



TECNOLÓGICO NACIONAL DE MÉXICO



INSTITUTO TECNOLÓGICO DEL VALLE DE OAXACA

Web Service Restful

ASIGNATURA: DESARROLLO DE APLICACIONES MÓVILES II

DOCENTE: L.I. AMBROSIO CARDOSO JIMÉNEZ

ALUMNO: CARRIZOSA BUSTAMANTE JOSÉ MANUEL

SEMESTRE: 9° GRUPO: A

CARRERA: INGENIERÍA INFORMÁTICA.

NUMERO DE CONTROL: 17920317



RESTful

Los servicios web Restful son aquellos en los que se accede a un recurso remoto en el backend de manera simple. Las características que tiene un servicio Restful son las siguientes (Castro 2018).

Características principales de un servicio Web RESTful

- Tiene cinco operaciones típicas: listar, crear, leer, actualizar y borrar
- Cada operación requiere de dos cosas: El método URI y HTTP
- El URI es un sustantivo que contiene el nombre del recurso
- El método HTTP es un verbo.

Se desarrolló una API en el entorno de ejecución Deno y con el lenguaje Typescript para proveer información relacionada a frases célebres de diversos autores. Para ello se cuenta con los siguientes endpoints.

Tabla 1: Operaciones de la API

Operación Método HTTP		URI	Parámetros	Resultado
Listar	GET	/quotes	No aplica	Se muestran todas las frases guardadas en la base de datos
Crear	POST	/quotes	Dentro del cuerpo en el POST en formato JSON	Se guarda una nueva frase
Leer	GET	/quotes/{quote_id}	No aplica	Devuelve una frase específica indicada por el id en la URI
Actualizar	PUT	/quotes/{quote_id}	Dentro del cuerpo de la petición en formato JSON	Se actualiza la frase correspondiente al id señalado en la URI
Borrar	DELETE	/quotes/{quote_id}	No aplica	Se elimina la frase en función al id

La primera vez que se ejecuta la aplicación y después de realizar un cambio en el código, se muestra el siguiente mensaje.



```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

at https://deno.land/std@0.106.0/node/events.ts:451:11

TS2578 [ERROR]: Unused '@ts-expect-error' directive.
// @ts-expect-error
at https://deno.land/std@0.106.0/node/events.ts:646:5

TS2578 [ERROR]: Unused '@ts-expect-error' directive.
// @ts-expect-error
at https://deno.land/std@0.106.0/node/events.ts:667:5

Found 11 errors.
```

Ilustración 1: Ejecución después de hacer un cambio

Sin embargo al repetir la operación, se muestra que la aplicación está corriendo y escuchando en el puerto especificado en el código.

```
PS C:\Users\carrizosa\proyectosjm\quotes-api-main\quotes-api-main> deno run --allow-net --allow-read --allow-env server.ts --config @ts-ignore
Server running on port 8080
```

Ilustración 2: Aplicación deno ejecutándose

Con la aplicación corriendo se pueden probar los endpoints con una herramienta para realizar peticiones HTTP, por ejemplo Postman.

Listar todas las frases

```
GET localhost:8080/api/v1/quotes

Status: 200 OK Time: 1517 ms Size: 2.06 KB

{
  "success": true,
  "message": "Retrive list quotes",
  "data": [
    {
      "_id": "6193fcd412e8fb8e0a79960c",
      "id": 2,
      "quote": "No temo a los ordenadores; lo que temo es quedarme sin ellos",
      "author": "Isaac Asimov"
    },
    {
      "_id": "6193fcd412e8fb8e0a79960d",
      "id": 3,
      "quote": "Pienso que los virus informáticos muestran la naturaleza humana: la única forma de vida que hemos creado hasta el momento es puramente destructiva",
      "author": "Isaac Asimov"
    }
  ]
}
```

Ilustración 3: Se muestran todas las frases



Crear una frase

POST localhost:8080/api/v1/quotes/ Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {  
2   "author": "Benito Juárez García",  
3   "quote": "El respeto al derecho ajeno es la paz"  
4 }
```

Body Cookies Headers (3) Test Results Status: 200 OK Time: 321 ms Size: 299 B Save

Pretty Raw Preview Visualize JSON

```
1 {  
2   "success": true,  
3   "message": "save quote successfull",  
4   "data": {  
5     "_id": "619729d60c63e20134a2aba4",  
6     "quote": "El respeto al derecho ajeno es la paz",  
7     "author": "Benito Juárez García"  
8   }  
9 }
```

Ilustración 4: Creación de una frase

Cuando falta información para la creación de la frase, como el autor o el texto de la frase.

POST localhost:8080/api/v1/quotes/ Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {  
2   "author": "El conocimiento es sólo una de las representaciones de la existencia"  
3 }
```

Body Cookies Headers (3) Test Results Status: 400 Bad Request Time: 23 ms

Pretty Raw Preview Visualize JSON

```
1 {  
2   "success": false,  
3   "message": "The request must have the citation and author.",  
4   "data": []  
5 }
```

Ilustración 5: Inserción incorrecta



Eliminar una frase.

DELETE localhost:8080/api/v1/quotes/10 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Body Cookies Headers (3) Test Results Status: 200 OK Time: 67 ms Size: 364 B

Pretty Raw Preview Visualize JSON ≡

```
1 {
2   "success": true,
3   "message": "Quote removed",
4   "data": [
5     {
6       "_id": "6193fcd412e8fb8e0a799614",
7       "id": 10,
8       "quote": "Enseñar a los niños el uso de software libre en las escuelas, formará individuos con sentido de libertad",
9       "author": "Richard Stallman"
10    }
11  ]
12 }
```

Ilustración 6: Eliminación de una frase

Actualizar una frase.

PUT localhost:8080/api/v1/quotes/9

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON ▼

```
1 {
2   "quote": "Hablar es fácil, muéstrame el código.",
3   "author": "Linus Benedict Torvalds"
4 }
```

Body Cookies Headers (3) Test Results Status: 200 OK

Pretty Raw Preview Visualize JSON ≡

```
1 {
2   "success": true,
3   "message": "Update for quote with id 9 was successful",
4   "data": []
5 }
```

Ilustración 7: Edición de una frase existente



Cuando no se indica la frase o el autor correspondiente se muestra se indica el fallo en la respuesta.

PUT localhost:8080/api/v1/quotes/9

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● GraphQL **JSON**

```
1 {
2
3 }
```

Body Cookies Headers (3) Test Results Status: 400 Bad Request Time: 19 ms

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "success": false,
3   "message": "The request must have the citation or author.",
4   "data": []
5 }
```

Ilustración 8: Error de inserción, datos incompletos

Cuando no se encuentra la frase correspondiente se devolverá la respuesta indicando que la operación no fue exitosa.

PUT localhost:8080/api/v1/quotes/123

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary ● GraphQL **JSON**

```
1 {
2   "author": "Linus B. Torvalds"
3 }
```

Body Cookies Headers (3) Test Results Status: 404 Not Found Time: 16 ms

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "success": false,
3   "message": "Quote with id: 123 not found",
4   "data": []
5 }
```

Ilustración 9: Error de inserción, no se encontró la frase



Mostrar una frase específica.

Se eligió esta acción para ejemplificar el uso de un middleware que permita la autenticación del usuario antes de acceder a un recurso.

Por ello también se tiene el endpoint para listar los usuarios que pueden utilizar la api.

The screenshot shows a REST client interface with a GET request to `localhost:8080/api/v1/users/`. The response status is 200 OK. The response body is a JSON object with the following structure:

```
1 {
2   "success": true,
3   "message": "list users",
4   "data": [
5     {
6       "id": 1,
7       "account": "josem"
8     },
9     {
10      "id": 6,
11      "account": "manuel"
12    },
13    {
14      "id": 7,
15      "account": "usr1111"
16    }
17  ]
18 }
```

Ilustración 10: Lista de usuarios registrados

Cada usuario debe generar un token para solicitar un recurso que tenga esté relacionado al middleware. Para ello realiza una petición POST con el nombre de usuario y contraseña en formato json dentro del cuerpo.

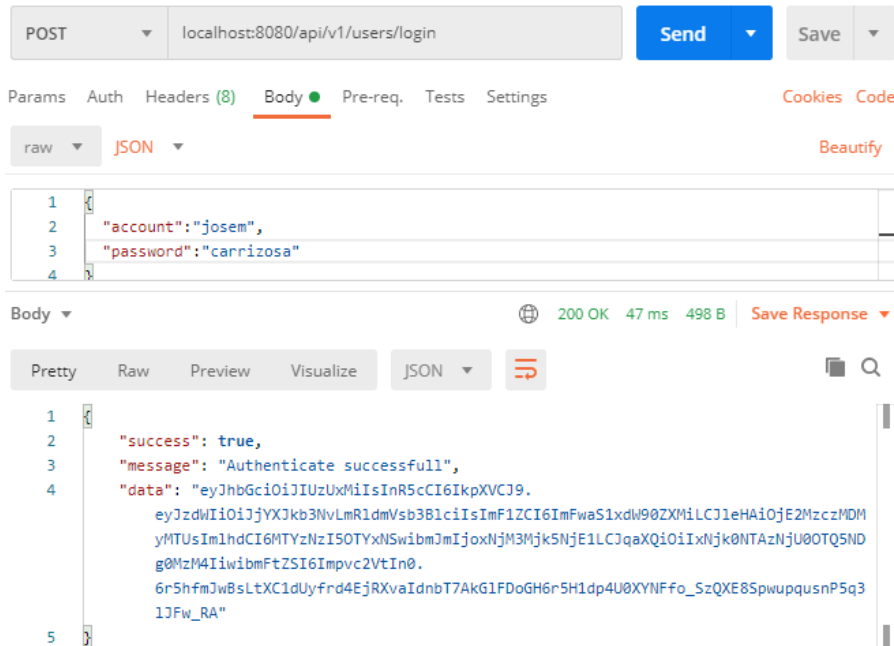


Ilustración 11: Generación de token de usuario

Una vez obtenido el token, deberá enviarlo en el Header de la petición con la clave **Authorization**

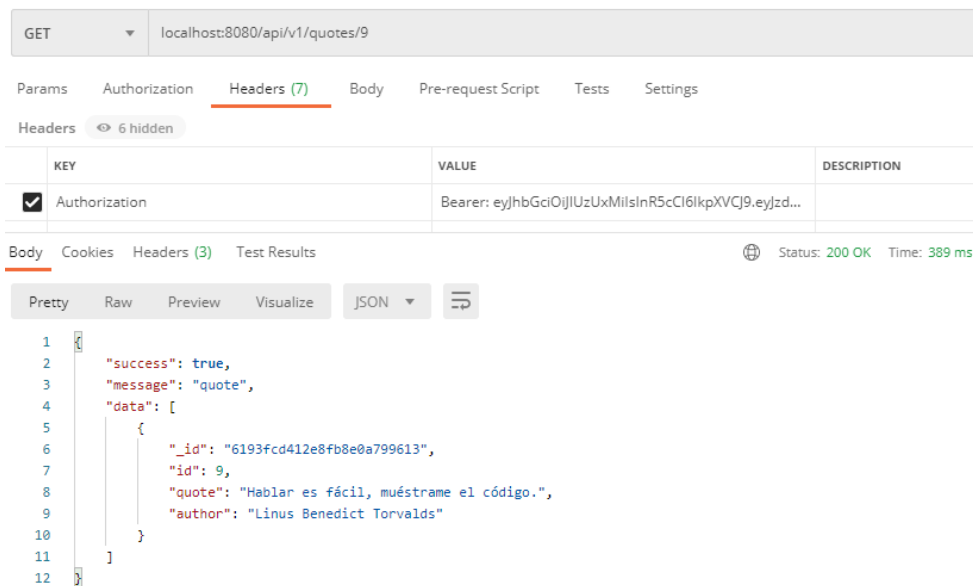


Ilustración 12: Se utiliza el token en el campo Authorization del Header



Cuando no se encuentra la frase con el id señalado se indica en la respuesta en la propiedad **success** que tendrá el valor **false**.

GET localhost:8080/api/v1/quotes/66

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Headers 6 hidden

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> Authorization	Bearer: eyJhbGciOiJIUzUxMiIsInR5cCI6IkpXVCJ9.eyJzdW...	
Key	Value	Description

Body Cookies Headers (3) Test Results

Status: 400 Bad Request Time: 28 ms

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": false,
3   "message": "Quote with id: 66 not found",
4   "data": []
5 }
```

Ilustración 13: No se encontró la frase que señalada

Referencias.

Castro, A. (2018, junio 19). Servicios Web: RESTful. Bi-geek.com. <https://blog.bi-geek.com/servicios-web-restful/>