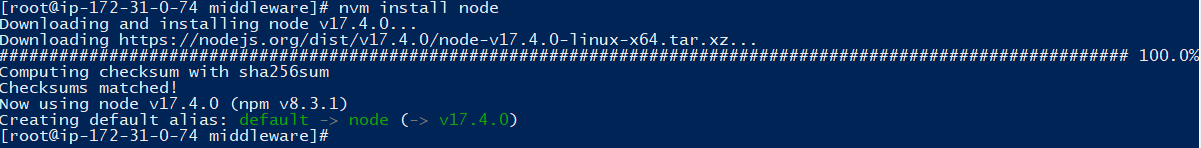
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.34.0/install.sh | bash



. ~/.nvm/nvm.sh



nvm install node



Switch to middleware source directory



Install dependencies

Une image contenant texte

Description générée automatiquement

Now everything is ready to start middleware.  


You can start it in ssh or make it a service (or anything you want)

For example, if you want to build a service then open /etc/system/system and create a service which automatically start middleware at OS start.

Une image contenant texte

Description générée automatiquement

Une image contenant texte

Description générée automatiquement

You should start it with



restart with



Stop with



Get log with

