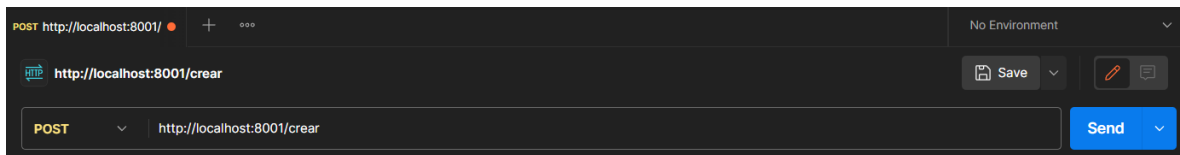


Evaluación Práctica Desarrollo de Microservicios.

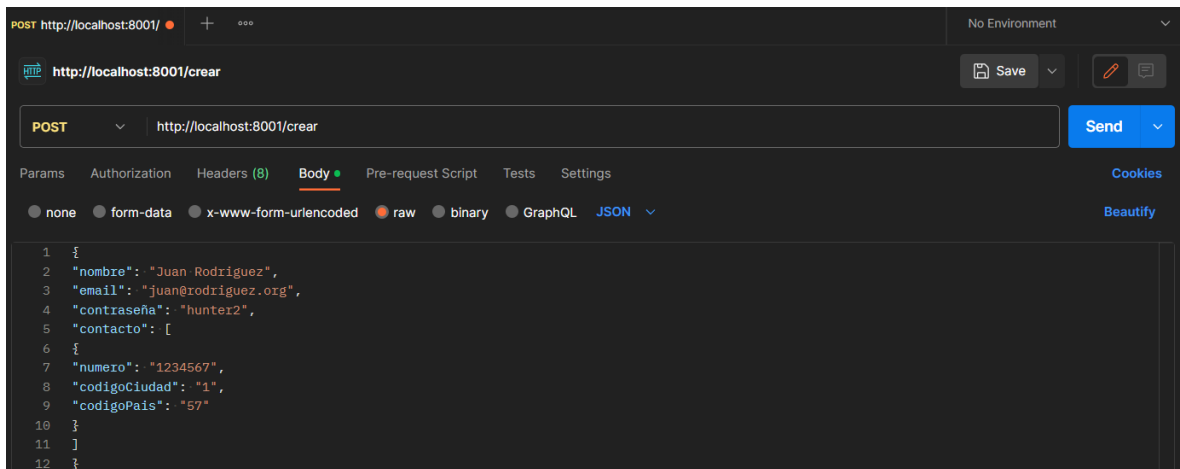
1. POST, registro de usuario.

Para ocupar correctamente la aplicación (en Postman) se debe ingresar un registro de usuario con la opción “POST” en el endpoint “<http://localhost:8001/crear>” con el siguiente formato:

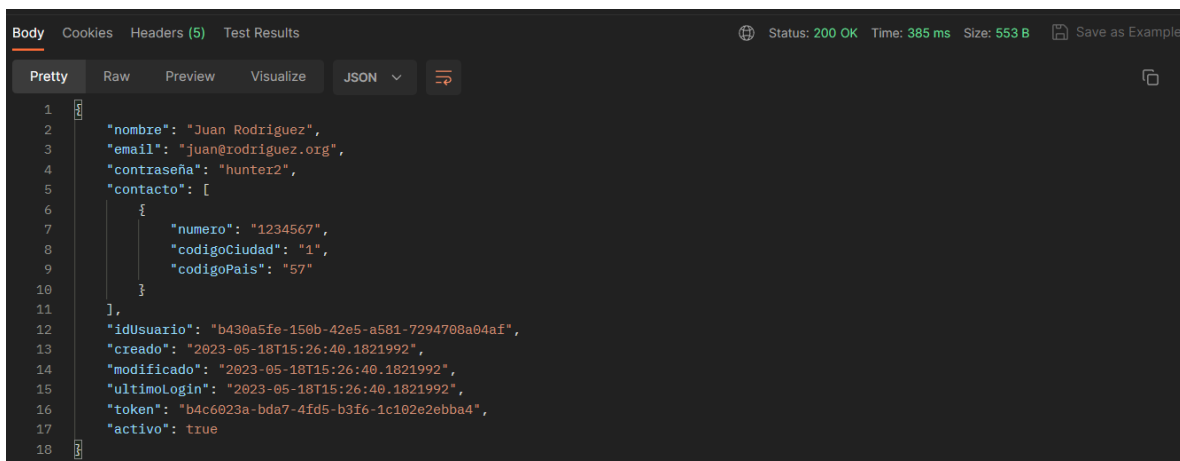
```
{
  "nombre": "Juan Rodriguez",
  "email": "juan@rodriguez.org",
  "contraseña": "hunter2",
  "contacto": [
    {
      "numero": "1234567",
      "codigoCiudad": "1",
      "codigoPais": "57"
    }
  ]
}
```



Este registro se escribe en el apartado “Body” (opción “raw”) ubicado bajo la barra del endpoint (al igual que todos los endpoint y en formato JSon):



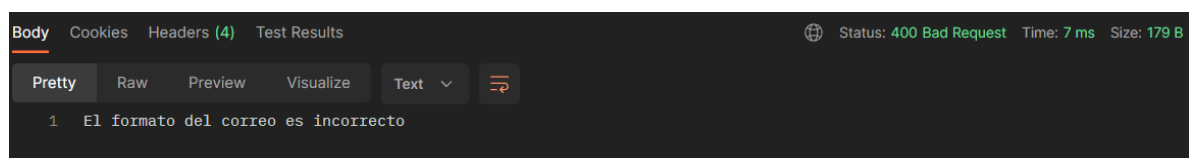
Posterior a escribir los datos se debe oprimir el botón “Send” para obtener en registro completo del usuario, lo cual devuelve el siguiente archivo JSon:



Esto contiene al usuario creado junto con el id de usuario (UUID), la fecha y hora de creación, fecha de modificación, su ultimo login, su respectivo token y si este usuario está activo.

Se debe tener en cuenta algunas consideraciones al registrar un usuario en la aplicación, tales como:

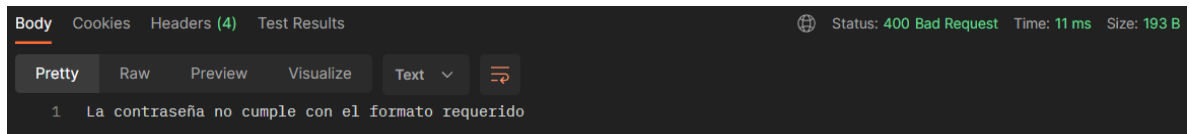
- El correo debe seguir un formato en específico (ejemplo: aaaaa@dominio.com), de no ser así, se verá el mensaje “El formato del correo es incorrecto”:



- El correo no se puede repetir, es decir, el correo no se puede registrar mas de una vez, de ser así, se verá el siguiente mensaje (“El correo ya está registrado”):



- La contraseña debe seguir un formato alfanumérico, de no ser así arroja el siguiente mensaje (“La contraseña no cumple con el formato requerido”):



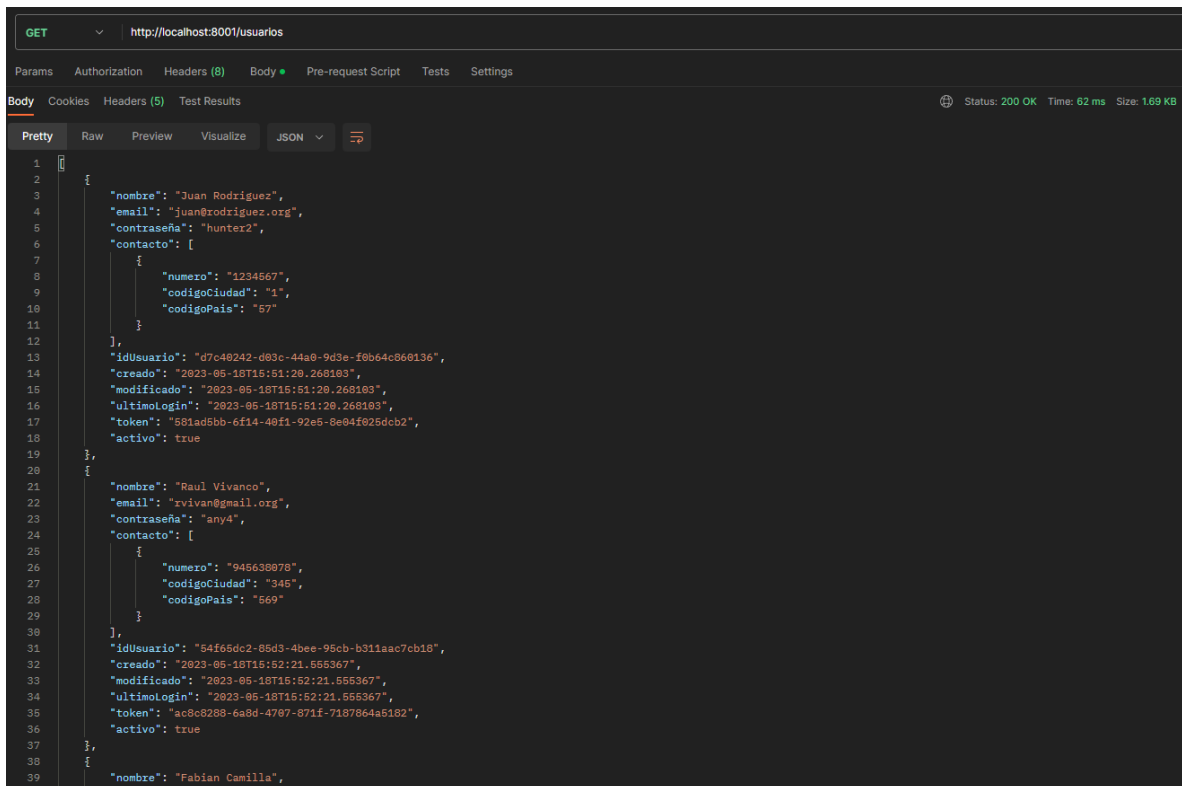
- (Adicional) La contraseña deberá ser nueva, es decir, que no se repita entre usuarios, si no se verá (“Contraseña en uso, opte por otra.”):



Si no hay ningún problema en el registro del usuario, este quedará guardado en la base de datos y podrá ser llamado (GET), editado (PUT) e incluso eliminado (DELETE).

2. GET, ver la lista de usuarios.

Con la opción “GET” y en el endpoint “<http://localhost:8001/usuarios>” (o por id con “usuario/{id}”) se puede ver la lista de usuarios registrados en la base de datos:



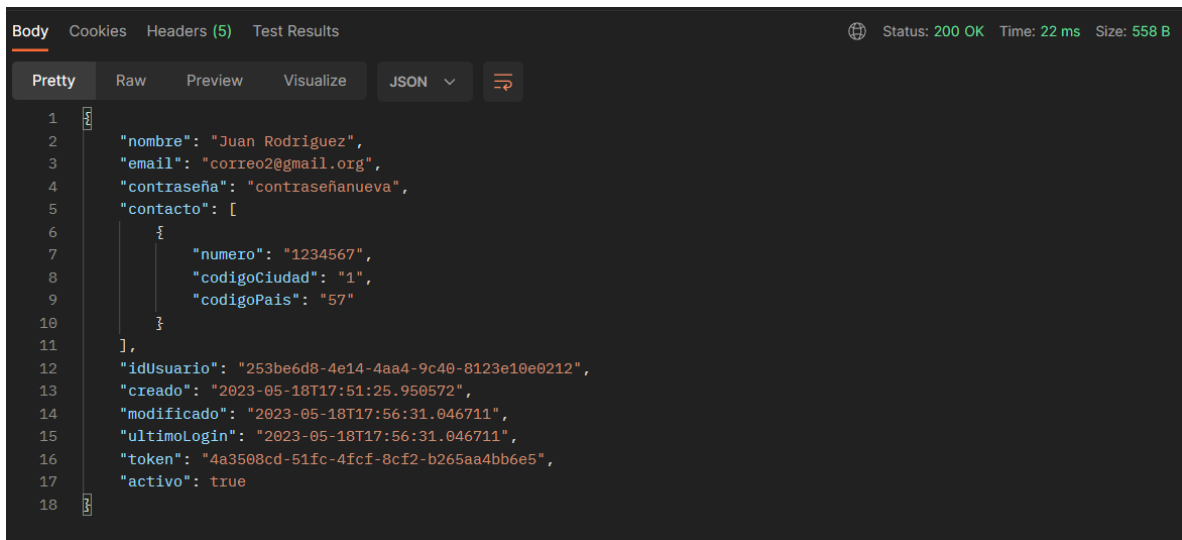
```
1 {
2   {
3     "nombre": "Juan Rodriguez",
4     "email": "juan@rodriguez.org",
5     "contraseña": "hunter2",
6     "contacto": [
7       {
8         "numero": "1234567",
9         "codigoCiudad": "1",
10        "codigoPais": "57"
11      }
12    ],
13    "idUsuario": "d7c40242-d03c-44a0-9d3e-f0b64c860136",
14    "creado": "2023-05-10T15:51:20.260103",
15    "modificado": "2023-05-10T15:51:20.260103",
16    "ultimoLogin": "2023-05-10T15:51:20.260103",
17    "token": "581ad5bb-6f14-40f1-92e5-8e04f025dcb2",
18    "activo": true
19  },
20  {
21    "nombre": "Raul Vivanco",
22    "email": "rvivan@gmail.org",
23    "contraseña": "any4",
24    "contacto": [
25      {
26        "numero": "945638078",
27        "codigoCiudad": "345",
28        "codigoPais": "569"
29      }
30    ],
31    "idUsuario": "54f65dc2-85d3-4bee-95cb-b311aac7cb18",
32    "creado": "2023-05-10T15:52:21.555367",
33    "modificado": "2023-05-10T15:52:21.555367",
34    "ultimoLogin": "2023-05-10T15:52:21.555367",
35    "token": "ac8c8288-6a8d-4707-871f-7187864a5182",
36    "activo": true
37  },
38  {
39    "nombre": "Fabian Camilla",
40    "email": "fcamilla@gmail.org",
41    "contraseña": "123456",
42    "contacto": [
43      {
44        "numero": "1234567",
45        "codigoCiudad": "1",
46        "codigoPais": "57"
47      }
48    ],
49    "idUsuario": "12345678-9abc-def0-1234-56789abcde",
50    "creado": "2023-05-10T15:51:20.260103",
51    "modificado": "2023-05-10T15:51:20.260103",
52    "ultimoLogin": "2023-05-10T15:51:20.260103",
53    "token": "12345678-9abc-def0-1234-56789abcde",
54    "activo": true
55  }
56 ]
```

3. PUT, editar usuario.

Para actualizar los datos de algún usuario se debe usar el método PUT, con el endpoint [“http://localhost:8001/editar/{id}”](http://localhost:8001/editar/{id}), en {id} se coloca el numero de orden correspondiente en la base de datos (1, 2, 3...):



```
1 {
2   "nombre": "Juan Rodriguez",
3   "email": "correo2@gmail.org",
4   "contraseña": "contraseñanueva",
5   "contacto": [
6     {
7       "numero": "1234567",
8       "codigoCiudad": "1",
9       "codigoPais": "57"
10    }
11  ]
12 }
```

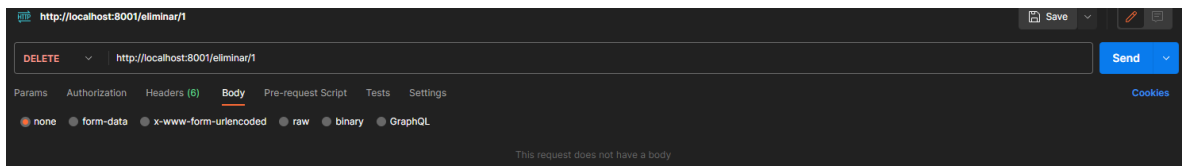


The screenshot shows a REST client interface with the 'Body' tab selected. The response is a JSON object representing a newly created user. The status is 200 OK, the time taken is 22 ms, and the size is 558 B. The JSON data is as follows:

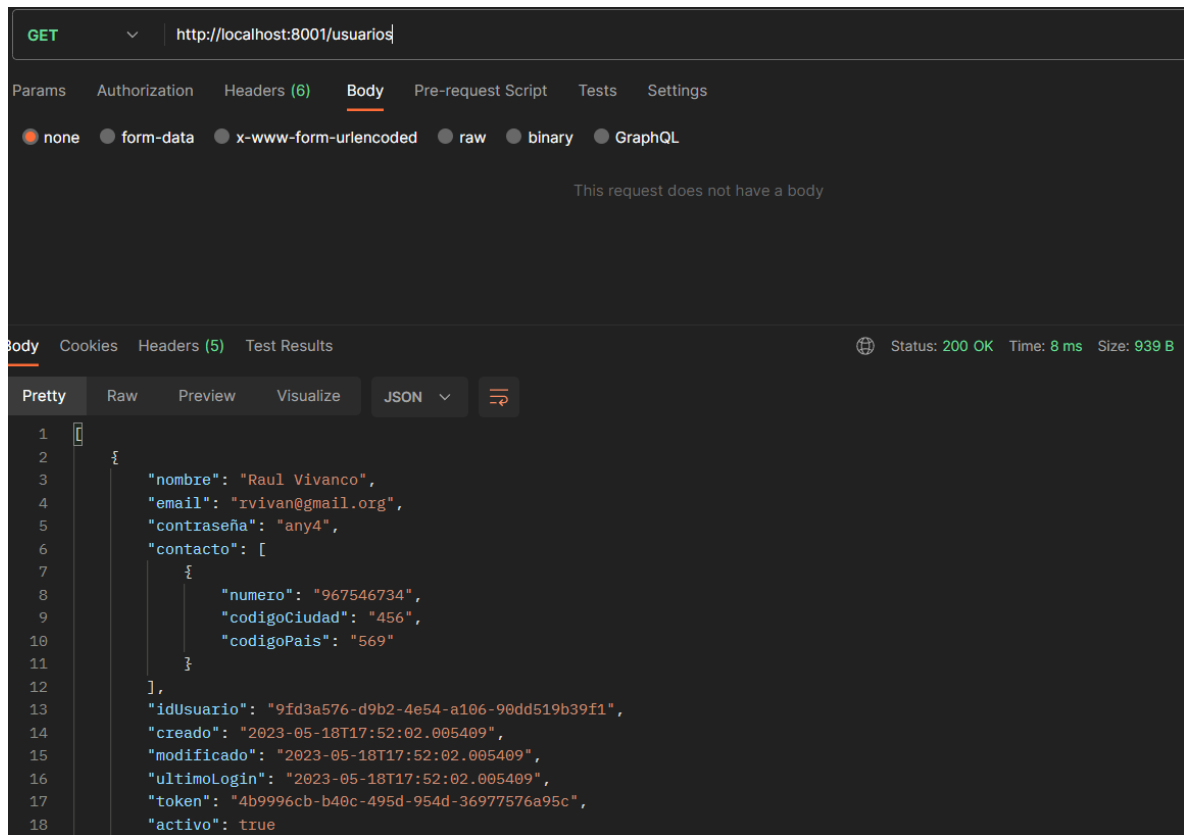
```
1  {
2    "nombre": "Juan Rodriguez",
3    "email": "correo2@gmail.org",
4    "contraseña": "contraseñanueva",
5    "contacto": [
6      {
7        "numero": "1234567",
8        "codigoCiudad": "1",
9        "codigoPais": "57"
10     }
11  ],
12  "idUsuario": "253be6d8-4e14-4aa4-9c40-8123e10e0212",
13  "creado": "2023-05-18T17:51:25.950572",
14  "modificado": "2023-05-18T17:56:31.046711",
15  "ultimoLogin": "2023-05-18T17:56:31.046711",
16  "token": "4a3508cd-51fc-4fcf-8cf2-b265aa4bb6e5",
17  "activo": true
18 }
```

4. DELETE, eliminar usuario.

Para eliminar a un usuario se debe usar la opción “DELETE”, junto al endpoint “<http://localhost:8001/eliminar/{id}>”, de este modo el usuario es borrado de la base de datos:



Cabe destacar que bajo la barra del endpoint se marca la opción “none”, ya que no se necesita registro JSon; al enviar, se elimina el usuario y desaparece de la lista:



En el ejemplo, “Raul Vivanco” pasa a primer lugar de la lista de usuarios al no existir “Juan Rodriguez”.