Ariel Arreola

Universidad Marista de GuaDalajara

Cloud Computing

19/02/2021

Documentación API REST TESTING



Tabla de contenido

[Testing for Api Server at UMG - Prueba de servidor API en UMG 1](#_Toc64616606)

[Made for Cloud Computing class 1](#_Toc64616607)

[Descripción 1](#_Toc64616608)

[Requisitos 2](#_Toc64616609)

[Instalación 2](#_Toc64616610)

[Documentos 2](#_Toc64616611)

[Prueba del código 2](#_Toc64616612)

[Rest 3](#_Toc64616613)

[Método mi primer objeto en Javascript 5](#_Toc64616614)

[Método Detector de sentimientos en texto 6](#_Toc64616615)

[Método Generador de Usuarios Aleatorios 9](#_Toc64616616)

[Método Acortador de URL 14](#_Toc64616617)

[Método Análisis de recetas 15](#_Toc64616618)

[Método Días Festivos 20](#_Toc64616619)

[Método Obtener Clima 24](#_Toc64616620)

[Método Obtener info música 27](#_Toc64616621)

[Método Obtener info Studio Ghibli 29](#_Toc64616622)

[Método Edad de nombres 32](#_Toc64616623)

# Testing for Api Server at UMG - Prueba de servidor API en UMG

## Made for Cloud Computing class

## Descripción

La presente Api está orientada a ser un intermediario entre páginas de Apis existentes y nuestro equipo. La ruta base para el acceso al presente servidor es mediante [localhost:8080/](https://localhost:3000/). En el presente proyecto definimos 5 accesos a métodos GET y 5 de acceso a POST.

## Requisitos

* Nodejs
* Postman
* Navegador Chrome (opcional)

## Instalación

En nodejs se requieren las siguientes dependencias:

* Axios 0.21.1
* body-parser 1.19.0
* express 4.17.1
* morgan 1.10.0
* node-fetch 2.6.1
* nodemon 2.0.7
* request 2.88.2

Al montar el presente servidor en Nodejs instalando sus debidas dependencias con **npm install** y mientras se esté ejecutando el server.js con el comando **"npm run dev"**, podrá acceder mediante el navegador (en el caso de las acciones GET) o en Postman (en GET y POST).

npm install express nodemon request morgan body-parser node-fetch axios

npm run dev

## Documentos

* **Server.js**: aqui se define cada método empleado en lenguaje Javascript, es el código fuente del proyecto
* **package.json**: lista de versión, dependencias, nombre, etcétera.
* **package-lock.json**: archivo requerido para la ejecución del package y dependencias, no modificar

## Prueba del código

Para acceder a cada uno de los métodos del servidor de prueba, seguido del **localhost:8080** se agrega cada uno de los puntos de entrada desarrollados en el proyecto, cada uno de ellos se describirá a continuación. Los resultados se configuraron para que aparezcan en la terminal de Postman, así como respondería un endpoint si usted lo envía por separado.

## Rest

URL Base: [http://localhost:8080/](https://localhost:3000/) Este sitio presenta una página de formato Html con la descripción básica de cada uno de los puntos de entrada de la presente aplicación. A continuación se describen los métodos GET y POST que soporta la aplicación y su uso. Cada método devuelve información en formato json, se recomienda tener en el navegador un plugin para una mejor lectura del texto. Para los métodos POST se requieren objetos tipo JSON de entrada, los cuales se pueden aplicar a Postman en la sección de Body.

Métodos GET

* **Días Festivos**: localhost:8080/diasfestivos/{año}/{codpais} Brinda una lista de las festividades más populares en un país y en que fecha se celebra dado un año
* **Obtener Clima**: localhost:8080/clima/{ciudad}/ Brinda toda la información sobre el clima actual de la ciudad indicada
* **Obtener info música**: localhost:8080/entretenimiento/musica/{banda o cantante}--si aplica, sustituir espacios en blanco por "%20". Despliega todos los objetos e información almacenada relacionada con la banda o el cantante
* **Obtener info Studio Ghibli**: localhost:8080/entretenimiento/studioghiblipeliculas/ Obtiene información en orden cronológico acerca de las peliculas de Studio Ghibli
* **Edad de nombres**: localhost:8080/entretenimiento/edades/{nombre} Dado un nombre, la página devuelve el índice de frecuencia del mismo y una posible edad asociada al nombre.

Los parámetros entre llaves indican que en este espacio pondremos un dato del tipo especificado por ellos. Métodos POST

* [Mi primer objeto de Javascript](https://localhost:3000/crearObjeto) Crea un objeto JSON con base en parámetros de entrada.
* [Detector de sentimientos en texto (en inglés)](https://localhost:3000/appfeelings) Recibe como entrada un texto en inglés y de salida interpreta si las frases dichas reflejan una emoción positiva o negativa (o neutra).
* [Análisis de recetas](https://localhost:3000/analisisreceta) Recibe como entrada una receta con parámetros puntuales y de salida despliega un análisis completo sobre atributos posibles de la receta, ingredientes, imágenes, alérgenos, etcétera.
* [Generador de usuarios aleatorios](https://localhost:3000/personarandom) Recibe un objeto con parámetros para generar usuarios ficticios de manera aleatoria. Útil para crear bases de datos de muestra o práctica.
* [Acortador de URL](https://localhost:3000/acortarURL) Dada una url cualquiera, devuelve una url corta originaria del propio endpoint.

NOTA: las URL se están marcando, sin embargo estas funcionan al ejecutarlas en postman con sus debidos objetos JSON de entrada.

POST

Para todos los métodos colocar en “Headers” lo siguiente: Key=Content-Type, Value=application/json. Postman facilita esta tarea, además está junto al lado de “Body” que es donde se puede agregar en crudo el objeto JSON a ingresar como entrada al método

NOTA: las fotoevidencias aparecen como puerto 3000 ya que inicialmente para pruebas se había configurado ese puerto, sin embargo, para acceder en la versión actual ya probada, es mediante localhost:8080 o bien desde <http://apipruebaariel.herokuapp.com/>.

# Método mi primer objeto en Javascript

**Descripción**: este método sirve para crear un objeto JSON a un repositorio existente, devolviendo el objeto creado junto con un id. Es importante tener nociones sobre la creación de objetos, por lo u e esta página, además de su documentación, es muy útil para saber cómo funcionan los métodos principales GET, POST e incluso otros, como PUT o PATCH.

/crearObjeto

**Documentacion:** https://jsonplaceholder.typicode.com/guide/

**Endpoints**: https://jsonplaceholder.typicode.com/posts

**Parámetros**

|  |  |  |  |
| --- | --- | --- | --- |
| Parámetro | Param. De entrada o salida? | Tipo | Descripción |
| id | Salida | numeric | Identificador de objeto basado en una lista de objetos predefinidos (generalmente es 101) |
| title | Ambos | string | Título del objeto a crear |
| body | Ambos | string | Información o texto para agregar al objeto |
| userId | Ambos | numeric | Un id de usuario |

**Ejemplo**

{

      "userId": 1,

      "title": "Prueba1",

      "body": "Hola mundo desde JSON, en espera de un id... :) "

    }

**Respuestas**

{

    "userId": "1",

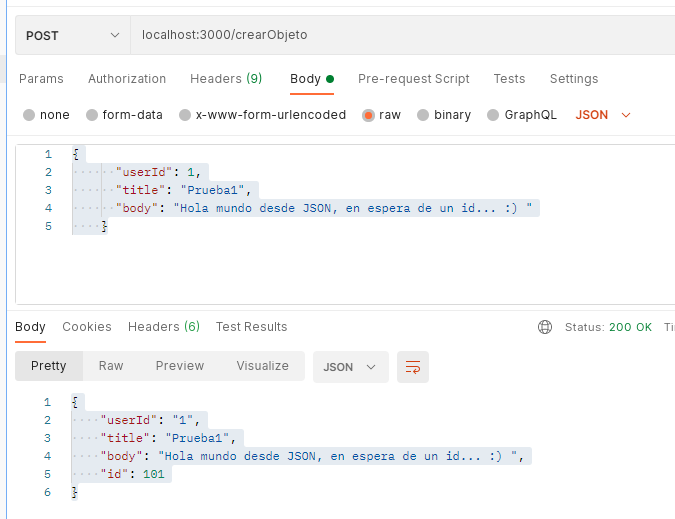
    "title": "Prueba1",

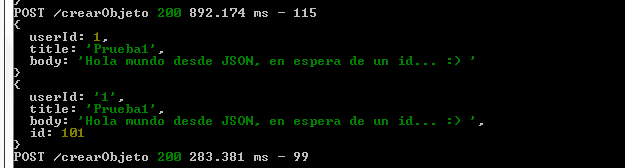
    "body": "Hola mundo desde JSON, en espera de un id... :) ",

    "id": 101

}

**Fotoevidencia**





# Método Detector de sentimientos en texto

/appfeelings

**Descripción:** Este método sirve para escribir como entrada un párrafo o texto cualquiera en inglés, y como respuesta por cada frase se interpretará si es un sentimiento positivo o negativo, asignando un puntaje de ‘polarity’ positivo o negativo, respectivamente. Es interesante este algoritmo ya que, bajo algunos márgenes de errores, interpreta oraciones completas de acuerdo con los adjetivos utilizados o incluso frases o expresiones en el inglés, lo cual puede extrapolarse a una aplicación de inteligencia artificial, no solo para el inglés, sino también en español, lo cual me hubiera gustado mucho que tuviera ese soporte.

**Endpoints:** https://sentim-api.herokuapp.com/api/v1/

**Documentación**: https://sentim-api.herokuapp.com/

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| Parámetro | Param. ¿De entrada, o salida? | Tipo | Descripción |
| “text” | Entrada | string | El texto para analizar, el endpoint separa cada frase al detectar un punto “.” |
| “result” | Salida | object | Resultado general de todo el texto con base en cada frase |
| “result.polarity” | Salida | numeric | Resultado numerico general del texto, si es menor a cero, es negativo, en otro caso es positivo o neutral |
| “result.type” | Salida | string | Resultado en letras: positive/negative/neutral |
| “sentences” | Salida | Array<object> | Conjunto de frases del texto y análisis de cada una |
| “sentences.sentence” | Salida | string | Fragmento del texto de entrada delimitado por un punto. |
| “sentences.sentiment” | Salida | object | Resultado de análisis en número y letras |
| “sentences.sentiment.polarity” | Salida |  | Resultado de análisis de frase en número |
| “sentences.sentiment.type” | Salida |  | Resultado de análisis de frase en letras |

**Ejemplo**

{

      "text":"I am sad. I don't like my carreer because it makes me difficult. But i could be happy when finished university while I do my better in this course"

    }

**Respuestas**

{

    "result": {

        "polarity": -0.12,

        "type": "negative"

    },

    "sentences": [

        {

            "sentence": "I am sad.",

            "sentiment": {

                "polarity": -0.5,

                "type": "negative"

            }

        },

        {

            "sentence": "I don't like my carreer because it makes me difficult.",

            "sentiment": {

                "polarity": -0.5,

                "type": "negative"

            }

        },

        {

            "sentence": "But i could be happy when finished university while I do my better in this course",

            "sentiment": {

                "polarity": 0.65,

                "type": "positive"

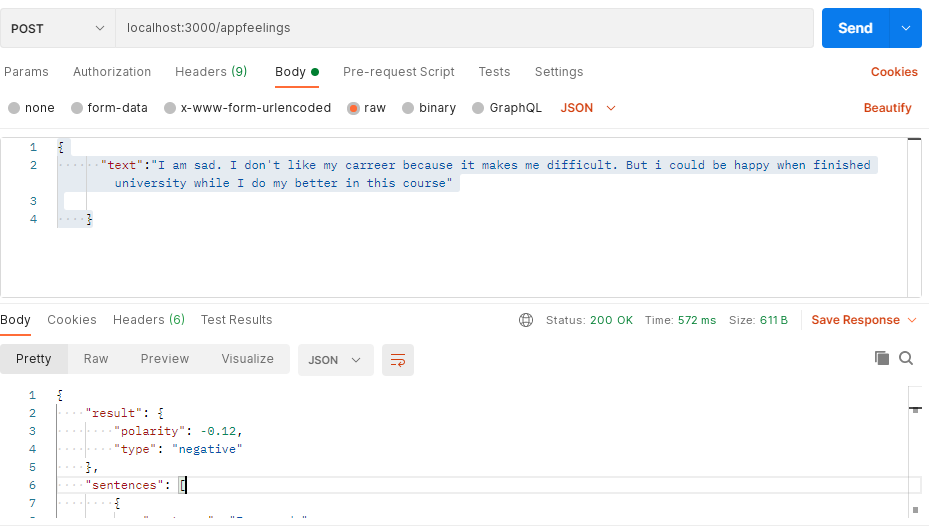
            }

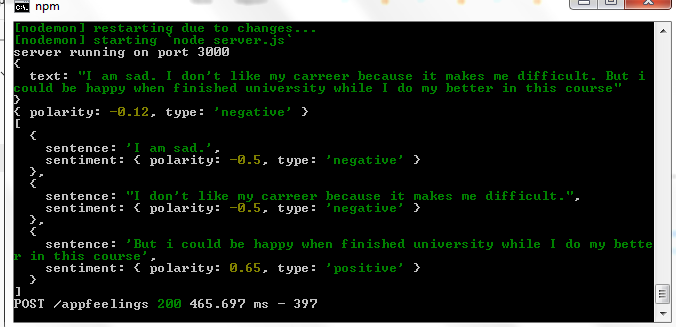
        }

    ]

}

**Fotoevidencia**





# Método Generador de Usuarios Aleatorios

**/personarandom**

**Descripción**: genera un usuario con datos ficticios de manera aleatoria. El usuario puede brindar una serie de parámetros específicos y el endpoint devolverá el usuario o usuarios creados. Útil para creación de muestras o ejercicios de bases de datos, ya que cuando se necesita y hay poco tiempo, a veces la imaginación suele ser limitada. Sería interesante utilizar una de estas APIS directamente montado sobre un query o consulta de bases de datos como SQL Server. Al ser de mucha utilidad brindarle un objeto de datos, convertí una configuración GET en una de POST, ya que nosotros solemos buscar determinados perfiles de personas o incluso más de una.

**Endpoints:**

URL base: https://randomuser.me/api

URL Extendida: https://randomuser.me/api/?password={password}&results={ results}&gender={ gender}&nat={nat}

Los datos entre llaves los rellena el método de acuerdo con la información del cuerpo del objeto de entrada.

**Documentación:** https://randomuser.me/documentation

**Parámetros**

Debido al número de parámetros obtenidos en la salida, se agregó una tabla de “contenido” que por cada objeto se describe nombre y tipo de parámetro, cuando aplique. Se agrega una categoría de ejemplos, mismos que pueden consultarse en la página indicada en “Documentación”. Se mencionarán todos, incluyendo las salidas, para futuras referencias.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parámetro | Param. De entrada o salida? | Tipo | Contenido | Descripcion | Ejemplo |
| “password” | Ambos | String | No aplica | Indica el formato de contraseña para nuestro(s) usuario(s).  Esta misma configuración se repite para todos los resultados de la consulta.  Puede ser vacío | “number”, “lower”,”upper”,  ”special”,”20”. Puede combinarse más de uno de los anteriores mediante comas |
| “results” | Entrada | String | No aplica | El número de usuarios aleatorios a crear. | Un número entero |
| “gender” | Ambos | String | No aplica | Género del usuario.  Esta misma configuración se repite para todos los resultados de la consulta.  Puede ser vacío para crear un grupo mixto de usuarios. | “male”, “female”, “ “. |
| “nat” | Ambos | string | No aplica | Código de nacionalidad del usuario.  Esta misma configuración se repite para todos los resultados de la consulta.  Puede ser vacío. | AU, BR, CA, CH, DE, DK, ES, FI, FR, GB, IE, IR, NO, NL, NZ, TR, US |
| “name” | Salida | Object | Title(string)  First(string)  Last(string) | El nombre completo del usuario, muestra un nombre, un apellido y un título o trato. | Title: “mr”  First:” Lauro”  Last: “Smith” |
| “location” | Salida | Object | Street(object)  City(string)  State(string)  Country(string)  Postcode(number)  Coordinates(object)  Timezone(object) | Todos los datos del usuario acerca de su vivienda. | "location": {  "street": {  "number": 1639,  "name": "Walnut Hill Ln"  },  "city": "Richardson",  "state": "Pennsylvania",  "country": "United States",  "postcode": 68013,  "coordinates": { "latitude": "-4.6317", "longitude": "34.2814"  }, |
| “email” | Salida | string | No aplica | Correo electrónico del usuario creado | "email":"landon.beck@example.com" |
| “login” | Salida | object | Uuid(string)  Username(string)  Password(string)  Salt(string)  Md5(string)  Sha(string)  sha256(string) | Datos de inicio de sesión del usuario | "login": {  "uuid": "052ebba5-1b75-4421-b5de-916c3fcf2ff4",  "username": "angrygorilla826",  "password": "3895147297919930322829471476816807005878848275935132865921812046",  "salt": "ugTBurlf",  "md5": "41265865df3b01db60f1aff4aaf7da0a",  "sha1": "0bf67137b751dbdc58017a655ae48815100085aa",  "sha256": "32220110cfb924827281179710bfa809a5a6c5d9f83e012cee35f5aa7bdfa378"  } |
| “dob” | Salida | object | Date(string)  Age(numeric) | Fecha de nacimiento y edad del usuario | "dob": {  "date": "1973-12-24T10:02:18.868Z",  "age": 48  } |
| “registered” | Salida | object | Date(string)  Age(numeric) | Fecha y años de registro | "registered": {  "date": "2008-01-18T22:07:55.497Z",  "age": 13  }, |
| “phone”/”cell” | salida | string | No aplica | Números telefónicos del usuario con lada de 3 números sumando 10 digitos separados por guiones | "phone": "(571)-550-2012",  "cell": "(871)-671-3035", |
| “id” | salida | object | Name(string)  Value(string) | Datos de identificación del usuario | "id": {  "name": "SSN",  "value": "295-99-8479"  }, |
| “picture” | salida | object | Large(string)  Médium(string)  Thumbnail(string) | URL de las imágenes del usuario en 3 distintos tamaños | "picture": {  "large": "https://randomuser.me/api/portraits/men/16.jpg",  "medium": "https://randomuser.me/api/portraits/med/men/16.jpg",  "thumbnail": "https://randomuser.me/api/portraits/thumb/men/16.jpg"  }, |
| “nat” | salida | string | No aplica | Código de nacionalidad del usuario | "nat": "US" |

**Ejemplo**

{

      "password": "number",

      "results": "1",

      "gender": "male",

      "nat": "CH"

    }

**Respuestas**

[

    {

        "gender": "male",

        "name": {

            "title": "Monsieur",

            "first": "Noël",

            "last": "Jean"

        },

        "location": {

            "street": {

                "number": 4523,

                "name": "Rue de L'Abbé-Soulange-Bodin"

            },

            "city": "Beggingen",

            "state": "Appenzell Ausserrhoden",

            "country": "Switzerland",

            "postcode": 8887,

            "coordinates": {

                "latitude": "58.6797",

                "longitude": "86.2930"

            },

            "timezone": {

                "offset": "+9:30",

                "description": "Adelaide, Darwin"

            }

        },

        "email": "noel.jean@example.com",

        "login": {

            "uuid": "7e7910f5-3291-43d9-83f9-e8a954f58d5b",

            "username": "blackrabbit411",

            "password": "2959565329358370948001879",

            "salt": "xnjJdnYM",

            "md5": "1c14468af4cf101a5e03beca2d262d6a",

            "sha1": "fbeb5eb77da9f1037851d0d83193344b783ab5b2",

            "sha256": "ba6127ae6ca325f453d6a2cef2ef694dadb4ffd94d3074b44fda3cea19d9f1a4"

        },

        "dob": {

            "date": "1974-08-18T20:32:07.731Z",

            "age": 47

        },

        "registered": {

            "date": "2006-01-31T21:05:22.886Z",

            "age": 15

        },

        "phone": "077 780 00 07",

        "cell": "075 135 21 80",

        "id": {

            "name": "AVS",

            "value": "756.9453.1191.00"

        },

        "picture": {

            "large": "https://randomuser.me/api/portraits/men/62.jpg",

            "medium": "https://randomuser.me/api/portraits/med/men/62.jpg",

            "thumbnail": "https://randomuser.me/api/portraits/thumb/men/62.jpg"

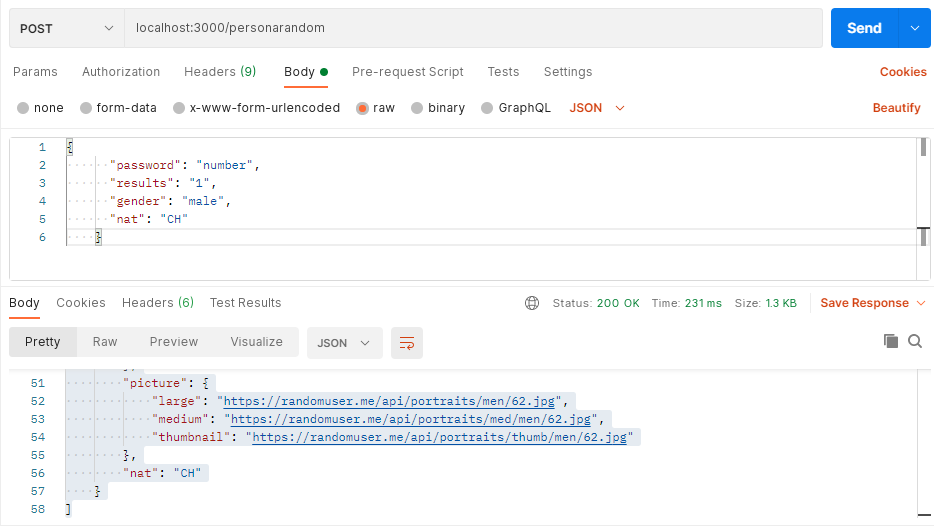
        },

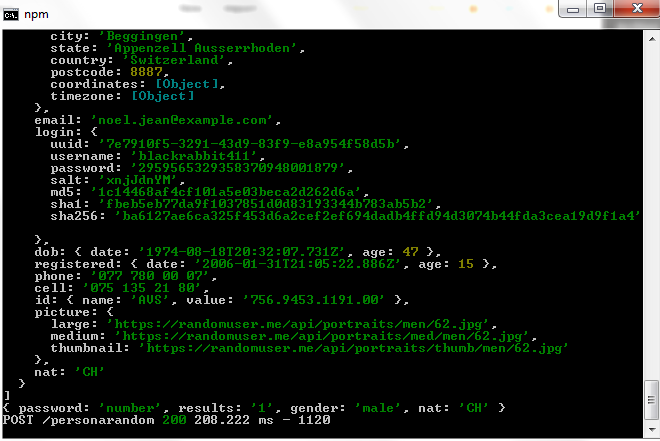
        "nat": "CH"

    }

]

**Fotoevidencia**





# Método Acortador de URL

/acortarURL

**Descripción**: este método recibe como entrada una url cualquiera, y el endpoint despliega una url acortada, generada por el propio sitio del endpoint. Útil para reducir tamaño de links y tener una presentación más elegante al adjuntarlos, así también de transcribirlos si consisten en textos planos.

**Endpoints:** https://cleanuri.com/api/v1/shorten

**Documentación:** https://cleanuri.com/docs

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| Parámetro | Param. De entrada o salida? | Tipo | Descripción |
| “url” | Entrada | String | Url cualquiera de internet |
| “result\_url” | Salida | string | Url acortada generada por el endpoint |

**Ejemplo**

{

"url":"https://mappinggis.com/2019/10/como-crear-mapas-con-google-my-maps/"

}

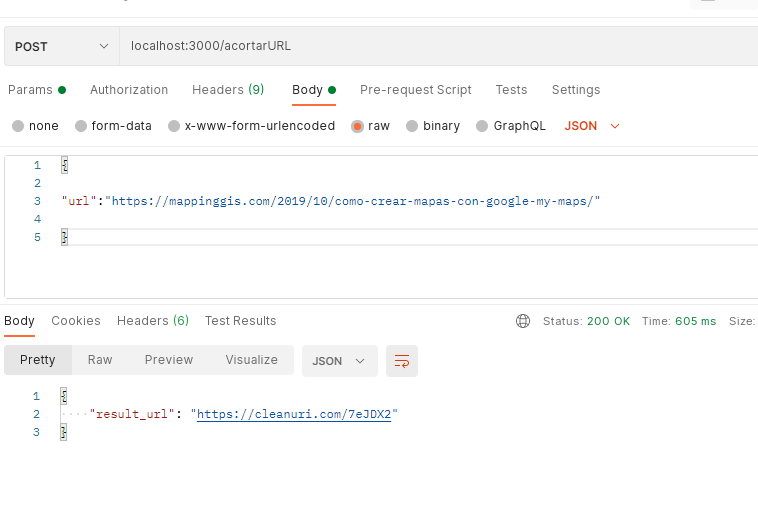
**Respuestas**

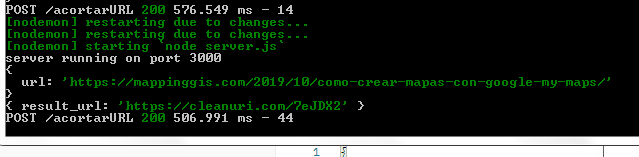
{

    "result\_url": "https://cleanuri.com/7eJDX2"

}

**Fotoevidencia**





# Método Análisis de recetas

/analisisreceta

**Descripción:** este método recibe como parámetro de entrada una receta existente o conocida, y el endpoint desglosa toda la información que almacena acerca del platillo y sus características. Es útil para las personas que poseen un recetario en línea o bien para programadores que estén desarrollando puntos de ventas extendidos para restaurantes.

**Endpoints:** https://api.spoonacular.com/recipes/analyze

**Documentación:** https://spoonacular.com/food-api/docs#Analyze-Recipe

**Parámetros de entrada**

|  |  |  |  |
| --- | --- | --- | --- |
| Parámetro | Tipo | Descripción | Ejemplo |
| “title” | String | Nombre del platillo | "Spaghetti Carbonara" |
| “servings” | Number | Raciones de la receta elegida | 2 |
| “ingredients” | Array<string> | Cantidades y nombre de ingredientes | "ingredients": [          "1 lb spaghetti",          "3.5 oz pancetta",          "2 Tbsps olive oil",          "1  egg",          "0.5 cup parmesan cheese"      ] |
| “instructions” | string | Instrucciones a seguir para cocinar el platillo | "instructions": "Bring a large pot of water to a boil and season generously with salt. Add the pasta to the water once boiling and cook until al dente. Reserve 2 cups of cooking water and drain the pasta. " |

Key: 272e1846d98d438296854d06a97c977d

Ejemplo

{

    "title": "Spaghetti Carbonara",

    "servings": 2,

    "ingredients": [

        "1 lb spaghetti",

        "3.5 oz pancetta",

        "2 Tbsps olive oil",

        "1  egg",

        "0.5 cup parmesan cheese"

    ],

    "instructions": "Bring a large pot of water to a boil and season generously with salt. Add the pasta to the water once boiling and cook until al dente. Reserve 2 cups of cooking water and drain the pasta. "

}

Respuestas

{

    "vegetarian": **false**,

    "vegan": **false**,

    "glutenFree": **false**,

    "dairyFree": **false**,

    "veryHealthy": **false**,

    "cheap": **false**,

    "veryPopular": **false**,

    "sustainable": **false**,

    "weightWatcherSmartPoints": 0,

    "gaps": "no",

    "lowFodmap": **false**,

    "aggregateLikes": 0,

    "spoonacularScore": 0,

    "healthScore": 0,

    "pricePerServing": 0,

    "extendedIngredients": [],

    "id": -1,

    "title": "Spaghetti Carbonara",

    "servings": 2,

    "summary": **null**,

    "cuisines": [],

    "dishTypes": [],

    "diets": [

        "paleolithic",

        "primal",

        "whole 30"

    ],

    "occasions": [],

    "instructions": "Bring a large pot of water to a boil and season generously with salt. Add the pasta to the water once boiling and cook until al dente. Reserve 2 cups of cooking water and drain the pasta.",

    "analyzedInstructions": [

        {

            "name": "",

            "steps": [

                {

                    "number": 1,

                    "step": "Bring a large pot of water to a boil and season generously with salt.",

                    "ingredients": [

                        {

                            "id": 14412,

                            "name": "water",

                            "localizedName": "water",

                            "image": "water.png"

                        },

                        {

                            "id": 2047,

                            "name": "salt",

                            "localizedName": "salt",

                            "image": "salt.jpg"

                        }

                    ],

                    "equipment": [

                        {

                            "id": 404752,

                            "name": "pot",

                            "localizedName": "pot",

                            "image": "stock-pot.jpg"

                        }

                    ]

                },

                {

                    "number": 2,

                    "step": "Add the pasta to the water once boiling and cook until al dente. Reserve 2 cups of cooking water and drain the pasta.",

                    "ingredients": [

                        {

                            "id": 20420,

                            "name": "pasta",

                            "localizedName": "pasta",

                            "image": "fusilli.jpg"

                        },

                        {

                            "id": 14412,

                            "name": "water",

                            "localizedName": "water",

                            "image": "water.png"

                        }

                    ],

                    "equipment": []

                }

            ]

        }

    ],

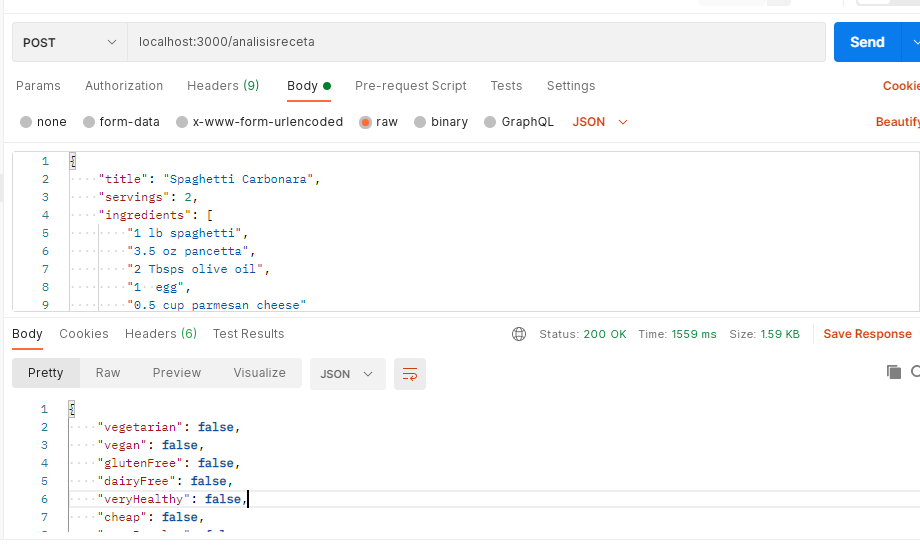
    "sourceName": **null**,

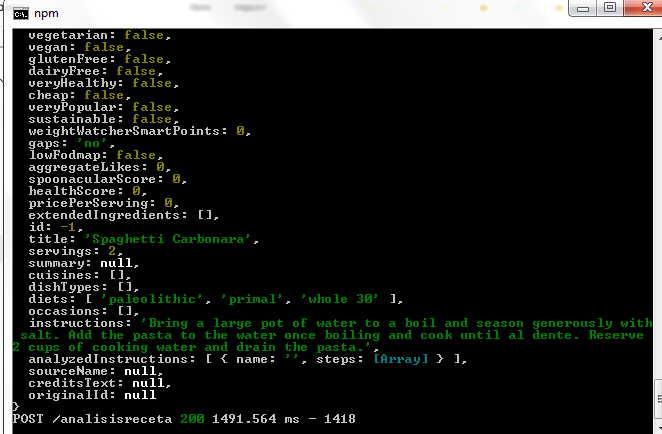
    "creditsText": **null**,

    "originalId": **null**

}

**Fotoevidencia**





GET

Para todos los métodos colocar en “Headers” lo siguiente: Key=Content-Type, Value=application/json. En caso de visualizarse en un navegador, se recomienda usar un plug in o programa para navegadores que permita visualizar de una manera limpia archivos json, por lo que ,para conseguir esta visualización, se recomienda utilizarlos también en Postman.

Los ejemplos de uso de los métodos get incluirán el uso de una URL de ejemplo, ya que no requieren de objetos de entrada, sino que solo son consultas que pueden tener parámetros o no.

# Método Días Festivos

/diasfestivos/{año}/{codpais}

**Descripción:** este método llama al endpoint y despliega las festividades por día más conocidas del país en cuestión. Puede ser útil para la creación de aplicaciones de calendarios o que usen los mismos, así también sirve como cultural para conocer las fiestas o conmemoraciones populares de otros países.

**Endpoints:** https://date.nager.at/api/v2/PublicHolidays/{año}/{codigopais}

Los datos que están en llaves los elige el usuario y los valida siempre y cuando existan

**Documentación**: https://date.nager.at/api

https://date.nager.at/swagger/index.html

**Parámetros**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parámetro | Param. De entrada o salida? | Tipo | Descripción | Ejemplo |
| {año} | Entrada | Param URL | Año de consulta (en varios países las celebraciones cambian a lo largo de los años) | 2012 |
| {codigopais} | Entrada | Param URL | Dos letras que simbolizan un país o región | UK |
| “date” | Salida | string | La fecha de celebración | 2021-11-25 |
| “localName” | Salida | string | Nombre en idioma original del país | “Thanksgiving Day” |
| “name” | Salida | string | Nombre traducido al inglés o reconocido de forma internacional | “Thanksgiving Day” |
| “countryCode” | Salida | string | Formato ISO 3166-1 alpha-2 | UK |
| “fixed” | Salida | Boolean | Si la festividad es siempre en la misma fecha cada año | True/false |
| “global” | Salida | Boolean | Si la festividad se celebra en todo el país | True/false |
| “counties” | Salida | array | Regiones en que se celebra | “null”,”US-NE” |
| “launchYear” | Salida | numeric | Año en que se ofició la fecha como festividad | “Null”, 1863 |
| “type” | Salida | string | Tipo de festividad | “Public” |

**Ejemplo**

localhost:8080/diasfestivos/2021/MX

**Respuestas**

[

    {

        "date": "2021-01-01",

        "localName": "Año Nuevo",

        "name": "New Year's Day",

        "countryCode": "MX",

        "fixed": **false**,

        "global": **true**,

        "counties": **null**,

        "launchYear": **null**,

        "type": "Public"

    },

    {

        "date": "2021-02-01",

        "localName": "Día de la Constitución",

        "name": "Constitution Day",

        "countryCode": "MX",

        "fixed": **false**,

        "global": **true**,

        "counties": **null**,

        "launchYear": **null**,

        "type": "Public"

    },

    {

        "date": "2021-03-15",

        "localName": "Natalicio de Benito Juárez",

        "name": "Benito Juárez's birthday",

        "countryCode": "MX",

        "fixed": **false**,

        "global": **true**,

        "counties": **null**,

        "launchYear": **null**,

        "type": "Public"

    },

    {

        "date": "2021-04-30",

        "localName": "Día del Trabajo",

        "name": "Labor Day",

        "countryCode": "MX",

        "fixed": **false**,

        "global": **true**,

        "counties": **null**,

        "launchYear": **null**,

        "type": "Public"

    },

    {

        "date": "2021-09-16",

        "localName": "Día de la Independencia",

        "name": "Independence Day",

        "countryCode": "MX",

        "fixed": **false**,

        "global": **true**,

        "counties": **null**,

        "launchYear": **null**,

        "type": "Public"

    },

    {

        "date": "2021-11-15",

        "localName": "Día de la Revolución",

        "name": "Revolution Day",

        "countryCode": "MX",

        "fixed": **false**,

        "global": **true**,

        "counties": **null**,

        "launchYear": **null**,

        "type": "Public"

    },

    {

        "date": "2021-12-25",

        "localName": "Navidad",

        "name": "Christmas Day",

        "countryCode": "MX",

        "fixed": **true**,

        "global": **true**,

        "counties": **null**,

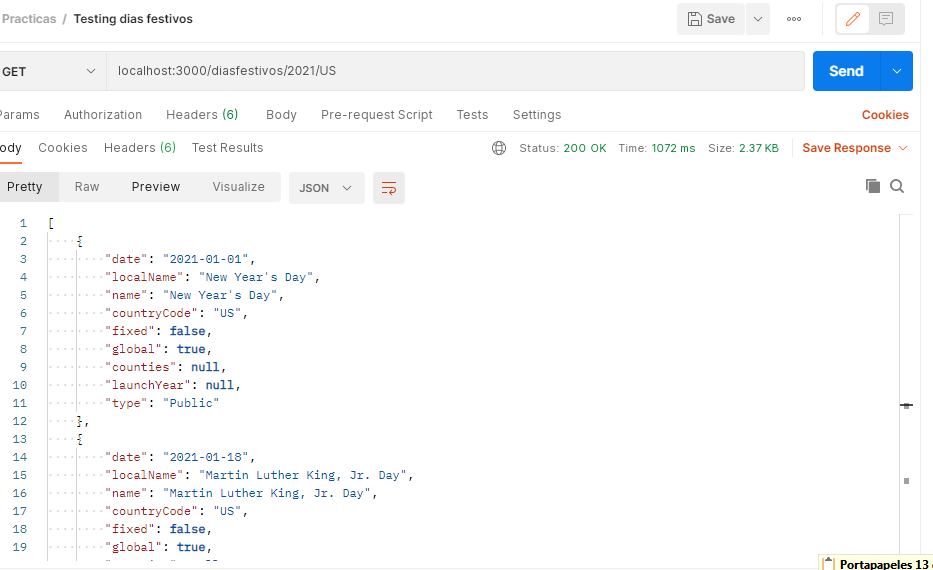
        "launchYear": **null**,

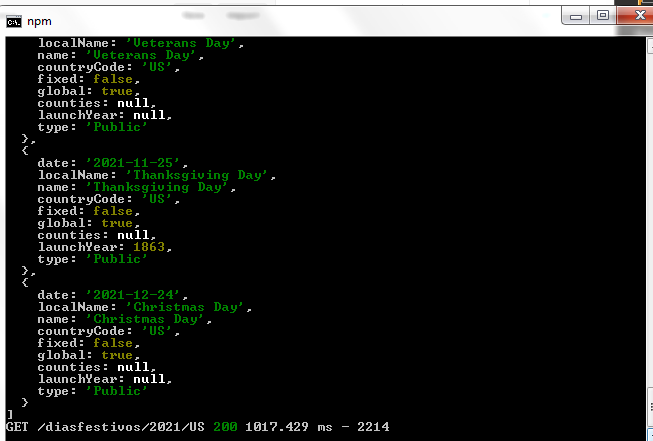
        "type": "Public"

    }

]

**Fotoevidencia**





# Método Obtener Clima

**/clima/{ciudad}**

**Descripción:** obtiene datos acerca del clima en general de una ciudad especificada, aplicable para cualquier país. Es útil para asistentes o widgets que se utilicen en equipos inteligentes como bocinas, teléfonos, tabletas u otros dispositivos.

**Endpoints:** http://api.openweathermap.org/data/2.5/weather?q={ ciudad}&units=metric&appid={key}&lang=es`

Key: f6902099c5861eb71633b6ef9828c452

**Documentación:** https://openweathermap.org/current

Parámetros:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parámetro | Param. De entrada o salida? | Tipo | Contenido | Descripción | Ejemplo |
| {ciudad} | Entrada | Param URL | No aplica | Nombre de la ciudad (de cualquier país) para obtener información del clima, usa estándar de nombres en inglés. | Seattle |
| “coord.” | Salida | Object | Lon(numeric)  Lat(numeric) | Muestra las coordenadas de latitud y longitud de la referencia de la ciudad | "coord": {  "lon": -122.3321,  "lat": 47.6062  } |
| “wheather” | Salida | Array | Id(numeric)  Main(string)  Description(string)  Icon(string) | Muestra datos principales de clima | "weather": [  {  "id": 800,  "main": "Clear",  "description": "cielo claro",  "icon": "01n"  }  ], |
| “base” | Salida | string | No aplica | Punto de referencia (escrito con letra) | “stations” |
| “main” | Salida | object | Temp(numeric)  Feels\_like(numeric)  Temp\_min(numeric)  Temp\_max(numeric)  Pressure(numeric)  Humidity(numeric) | Datos de temperatura, humedad, presión, sensación térmica | "main": {  "temp": 2.54, "feels\_like": -0.43,  "temp\_min": 1.11, "temp\_max": 4, "pressure": 1026, "humidity": 87  }, |
| “visibility” | Salida | numeric | No aplica | Índice de visibilidad | 10000 |
| “wind” | Salida | object | Speed(numeric)  Deg(numeric) | Índices de viento | "wind": {  "speed":1.54, "deg":150  } |
| “clouds” | Salida | object | Flag “all” | Índice de nubes | All:1 |
| “dt” | Salida | numeric | No aplica | Fecha de toma de estadísticas (UNIX) | 1613639402 |
| “sys” | Salida | object | Type(numeric)  Id(numeric)  Country(string)  Sunrise(numeric)  Sunset(numeric) | Registro de país, hora de amanecer (formato UNIX) | "sys": {  "type": 1,  "id": 3417, "country": "US",  "sunrise": 1613660967,  "sunset": 1613698638  } |
| “timezone” | Salida | numeric | No aplica | Zona horaria de referencia | -28800 |
| “id” | Salida | numeric | No aplica | Identificador de solicitud | 5809844 |
| “name” | Salida | string | No aplica | Nombre de la ciudad solicitada | Seattle |
| “cod” | Salida | Numeric | No aplica | Código de ciudad | 200 |

**Ejemplo**

localhost:8080/clima/Seattle

**Respuestas**

{

    "coord": {

        "lon": -122.3321,

        "lat": 47.6062

    },

    "weather": [

        {

            "id": 800,

            "main": "Clear",

            "description": "cielo claro",

            "icon": "01n"

        }

    ],

    "base": "stations",

    "main": {

        "temp": 2.54,

        "feels\_like": -0.43,

        "temp\_min": 1.11,

        "temp\_max": 4,

        "pressure": 1026,

        "humidity": 87

    },

    "visibility": 10000,

    "wind": {

        "speed": 1.54,

        "deg": 150

    },

    "clouds": {

        "all": 1

    },

    "dt": 1613639402,

    "sys": {

        "type": 1,

        "id": 3417,

        "country": "US",

        "sunrise": 1613660967,

        "sunset": 1613698638

    },

    "timezone": -28800,

    "id": 5809844,

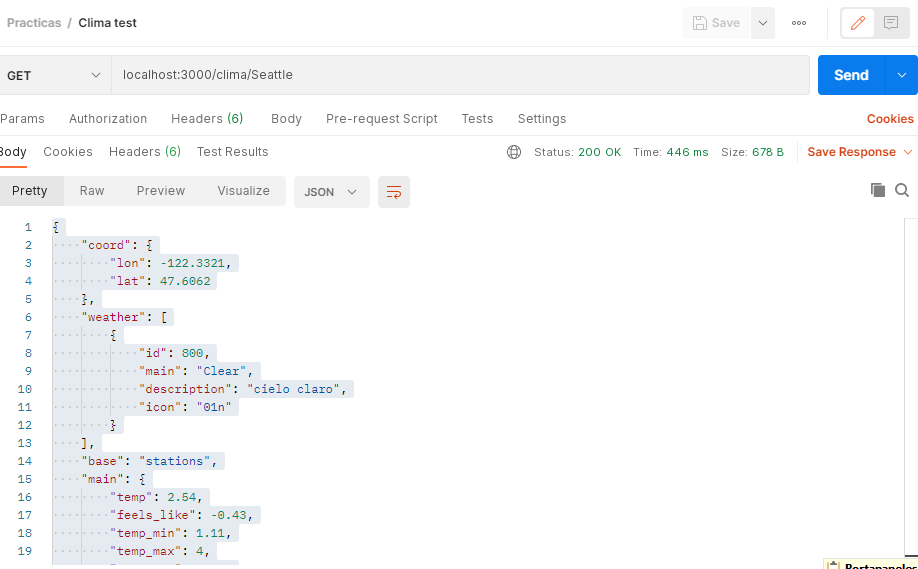
    "name": "Seattle",

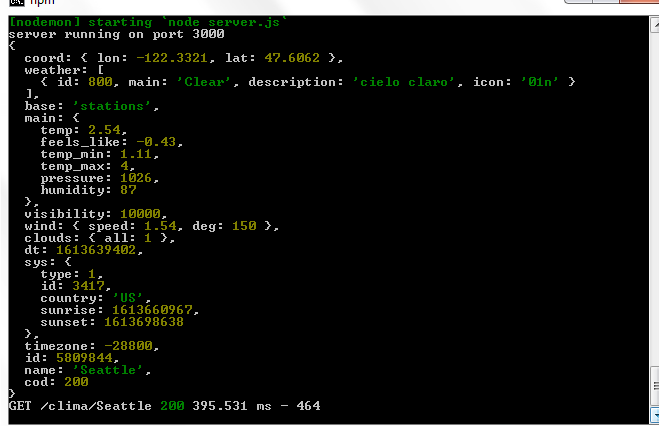
    "cod": 200

}

Los datos en fecha y hora están codificados en formato UNIX, se recomienda usar un convertidor para interpretarlos, por ejemplo: **https://www.cdmon.com/es/conversor-timestamp**

**Fotoevidencia**





# Método Obtener info música

/entretenimiento/musica/{banda}

Si la banda o cantante posee espacios, utilizar %20

**Descripción**: este método obtiene información general sobre una banda, grupo musical o cantante famoso que esté registrado en la base de datos del endpoint.

**Endpoints**: https://musicbrainz.org/ws/2/artist/?query={banda o cantante}

**Documentación**: https://musicbrainz.org/doc/Development

**Parametros:**

|  |  |  |  |
| --- | --- | --- | --- |
| Parametro | Param. De entrada o salida? | Tipo | Descripción |
| {banda} | Entrada | Param URL | El nombre de un cantante o banda que se desee buscar, por cada uno de los espacios sustituir carácter espaciador por %20, por ejemplo: Spandau%20Ballet |

**Descripción de salida:** se reciben resultados sobre todas las coincidencias del nombre de banda o cantante, ya sea que contengan el nombre o bien, se relacionen. El endpoint usa una base de datos que trae todas las coincidencias. La información que arroja incluye nombre de banda, país de origen, si ya se retiraron de la música, géneros musicales, miembros, etcétera.

**Ejemplo**

localhost:8080 /entretenimiento/musica/dokken

Respuestas

"created": "2021-02-18T09:54:41.274Z",

    "count": 9,

    "offset": 0,

    "artists": [

        {

            "id": "09ade3bd-b110-4277-b923-7dc32e6a3f92",

            "type": "Group",

            "type-id": "e431f5f6-b5d2-343d-8b36-72607fffb74b",

            "score": 100,

            "name": "Dokken",

            "sort-name": "Dokken",

            "country": "US",

            "area": {

                "id": "489ce91b-6658-3307-9877-795b68554c98",

                "type": "Country",

                "type-id": "06dd0ae4-8c74-30bb-b43d-95dcedf961de",

                "name": "United States",

                "sort-name": "United States",

                "life-span": {

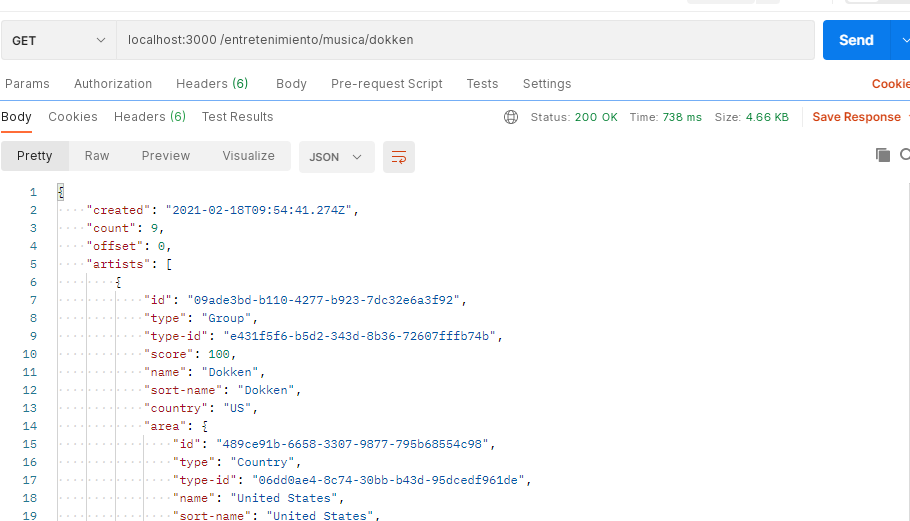
                    "ended": **null**

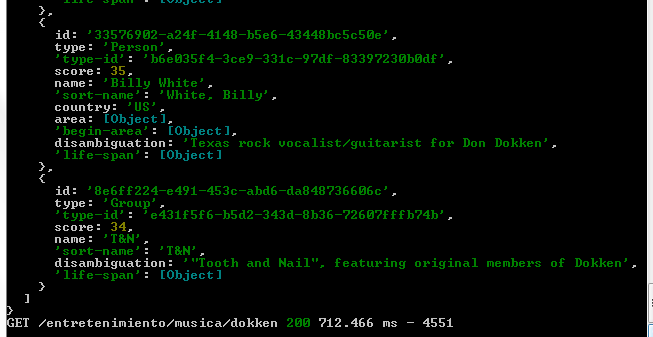
                }

            },

…(continua)

**Fotoevidencia**





# Método Obtener info Studio Ghibli

/entretenimiento/studioghiblipeliculas/{película}

**Descripción:** este método despliega una película específica o bien varias de las películas de Studio Ghibli comunicándose con el endpoint. Puede atender la solicitud sin {película} o en su defecto, atender 3: “Mi\_vecino\_Totoro”, “La\_tuma\_de\_las\_luciernagas” y “Castillo ambulante”.

**Endpoints:**

<https://ghibliapi.herokuapp.com/films/2baf70d1-42bb-4437-b551-e5fed5a87abe> (Castillo ambulante)

<https://ghibliapi.herokuapp.com/films/58611129-2dbc-4a81-a72f-77ddfc1b1b49> (Mi vecino Totoro)

<https://ghibliapi.herokuapp.com/films/12cfb892-aac0-4c5b-94af-521852e46d6a> (La tumba de las luciérnagas)

https://ghibliapi.herokuapp.com/films

**Documentación**: https://ghibliapi.herokuapp.com

**Parámetros:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parámetro | Param. De entrada o salida? | Tipo | Descripción | Opciones preconfiguradas |
| {película} | Entrada | ParamURL | Consulta una película específica de Studio Ghibli | Mi\_vecino\_Totoro  Castillo\_Ambulante  La\_Tumba\_De\_las\_Luciernagas |
| Sin parametro | Entrada | ParamURL | Devuelve todas las películas de Studio Ghibli | No aplica. |

Al ingresar al método, la API Rest soporta dejar sin parámetro “película” al path y devuelve varias de las películas de studio Ghibli.

**Ejemplo**

localhost:8080/entretenimiento/studioghiblipeliculas/Mi\_vecino\_Totoro

**Respuestas**

{

    "id": "58611129-2dbc-4a81-a72f-77ddfc1b1b49",

    "title": "My Neighbor Totoro",

    "description": "Two sisters move to the country with their father in order to be closer to their hospitalized mother, and discover the surrounding trees are inhabited by Totoros, magical spirits of the forest. When the youngest runs away from home, the older sister seeks help from the spirits to find her.",

    "director": "Hayao Miyazaki",

    "producer": "Hayao Miyazaki",

    "release\_date": "1988",

    "rt\_score": "93",

    "people": [

        "https://ghibliapi.herokuapp.com/people/986faac6-67e3-4fb8-a9ee-bad077c2e7fe",

        "https://ghibliapi.herokuapp.com/people/d5df3c04-f355-4038-833c-83bd3502b6b9",

        "https://ghibliapi.herokuapp.com/people/3031caa8-eb1a-41c6-ab93-dd091b541e11",

        "https://ghibliapi.herokuapp.com/people/87b68b97-3774-495b-bf80-495a5f3e672d",

        "https://ghibliapi.herokuapp.com/people/d39deecb-2bd0-4770-8b45-485f26e1381f",

        "https://ghibliapi.herokuapp.com/people/591524bc-04fe-4e60-8d61-2425e42ffb2a",

        "https://ghibliapi.herokuapp.com/people/c491755a-407d-4d6e-b58a-240ec78b5061",

        "https://ghibliapi.herokuapp.com/people/f467e18e-3694-409f-bdb3-be891ade1106",

        "https://ghibliapi.herokuapp.com/people/08ffbce4-7f94-476a-95bc-76d3c3969c19",

        "https://ghibliapi.herokuapp.com/people/0f8ef701-b4c7-4f15-bd15-368c7fe38d0a"

    ],

    "species": [

        "https://ghibliapi.herokuapp.com/species/af3910a6-429f-4c74-9ad5-dfe1c4aa04f2",

        "https://ghibliapi.herokuapp.com/species/603428ba-8a86-4b0b-a9f1-65df6abef3d3",

        "https://ghibliapi.herokuapp.com/species/74b7f547-1577-4430-806c-c358c8b6bcf5"

    ],

    "locations": [

        "https://ghibliapi.herokuapp.com/locations/"

    ],

    "vehicles": [

        "https://ghibliapi.herokuapp.com/vehicles/"

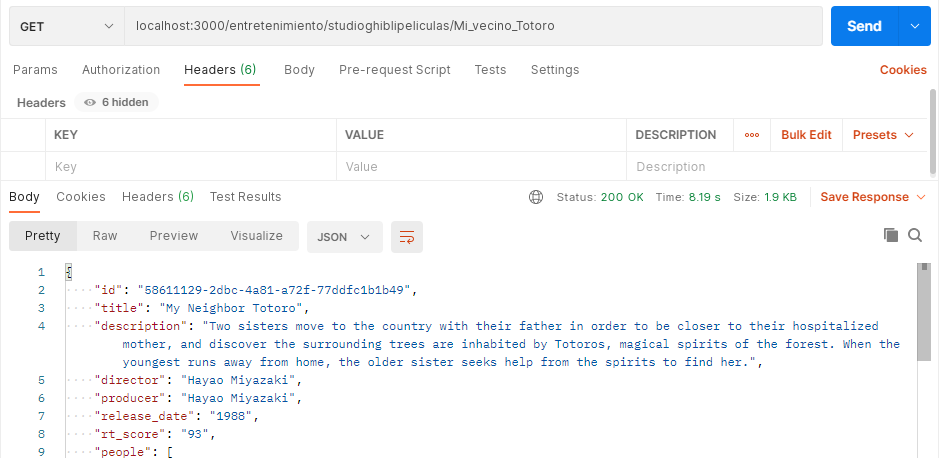
    ],

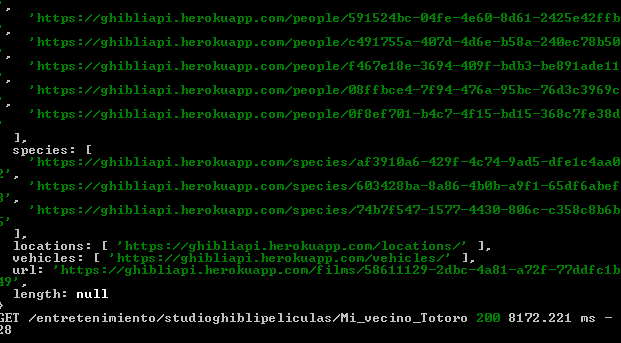
    "url": "https://ghibliapi.herokuapp.com/films/58611129-2dbc-4a81-a72f-77ddfc1b1b49",

    "length": **null**

}

**Fotoevidencia**





# Método Edad de nombres

/entretenimiento/edades/{nombre}

**Descripción:** este método estima la edad de una persona leyendo un nombre propio. Su uso es para entretenimiento.

**Endpoints**: https://api.agify.io?name={ nombre}

**Documentación**: <https://agify.io/>

**Parámetros:**

|  |  |  |  |
| --- | --- | --- | --- |
| Parámetro | Param. De entrada o salida? | Tipo | Descripción |
| {nombre} | entrada | Param URL | Nombre que ingresa el usuario |
| name | Salida | string | Nombre propio detectado |
| age | salida | numeric | Edad calculada con base en el nombre |
| count | salida | numeric | Frecuencia del nombre en la base de datos |

**Ejemplo**

localhost:8080/entretenimiento/edades/mary

**Respuestas**

{

    "name": "mary",

    "age": 63,

    "count": 140958

}

**Fotoevidencia**

