Práctica RESTful

Sistemas Orientados a Servicios

DLSIIS

Vidal Peña, Arturo
Melero Deza, Javier

10 de julio de 2019

Índice de contenidos

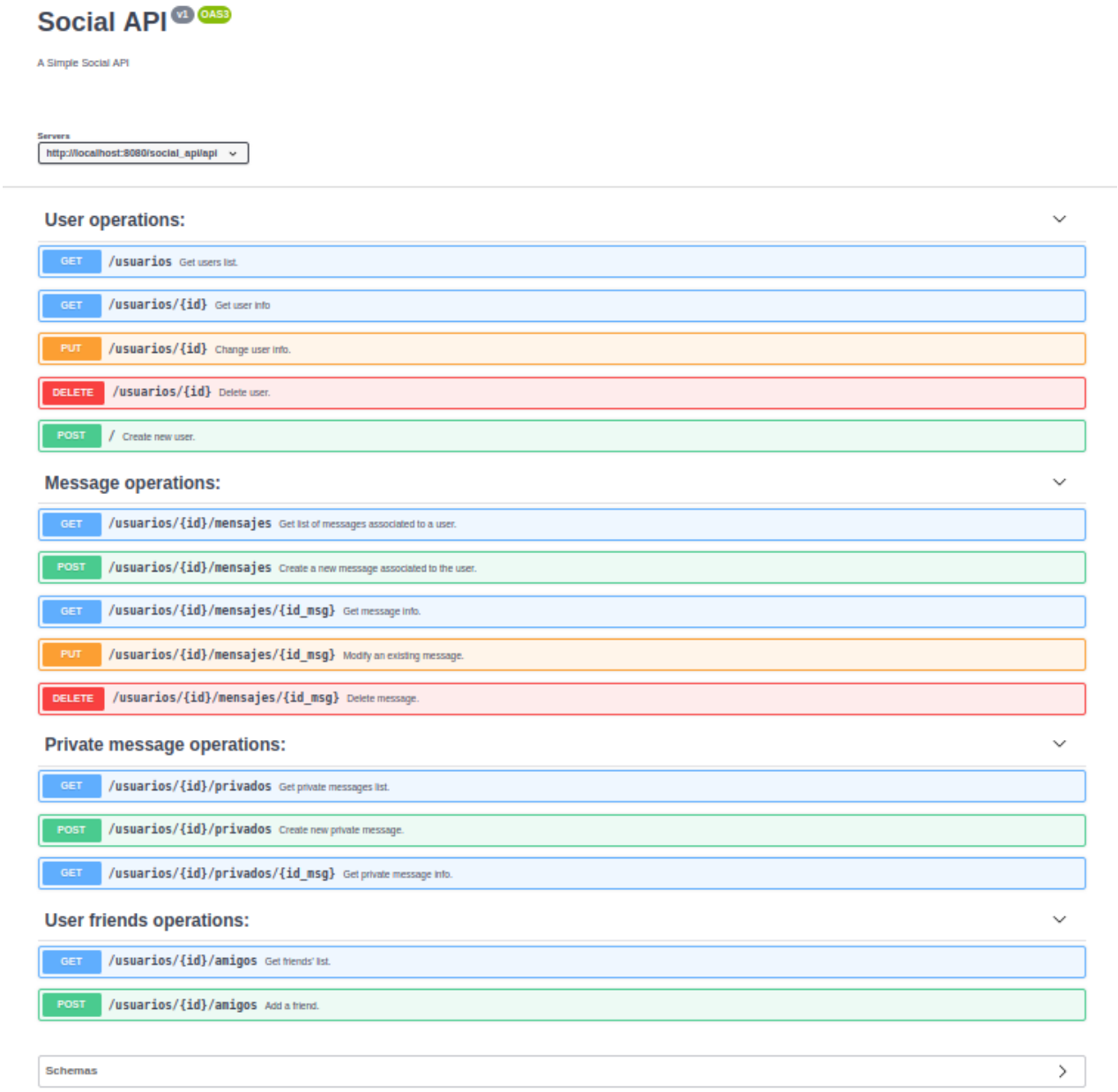
1 Resumen del diseño del servicio **1**
1.1 Métodos de usuarios 2
1.1.1 Crear un usuario 2
1.1.2 Obtener info del usuario 3
1.1.3 Obtener todos los usuarios 4
1.1.4 Modificar un usuario 5
1.1.5 Eliminar un usuario 6
1.2 Métodos de amistad 7
1.2.1 Crear una amistad 7
1.2.2 Obtener amigos de un usuario 8
1.3 Métodos de mensajes 9
1.3.1 Crear un mensaje 9
1.3.2 Obtener info del mensaje 10
1.3.3 Obtener todos los mensajes de un usuario 11
1.3.4 Modificar un mensaje 12
1.3.5 Eliminar un mensaje 13
1.4 Métodos de mensajes privados 14
1.4.1 Crear un mensaje privado 14
1.4.2 Obtener info del mensaje privado 15
1.4.3 Obtener todos los mensajes privados de un usuario 16
1.5 Fichero YAML del servicio 17
1.5.1 Definición de los métodos de usuario 18
1.5.2 Definición de los métodos de mensajes 22
1.5.3 Definición de los métodos de mensajes privados 27
1.5.4 Definición de los métodos de amistades 30
1.5.5 Definición de los esquemas 32

2 Capturas de la ejecución en el cliente Postman **34**
2.1 Métodos de usuarios 34
2.1.1 POST 34

2.1.2 GET . . . 36
2.1.3 PUT . . . 37
2.1.4 DELETE . . . 38
2.2 Métodos de mensajes . . . 39
2.2.1 POST . . . 39
2.2.2 GET . . . 41
2.2.3 PUT . . . 42
2.2.4 DELETE . . . 43
2.3 Métodos de Mensajes Privados . . . 44
2.3.1 POST . . . 44
2.3.2 GET . . . 44
2.4 Métodos de Amistad . . . 46
2.4.1 POST . . . 46
2.4.2 GET . . . 47
2.4.3 DELETE . . . 48
2.5 Obtención de mensajes similar a FaceBook . . . 49
2.6 Aplicación móvil . . . 50

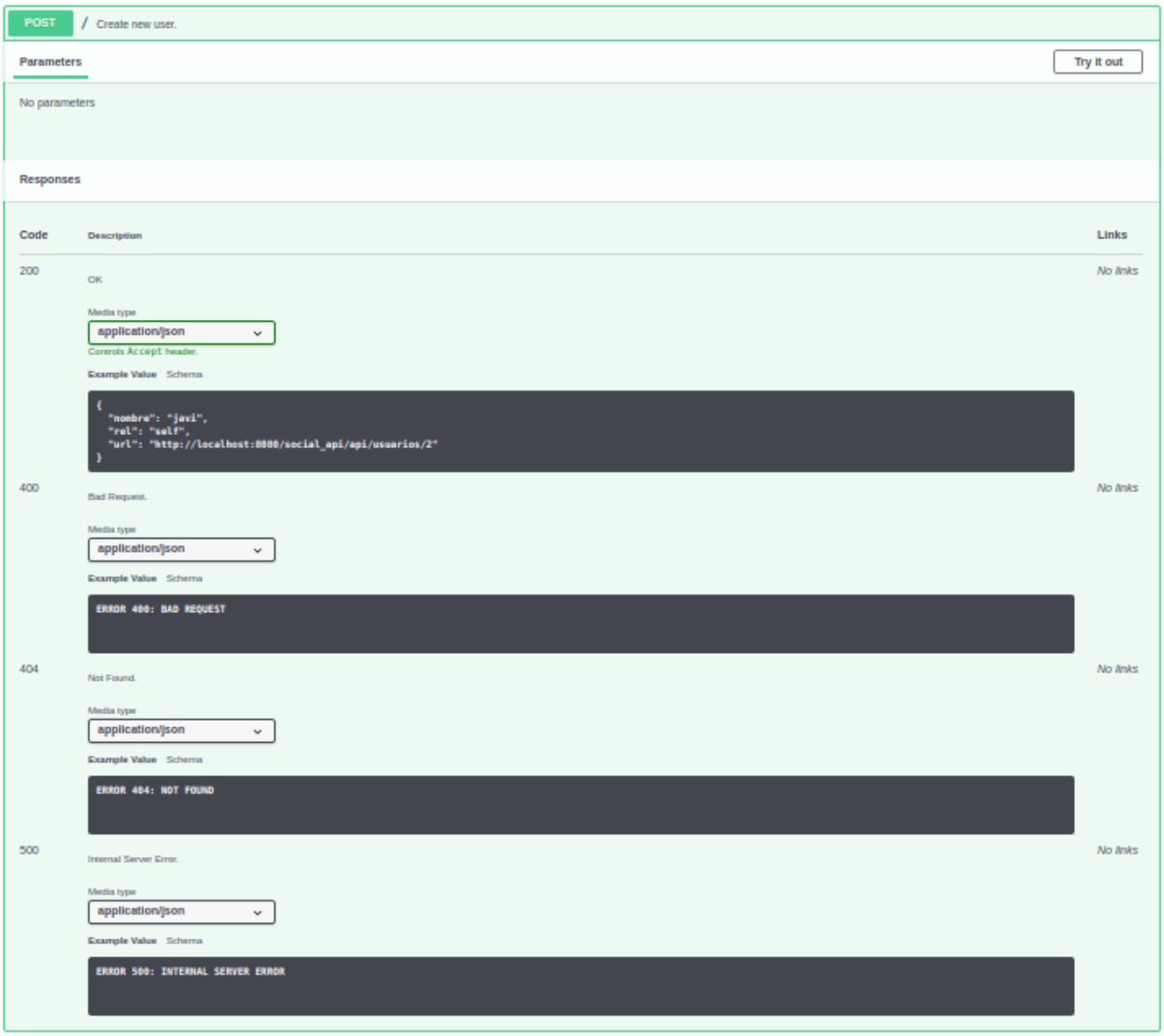
3 ANEXO: Problemas encontrados . . . 51

1. Resumen del diseño del servicio

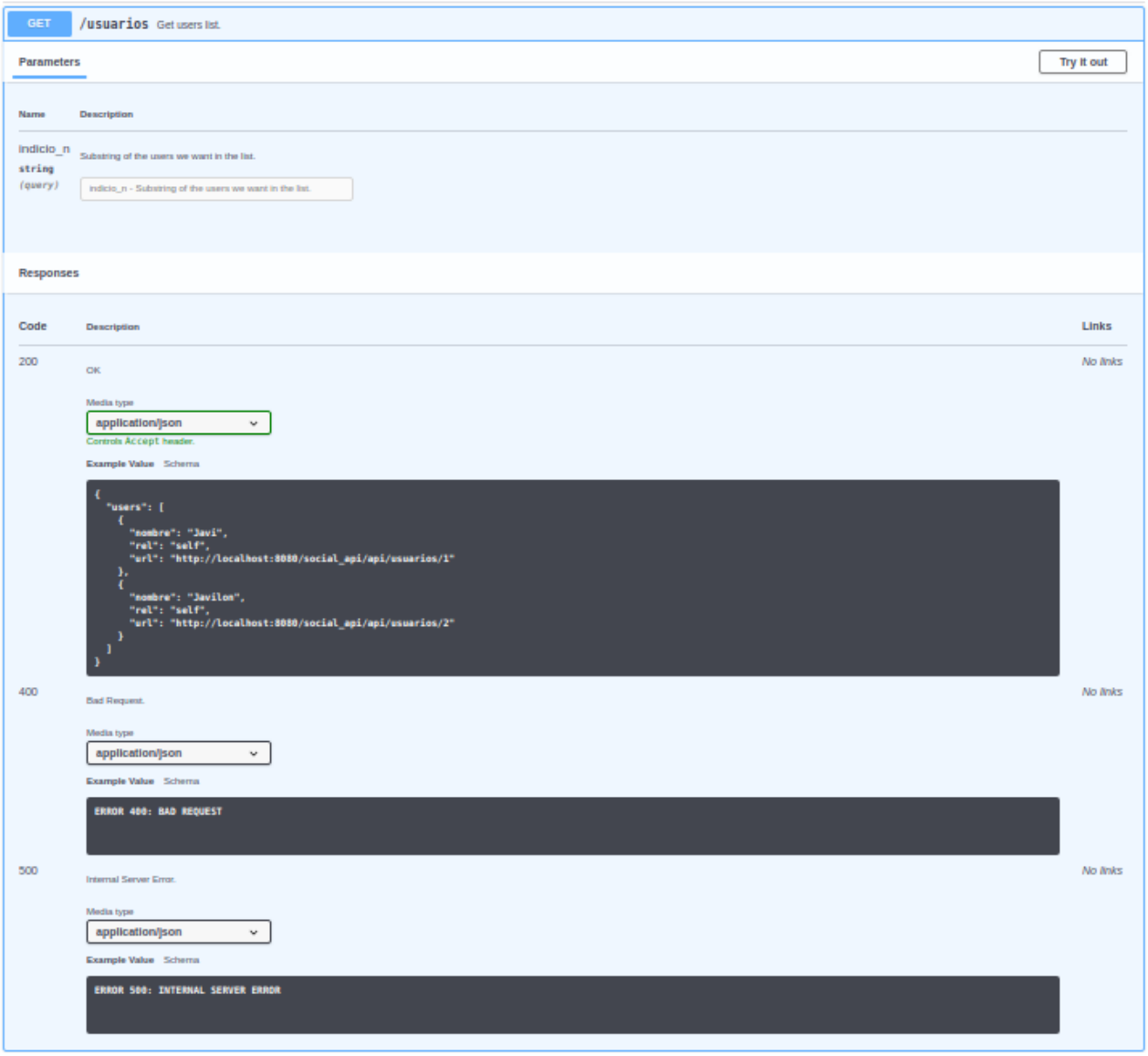


1.1. Métodos de usuarios

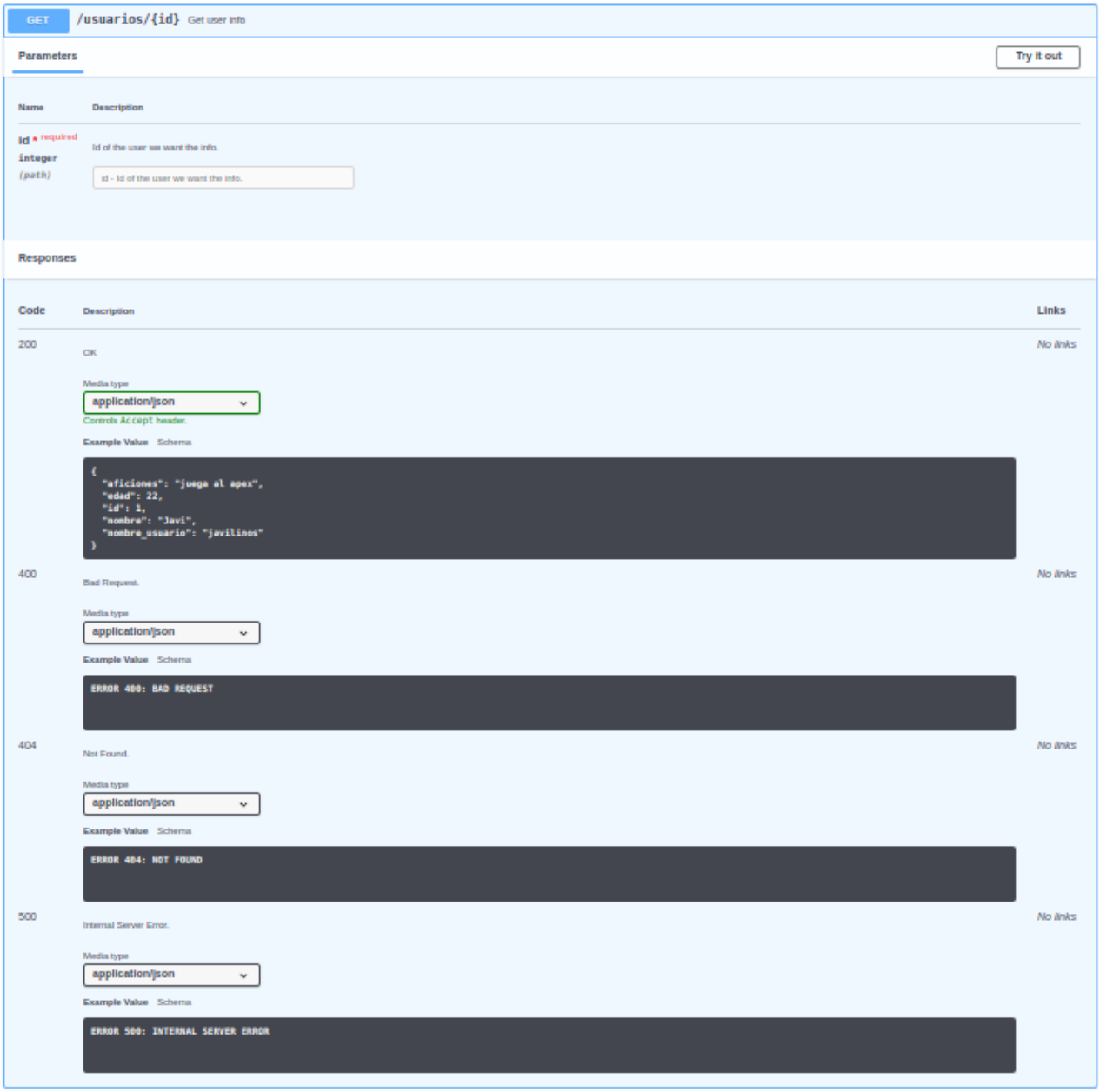
1.1.1. Crear un usuario



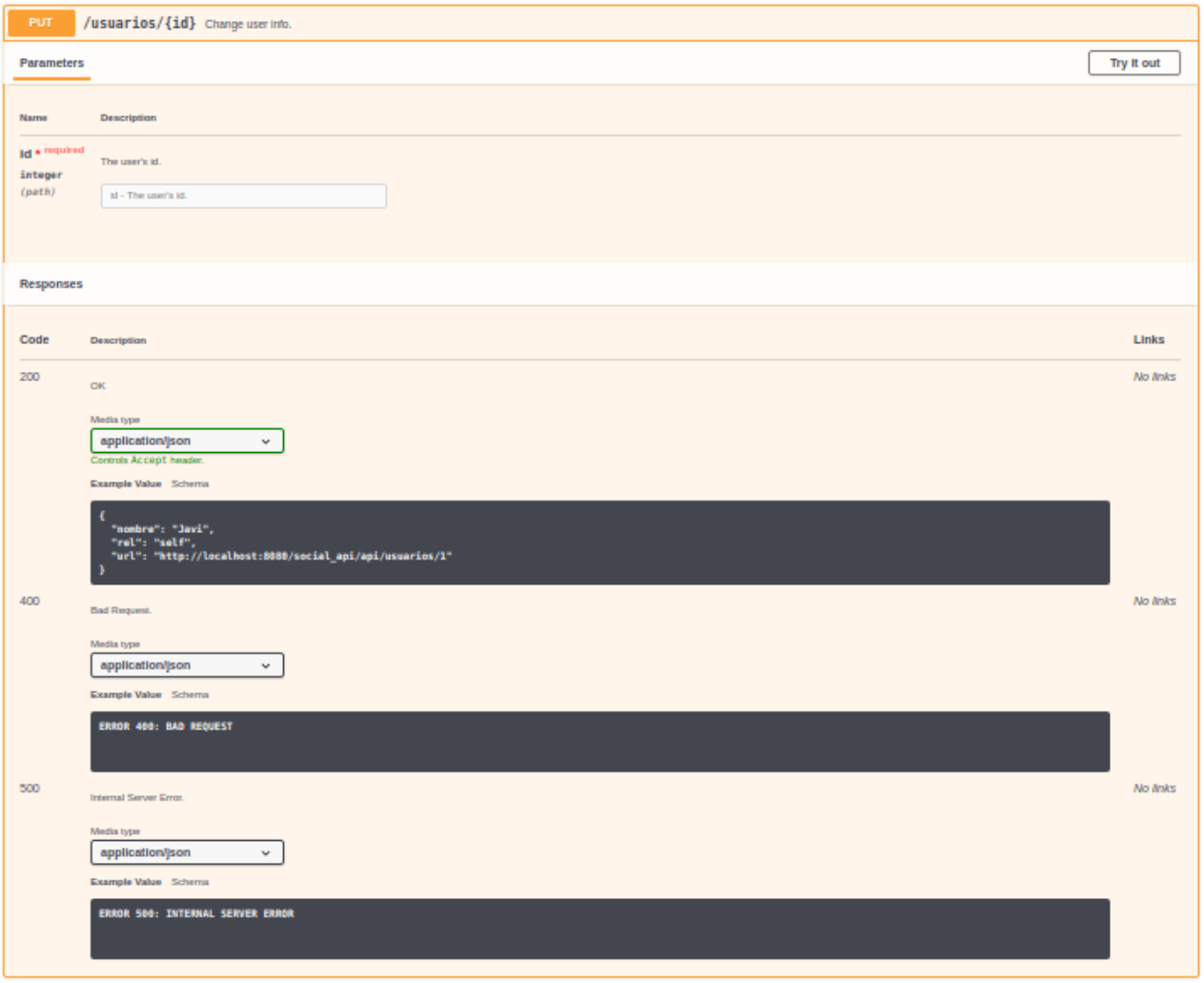
1.1.2. Obtener info del usuario



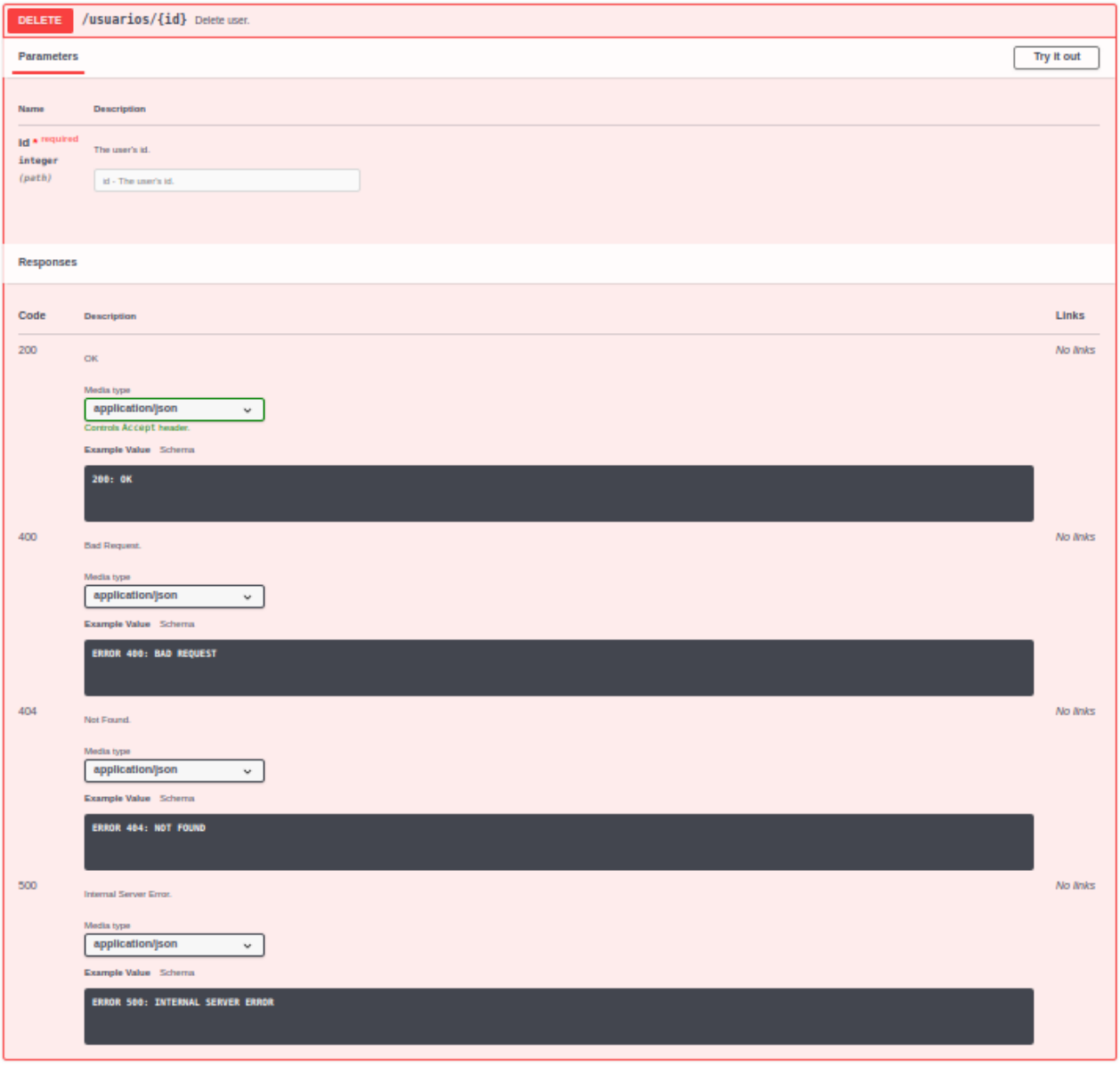
1.1.3. Obtener todos los usuarios



1.1.4. Modificar un usuario

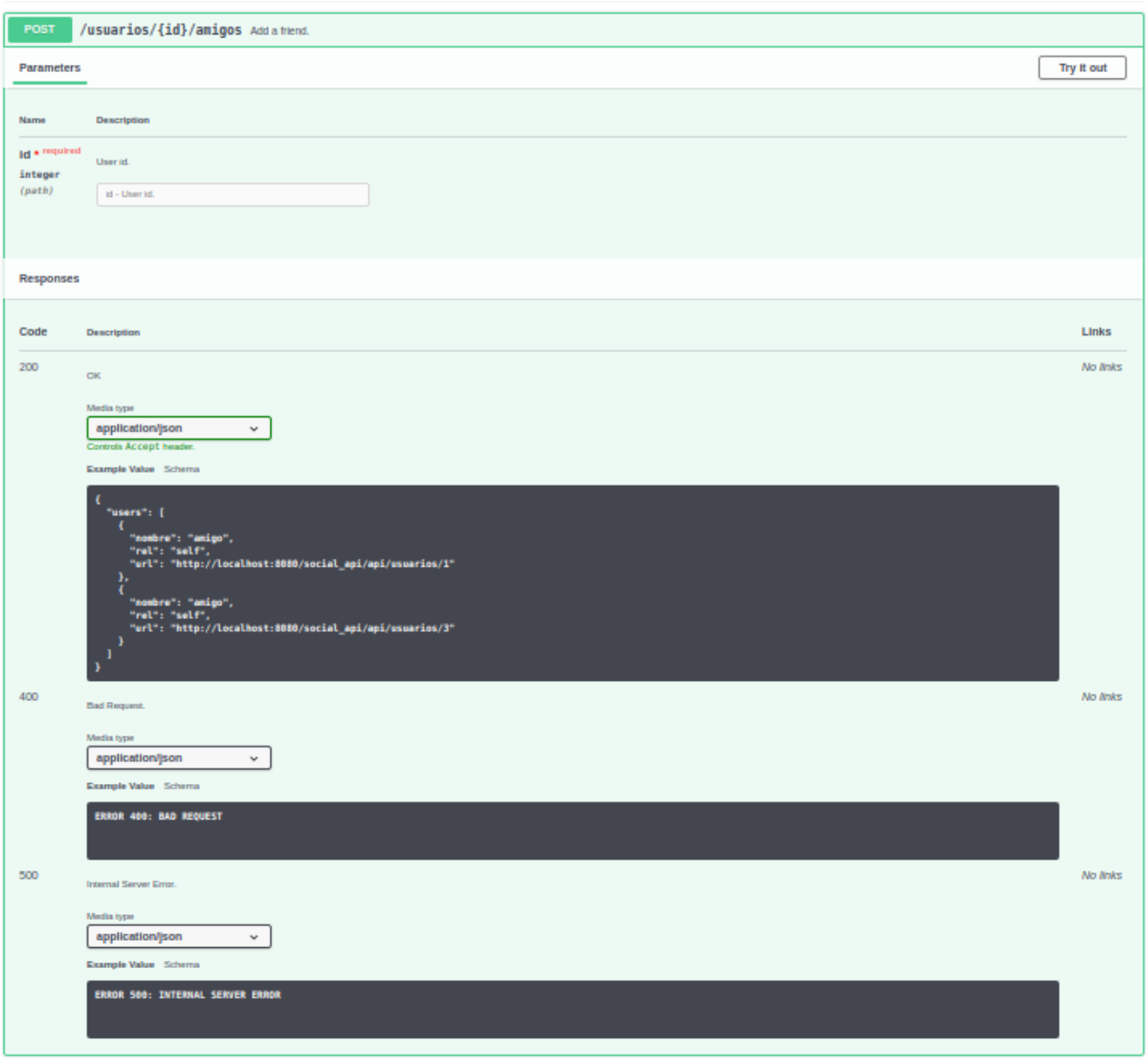


1.1.5. Eliminar un usuario

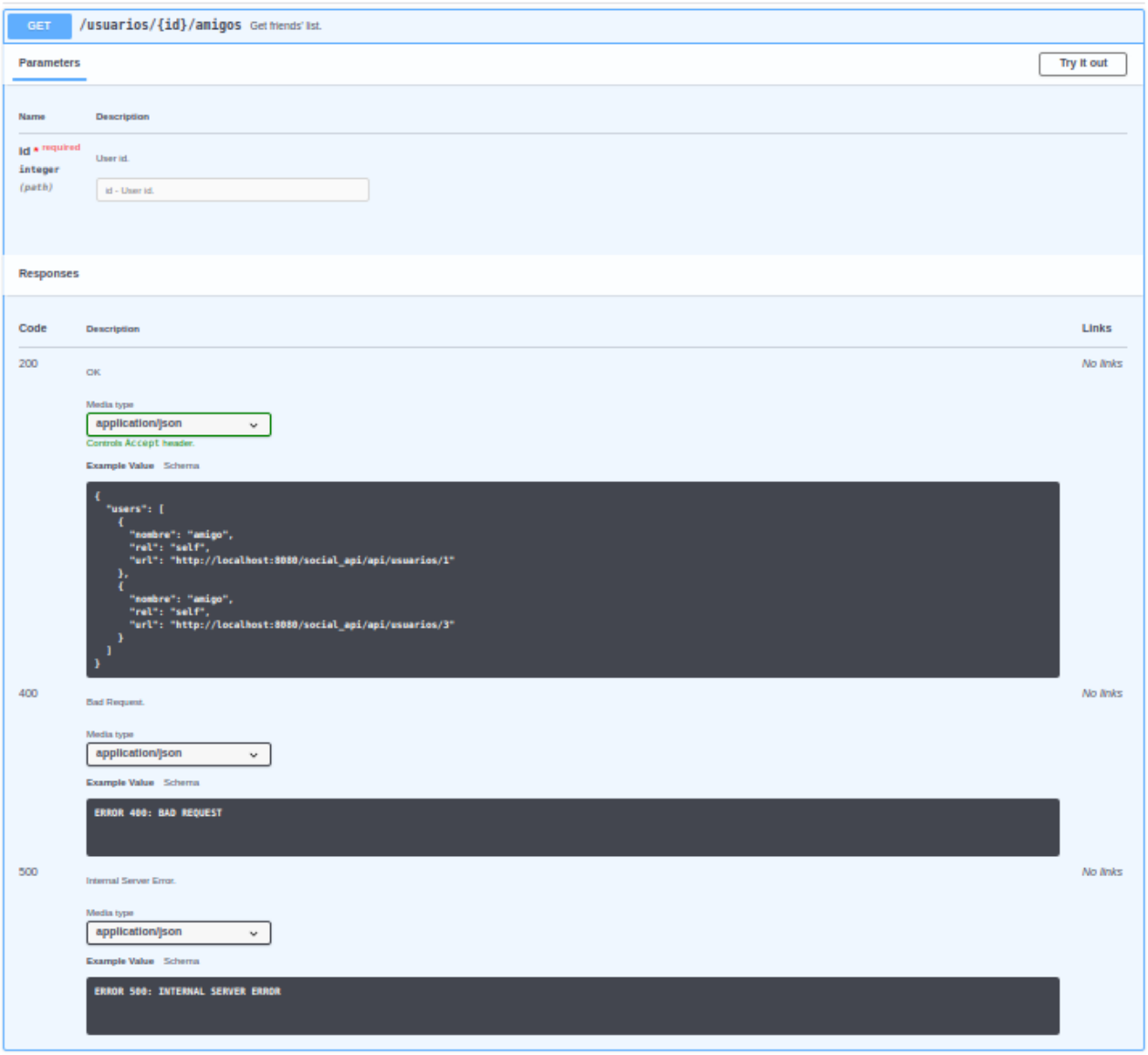


1.2. Métodos de amistad

1.2.1. Crear una amistad

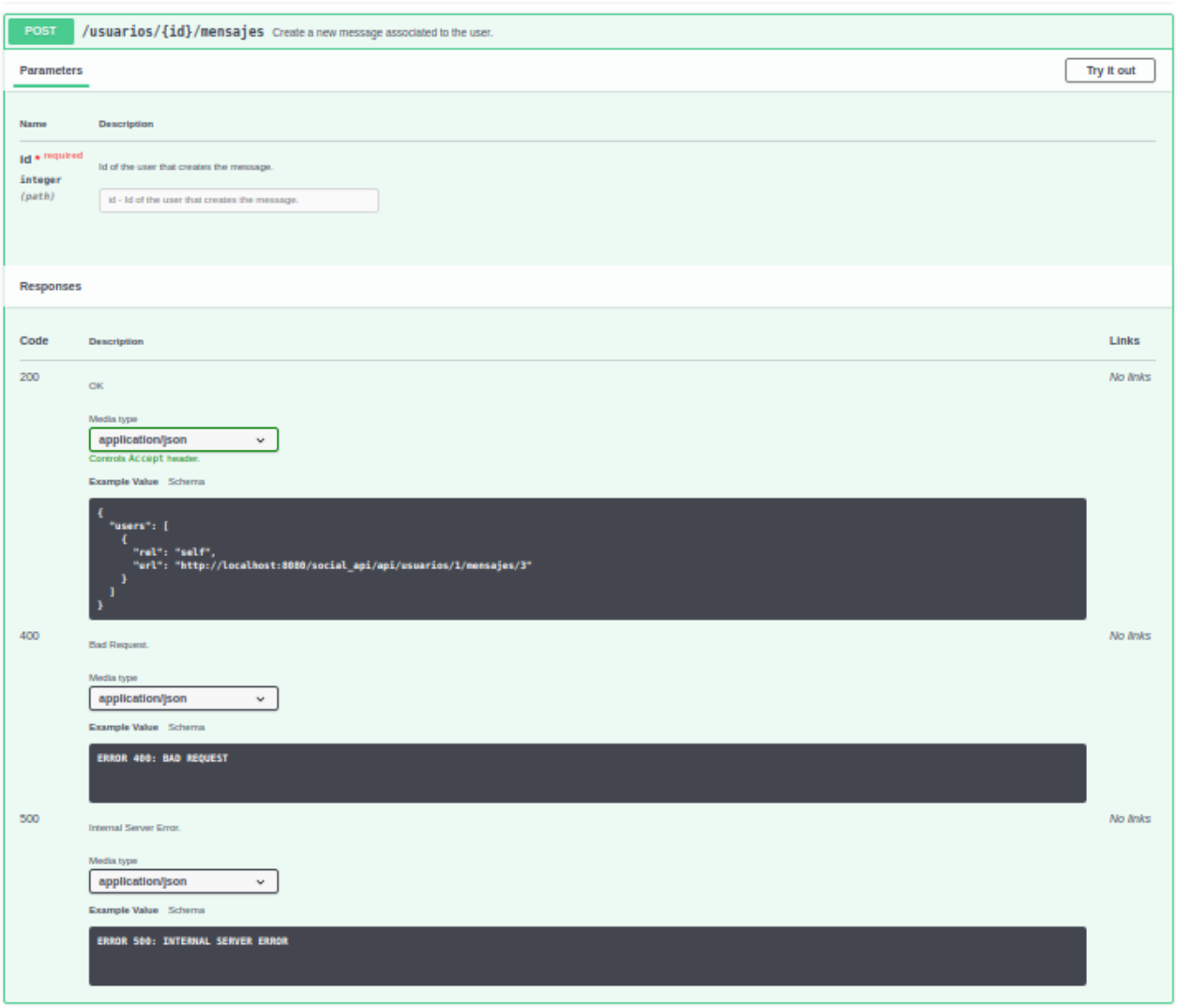


1.2.2. Obtener amigos de un usuario

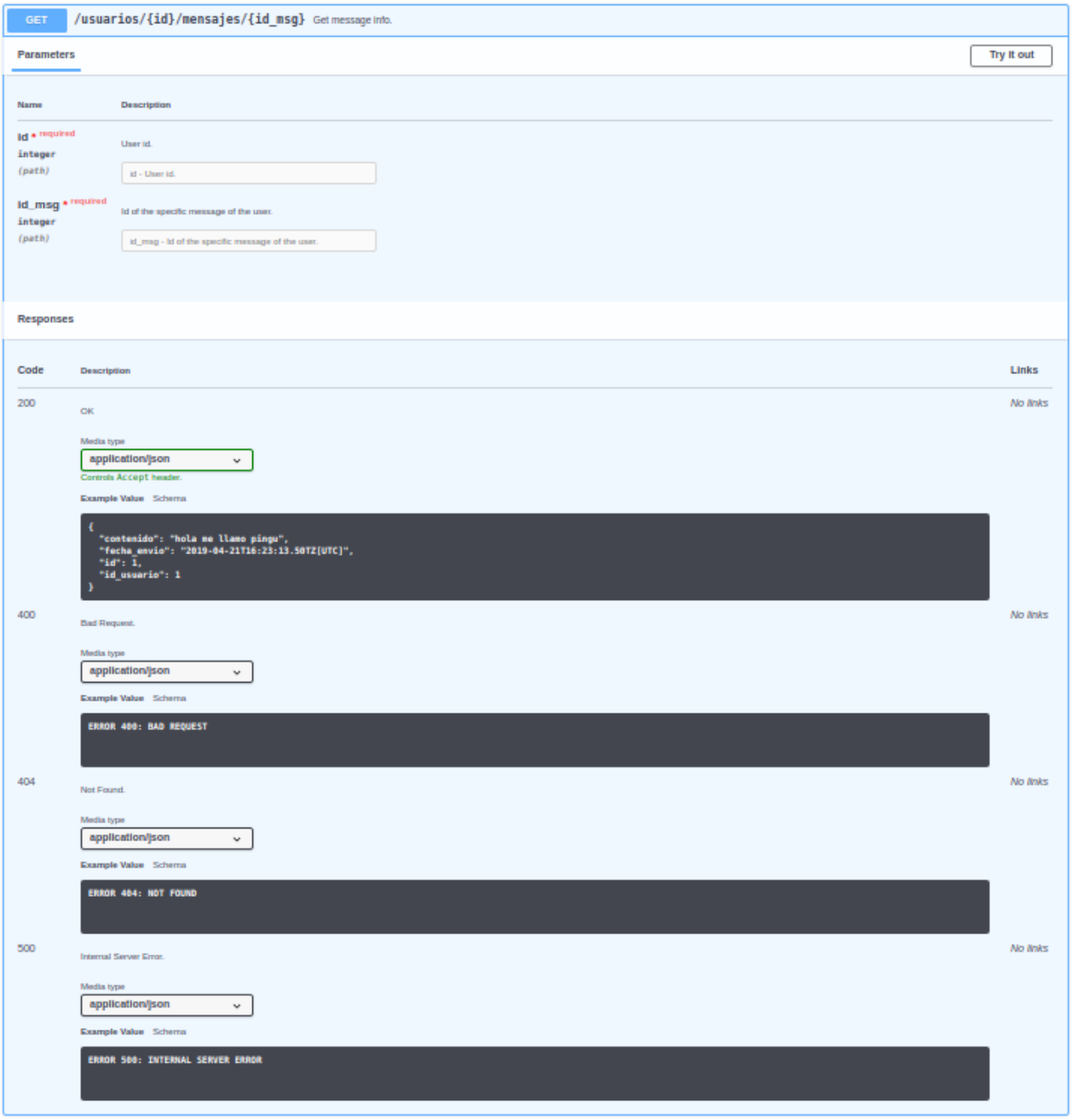


1.3. Métodos de mensajes

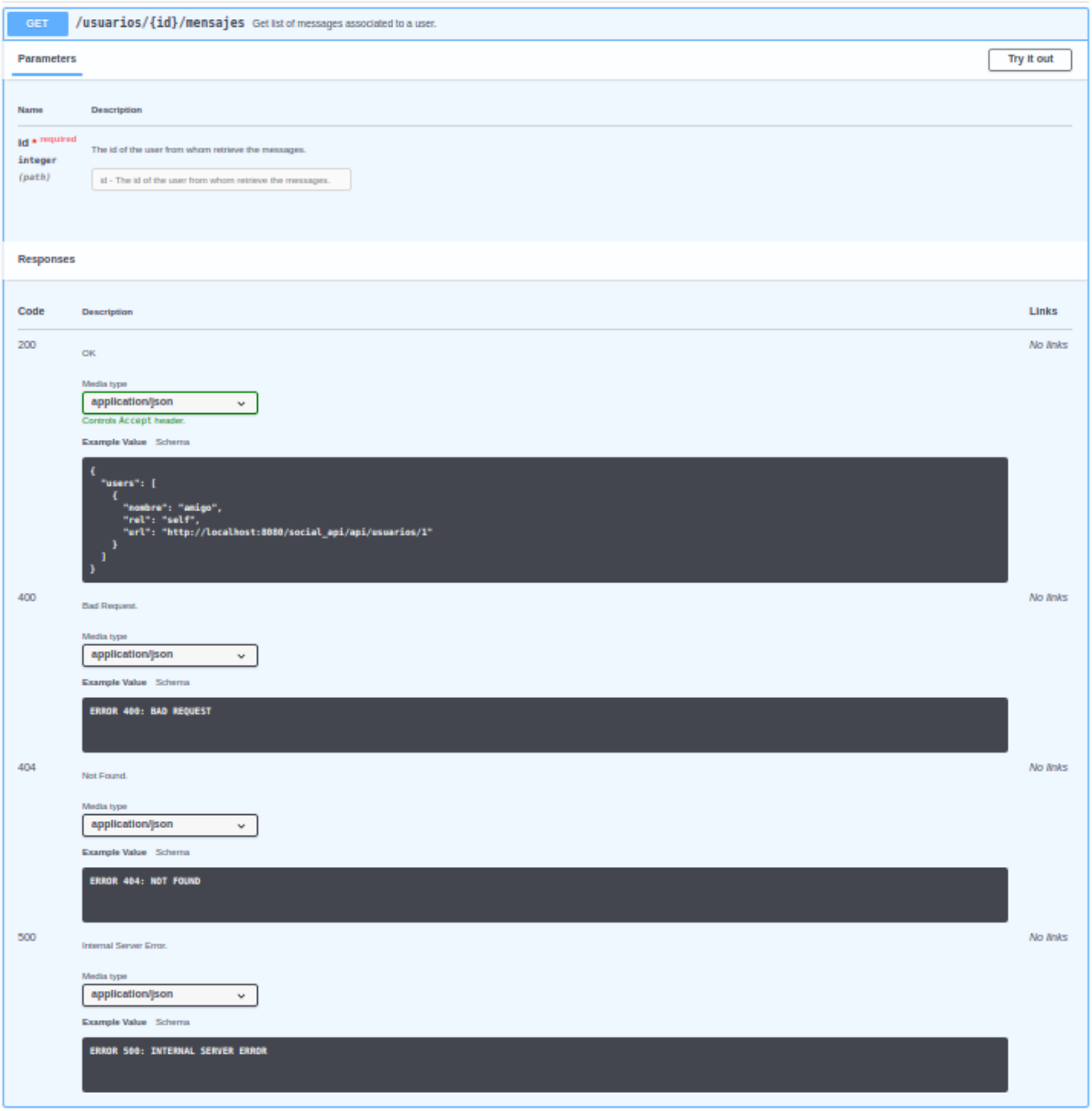
1.3.1. Crear un mensaje



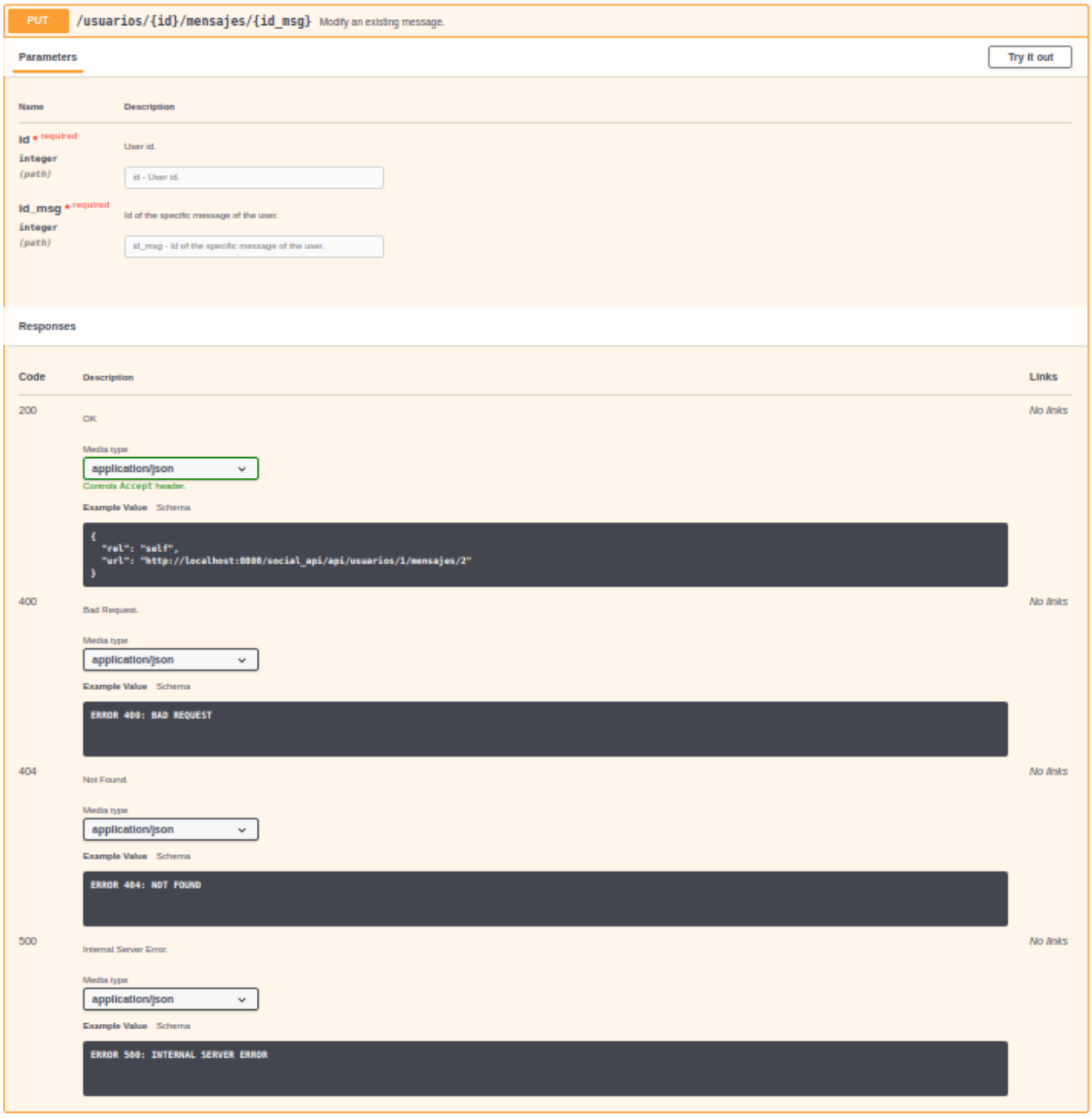
1.3.2. Obtener info del mensaje



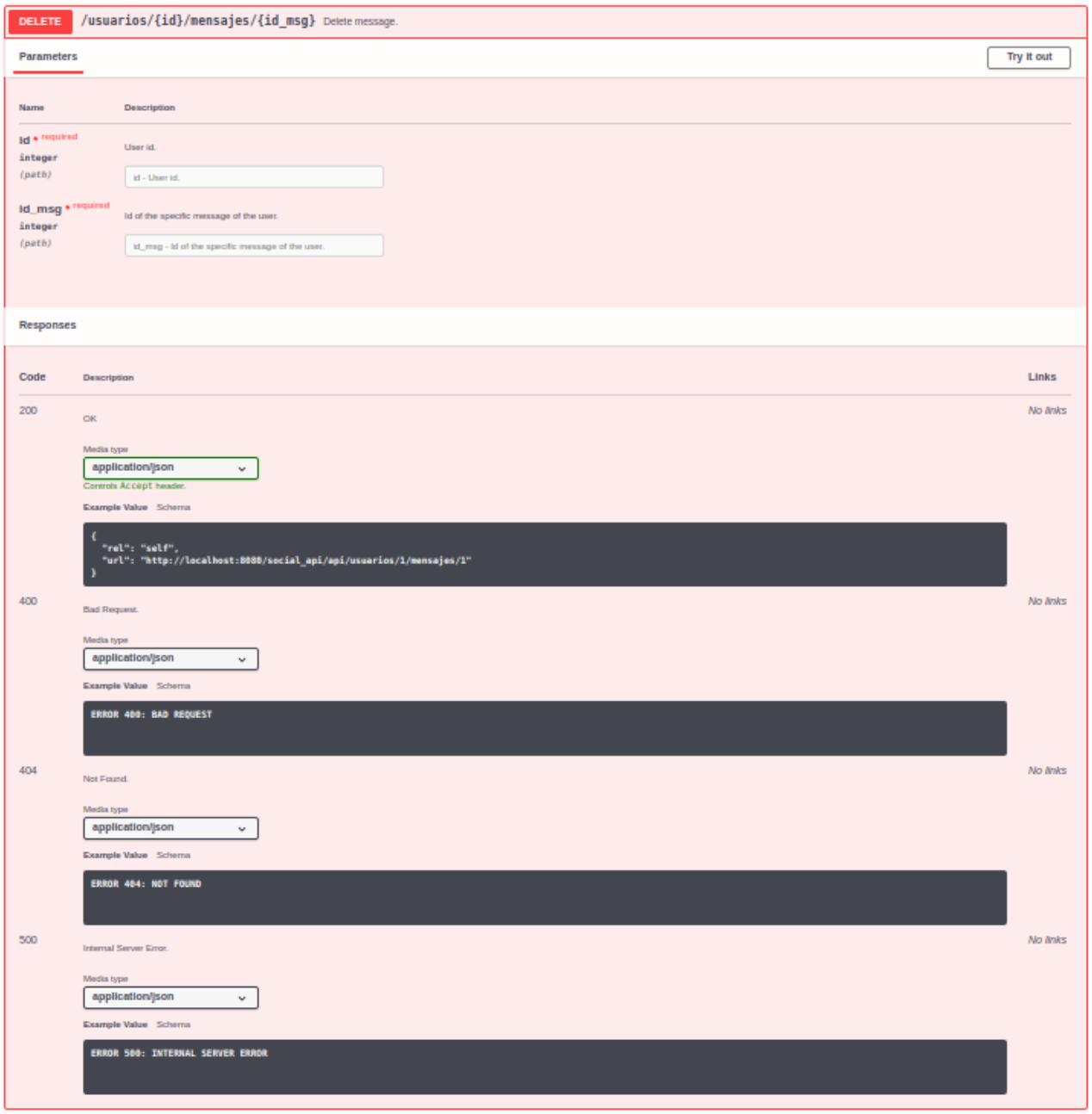
1.3.3. Obtener todos los mensajes de un usuario



1.3.4. Modificar un mensaje

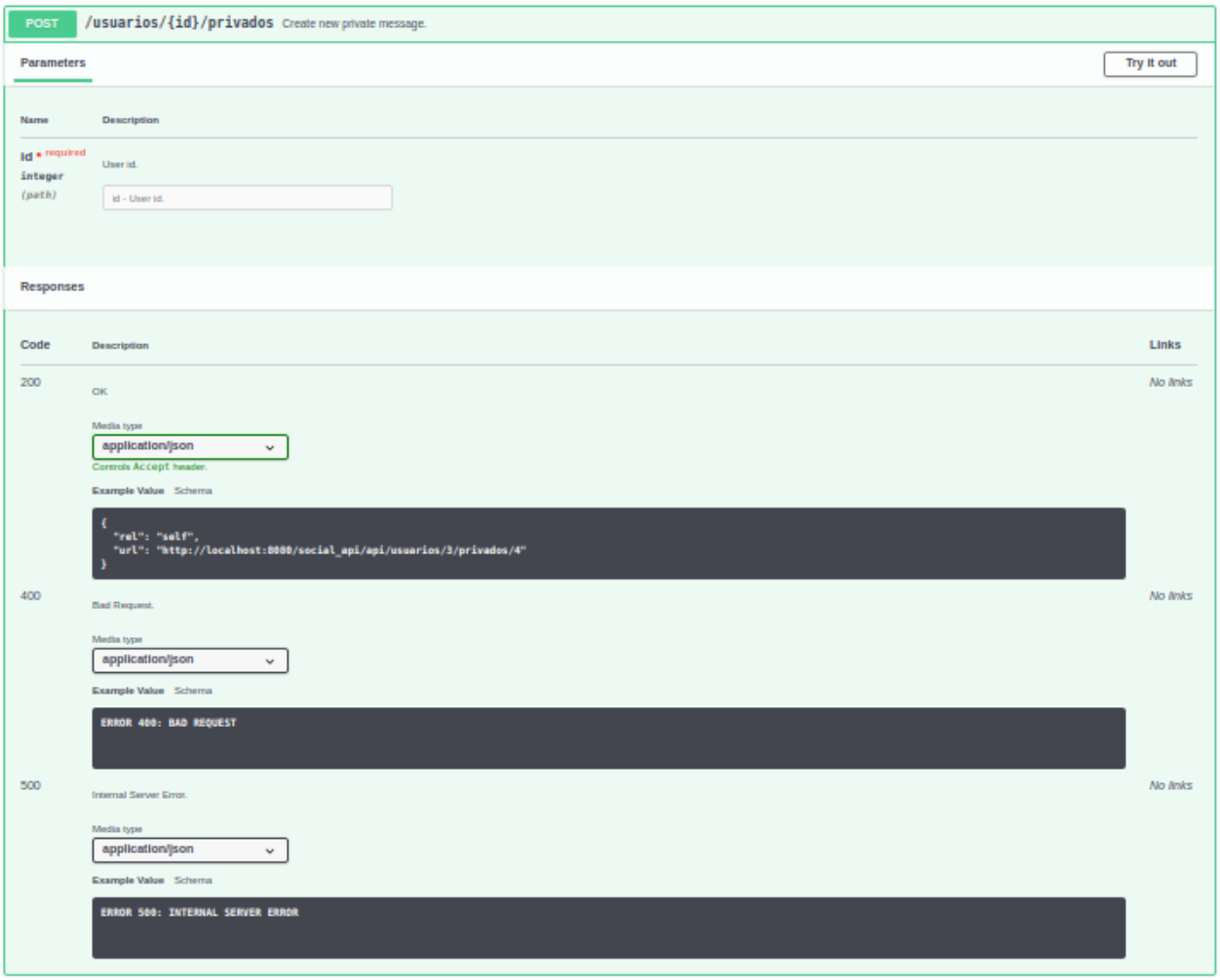


1.3.5. Eliminar un mensaje

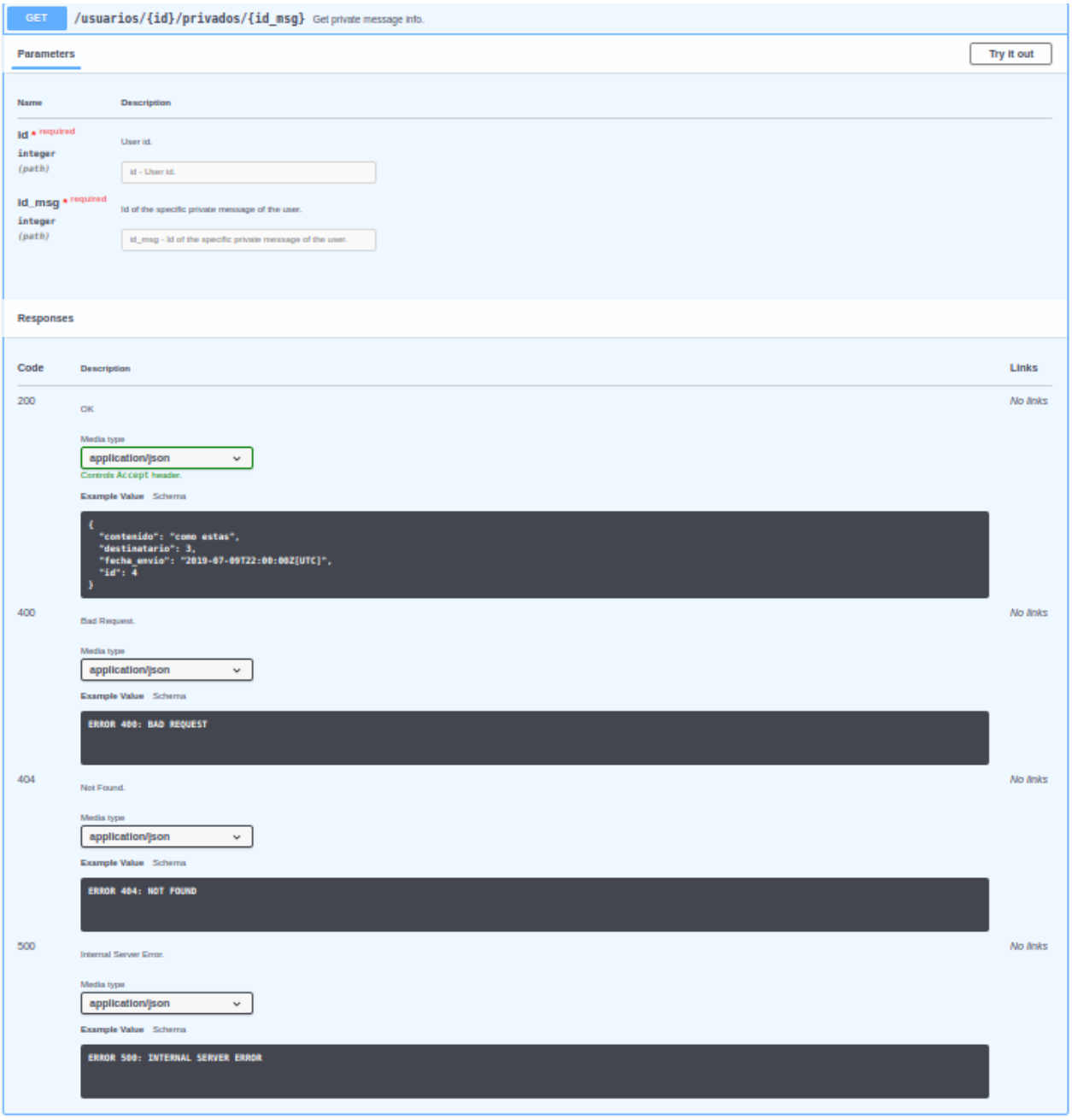


1.4. Métodos de mensajes privados

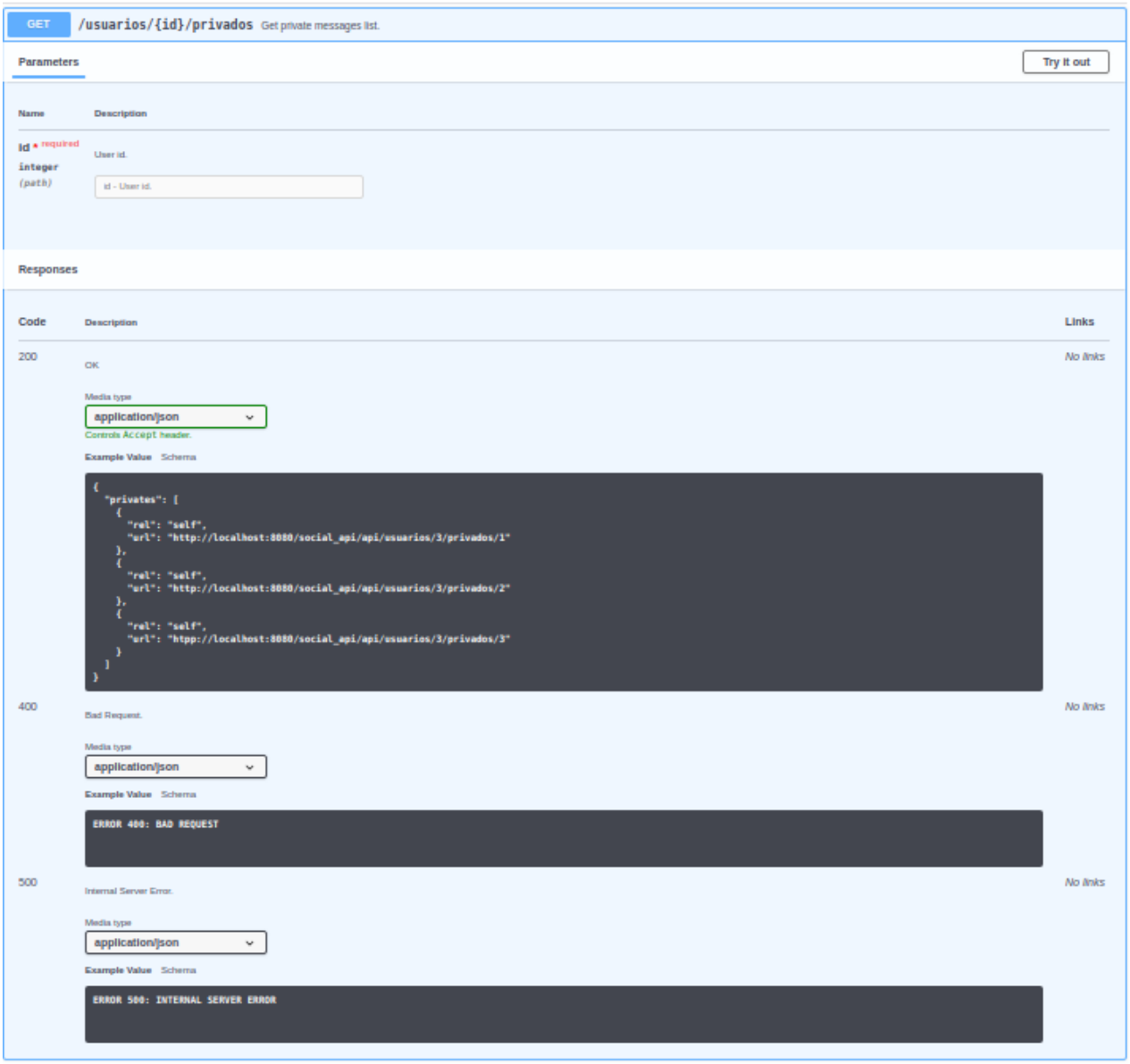
1.4.1. Crear un mensaje privado



1.4.2. Obtener info del mensaje privado



1.4.3. Obtener todos los mensajes privados de un usuario



1.5. Fichero YAML del servicio

El siguiente código se ha realizado usando la herramienta Swagger Editor. El fichero se encuentra adjunto a este documento y las capturas previas.

```yaml
openapi: '3.0.0'
info:
  description: A Simple Social API
  title: Social API
  version: v1

servers:
  - url: 'http://localhost:8080/social_api/api'

paths:
```

1.5.1. Definición de los métodos de usuario

```
/usuarios:
  get:
    parameters:
      - name: indicio_n
        in: query
        description: "Substring of the users we want in
           the list."
        schema:
          type: string
    tags:
      - "User operations:"
    summary: "Get users list."
    responses:
      '200':
        description: OK
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Users'
            example:
              users:
              - nombre: "Javi"
                rel: "self"
                url: "http://localhost:8080/social_api/
                   api/usuarios/1"
              - nombre: "Javilon"
                rel: "self"
                url: "http://localhost:8080/social_api/
                   api/usuarios/2"

      '400':
        description: Bad Request.
        content:
          application/json:
            example: 'ERROR 400: BAD REQUEST'
      '500':
        description: Internal Server Error.
        content:
          application/json:
            example: 'ERROR 500: INTERNAL SERVER ERROR'

/usuarios/{id}:
  get:
    parameters:
      - name: id
        in: path
        description: 'Id of the user we want the info.'
```

```yaml
          schema:
            type: integer
          required: true
      tags:
        - "User operations:"
      summary: 'Get user info'
      responses:
        '200':
          description: OK
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/UserInfo'
              example:
                aficiones: "juega al apex"
                edad: 22
                id: 1
                nombre: "Javi"
                nombre_usuario: "javilinos"
        '400':
          description: Bad Request.
          content:
            application/json:
              example: 'ERROR 400: BAD REQUEST'
        '404':
          description: Not Found.
          content:
            application/json:
              example: 'ERROR 404: NOT FOUND'
        '500':
          description: Internal Server Error.
          content:
            application/json:
              example: 'ERROR 500: INTERNAL SERVER ERROR'

    put:
      parameters:
        - name: id
          in: path
          description: "The user's id."
          schema:
            type: integer
          required: true
      tags:
        - "User operations:"
      summary: "Change user info."
      responses:
        '200':
```

```
93            description: OK
94            content:
95              application/json:
96                schema:
97                  $ref: '#/components/schemas/UserInfo'
98                example:
99                  nombre: "Javi"
100                 rel: "self"
101                 url: "http://localhost:8080/social_api/api/
                      usuarios/1"
102         '400':
103           description: Bad Request.
104           content:
105             application/json:
106               example: 'ERROR 400: BAD REQUEST'
107         '500':
108           description: Internal Server Error.
109           content:
110             application/json:
111               example: 'ERROR 500: INTERNAL SERVER ERROR'
112
113     delete:
114       parameters:
115         - name: id
116           in: path
117           description: "The user's id."
118           schema:
119             type: integer
120           required: true
121       tags:
122         - "User operations:"
123       summary: "Delete user."
124       responses:
125         '200':
126           description: OK
127           content:
128             application/json:
129               example: "200: OK"
130         '400':
131           description: Bad Request.
132           content:
133             application/json:
134               example: 'ERROR 400: BAD REQUEST'
135         '404':
136           description: Not Found.
137           content:
138             application/json:
139               example: 'ERROR 404: NOT FOUND'
```

```
        '500':
          description: Internal Server Error.
          content:
            application/json:
              example: 'ERROR 500: INTERNAL SERVER ERROR'

  /:
    post:
      tags:
        - "User operations:"
      summary: "Create new user."
      responses:
        '200':
          description: OK
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/User'
              example:
                nombre: "javi"
                rel: "self"
                url: "http://localhost:8080/social_api/api/
                   usuarios/2"
        '400':
          description: Bad Request.
          content:
            application/json:
              example: 'ERROR 400: BAD REQUEST'
        '404':
          description: Not Found.
          content:
            application/json:
              example: 'ERROR 404: NOT FOUND'
        '500':
          description: Internal Server Error.
          content:
            application/json:
              example: 'ERROR 500: INTERNAL SERVER ERROR'
```

1.5.2. Definición de los métodos de mensajes

```
/usuarios/{id}/mensajes:
  get:
    parameters:
      - name: id
        in: path
        description: "The id of the user from whom
           retrieve the messages."
        schema:
          type: integer
        required: true
    tags:
      - "Message operations:"
    summary: "Get list of messages associated to a user."
    responses:
      '200':
        description: OK
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Users'
            example:
              users:
                - nombre: "amigo"
                  rel: "self"
                  url: "http://localhost:8080/social_api/
                     api/usuarios/1"
      '400':
        description: Bad Request.
        content:
          application/json:
            example: 'ERROR 400: BAD REQUEST'
      '404':
        description: Not Found.
        content:
          application/json:
            example: 'ERROR 404: NOT FOUND'
      '500':
        description: Internal Server Error.
        content:
          application/json:
            example: 'ERROR 500: INTERNAL SERVER ERROR'
  post:
    parameters:
      - name: id
        in: path
        description: "Id of the user that creates the
           message."
```

```
          schema:
            type: integer
          required: true
      tags:
        - "Message operations:"
      summary: "Create a new message associated to the user."
      responses:
        '200':
          description: OK
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Users'
              example:
                users:
                  - rel: "self"
                    url: "http://localhost:8080/social_api/api/usuarios/1/mensajes/3"
        '400':
          description: Bad Request.
          content:
            application/json:
              example: 'ERROR 400: BAD REQUEST'
        '500':
          description: Internal Server Error.
          content:
            application/json:
              example: 'ERROR 500: INTERNAL SERVER ERROR'

  /usuarios/{id}/mensajes/{id_msg}:
    get:
      parameters:
        - name: id
          in: path
          description: 'User id.'
          schema:
            type: integer
          required: true
        - name: id_msg
          in: path
          description: 'Id of the specific message of the user.'
          schema:
            type: integer
          required: true
      tags:
        - "Message operations:"
```

```
90        summary: "Get message info."
91        responses:
92          '200':
93            description: OK
94            content:
95              application/json:
96                schema:
97                  $ref: '#/components/schemas/Message'
98                example:
99                  contenido: "hola me llamo pingu"
100                 fecha_envio: "2019-04-21T16:23:13.50TZ[UTC]
                      "
101                 id: 1
102                 id_usuario: 1
103         '400':
104           description: Bad Request.
105           content:
106             application/json:
107               example: "ERROR 400: BAD REQUEST"
108         '404':
109           description: Not Found.
110           content:
111             application/json:
112               example: "ERROR 404: NOT FOUND"
113         '500':
114           description: Internal Server Error.
115           content:
116             application/json:
117               example: "ERROR 500: INTERNAL SERVER ERROR"
118
119     put:
120       parameters:
121         - name: id
122           in: path
123           description: 'User id.'
124           schema:
125             type: integer
126           required: true
127         - name: id_msg
128           in: path
129           description: 'Id of the specific message of the
                user.'
130           schema:
131             type: integer
132           required: true
133       tags:
134         - "Message operations:"
135       summary: "Modify an existing message."
```

```
136       responses:
137         '200':
138           description: OK
139           content:
140             application/json:
141               schema:
142                 $ref: '#/components/schemas/Message'
143               example:
144                 rel: "self"
145                 url: "http://localhost:8080/social_api/api/
                      usuarios/1/mensajes/2"
146         '400':
147           description: Bad Request.
148           content:
149             application/json:
150               example: "ERROR 400: BAD REQUEST"
151         '404':
152           description: Not Found.
153           content:
154             application/json:
155               example: "ERROR 404: NOT FOUND"
156         '500':
157           description: Internal Server Error.
158           content:
159             application/json:
160               example: "ERROR 500: INTERNAL SERVER ERROR"
161
162     delete:
163       parameters:
164         - name: id
165           in: path
166           description: 'User id.'
167           schema:
168             type: integer
169           required: true
170         - name: id_msg
171           in: path
172           description: 'Id of the specific message of the
                user.'
173           schema:
174             type: integer
175           required: true
176       tags:
177         - "Message operations:"
178       summary: "Delete message."
179       responses:
180         '200':
181           description: OK
```

```
182             content:
183               application/json:
184                 schema:
185                   $ref: '#/components/schemas/Message'
186                 example:
187                   rel: "self"
188                   url: "http://localhost:8080/social_api/api/
                        usuarios/1/mensajes/1"
189           '400':
190             description: Bad Request.
191             content:
192               application/json:
193                 example: "ERROR 400: BAD REQUEST"
194           '404':
195             description: Not Found.
196             content:
197               application/json:
198                 example: "ERROR 404: NOT FOUND"
199           '500':
200             description: Internal Server Error.
201             content:
202               application/json:
203                 example: "ERROR 500: INTERNAL SERVER ERROR"
```

1.5.3. Definición de los métodos de mensajes privados

```yaml
/usuarios/{id}/privados:
  get:
    parameters:
      - name: id
        in: path
        description: 'User id.'
        schema:
          type: integer
        required: true
    tags:
      - "Private message operations:"
    summary: "Get private messages list."
    responses:
      '200':
        description: OK
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Message'
            example:
              privates:
                - rel: "self"
                  url: "http://localhost:8080/social_api/
                    api/usuarios/3/privados/1"
                - rel: "self"
                  url: "http://localhost:8080/social_api/
                    api/usuarios/3/privados/2"
                - rel: "self"
                  url: "htpp://localhost:8080/social_api/
                    api/usuarios/3/privados/3"
      '400':
        description: Bad Request.
        content:
          application/json:
            example: "ERROR 400: BAD REQUEST"
      '500':
        description: Internal Server Error.
        content:
          application/json:
            example: "ERROR 500: INTERNAL SERVER ERROR"

  post:
    parameters:
      - name: id
        in: path
        description: 'User id.'
        schema:
```

```
            type: integer
          required: true
      tags:
        - "Private message operations:"
      summary: "Create new private message."
      responses:
        '200':
          description: OK
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Message'
              example:
                rel: "self"
                url: "http://localhost:8080/social_api/api/
                   usuarios/3/privados/4"
        '400':
          description: Bad Request.
          content:
            application/json:
              example: "ERROR 400: BAD REQUEST"
        '500':
          description: Internal Server Error.
          content:
            application/json:
              example: "ERROR 500: INTERNAL SERVER ERROR"

  /usuarios/{id}/privados/{id_msg}:
    get:
      parameters:
        - name: id
          in: path
          description: 'User id.'
          schema:
            type: integer
          required: true
        - name: id_msg
          in: path
          description: 'Id of the specific private message
             of the user.'
          schema:
            type: integer
          required: true
      tags:
        - "Private message operations:"
      summary: "Get private message info."
      responses:
        '200':
```

```
            description: OK
            content:
              application/json:
                schema:
                  $ref: '#/components/schemas/Message'
                example:
                  contenido: "como estas"
                  destinatario: 3
                  fecha_envio: "2019-07-09T22:00:00Z[UTC]"
                  id: 4
          '400':
            description: Bad Request.
            content:
              application/json:
                example: "ERROR 400: BAD REQUEST"
          '404':
            description: Not Found.
            content:
              application/json:
                example: "ERROR 404: NOT FOUND"
          '500':
            description: Internal Server Error.
            content:
              application/json:
                example: "ERROR 500: INTERNAL SERVER ERROR"
```

1.5.4. Definición de los métodos de amistades

```
/usuarios/{id}/amigos:
  get:
    parameters:
      - name: id
        in: path
        description: 'User id.'
        schema:
          type: integer
        required: true
    tags:
      - "User friends operations:"
    summary: "Get friends' list."
    responses:
      '200':
        description: OK
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Message'
            example:
              users:
              - nombre: "amigo"
                rel: "self"
                url: "http://localhost:8080/social_api/
                   api/usuarios/1"
              - nombre: "amigo"
                rel: "self"
                url: "http://localhost:8080/social_api/
                   api/usuarios/3"
      '400':
        description: Bad Request.
        content:
          application/json:
            example: "ERROR 400: BAD REQUEST"
      '500':
        description: Internal Server Error.
        content:
          application/json:
            example: "ERROR 500: INTERNAL SERVER ERROR"

  post:
    parameters:
      - name: id
        in: path
        description: 'User id.'
        schema:
          type: integer
```

```
46            required: true
47        tags:
48          - "User friends operations:"
49        summary: "Add a friend."
50        responses:
51          '200':
52            description: OK
53            content:
54              application/json:
55                schema:
56                  $ref: '#/components/schemas/Message'
57                example:
58                  users:
59                  - nombre: "amigo"
60                    rel: "self"
61                    url: "http://localhost:8080/social_api/
                        api/usuarios/1"
62                  - nombre: "amigo"
63                    rel: "self"
64                    url: "http://localhost:8080/social_api/
                        api/usuarios/3"
65          '400':
66            description: Bad Request.
67            content:
68              application/json:
69                example: "ERROR 400: BAD REQUEST"
70          '500':
71            description: Internal Server Error.
72            content:
73              application/json:
74                example: "ERROR 500: INTERNAL SERVER ERROR"
```

1.5.5. Definición de los esquemas

```yaml
components:
  schemas:
    Users:
      type: array
      items:
        anyOf:
          - $ref: "#/components/schemas/User"

    UserInfo:
      type: object
      properties:
        aficiones:
          type: string
          description: "The user's hobbies.."
        edad:
          type: integer
          description: "The user's age."
        id:
          type: integer
          description: "The user's id."
        nombre:
          type: string
          description: "The user's first name."
        nombre_usuario:
          type: string
          description: "The user's username."

    User:
      type: object
      properties:
        nombre:
          type: string
          description: "The user's first name."
        rel:
          type: string
          description: "The relationship with the calling
            user."
        url:
          type: string
          description: "The user's URL."

    Message:
      type: object
      properties:
        contenido:
          type: string
          description: "The message's contents."
```

```
47          fecha:
48            type: string
49            description: "The date the message was sent."
50          id:
51            type: integer
52            description: "The message identifier."
53          id_usuario:
54            type: integer
55            description: "The id of the user who sent the
                message."
```

2. Capturas de la ejecución en el cliente Postman

Hay que poner detalles tanto de la invocación como del resultado de la operación.

2.1. Métodos de usuarios

2.1.1. POST

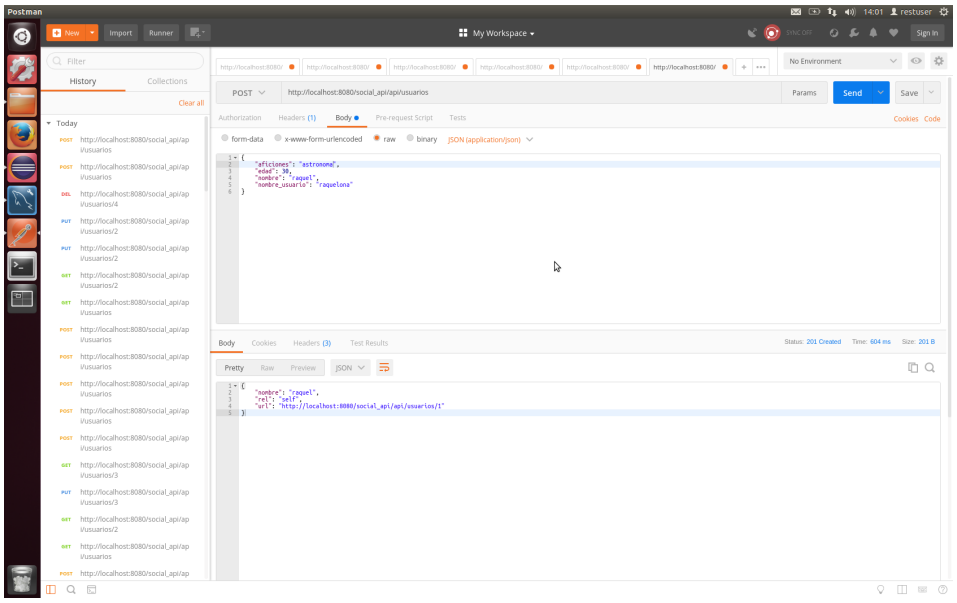


Figura 1: Se añade el usuario 1 de nombre Raquel.

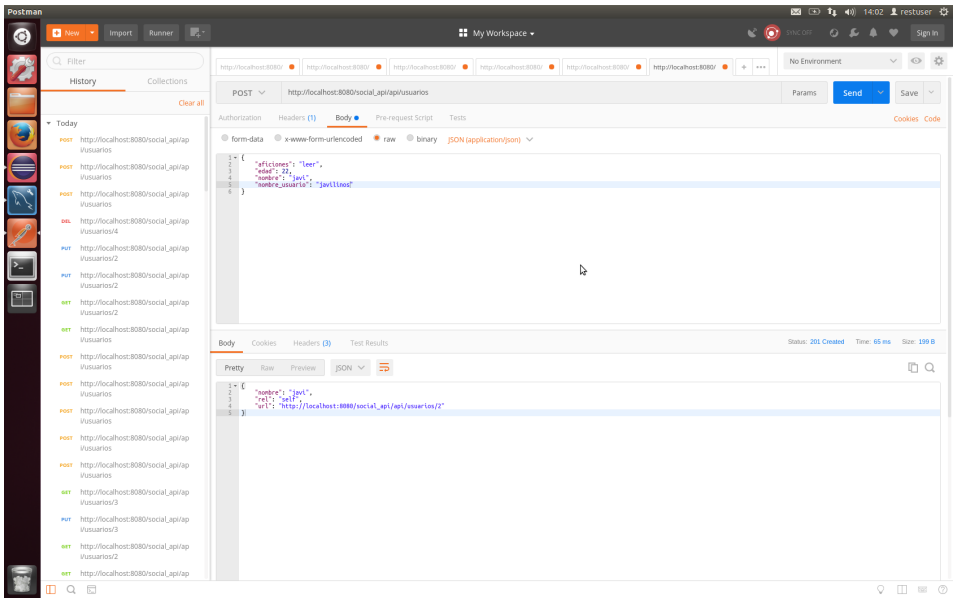


Figura 2: Se añade el usuario 2 de nombre Javi.

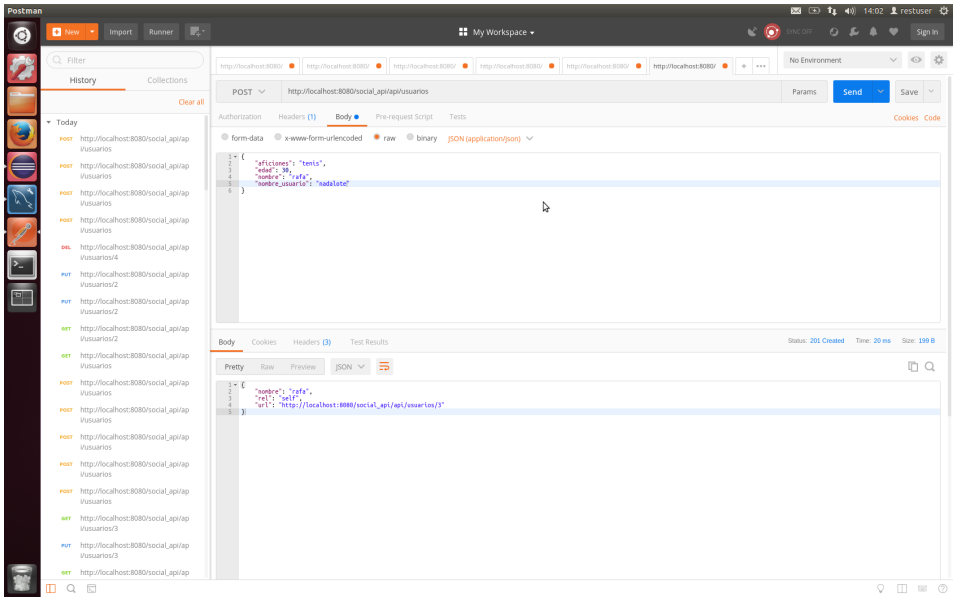


Figura 3: Se añade el usuario 3 de nombre Rafa.

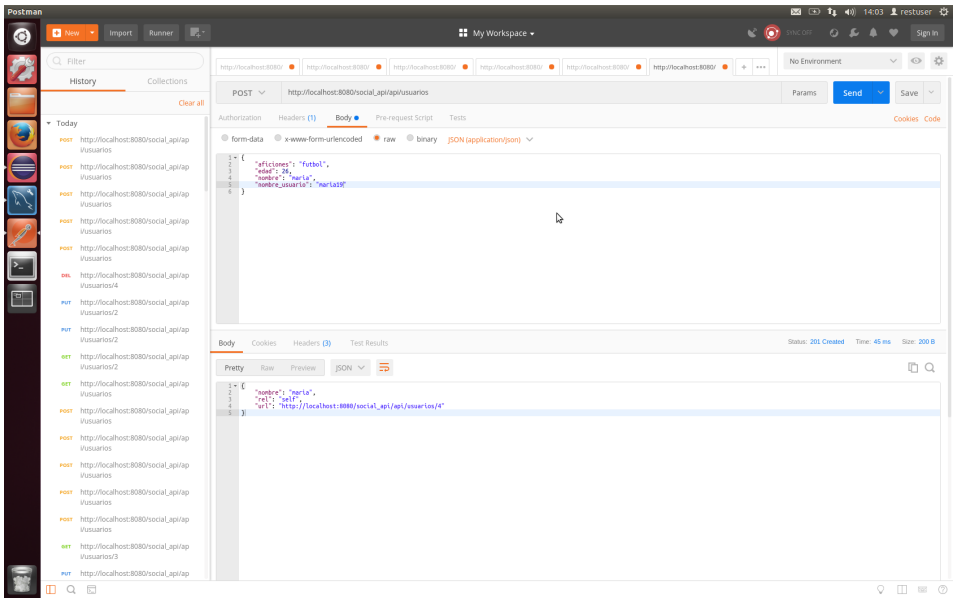


Figura 4: Se añade el usuario 4 de nombre María.

2.1.2. GET

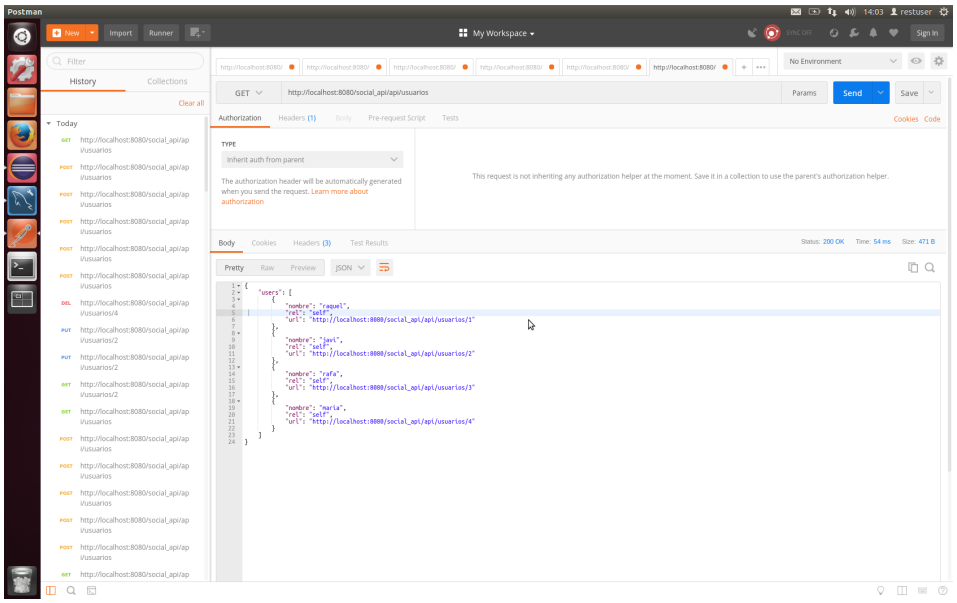


Figura 5: Se obtienen todos los usuarios.

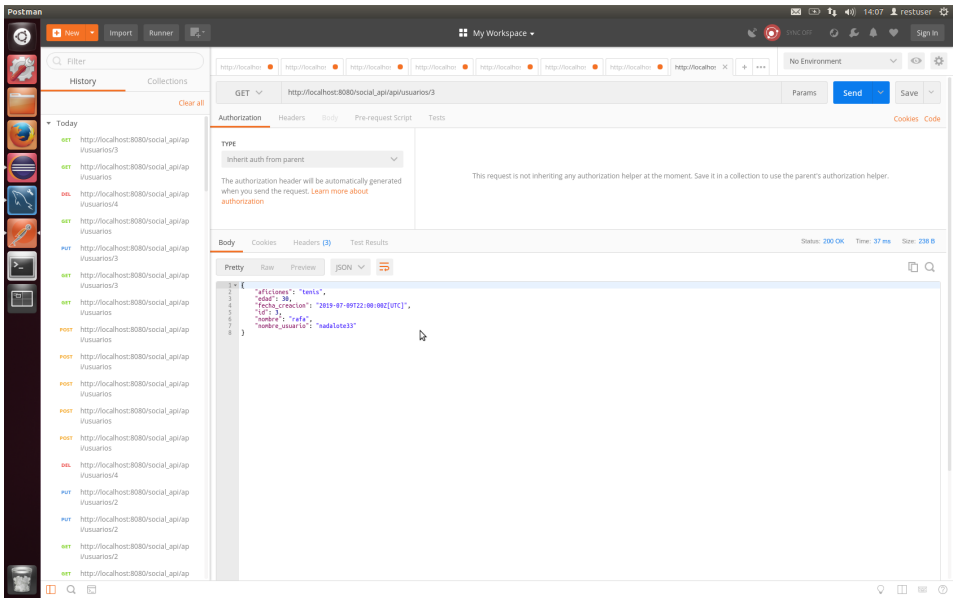


Figura 6: Obtención del usuario 3 después de ser modificado.

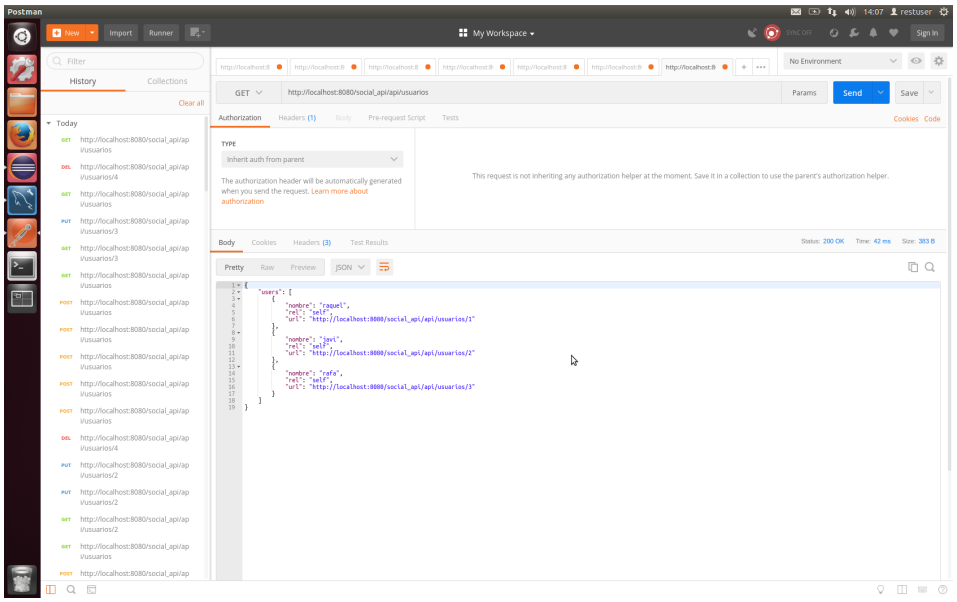


Figura 7: Se obtienen todos los usuarios después de eliminar el usuario 4 (Figura 10).

2.1.3. PUT

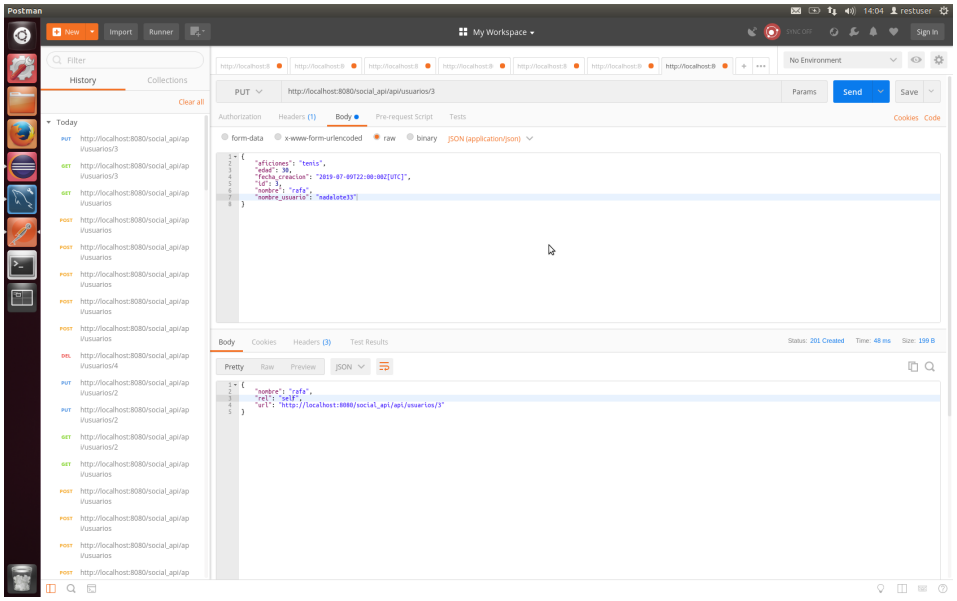


Figura 8: Modificación del nombre de usuario 3.

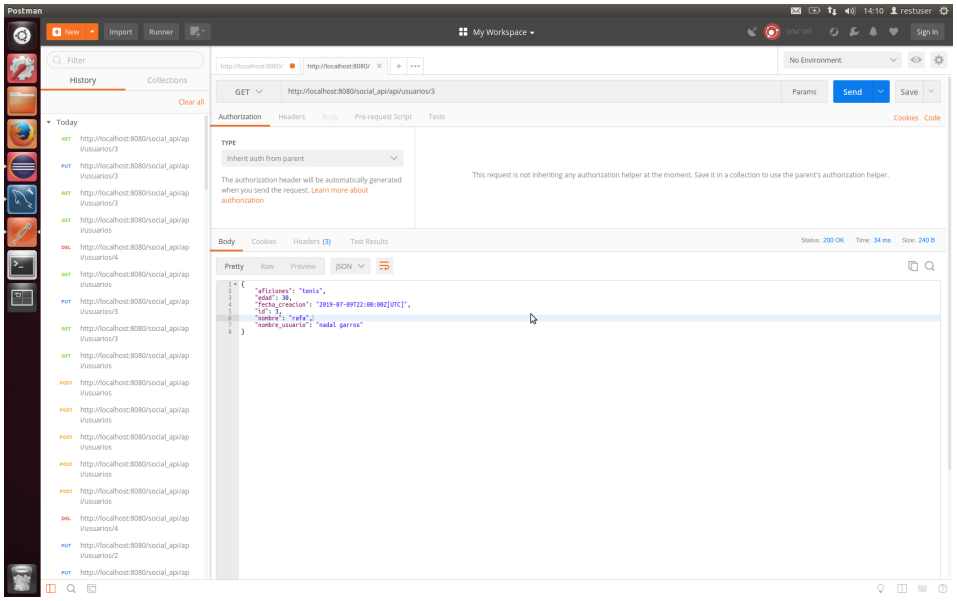


Figura 9: Obtención de los datos después de la modificación anterior.

2.1.4. DELETE

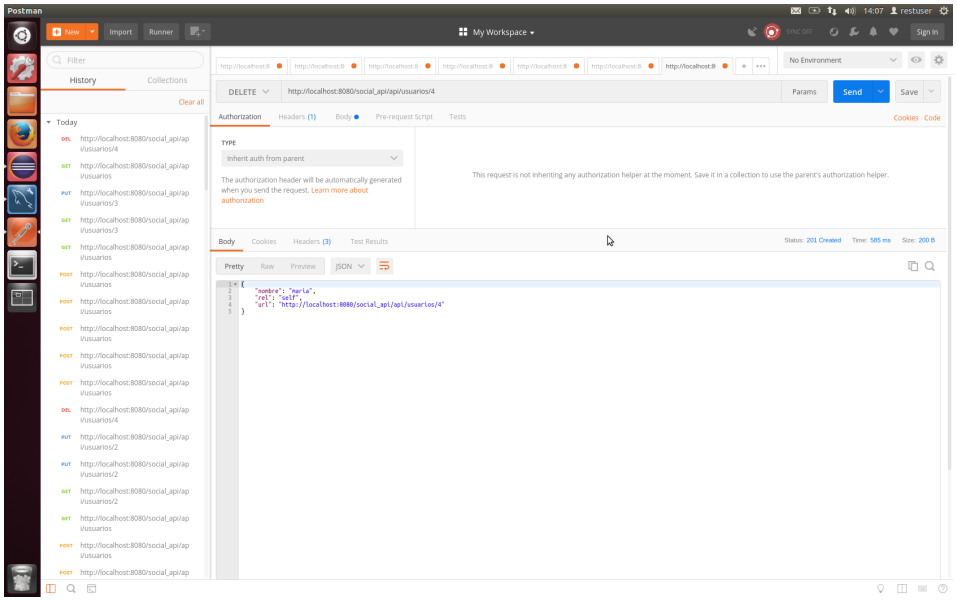


Figura 10: Eliminación del usuario 4.

2.2. Métodos de mensajes

2.2.1. POST

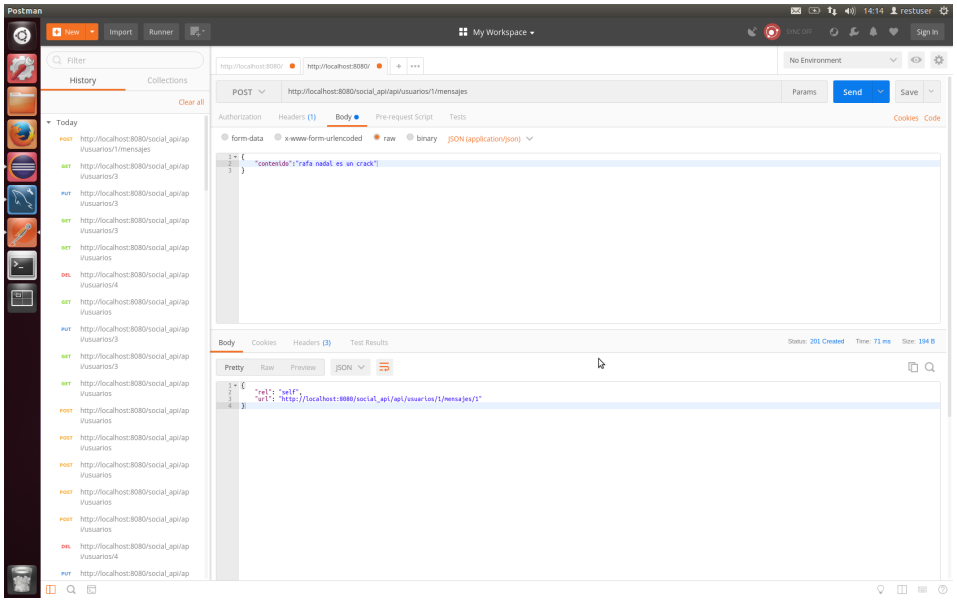


Figura 11: Creación del mensaje 1 asociado al usuario 1.

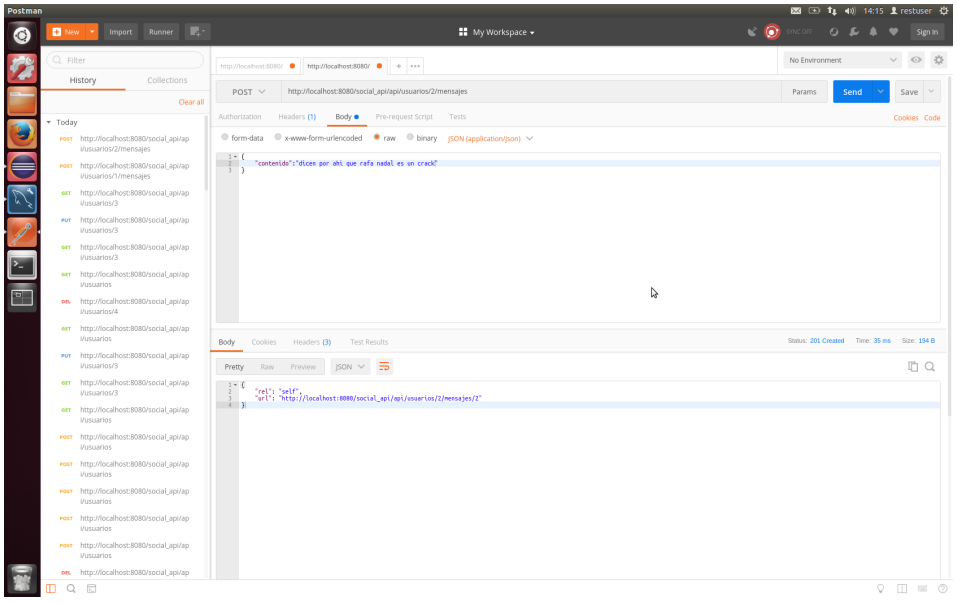


Figura 12: Creación del mensaje 2 asociado al usuario 2.

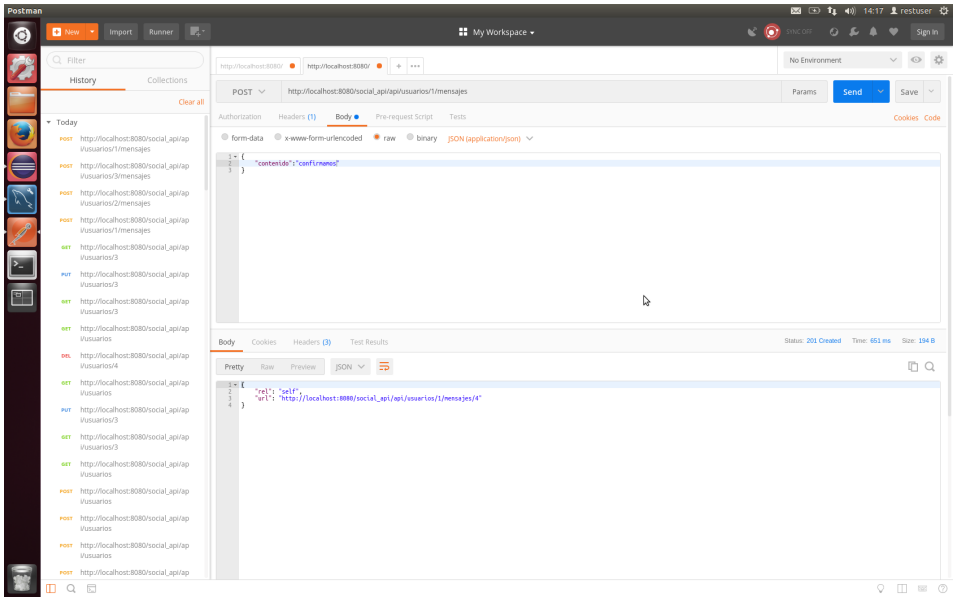


Figura 13: Creación del mensaje 4 asociado al usuario 1.

2.2.2. GET

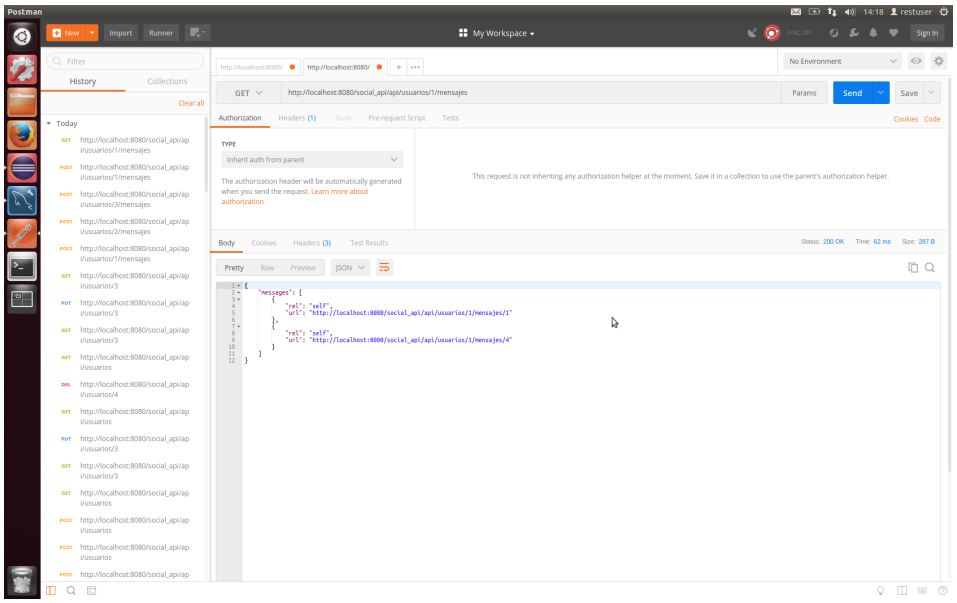


Figura 14: Se obtienen todos mensajes asociados al usuario 1.

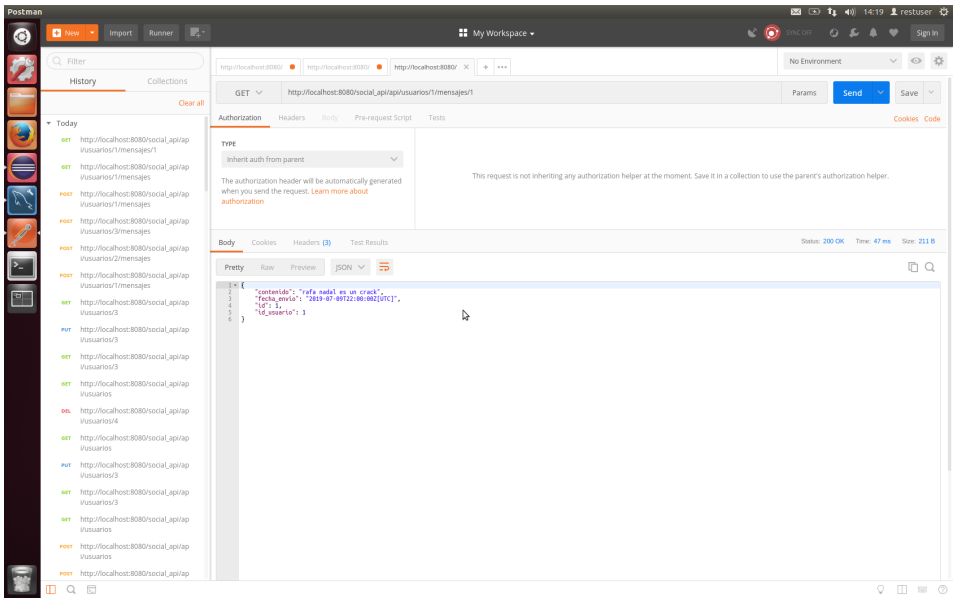


Figura 15: Obtención del mensaje 1 del usuario 1.

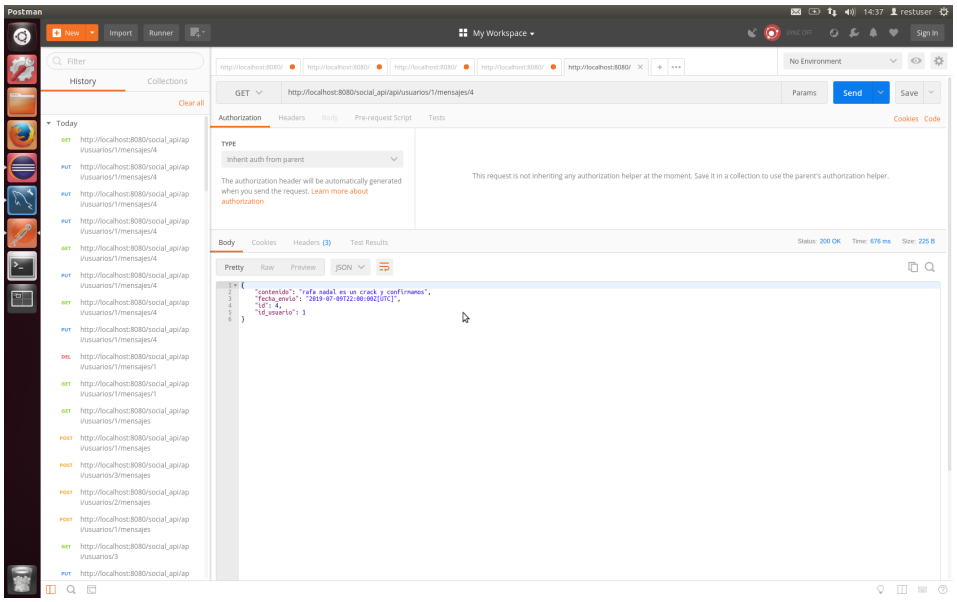


Figura 16: Obtención del mensaje 4 después de ser modificado (Figura 17).

2.2.3. PUT

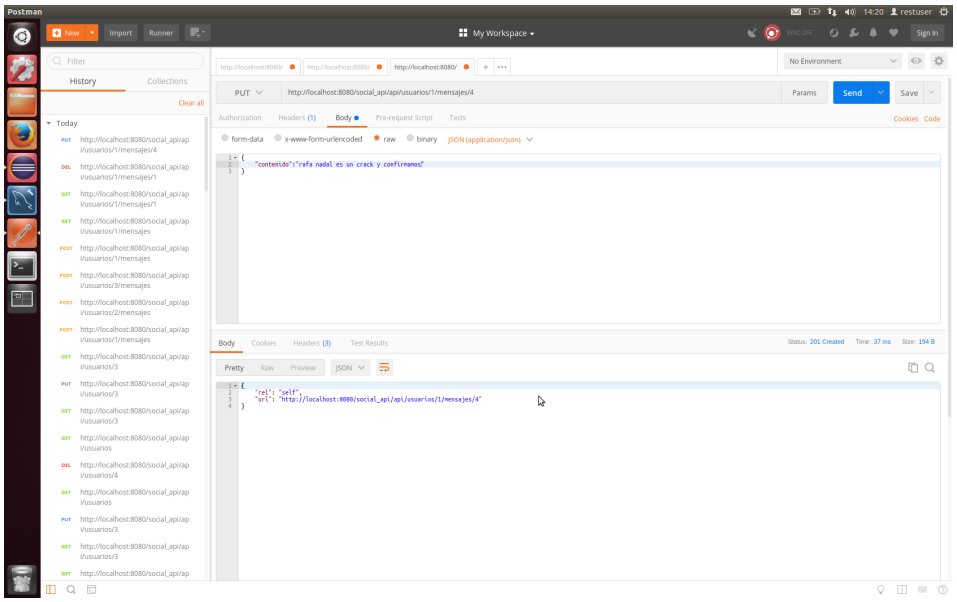


Figura 17: Modificación del mensaje 4 asociado al usuario 1.

2.2.4. DELETE

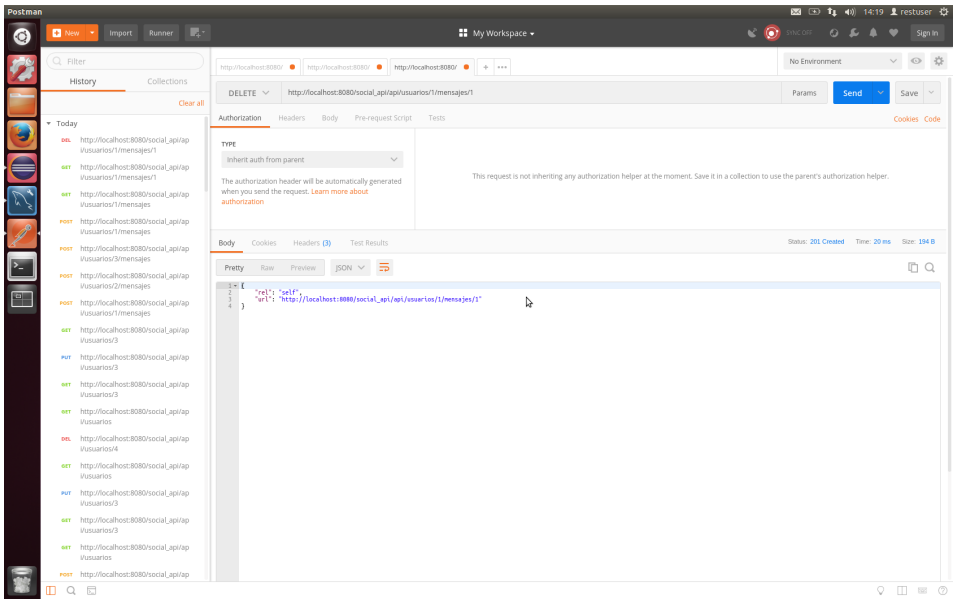


Figura 18: Eliminación del mensaje 1 del usuario 1.

2.3. Métodos de Mensajes Privados

2.3.1. POST

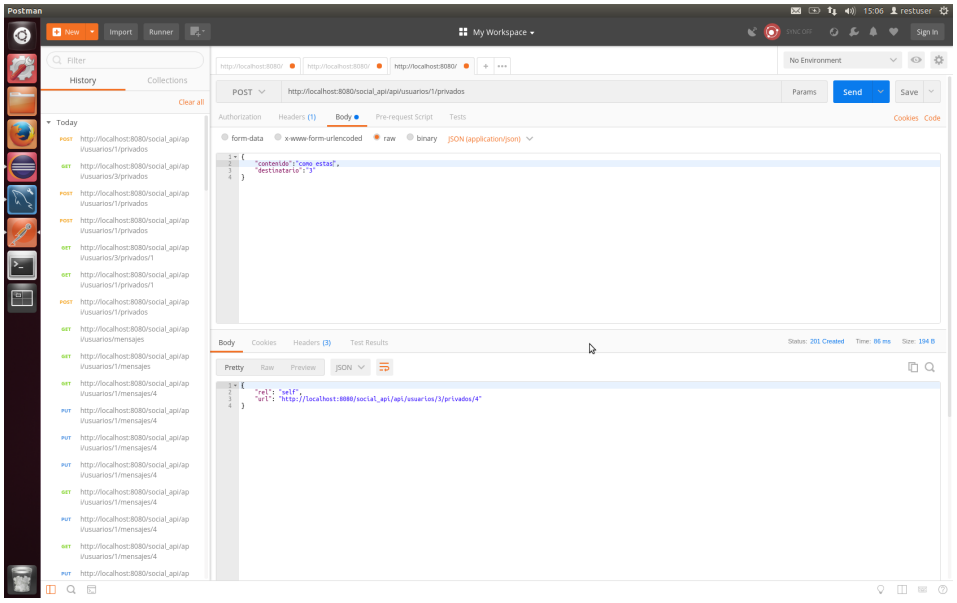


Figura 19: Creación del cuarto mensaje privado del usuario 1 al 3.

2.3.2. GET

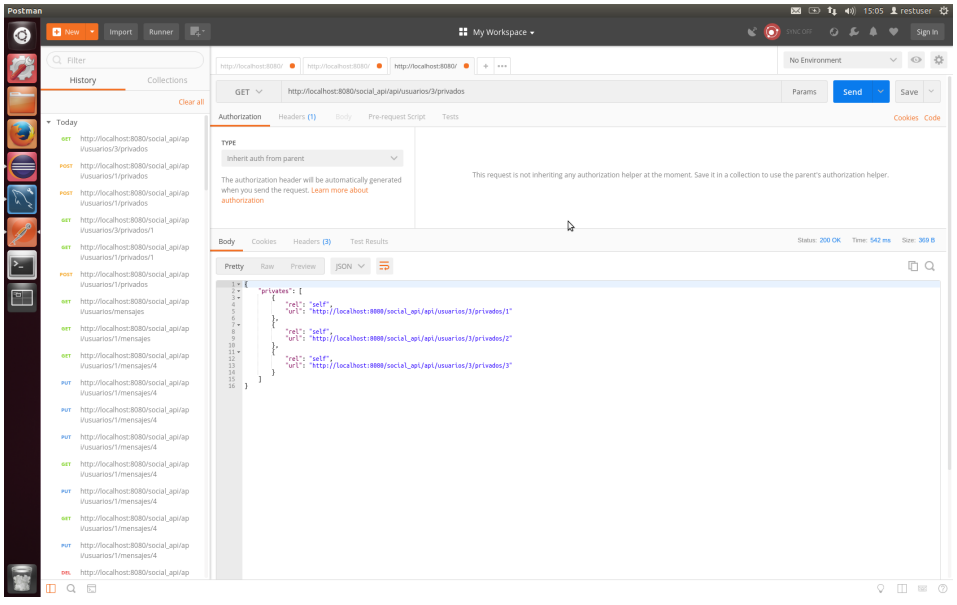


Figura 20: Obtención de todos los mensajes privados escritos por el usuario 3.

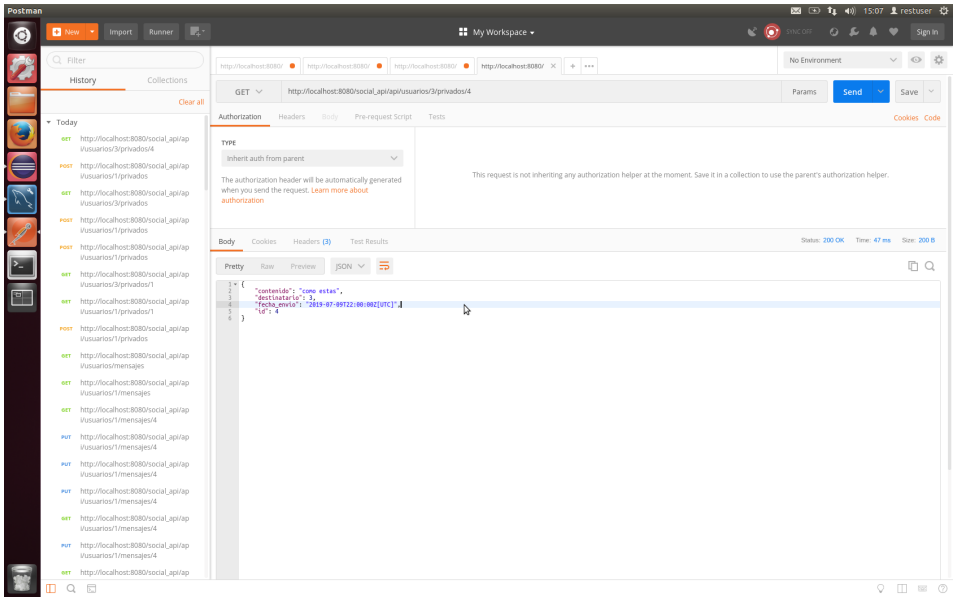


Figura 21: Obtención del mensaje privado 4 del usuario 3.

2.4. Métodos de Amistad

2.4.1. POST

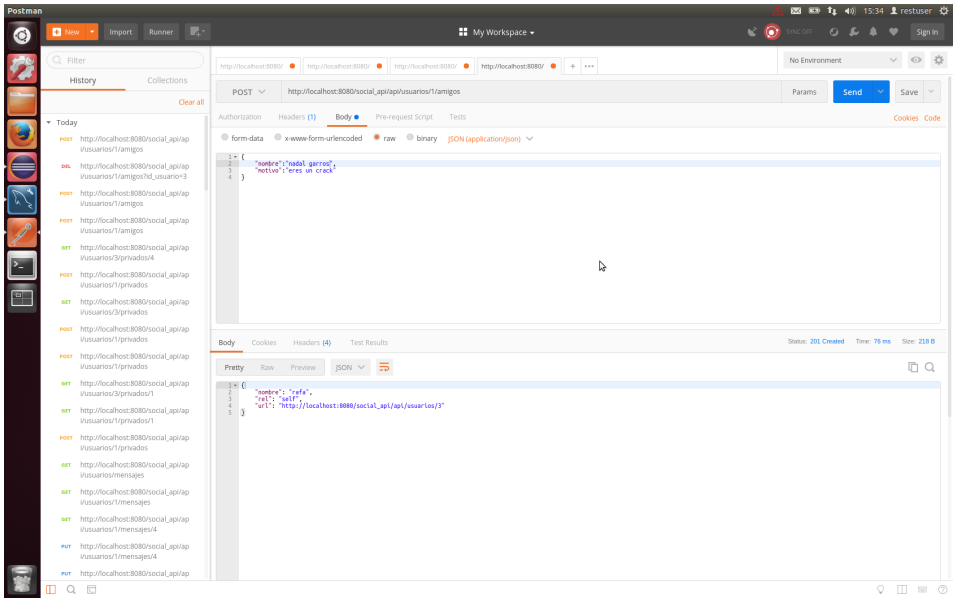


Figura 22: Se añade el usuario 3 como amigo del usuario 1.

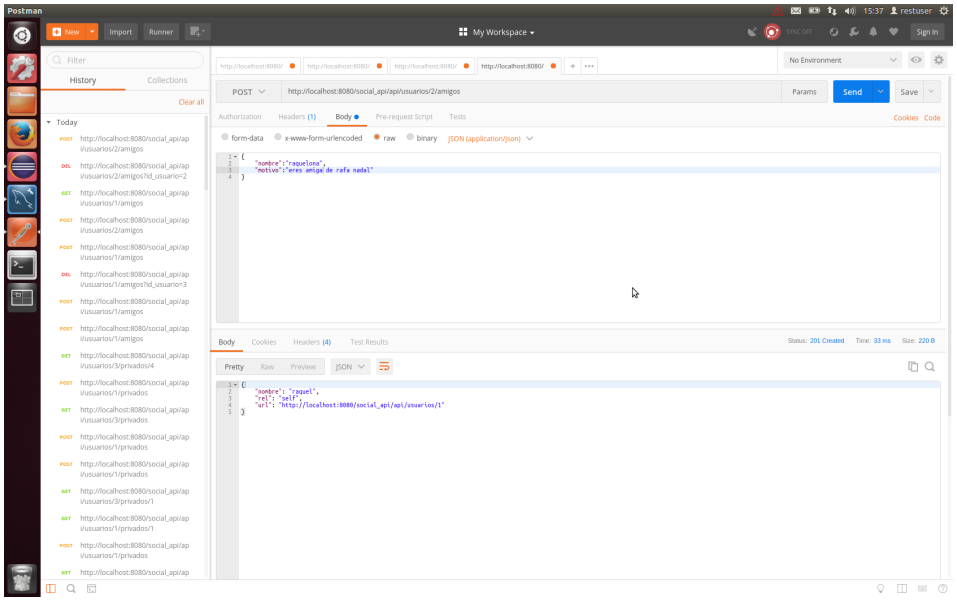


Figura 23: Se añade el usuario 1 como amigo del usuario 2.

2.4.2. GET

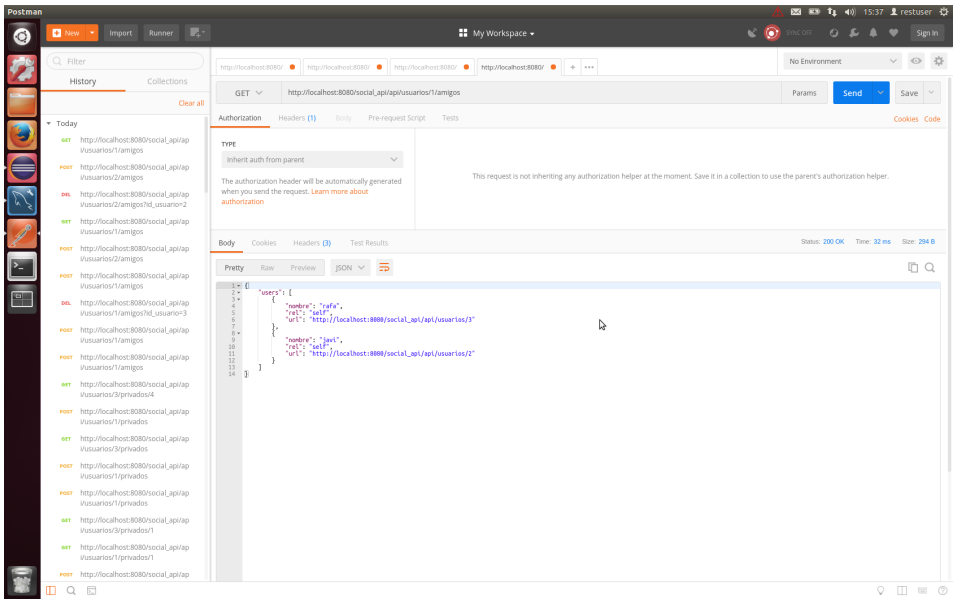


Figura 24: Obtención de todos los usuarios amigos del usuario 1.

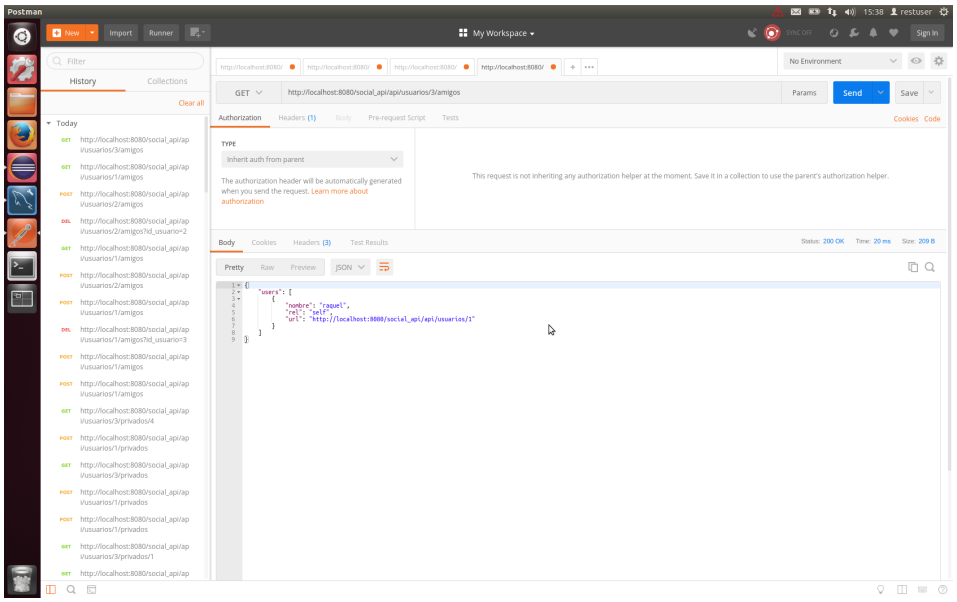


Figura 25: Obtención de todos los usuarios amigos del usuario 3.

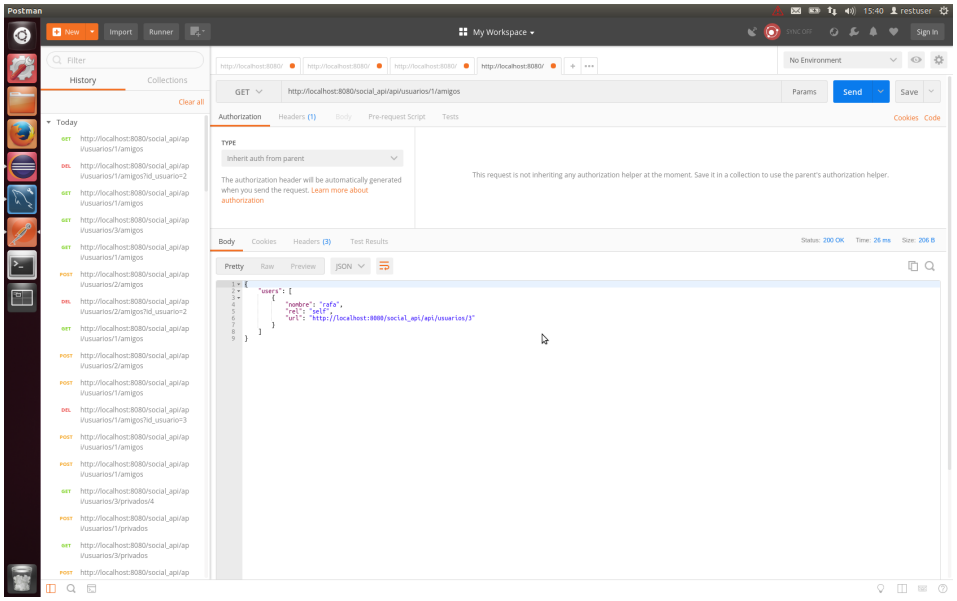


Figura 26: Obtención de los amigos del usuario 1 después de eliminar uno de ellos (Figura 27).

2.4.3. DELETE

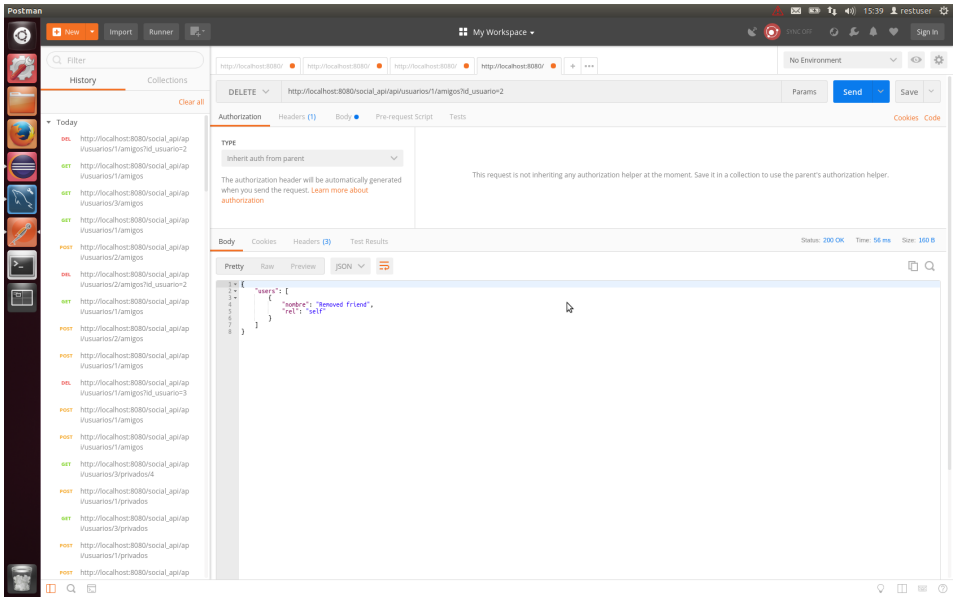


Figura 27: Eliminación del usuario 2 de los amigos del usuario 1.

2.5. Obtención de mensajes similar a FaceBook

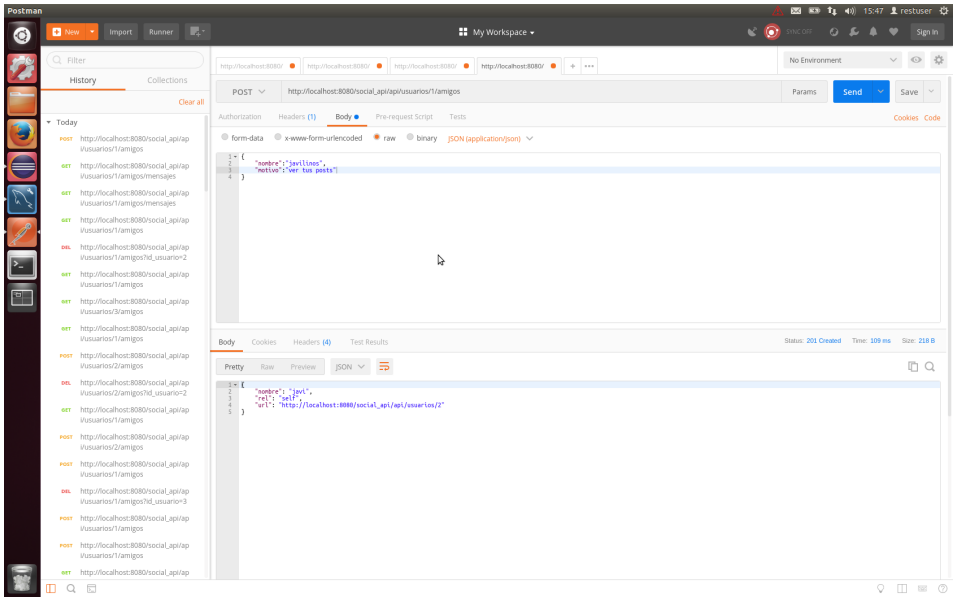


Figura 28: Obtención de los amigos del usuario 1.

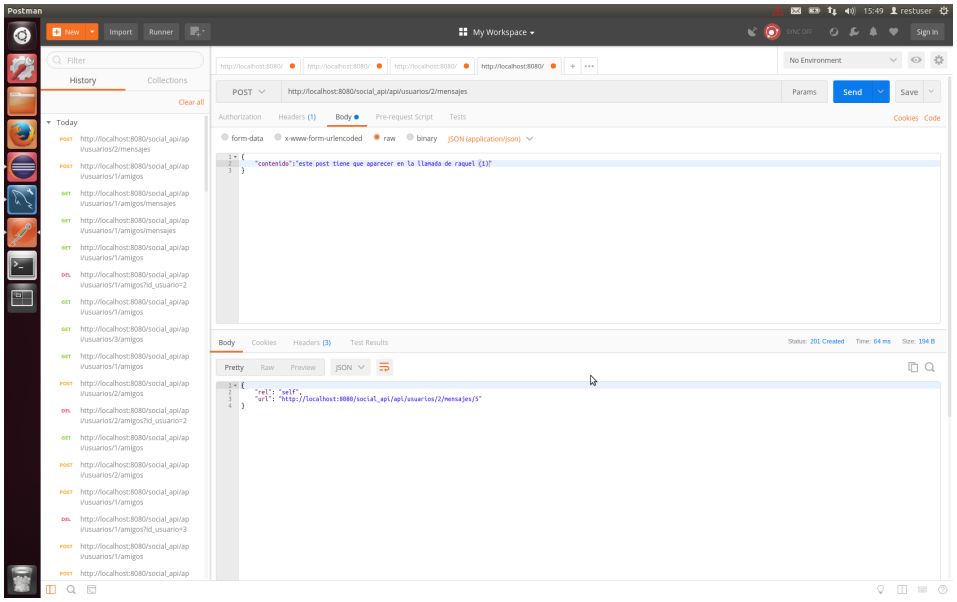


Figura 29: Creación de un mensaje del usuario 2.

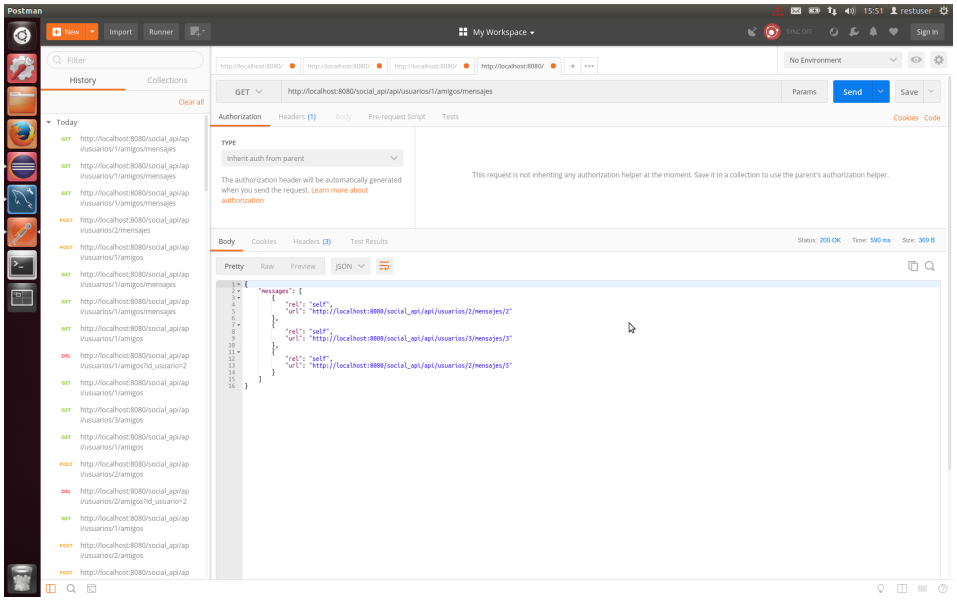


Figura 30: Obtención de los mensajes de los amigos del usuario 1.

2.6. Aplicación móvil

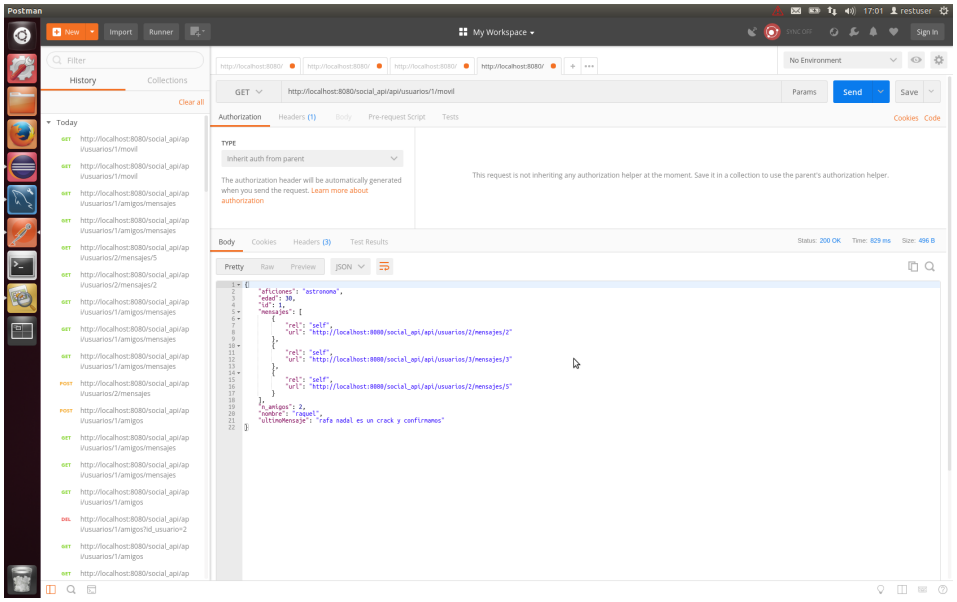


Figura 31: Obtención de la información del usuario 1 simulando una aplicación móvil.

3. ANEXO: Problemas encontrados

Los fallos encontrados referentes a la Máquina virtual son:

- Con la aplicación Postman, al probar los métodos HTTP.
- Con la aplicación Eclipse, que algunas veces no encontraba las URIs implementadas.
- Con la baja calidad de las capturas de pantalla del SO utilizado (probablemente por la antigüedad del mismo).

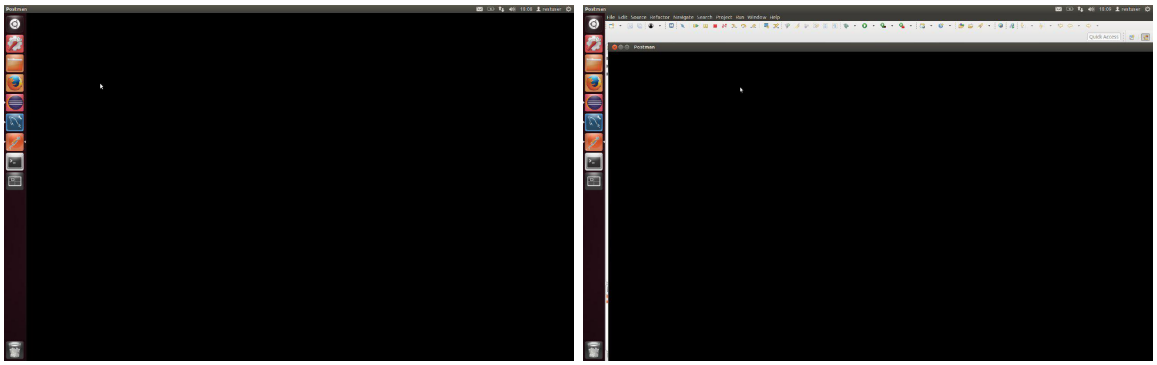


Figura 32: Fallo de la aplicación Postman al probar métodos.