

Sesión # 9 Componente Práctico

Desarrollo de Front-end web con React

Crea una aplicación en React donde muestres 20 de los personajes del mundo Marvel con su nombre e imagen. Para ello, sigue los siguientes pasos:

1. Ingresa a www.marvel.com/signin y crea una cuenta
2. Accede al correo electrónico registrado y confirma tu cuenta
3. Una vez confirmada accede a [Get a Key](#), sigue los pasos señalados y obtén tus llaves para hacer peticiones a la API de Marvel. Es probable que este paso demore unos minutos cargando, no te preocupes.



MY DEVELOPER ACCOUNT

Hi kriste783282115!
Here's your personal Marvel Comics API information:

Your public key

acee3ebcf5e11d64f4b19f6143e5b812

Your private key

8ce43aae2de470485447788dfdd91eb6c867a87

Read more about how to use your keys to sign requests. [»](#)

Your rate limit: **3000** Number of calls your application can make per day.

Your authorized referrers

List any domains that can make calls to the Marvel Comics API using your API key here:



4. Sin cerrar la pestaña anterior, accede a la [documentación de la API](#) y revisa el endpoint **GET /v1/public/characters**, este es el que usaremos en nuestro proyecto. Haz click sobre el botón *Try it out!* y mira un ejemplo de la Request URL y la respuesta obtenida al consumir este endpoint.

Try it out! Hide Response

Request URL
<https://gateway.marvel.com:443/v1/public/characters?apikey=acee3ebcf5e11d64f4b19f6143e5b812>

Response Body

```
"etag": "ab47c74e9c6e6e9027d52841a9587a14c2b1a2f1",
"data": {
  "offset": 0,
  "limit": 20,
  "total": 1559,
  "count": 20,
  "results": [
    {
      "id": 1011334,
      "name": "3-D Man",
      "description": "",
      "modified": "2014-04-29T14:18:17-0400",
      "thumbnail": {
        "path": "http://i.annihil.us/u/prod/marvel/i/mg/c/e0/535fecbbb9784",
        "extension": "jpg"
      },
      "resourceURI": "http://gateway.marvel.com/v1/public/characters/1011334",
      "comics": {
        "available": 12,
        "collectionURI": "http://gateway.marvel.com/v1/public/characters/1011334/comics"
      }
    }
  ]
}
```

Response Code
200

Response Headers

```
{
  "Content-Type": "application/json; charset=utf-8",
  "Date": "Mon, 22 Nov 2021 03:59:39 GMT"
}
```

- Crea un nuevo proyecto en React, utiliza la siguiente plantilla como base de tu solución: <https://github.com/Misiontic-Ciclo-4A/react-template>
- Corre el proyecto y verifica que todo esté en orden. Hasta este paso, debes poder visualizar 'Hola mundo' en nuestro caso, 'Marvel'



- Una vez tengas tu proyecto, instala las siguiente librería desde tu terminal:

Para realizar peticiones

npm i axios

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL
~/L/Mo/com~apple~C/Doc/CICLO4A/sesion9-solucion master ?6 > npm i axios
* axios
```

Para los estilos

npm i bootstrap --save

```
Run npm audit fix to fix many of these, or npm audit --fix for details
~/L/Mo/com~apple~C/Doc/CICLO4A/sesion9-solucion master ?7 > npm i bootstrap
```

** Esto puede demorar un poco según tu conexión a internet

- Ahora, desde el archivo App.js copia a manera de comentario la Request URL del endpoint mencionado en el paso 4 y tus llaves generadas al registrarse. En mi caso son las siguientes:

```
src > JS App.js > ...
  1 import "./App.css";
  2
  3 //https://gateway.marvel.com:443/v1/public/characters?apikey=acee3ebcf5e11d64f4b19f6143e5b812
  4 //key privada: 8ce43aae2de470485447788dfddf91eb6c867a87
  5 //key publica: acee3ebcf5e11d64f4b19f6143e5b812
  6
```

- Para conectarnos a nuestra API primeramente tendremos que realizar un proceso de autenticación, el cual según la [documentación](#) debe seguir el siguiente formato:

md5(ts+privateKey+publicKey), donde ts es un timestamp

Por ejemplo, un usuario con una clave pública de "1234" y una clave privada de "abcd" podría construir una llamada válida de la siguiente manera:

`http://gateway.marvel.com/v1/public/comics?ts=1&apikey=1234&hash=ffd275c5130566a2916217b101f26150` (el valor hash es el resumen md5 de `1abcd1234`).

- Según este ejemplo, crea tu formato con tus keys asignadas y un ts=1.

En mi caso sería:

`18ce43aae2de470485447788dfddf91eb6c867a87acee3ebcf5e11d64f4b19f6143e5b812`

- Ahora, utiliza <https://hash-generator.io/online-hash-generator.php?hl=es> para convertir el string anterior en un hash md5. Este hash será nuestra contraseña para conectarnos a la API, copiala y pegala como comentario junto con el resto de nuestras llaves para tenerla a la mano.

The screenshot shows a web browser window with the URL `hash-generator.io/online-hash-generator.php?hl=es`. The page title is "hash-generator.io". A text input field contains the string `18ce43aae2de470485447788dfddf91eb6c867a87acee3ebcf5e11d64f4b19f6143e5b812`. Below the input is a button labeled "Crear código hash". To the right, there are sections for MD2, MD4, and MD5 hash generation. The MD5 section is highlighted with a red border and contains the value `46dc15bd07661ab1f50e6127d7668994`.

- Importa bootstrap en el archivo index.js y verifica que la tipografía de nuestra app ha cambiado

```
import "bootstrap";
import "bootstrap/dist/css/bootstrap.css";
```

The screenshot shows a browser window with the URL `localhost:3001`. The page displays the word "MARVEL" in a large, bold font. The font appears to be a sans-serif typeface, which is different from the default monospace font used in the previous step.

- Añade las siguientes importaciones a tu archivo App.js, para la conexión a la API

```
import axios from "axios";
import { useState, useEffect } from "react";
```

- En el archivo App.js utilizaremos el hook **useEffect** para crear la conexión a la API inmediatamente actualicemos o corramos nuestra aplicación y **useState** para almacenar el estado de nuestro componente. useState acepta un valor inicial para esa variable y devuelve un array con dos elementos, el valor de la variable y la función para modificarla.

Ten en cuenta la estructura de la url:

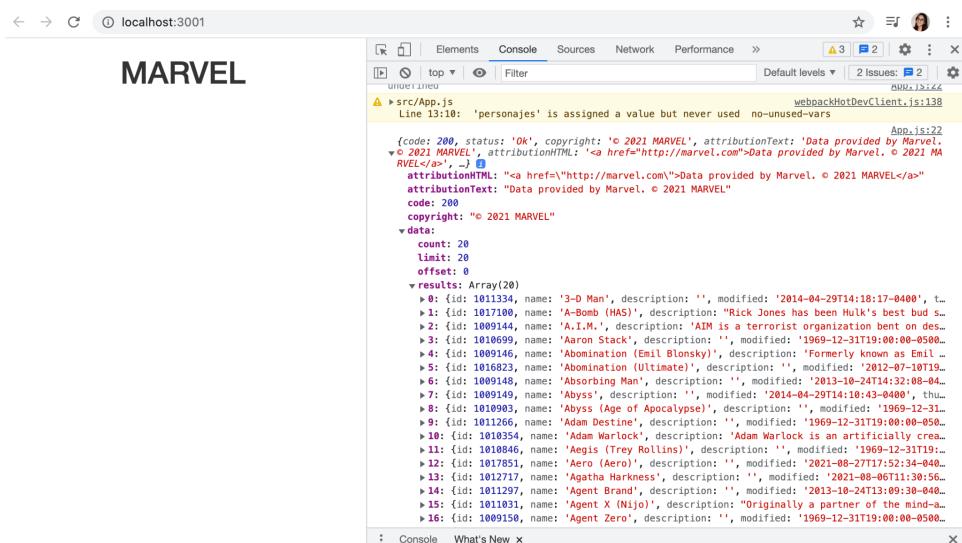
```
https://gateway.marvel.com:443/v1/public/characters?ts=1&  
apikey=publicKey&hash=md5pass
```

- Implementa la petición. Quedará así:

```
const [personajes, setPersonajes] = useState([]);  
useEffect(  
() =>  
  axios  
    .get(  
      "https://gateway.marvel.com:443/v1/public/characters?ts=1&apikey=acee3ebcf5e11d64f4b19f6143e5b812&hash=46dc15bd07661ab1f50e6127d7668994"  
    )  
    .then((res) => {  
      setPersonajes(res.data);  
      console.log(res.data);  
    })  
    .catch((error) => console.log(error)),  
  []  
);
```

- Ve a tu aplicación corriendo en el navegador y mediante la herramienta de inspección observa el resultado en consola.

Como ves, ya tenemos acceso a la data proporcionada por la API.



- Ahora, guarda sólo el objeto de results (teniendo en cuenta que es lo que nos interesa)

```
setPersonajes(res.data.data.results);
console.log(personajes);
```

MARVEL

```
etag: "ab42c74e9c6e6e9027d52841a9587a14c2b1a2f1"
status: "OK"
[[{"id": 1011334, "name": "3-D Man", "description": "", "modified": "2014-04-29T14:18:17-0400", "thumbnail.path": "https://i.imgur.com/3DMan.jpg", "thumbnail.extension": "jpg"}, {"id": 1017108, "name": "A-Bomb (HAS)", "description": "Rick Jones has been Hulk's best bud since the early 1970s.", "modified": "2014-04-29T14:18:17-0400", "thumbnail.path": "https://i.imgur.com/ABomb.jpg", "thumbnail.extension": "jpg"}, {"id": 1009144, "name": "A.I.M.", "description": "AIM is a terrorist organization bent on destroying the world.", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/AIM.jpg", "thumbnail.extension": "jpg"}, {"id": 1010699, "name": "Aaron Stack", "description": "", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/AaronStack.jpg", "thumbnail.extension": "jpg"}, {"id": 1016823, "name": "Abomination (Emil Blonsky)", "description": "Formerly known as Emil Blonsky, he was a scientist who turned into a giant green monster.", "modified": "2012-07-10T19:11:32-0400", "thumbnail.path": "https://i.imgur.com/Abomination.jpg", "thumbnail.extension": "jpg"}, {"id": 1009146, "name": "Abomination (Ultimate)", "description": "", "modified": "2014-04-29T14:18:17-0400", "thumbnail.path": "https://i.imgur.com/AbominationUltimate.jpg", "thumbnail.extension": "jpg"}, {"id": 1009148, "name": "Absorbing Man", "description": "", "modified": "2013-10-24T14:32:08-0400", "thumbnail.path": "https://i.imgur.com/AbsorbingMan.jpg", "thumbnail.extension": "jpg"}, {"id": 1009149, "name": "Abysse", "description": "", "modified": "2014-04-29T14:10:43-0400", "thumbnail.path": "https://i.imgur.com/Abysse.jpg", "thumbnail.extension": "jpg"}, {"id": 1010903, "name": "Abyss (Age of Apocalypse)", "description": "", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/Abyss.jpg", "thumbnail.extension": "jpg"}, {"id": 1011266, "name": "Adam Destine", "description": "", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/AdamDestine.jpg", "thumbnail.extension": "jpg"}, {"id": 1018354, "name": "Adam Warlock", "description": "Adam Warlock is an artificially created being.", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/AdamWarlock.jpg", "thumbnail.extension": "jpg"}, {"id": 1018846, "name": "Aegis (Trey Rollins)", "description": "", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/Aegis.jpg", "thumbnail.extension": "jpg"}, {"id": 1017851, "name": "Aero (Aero)", "description": "", "modified": "2021-08-27T17:52:34-0400", "thumbnail.path": "https://i.imgur.com/Aero.jpg", "thumbnail.extension": "jpg"}, {"id": 1012717, "name": "Agatha Harkness", "description": "", "modified": "2021-08-06T11:30:56-0400", "thumbnail.path": "https://i.imgur.com/AgathaHarkness.jpg", "thumbnail.extension": "jpg"}, {"id": 1011297, "name": "Agent Brand", "description": "", "modified": "2013-10-24T13:09:30-0400", "thumbnail.path": "https://i.imgur.com/AgentBrand.jpg", "thumbnail.extension": "jpg"}, {"id": 1011031, "name": "Agent X (Nijilo)", "description": "Originally partner of the mind-altering Agent Zero.", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/AgentX.jpg", "thumbnail.extension": "jpg"}, {"id": 1009150, "name": "Agent Zero", "description": "", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/AgentZero.jpg", "thumbnail.extension": "jpg"}, {"id": 1011198, "name": "Agent Object", "description": "", "modified": "2016-02-03T10:25:22-0500", "thumbnail.path": "https://i.imgur.com/AgentObject.jpg", "thumbnail.extension": "jpg"}, {"id": 1011175, "name": "Aglarion", "description": "", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/Aglarion.jpg", "thumbnail.extension": "jpg"}, {"id": 1011136, "name": "Air-Walker (Gabriel Lan)", "description": "", "modified": "1969-12-31T19:00:00-0500", "thumbnail.path": "https://i.imgur.com/AirWalker.jpg", "thumbnail.extension": "jpg"}]]
```

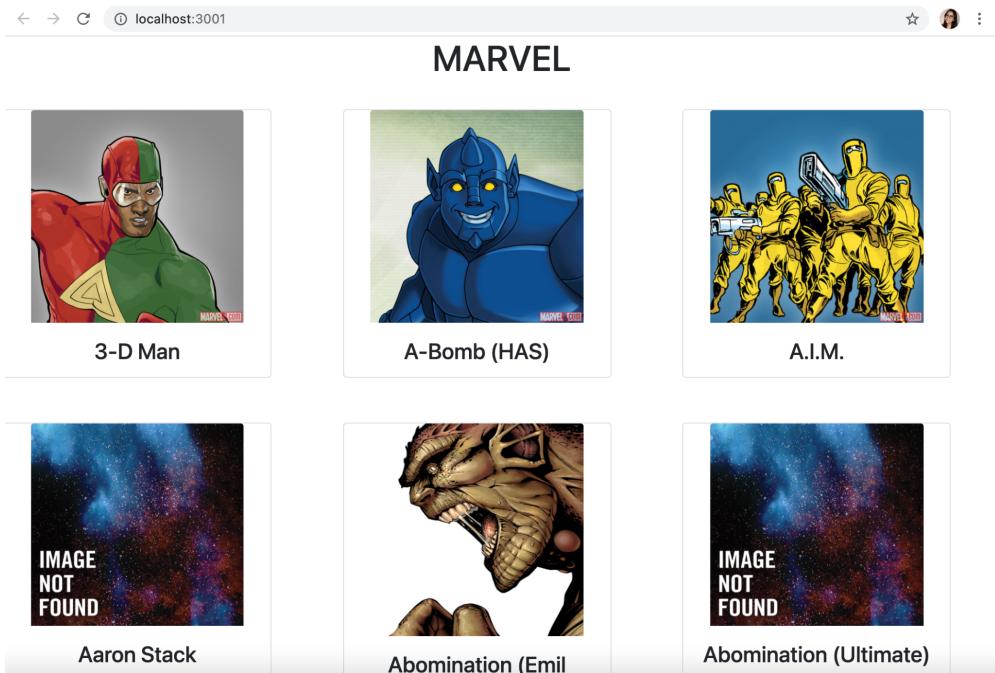
- ¡Listo! Ya nos estamos conectando a la API, ahora lo que nos queda es mostrar la información obtenida. Para ello, en el render del archivo App.js recorre el arreglo de Personajes y con ayuda de bootstrap crea una grid para mostrar la imagen de cada personaje y su nombre.

```
return (
  <div className="App">
    <h1>MARVEL</h1>

    {/* GRID DE IMAGENES */}
    {/* Elemento extraido de bootstrap, doc: https://getbootstrap.com/docs/5.1/components/card/#card-layout */}

    <div className="row row-cols-1 row-cols-md-3 g-4">
      {/* Recorriendo el arreglo de datos */}
      {personajes.map((p) =>
        <div className="col mt-5" key={p.id}>
          <div
            className="card align-items-center"
            style={{ width: "18rem", height: "18rem" }}
          >
            <img
              src={`${p.thumbnail.path}.${p.thumbnail.extension}`}
              className="card-img-top"
              style={{ width: "80%", height: "80%" }}
            />
            <div className="card-body">
              <h4 className="card-title">{p.name}</h4>
            </div>
          </div>
        </div>
      ))}
    </div>
  </div>
);
```

Resultado:



VALIDA TU SOLUCIÓN <https://github.com/Misiontic-Ciclo-4A/sesion9-solucion>