

Actividad de Postman

Estudiante:

Sofia Alejandra Benavides

Instructoras:

Dalila Mistura

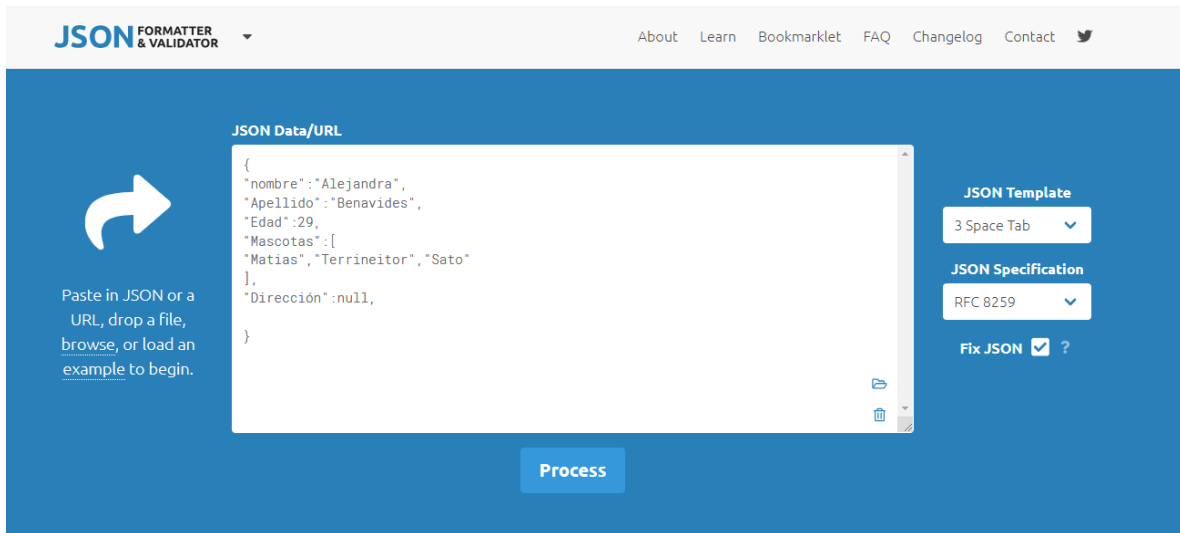
Blanca Vargas

Minhub

Septiembre 14 2024

Ejercicio 1:

1. Crear el Json en la pagina suministrada

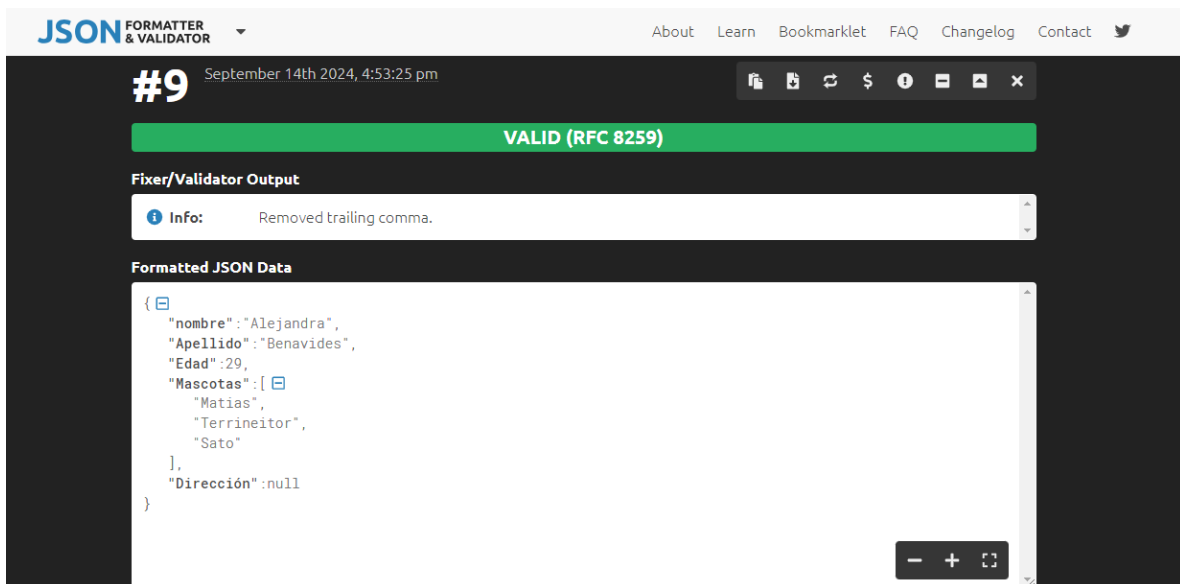


The screenshot shows the JSON Formatter & Validator website. The main area is a blue box with a white text area for input. On the left, there is a large white arrow pointing right and the text: "Paste in JSON or a URL, drop a file, browse, or load an example to begin." The input area contains the following JSON:

```
{
  "nombre": "Alejandra",
  "Apellido": "Benavides",
  "Edad": 29,
  "Mascotas": [
    "Matias", "Terrineitor", "Sato"
  ],
  "Dirección": null,
}
```

On the right side, there are settings for "JSON Template" (3 Space Tab), "JSON Specification" (RFC 8259), and a "Fix JSON" checkbox which is checked. At the bottom center, there is a blue "Process" button.

2. Después de 9 intentos el Json es valido por la plataforma. Los errores se presentaron en la sintaxis como falta de comas y exceso de espacios.



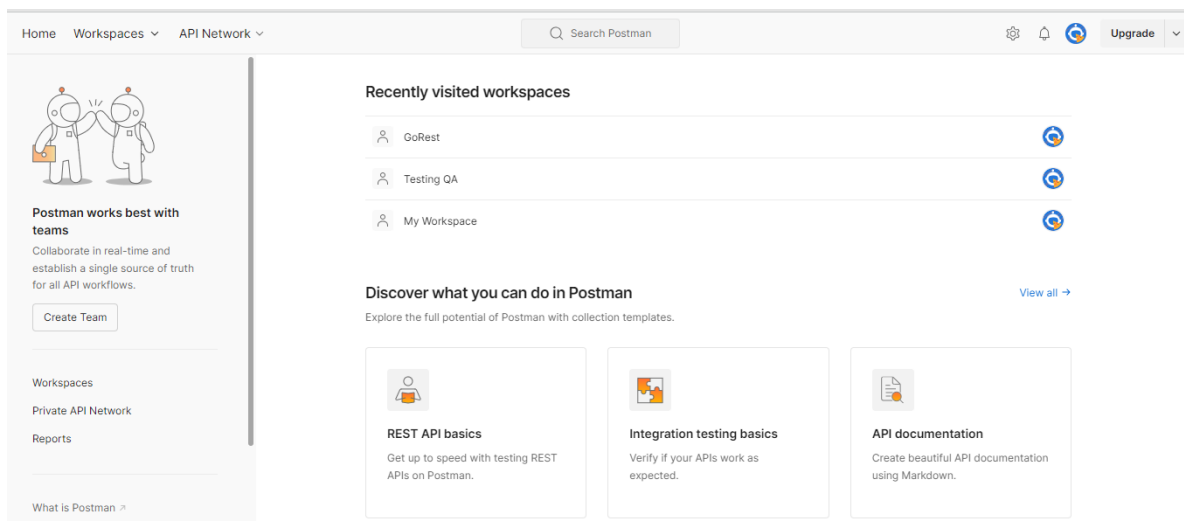
The screenshot shows the JSON Formatter & Validator website after 9 attempts. The top bar is black with the JSON logo and navigation links. Below the top bar, there is a green bar with the text "VALID (RFC 8259)". Underneath, there is a section titled "Fixer/Validator Output" with a message: "Info: Removed trailing comma." Below this, there is a section titled "Formatted JSON Data" with the following formatted JSON:

```
{
  "nombre": "Alejandra",
  "Apellido": "Benavides",
  "Edad": 29,
  "Mascotas": [
    "Matias",
    "Terrineitor",
    "Sato"
  ],
  "Dirección": null
}
```

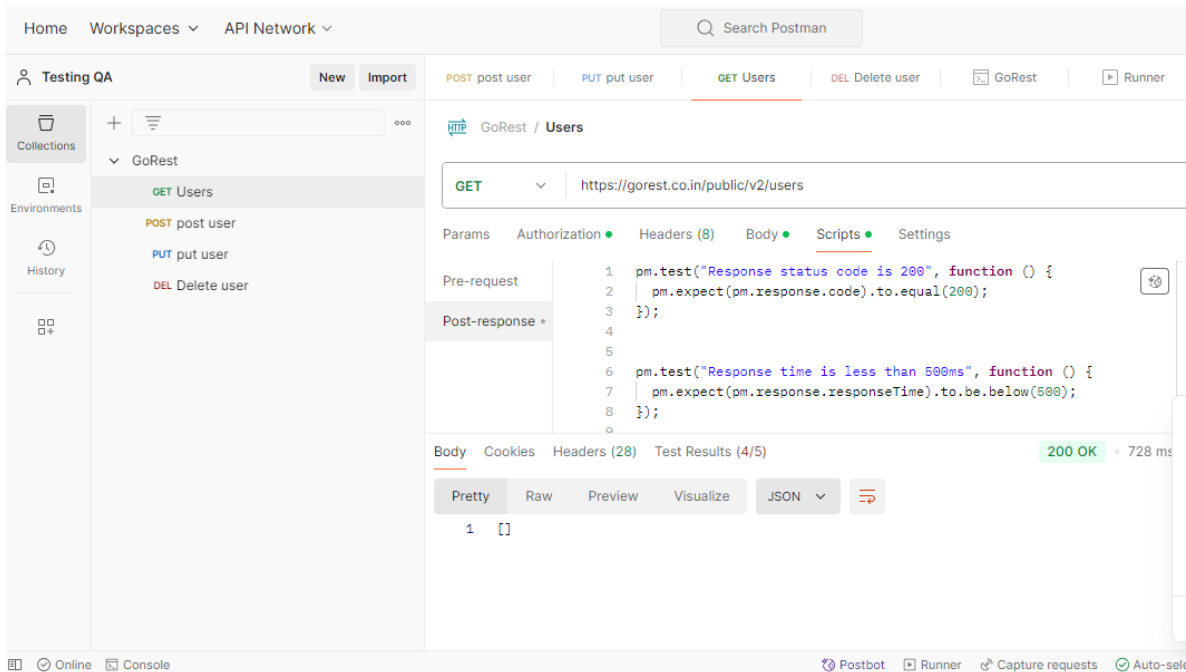
At the bottom right, there are zoom controls (minus, plus, and a square icon).

Ejercicio 2:

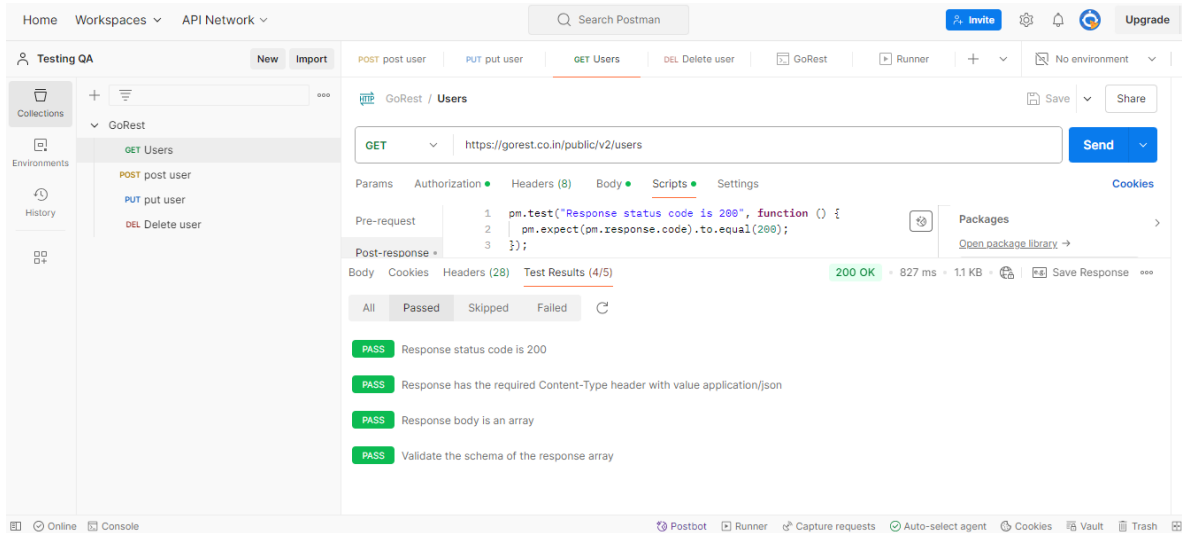
1. Se crean los workspace y la colección en la página web de postman



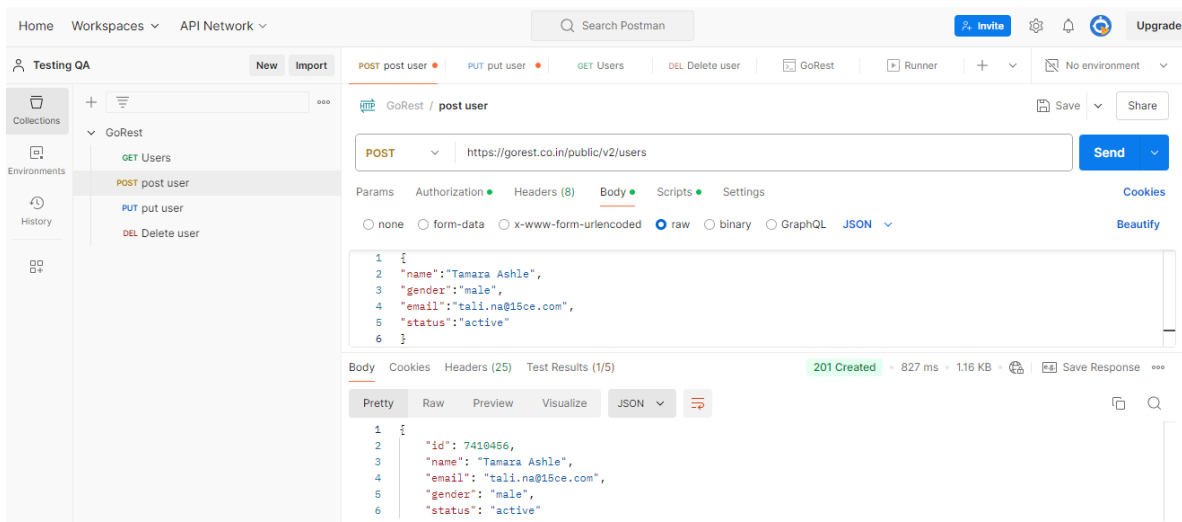
2. Se crean los request solicitados por el cliente, el GET, POST, PUT Y DELETE.



3. Se crea y manda el request de GET que arroja código 200 aprobado.



4. Se crea y manda el request de POST



5. Se crea y manda el request de PUT

The screenshot shows the Postman interface for a collection named "Testing QA". The "PUT put user" request is selected in the left sidebar. The main panel displays the request details for the endpoint `https://gorest.co.in/public/v2/users/7410456`. The request method is "PUT". The "Body" tab is active, showing a JSON payload:

```
1 {
2   "name": "Tamara Ashle",
3   "gender": "female",
4   "email": "tali.na@15ce.com",
5   "status": "active"
6 }
```

The response section shows a status of **200 OK** with a response time of 908 ms and a size of 1.13 KB. The "Body" tab is active, showing the response JSON:

```
1 {
2   "email": "tali.na@15ce.com",
3   "name": "Tamara Ashle",
4   "gender": "female",
5   "status": "active",
6   "id": 7410456
}
```

6. Se crea y manda el request de DELETE

The screenshot shows the Postman interface for the same collection "Testing QA". The "DEL Delete user" request is selected in the left sidebar. The main panel displays the request details for the endpoint `https://gorest.co.in/public/v2/users/7410456`. The request method is "DELETE". The "Body" tab is active, showing a JSON payload:

```
1 {
2   "name": "Tenali Ramakrishna",
3   "gender": "male",
4   "email": "tenali.ramakrishna@15ce.com",
5   "status": "active"
6 }
```

The response section shows a status of **204 No Content** with a response time of 532 ms and a size of 905 B. The "Text" tab is active, showing the response content:

```
1
```

7. Se realizan los test de los request

The image displays two screenshots of the Postman interface, specifically the 'GoRest' collection, showing the 'Tests' tab for different API requests.

Top Screenshot: GET Users and POST post user tests

Requests	Tests
GET Users	<ul style="list-style-type: none">> Response status code is 200> Response time is less than 500ms> Response has the required Content-Type header with value application/json> Response body is an array> Validate the schema of the response array
POST post user	<ul style="list-style-type: none">> Response status code is 201> Response has the required fields> Name and email are in a valid format> Response time is less than 500ms

Bottom Screenshot: PUT put user and DELETE Delete user tests

Requests	Tests
PUT put user	<ul style="list-style-type: none">> Response status code is 200> Response time is less than 200ms> Response has the required fields> Email is in a valid email format> Id is a non-zero positive integer
DELETE Delete user	<ul style="list-style-type: none">> Response status code is 204> Response time is less than 200ms> Verify Content-Type is text/xml> Validate that the response body is null

Después de mucho probar y de ver nuevamente los videos de las clases grabadas el ejercicio arrojo respuestas validas por el servidor.

¡Muchas gracias!