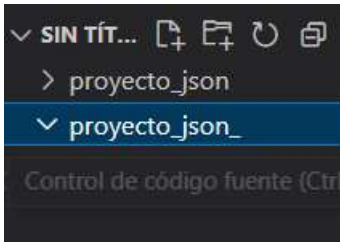


API REST JSON

En este documento podemos encontrar una pequeña documentación de cómo crear una API REST de ejemplo para depuración y pruebas

Para crear api de prueba abrimos vacia una carpeta dentro de Visual Studio Code en mi caso proyecto_json_ .



Necesitamos también el NODE.JS deberíamos descargar e instalar

 A screenshot of the Node.js download page (nodejs.org/es/download). The page is in Spanish and shows the download options for Node.js version 18.15.0. It includes a navigation bar with links like INICIO, ACERCA, DESCARGAS, DOCUMENTACIÓN, PARTICIPE, SEGURIDAD, CERTIFICACIÓN, and NOTICIAS. The main content area is titled 'Descargas' and mentions 'Versión actual: 18.15.0 (includes npm 9.5.0)'. It instructs users to download the source code or a pre-compiled installer. Below this, there are two main sections: 'LTS' (Recommended for the majority) and 'Actual' (Latest features). Under 'LTS', there are links for 'Instalador Windows' (node-v18.15.0-x64.msi), 'Instalador macOS' (node-v18.15.0.pkg), and 'Código Fuente' (node-v18.15.0.tar.gz). Under 'Actual', there are similar links. At the bottom, there is a table summarizing the available binaries and their architectures.

Instalador Windows (.msi)	32-bit	64-bit
Binario Windows (.zip)	32-bit	64-bit
Instalador macOS (.pkg)	64-bit / ARM64	
Binario macOS (.tar.gz)	64-bit	ARM64
Binario Linux (x64)	64-bit	
Binario Linux (ARM)	ARMv7	ARMv8
Código Fuente	node-v18.15.0.tar.gz	

Una vez instalada vamos al Visual Studio Code y abrimos una nueva linea de comando Inicializamos un paquete de node.js por defecto con el comando `npm init -y`

```
PS C:\Users\ArnauMB\Desktop\DAM\M04\proyecto_json_> npm init -y
Wrote to C:\Users\ArnauMB\Desktop\DAM\M04\proyecto_json_\package.json:

{
  "name": "proyecto_json_",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

PS C:\Users\ArnauMB\Desktop\DAM\M04\proyecto_json_> █
```

Esto sirve para gestionar los paquetes npm y las dependencias vamos a instalar la dependencia que queremos `npm i json-server -D`

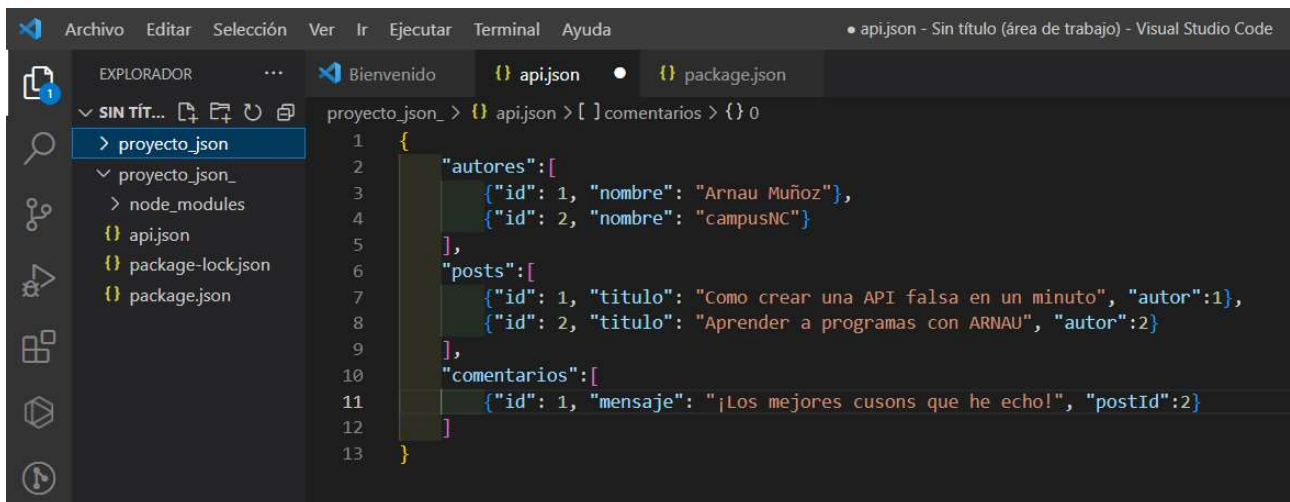
```
PS C:\Users\ArnauMB\Desktop\DAM\M04\proyecto_json_> npm i json-server -D
added 315 packages, and audited 316 packages in 20s

92 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\ArnauMB\Desktop\DAM\M04\proyecto_json_> █
```

Con esto lo que hemos conseguido es tener una dependencia para el tipo de desarrollo para este paquete de json server.

Vamos a crear nuestra base de datos y una tarea para lanzarlo
creamos el documento api.json



The screenshot shows the Visual Studio Code editor with the 'api.json' file open. The Explorer sidebar on the left shows the project structure: 'proyecto_json' (selected), 'proyecto_json_', 'node_modules', 'api.json', 'package-lock.json', and 'package.json'. The main editor area displays the content of 'api.json' with the following JSON structure:

```
1 {
2   "autores": [
3     { "id": 1, "nombre": "Arnau Muñoz" },
4     { "id": 2, "nombre": "campusNC" }
5   ],
6   "posts": [
7     { "id": 1, "titulo": "Como crear una API falsa en un minuto", "autor": 1 },
8     { "id": 2, "titulo": "Aprender a programar con ARNAU", "autor": 2 }
9   ],
10  "comentarios": [
11    { "id": 1, "mensaje": "¡Los mejores cursos que he echo!", "postId": 2 }
12  ]
13 }
```

Para poder lanzar esta api, para ello volvemos al package.json y creamos una nueva tarea dentro de la rama scripts



The screenshot shows the Visual Studio Code editor with the 'package.json' file open. The Explorer sidebar on the left shows the project structure: 'proyecto_json' (selected), 'proyecto_json_', 'node_modules', 'api.json', 'package-lock.json', and 'package.json'. The main editor area displays the content of 'package.json' with the following JSON structure:

```
1 {
2   "name": "proyecto_json_",
3   "version": "1.0.0",
4   "description": "",
5   "main": "index.js",
6   "scripts": {
7     "servicio": "json-server -- watch api.json "
8   },
9   "keywords": [],
10  "author": "",
11  "license": "ISC",
12  "devDependencies": {
13    "json-server": "^0.17.3"
14  }
15 }
```

Solamente con esto ya podemos lanzar el servidor y tener nuestro servicio en marcha volvemos a la línea de comandos y escribimos `npm run servicio`

```
PS C:\Users\ArnauMB\Desktop\DAM\M04\proyecto_json_> npm run servicio

> proyecto_json_@1.0.0 servicio
> json-server --watch api.json

\{^_^}/ hi!

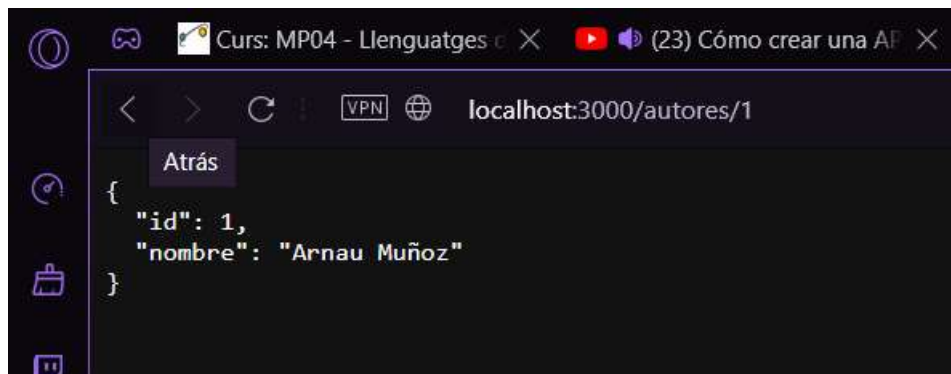
Loading api.json
Done

Resources
http://localhost:3000/autores
http://localhost:3000/posts
http://localhost:3000/comentarios

Home
http://localhost:3000

Type s + enter at any time to create a snapshot of the database
Watching...
```

Como podemos ver se crean en el puerto 3000 tres endpoints uno para autores otro para post y otro para comentarios que coinciden con las propiedades del archivo `api.json`

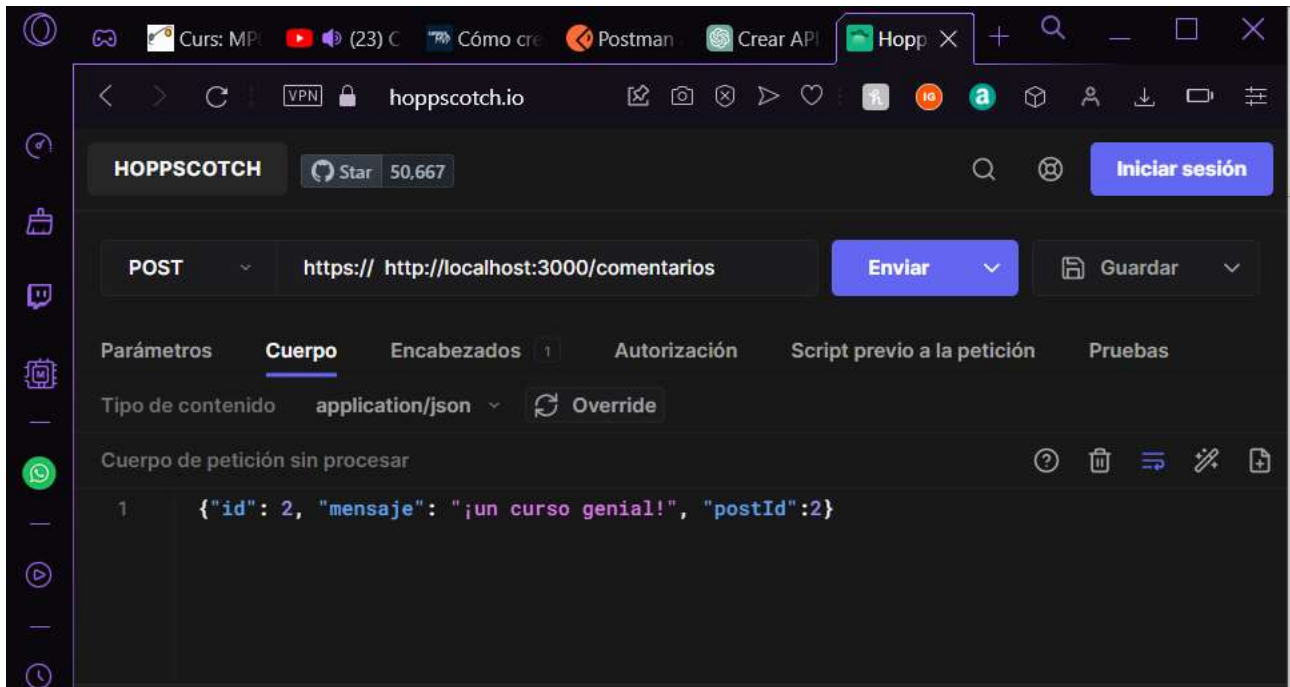


En este caso estamos viendo los autores con id 1

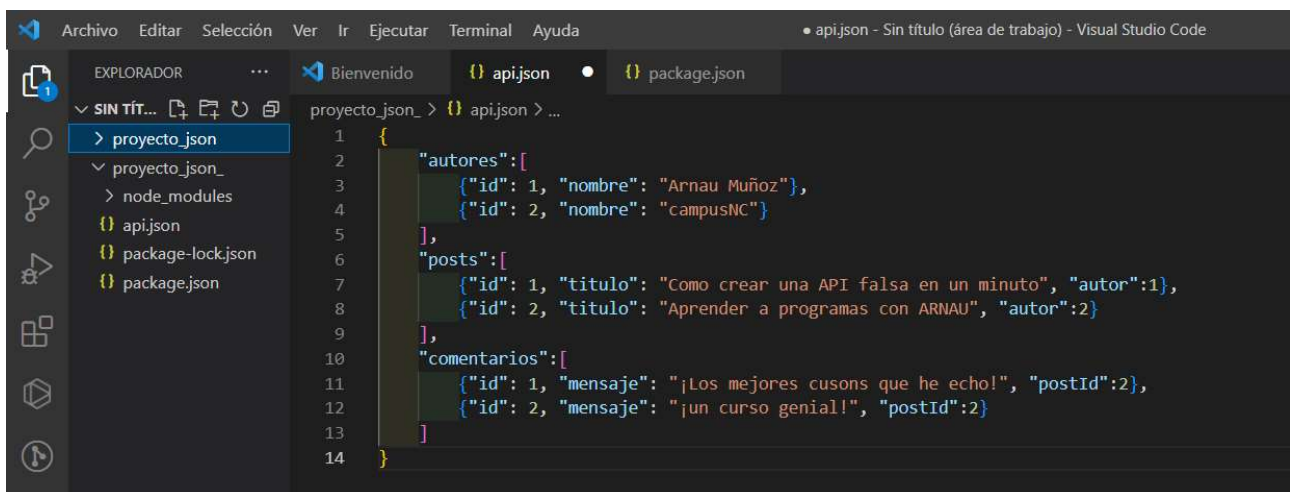
En la web de json server de gitHub podremos encontrar todas las formas de acceder a la información

github.com/typicode/json-server

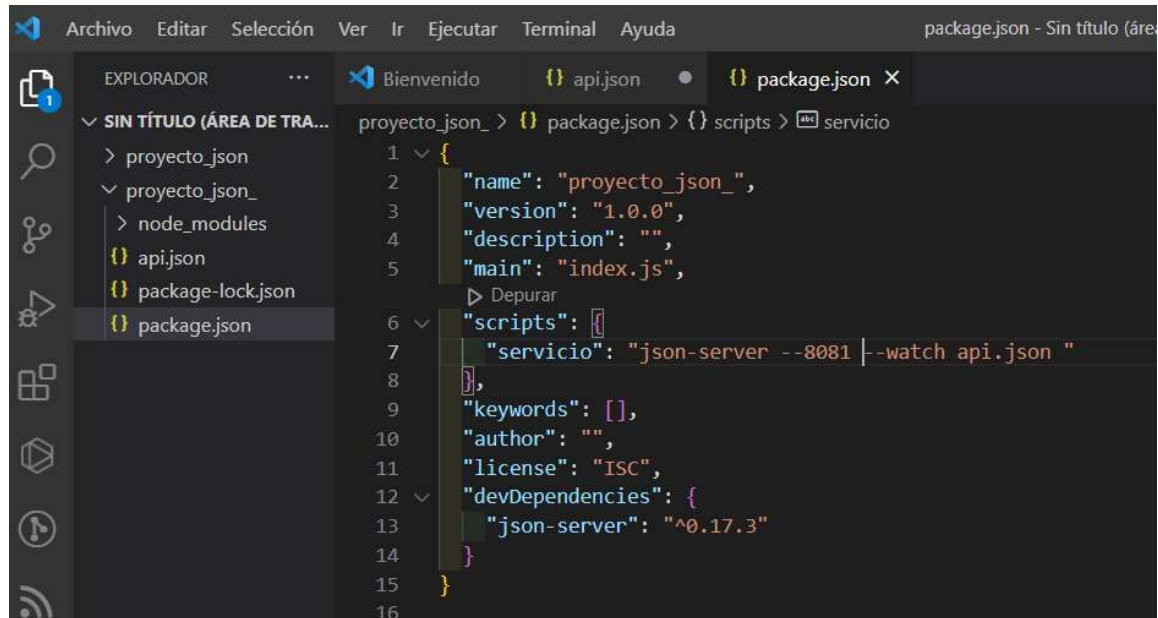
Ahora vamos a ver como introducir datos en esta API
accedemos a una <https://hoppscotch.io> que es una herramienta para acceder a la api y simular llamadas y servicios.



Como vemos ya a sido introducido



Por ultimo vamos a ver como cambiar el puerto por si esta en uso o para tener varios servicios con el comando `--port` y el puerto nuevo, en este caso el 8081, dentro del servicio en scripts



Para que funcione reiniciamos el servicios

```

PS C:\Users\ArnauMB\Desktop\DAM\M04\proyecto_json> npm run servicio

> proyecto_json_@1.0.0 servicio
> json-server --port 8081 --watch api.json

\{^_^}/ hi!

Loading api.json
Done

Resources
http://localhost:8081/autores
http://localhost:8081/posts
http://localhost:8081/comentarios

Home
http://localhost:8081

Type s + enter at any time to create a snapshot of the database
Watching...

```

Y finalmente aquí podemos ver el servicio con el nuevo puerto introducido