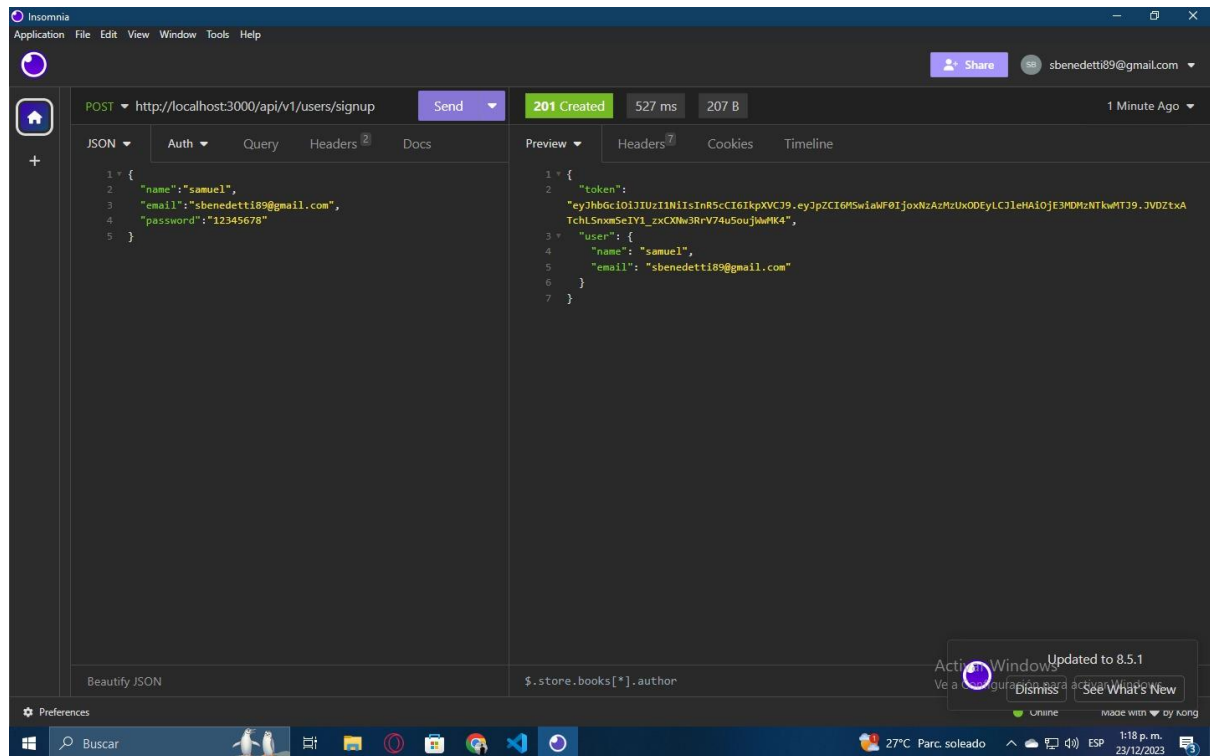
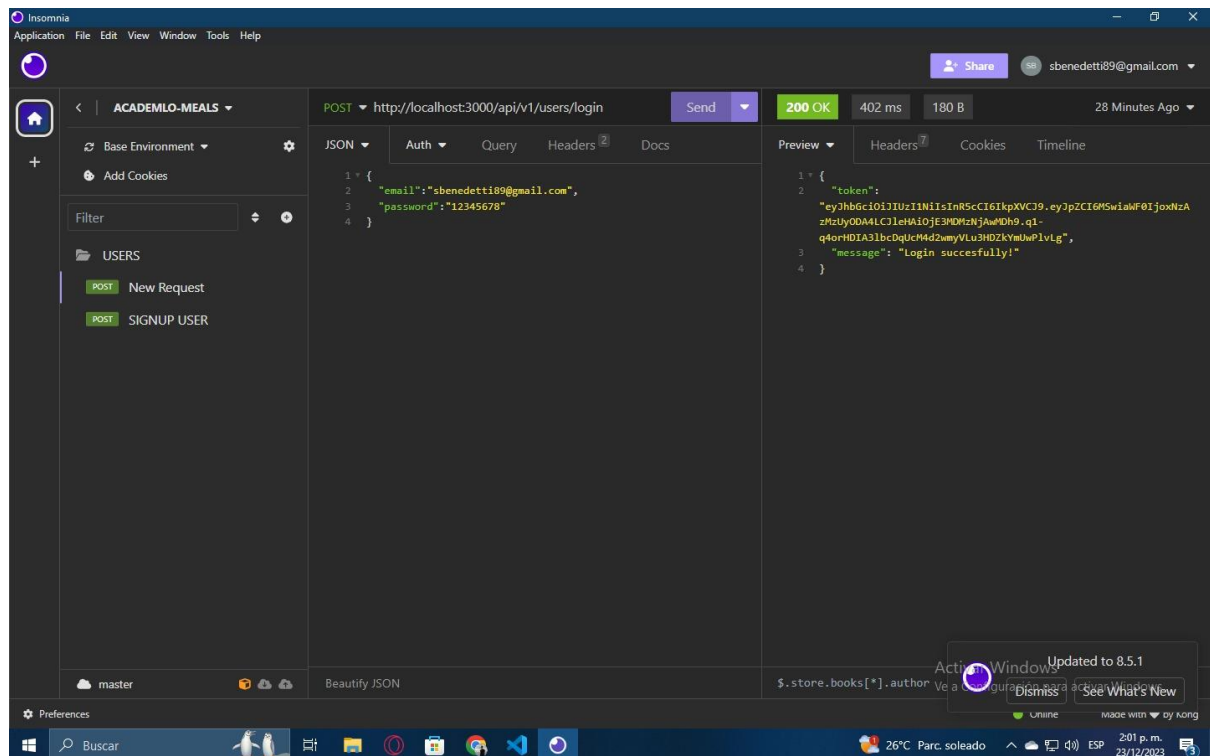


## USERS

## POST SIGNUP



## POST LOGIN



## UPDATE USER

The screenshot shows the Insomnia application interface. On the left sidebar, under the 'ACADEMLO-MEALS' collection, the 'USERS' folder is expanded, showing a 'PATCH' request named 'New Request'. The main panel displays the details of this request: the method is 'PATCH', the URL is 'http://localhost:3000/api/v1/users/1', and the status is '200 OK' with a response time of 778 ms and 57 B of data. The 'JSON' tab is selected, showing the request body as an object with 'name' and 'email' fields. The 'Preview' tab on the right shows the response body, which is a JSON object with 'status' and 'message' fields. The Windows taskbar at the bottom shows the system clock as 2:11 p.m. on 23/12/2023.

```
1 {
2   "name": "Samuel",
3   "email": "sbenedetti89@gmail.com"
4 }
```

```
1 {
2   "status": "success",
3   "message": "User updated succesfully"
4 }
```

## DELETE USER

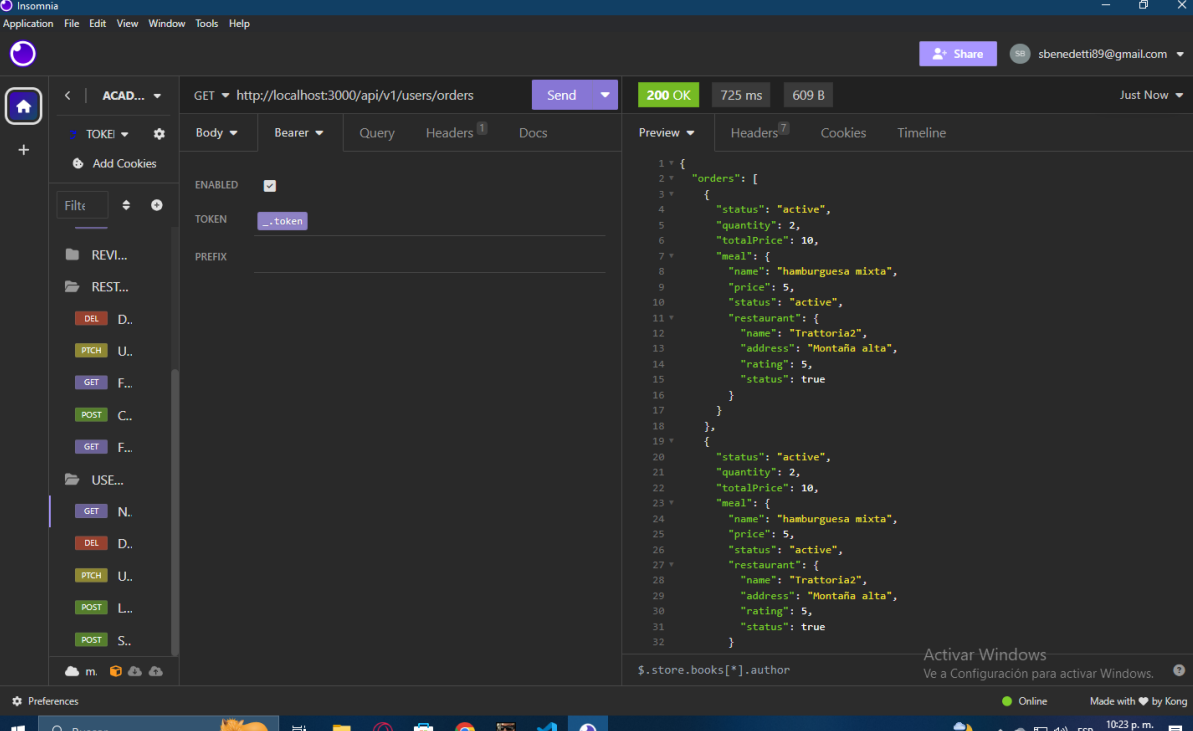
The screenshot shows the Insomnia application interface. On the left sidebar, under the 'ACADEMLO-MEALS' collection, the 'USERS' folder is expanded, showing a 'DELETE' request named 'DELETE USER'. The main panel displays the details of this request: the method is 'DELETE', the URL is 'http://localhost:3000/api/v1/users/1', and the status is '204 No Content' with a response time of 737 ms and 0 B of data. The 'Body' tab is selected, showing a large grey bug icon and a message: 'Enter a URL and send to get a response'. The 'Preview' tab on the right shows the response body, which is 'No body returned for response'. The Windows taskbar at the bottom shows the system clock as 2:12 p.m. on 23/12/2023.

Enter a URL and send to get a response

Select a body type from above to send data in the body of a request

Introduction to Insomnia

## GET SESSION USER ORDERS



Insomnia REST client interface showing a GET request to `http://localhost:3000/api/v1/users/orders`. The response is a 200 OK status with a JSON body containing an array of orders.

Request Details:

- Method: GET
- URL: `http://localhost:3000/api/v1/users/orders`
- Body: Bearer
- Token: `...token`

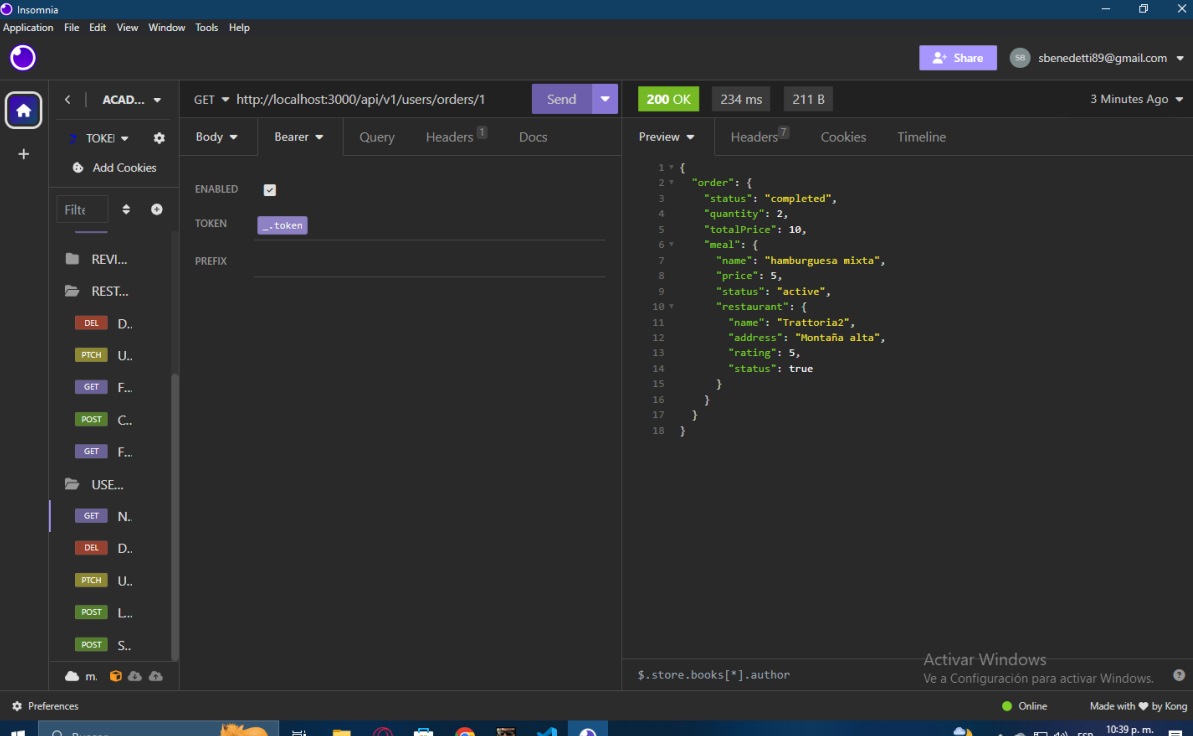
Response Details:

- Status: 200 OK
- Time: 725 ms
- Size: 609 B

Response Body (JSON):

```
1 {
2   "orders": [
3     {
4       "status": "active",
5       "quantity": 2,
6       "totalPrice": 10,
7       "meal": {
8         "name": "hamburguesa mixta",
9         "price": 5,
10        "status": "active",
11        "restaurant": {
12          "name": "Trattoria2",
13          "address": "Montaña alta",
14          "rating": 5,
15          "status": true
16        }
17      }
18    },
19    {
20      "status": "active",
21      "quantity": 2,
22      "totalPrice": 10,
23      "meal": {
24        "name": "hamburguesa mixta",
25        "price": 5,
26        "status": "active",
27        "restaurant": {
28          "name": "Trattoria2",
29          "address": "Montaña alta",
30          "rating": 5,
31          "status": true
32        }
33      }
34    }
35  ]
36 }
```

## GET SESSION USER ORDER



Insomnia REST client interface showing a GET request to `http://localhost:3000/api/v1/users/orders/1`. The response is a 200 OK status with a JSON body containing a single order object.

Request Details:

- Method: GET
- URL: `http://localhost:3000/api/v1/users/orders/1`
- Body: Bearer
- Token: `...token`

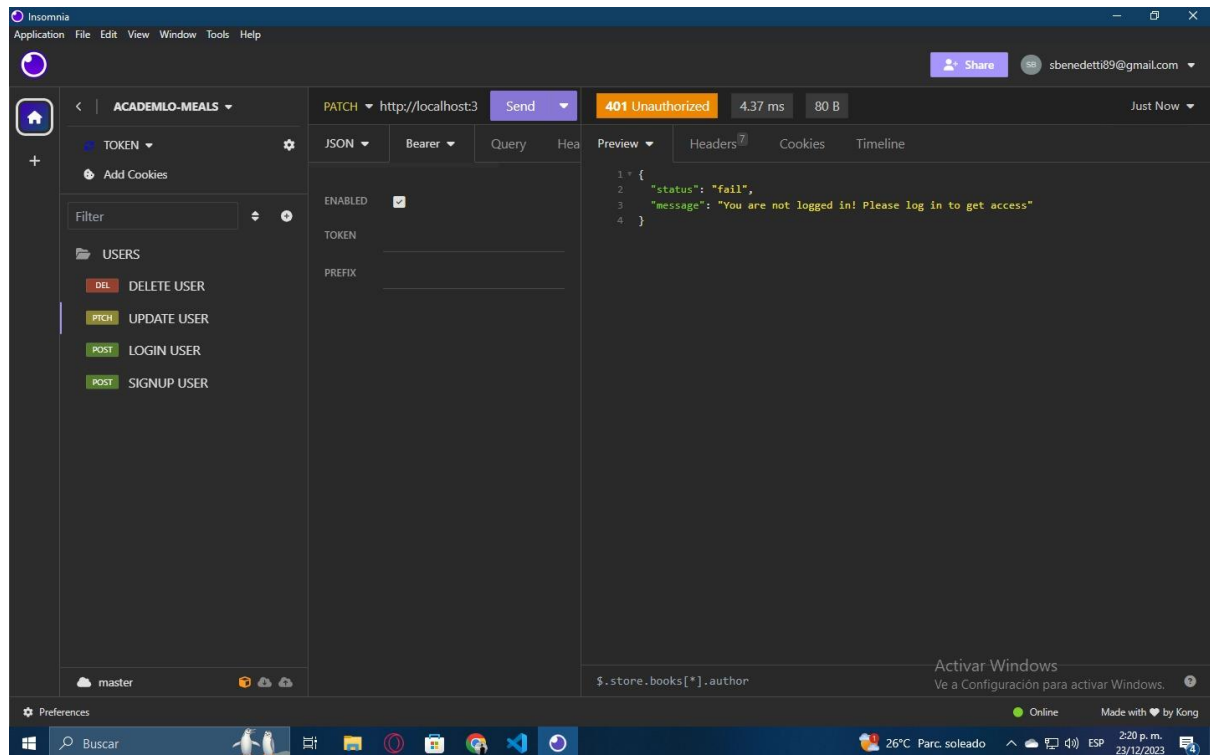
Response Details:

- Status: 200 OK
- Time: 234 ms
- Size: 211 B

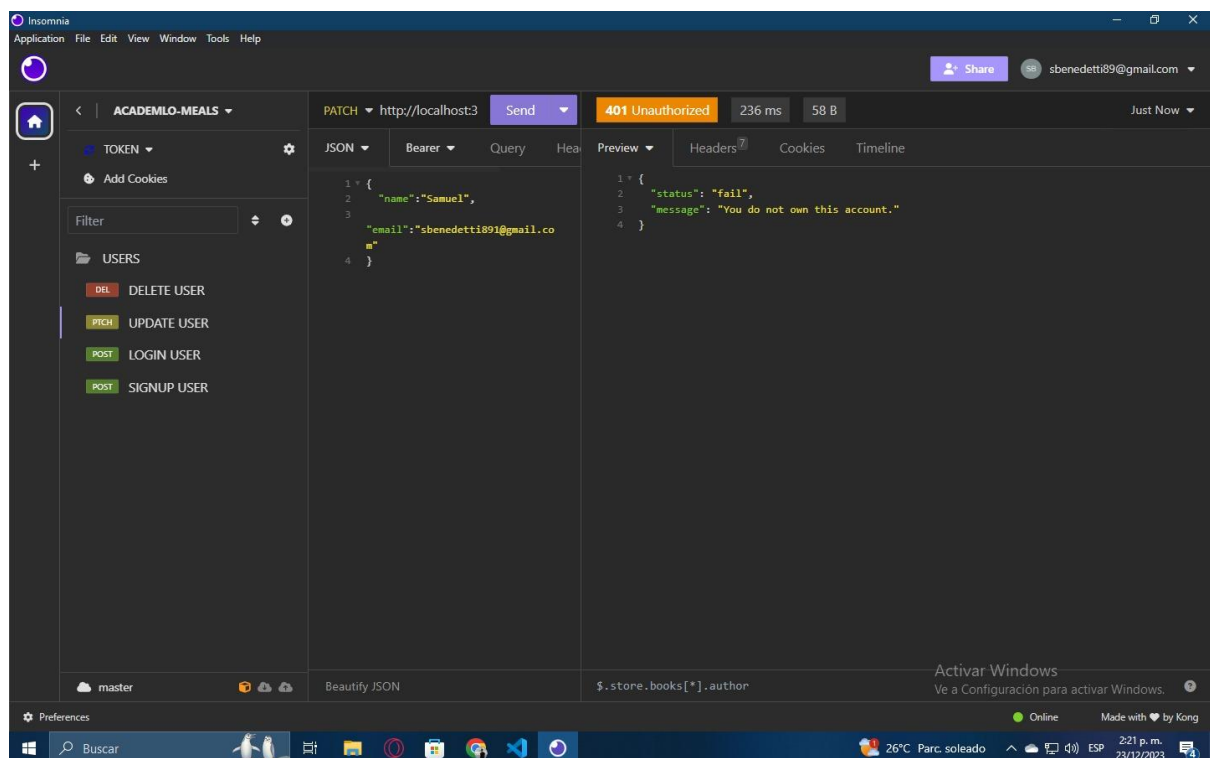
Response Body (JSON):

```
1 {
2   "order": {
3     "status": "completed",
4     "quantity": 2,
5     "totalPrice": 10,
6     "meal": {
7       "name": "hamburguesa mixta",
8       "price": 5,
9       "status": "active",
10      "restaurant": {
11        "name": "Trattoria2",
12        "address": "Montaña alta",
13        "rating": 5,
14        "status": true
15      }
16    }
17  }
18 }
```

# PROTECT JWT

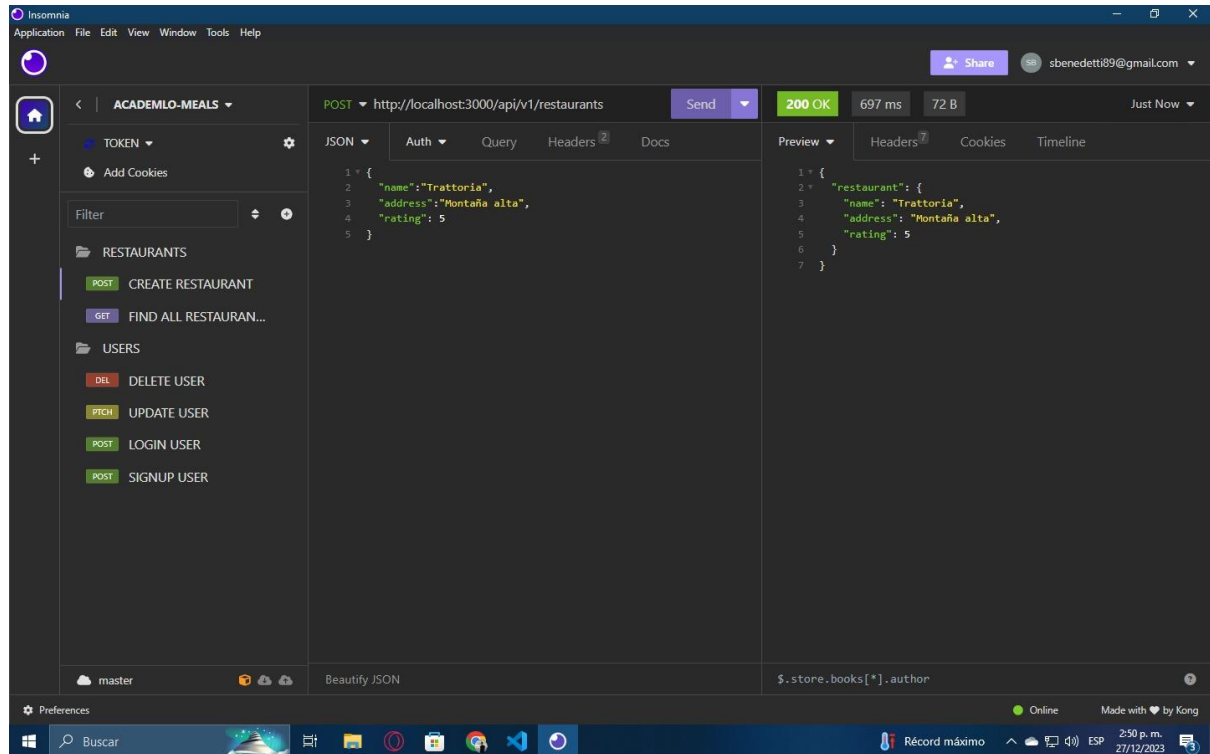


# PROTECT ACCOUNT OWNER

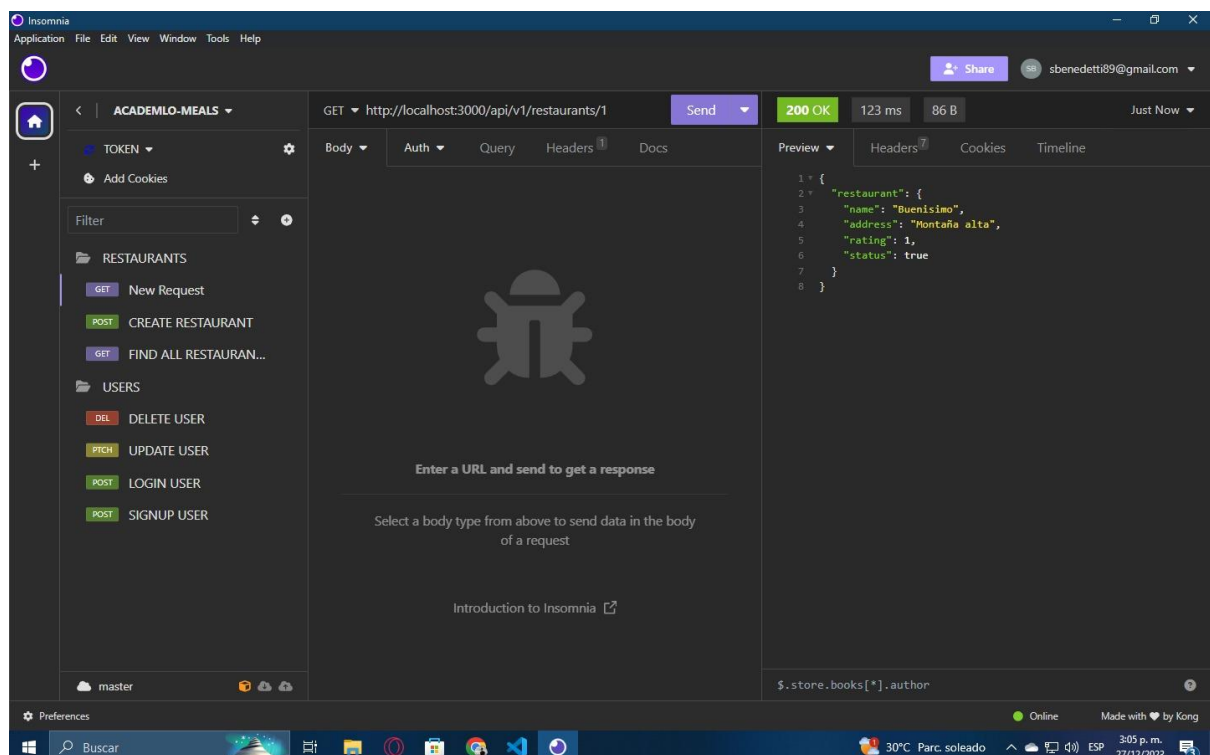


# RESTAURANTS

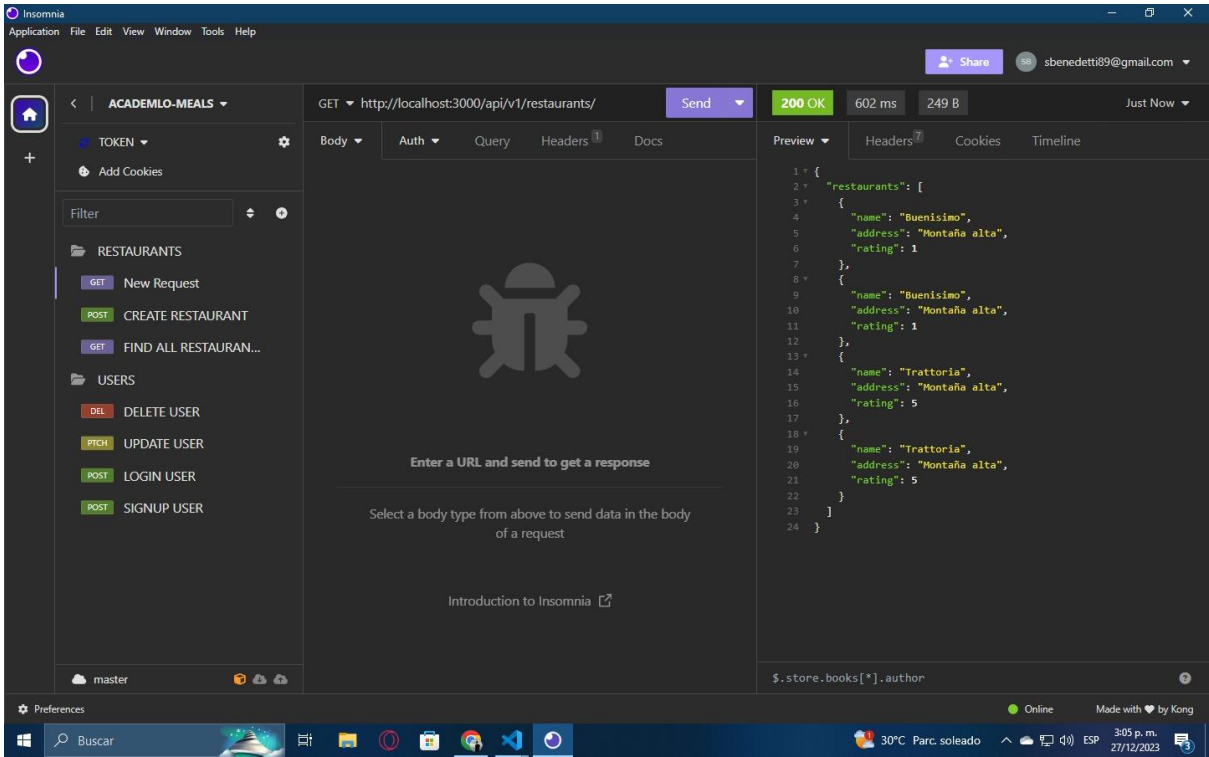
## POST CREATE RESTAURANT



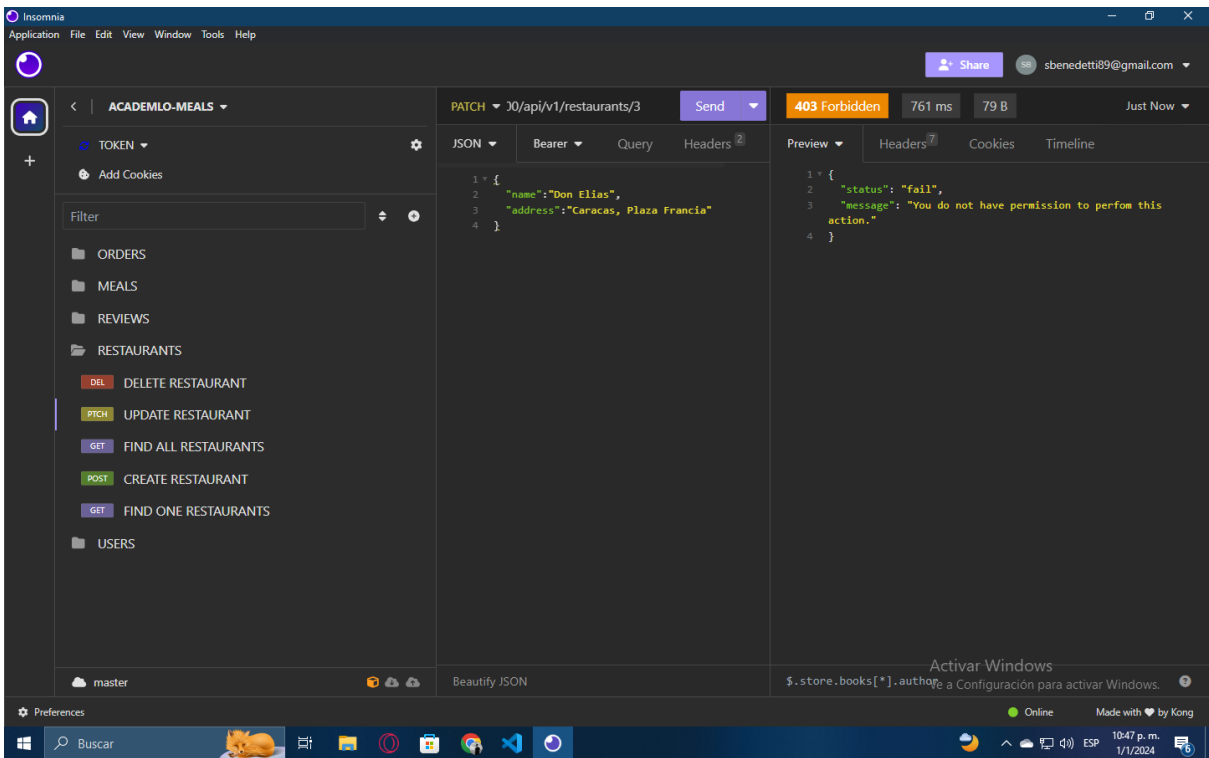
## FIND ONE RESTAURANT



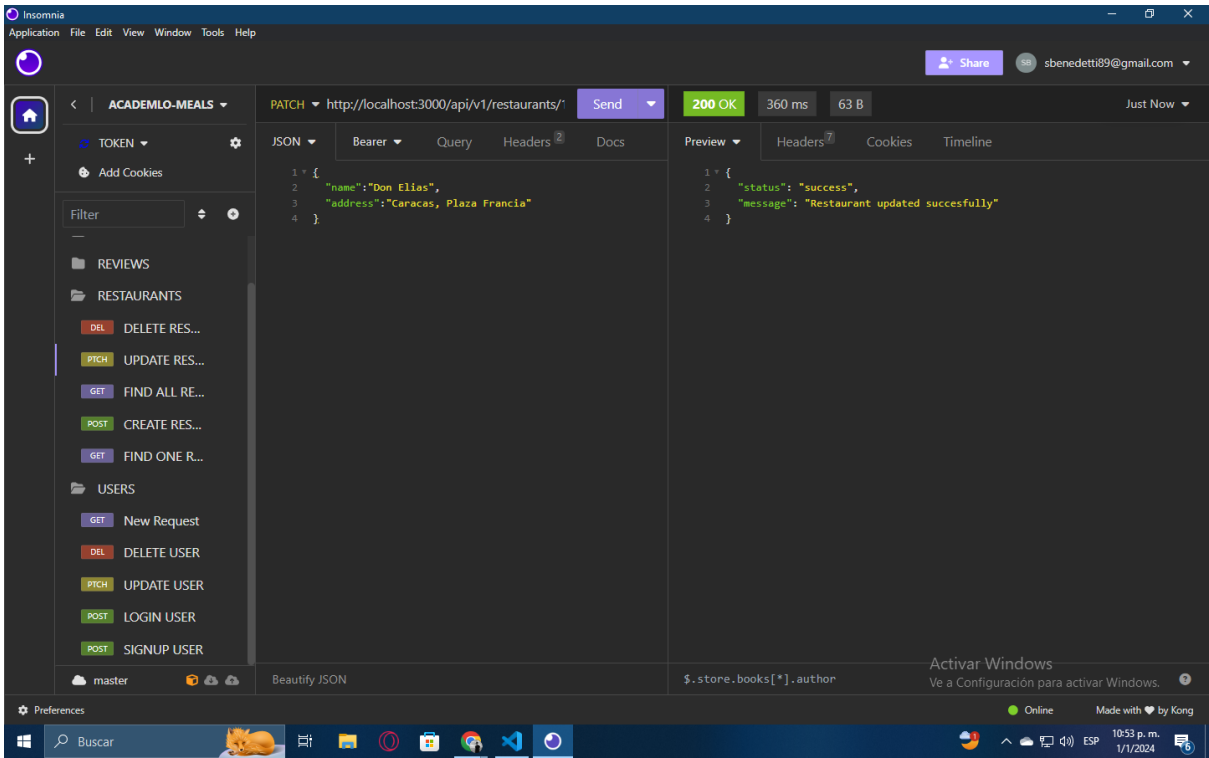
# FIND ALL RESTAURANTS



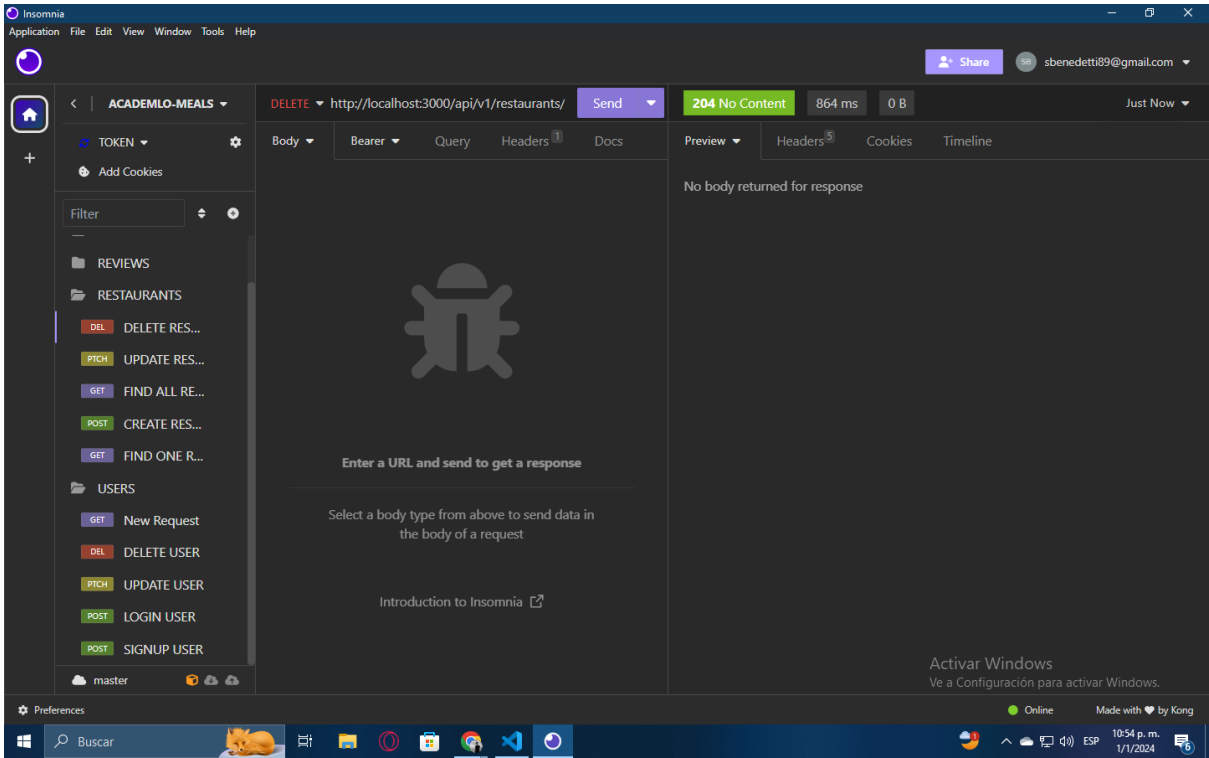
# RESTRICT TO



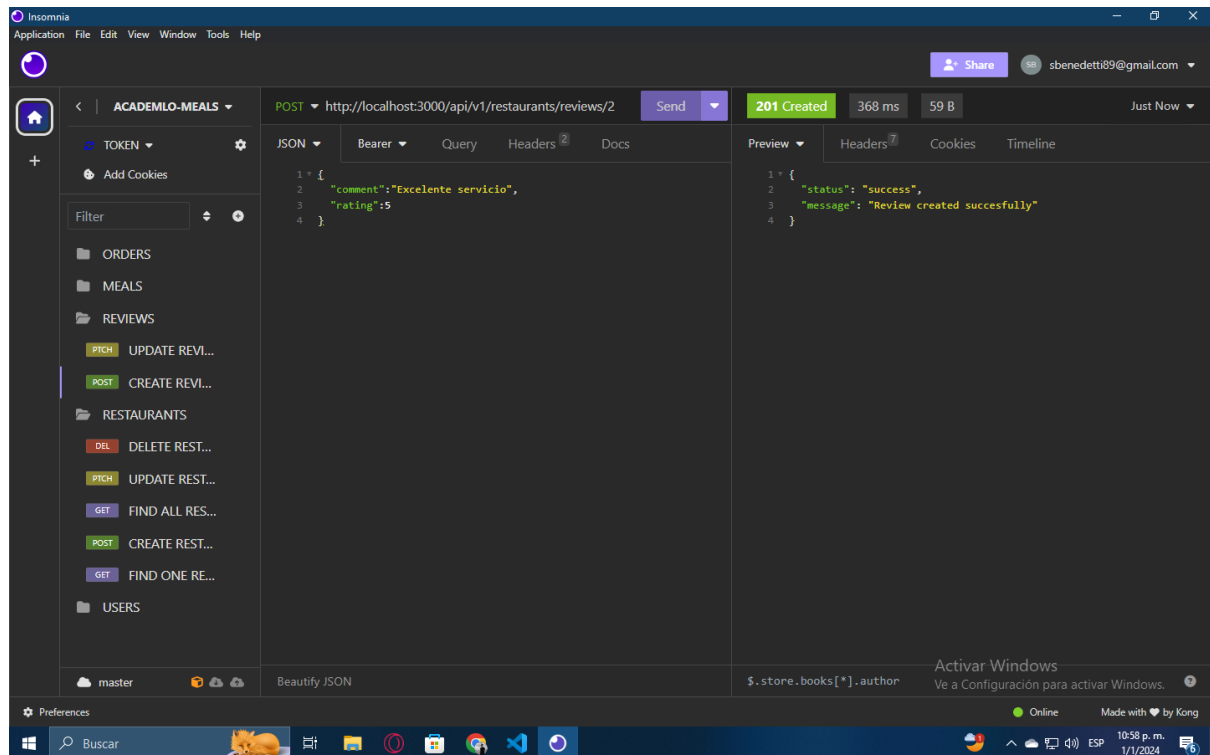
# UPDATE RESTAURANT



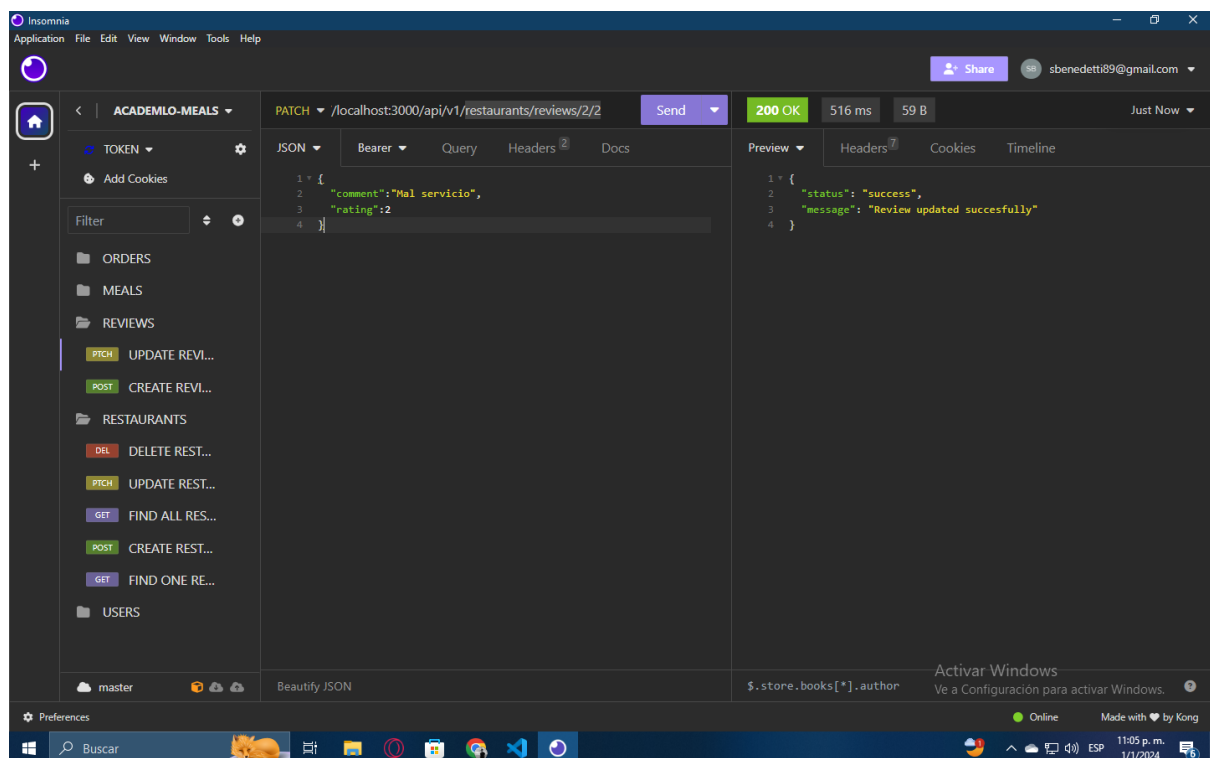
# DELETE RESTAURANT



## POST CREATE REVIEW

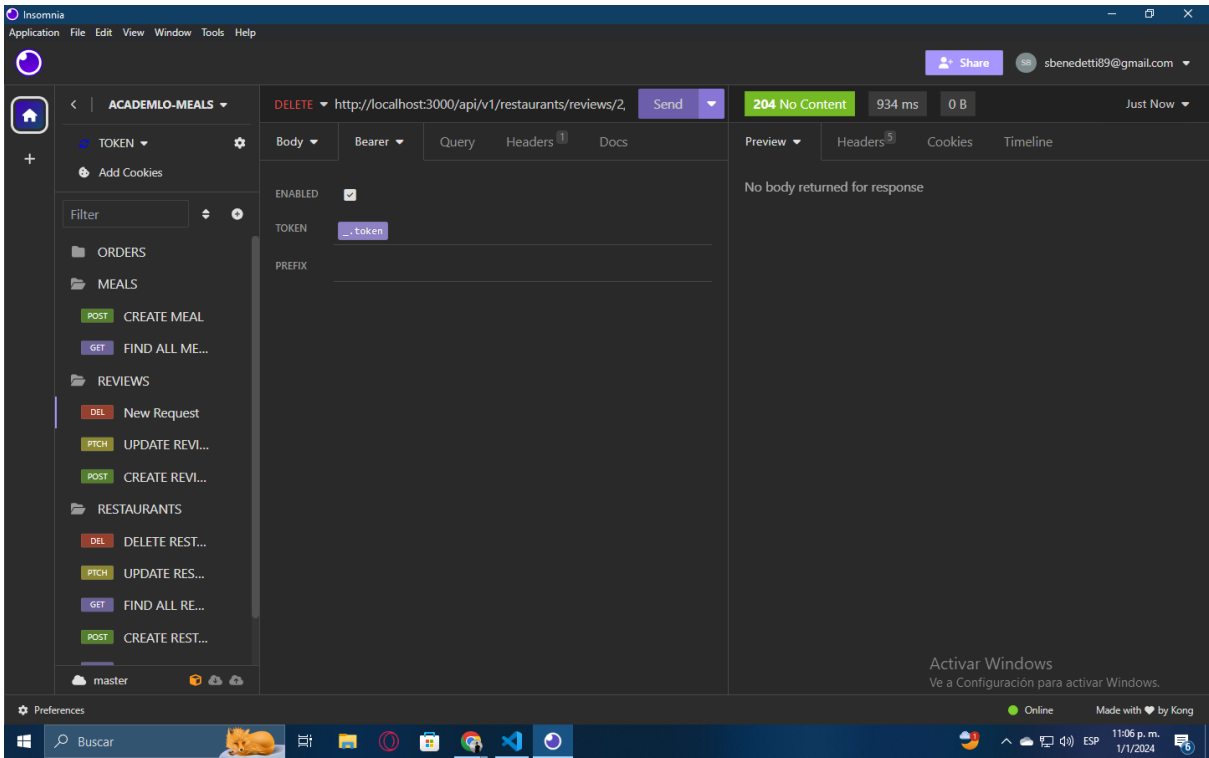


## UPDATE REVIEW



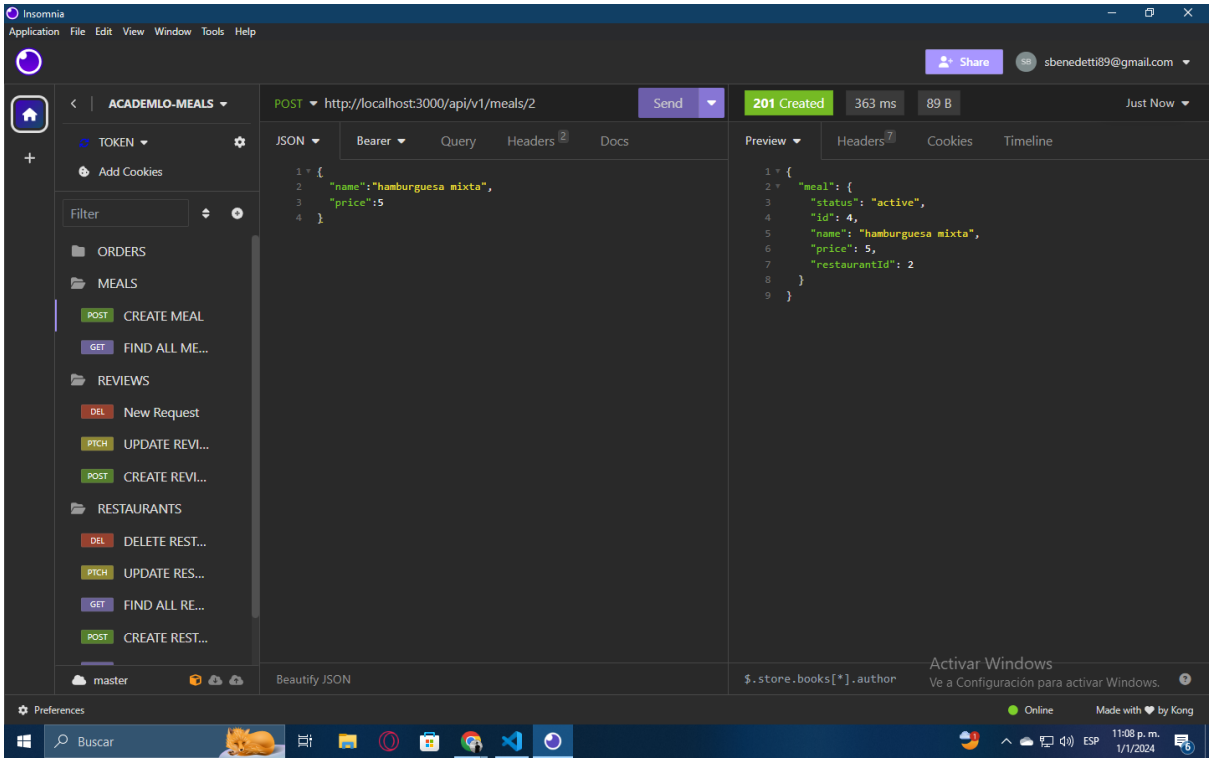


# DELETE REVIEW

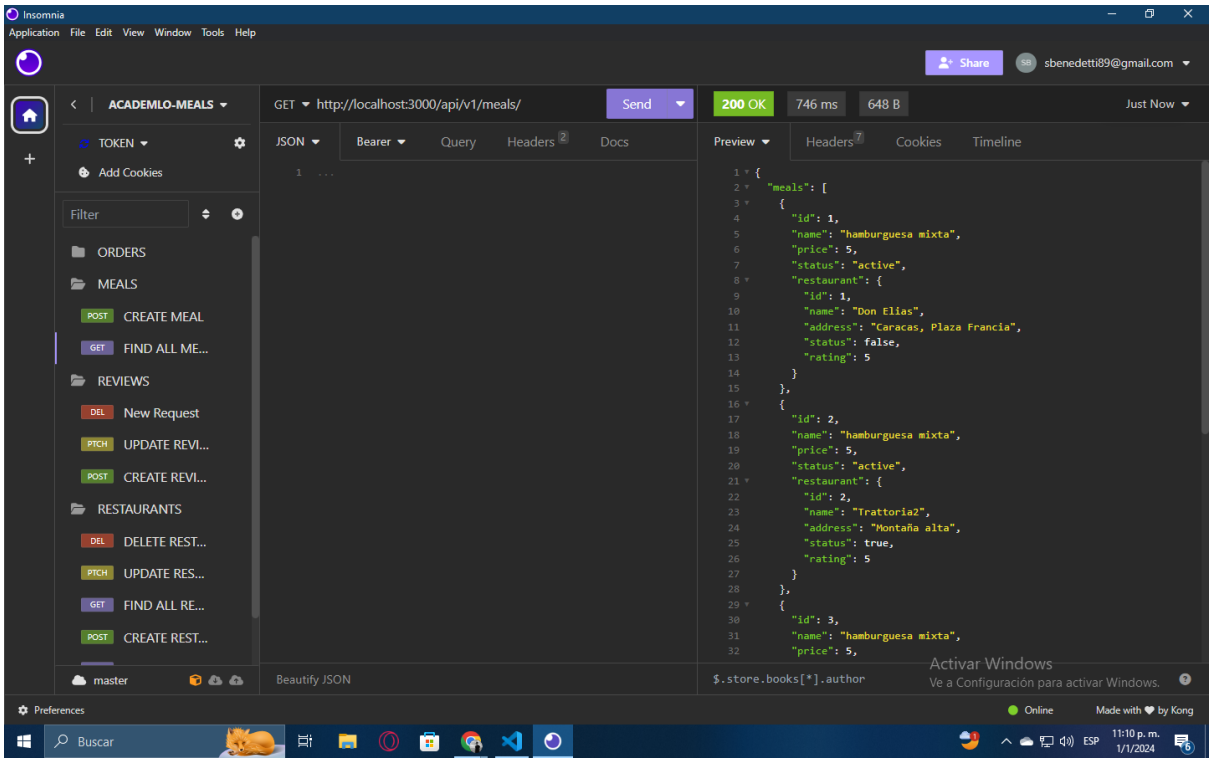


# MEALS

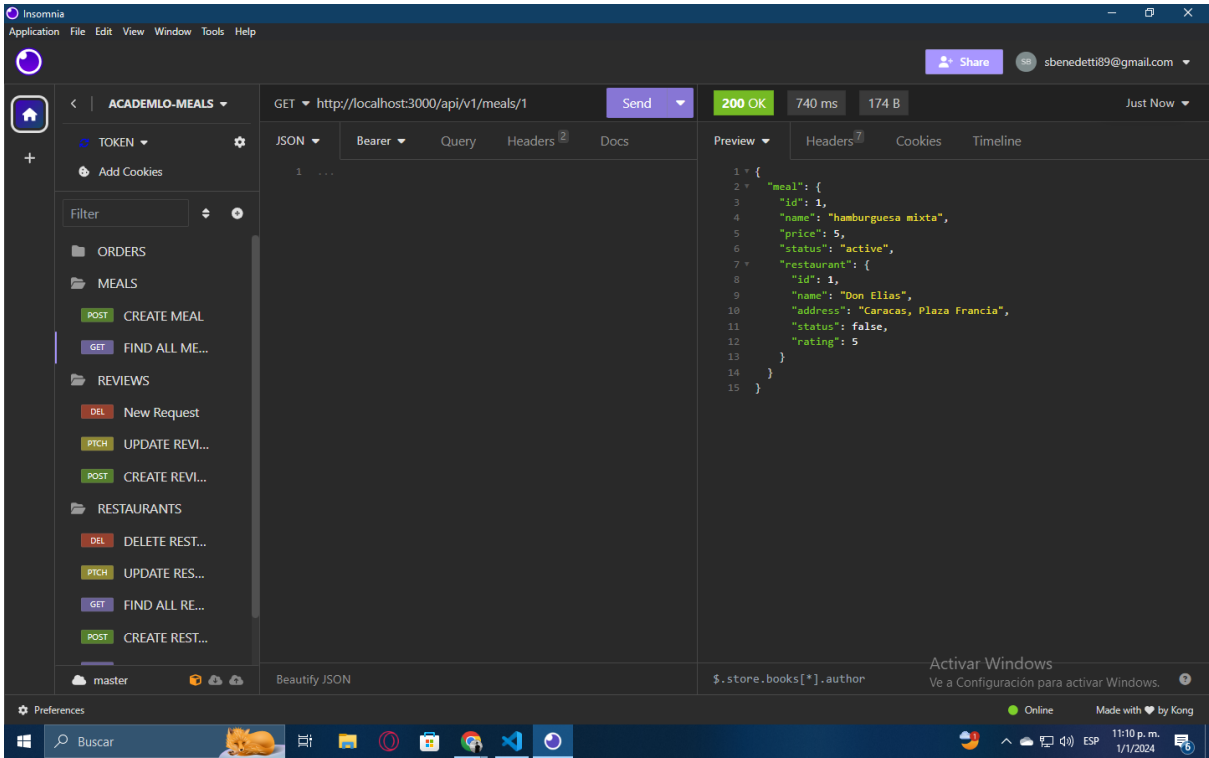
## POST CREATE MEAL



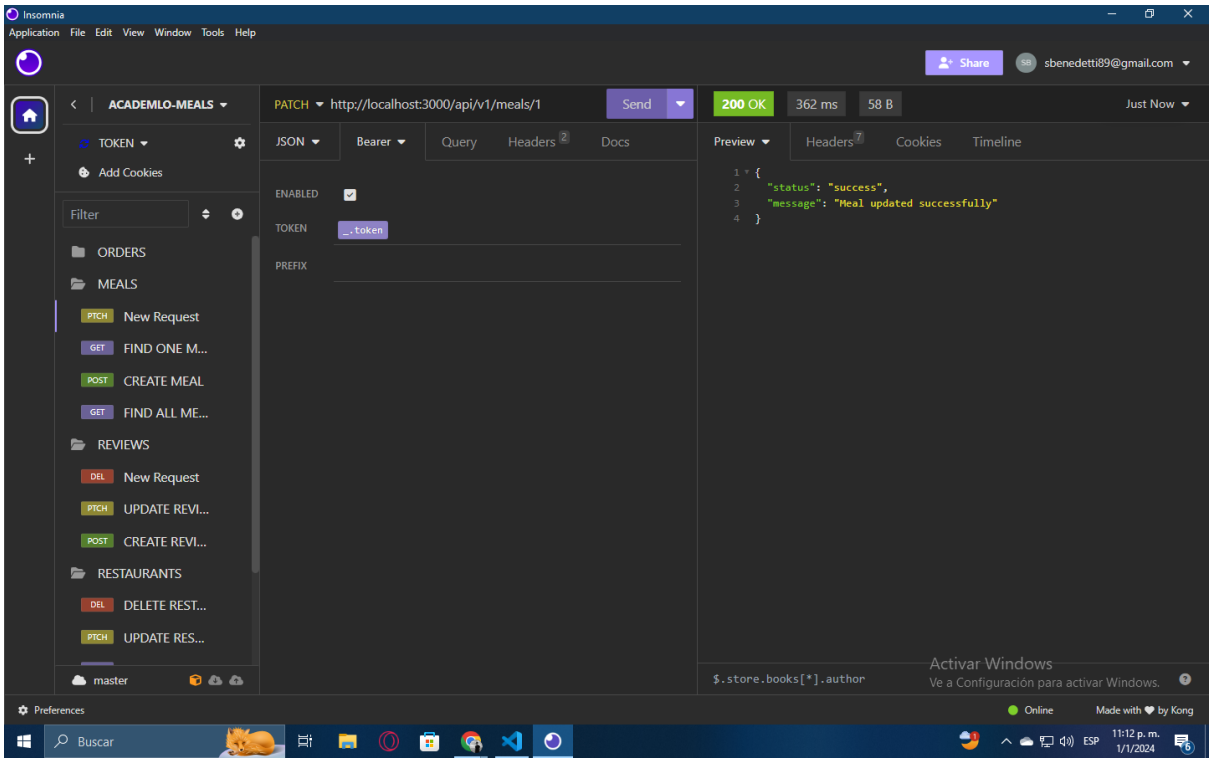
# FIND ALL MEALS



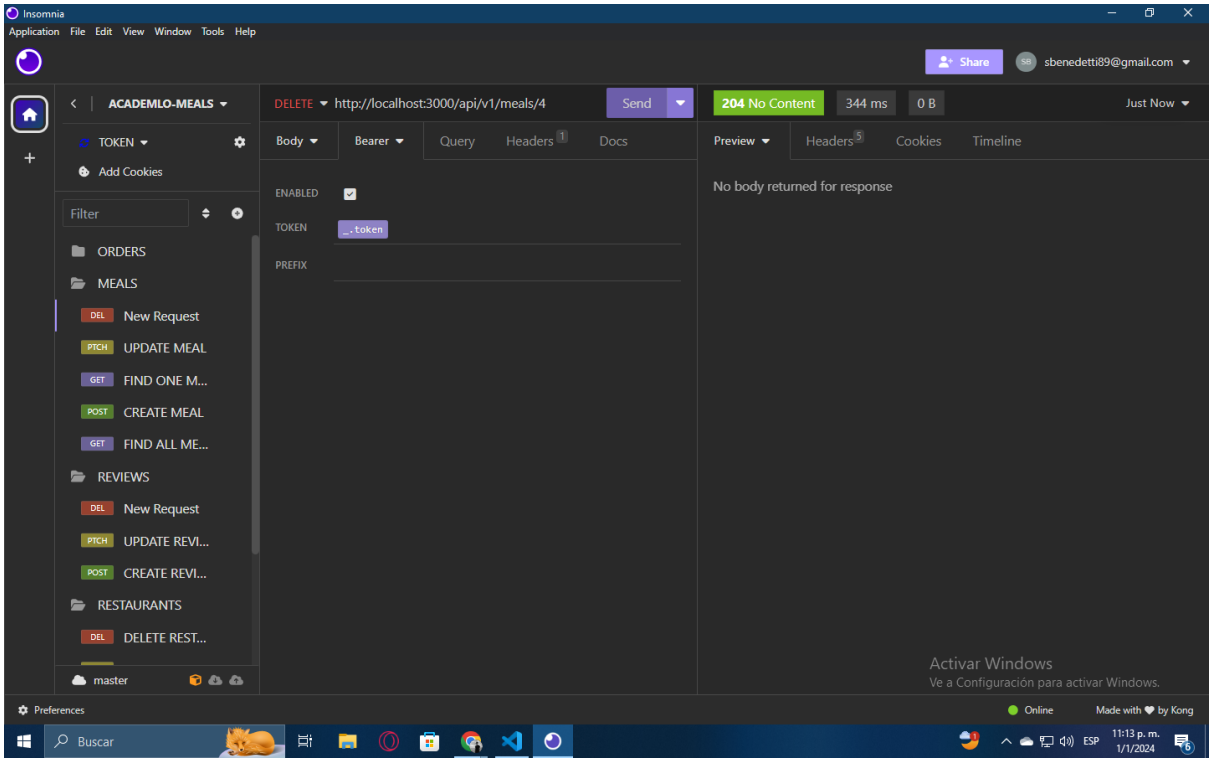
# FIND ONE MEAL



# UPDATE MEAL

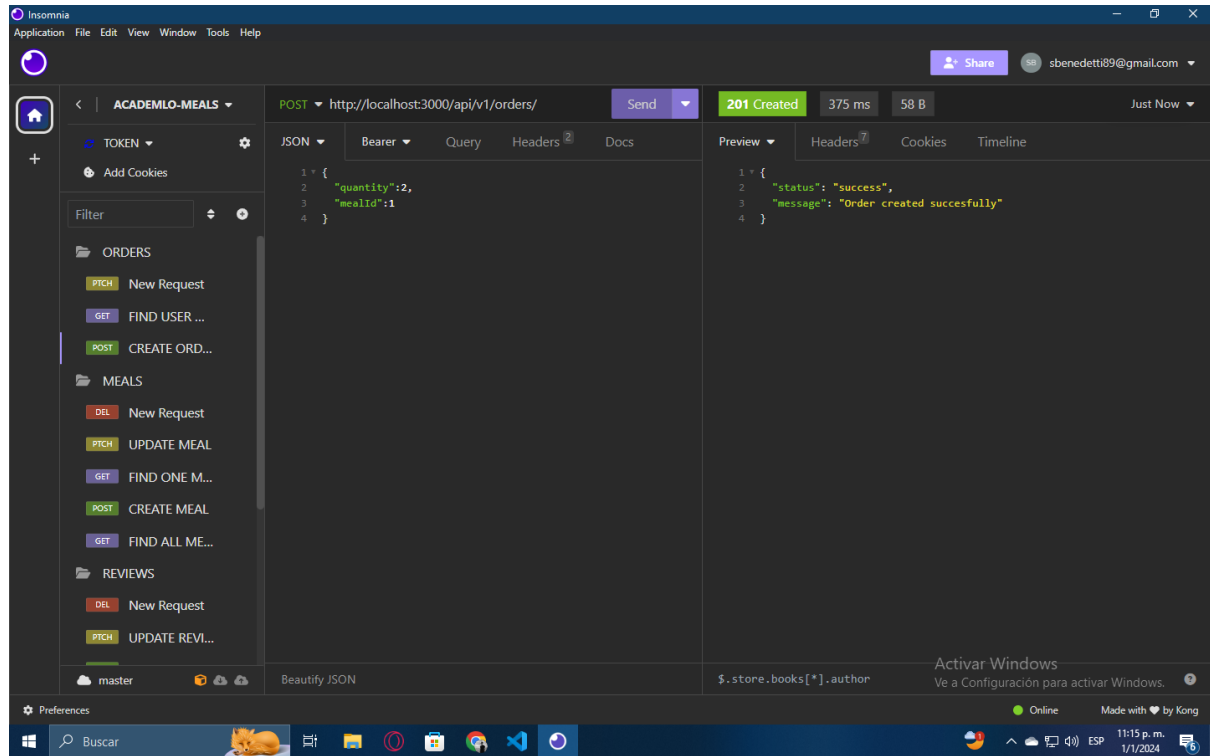


# DELETE MEAL

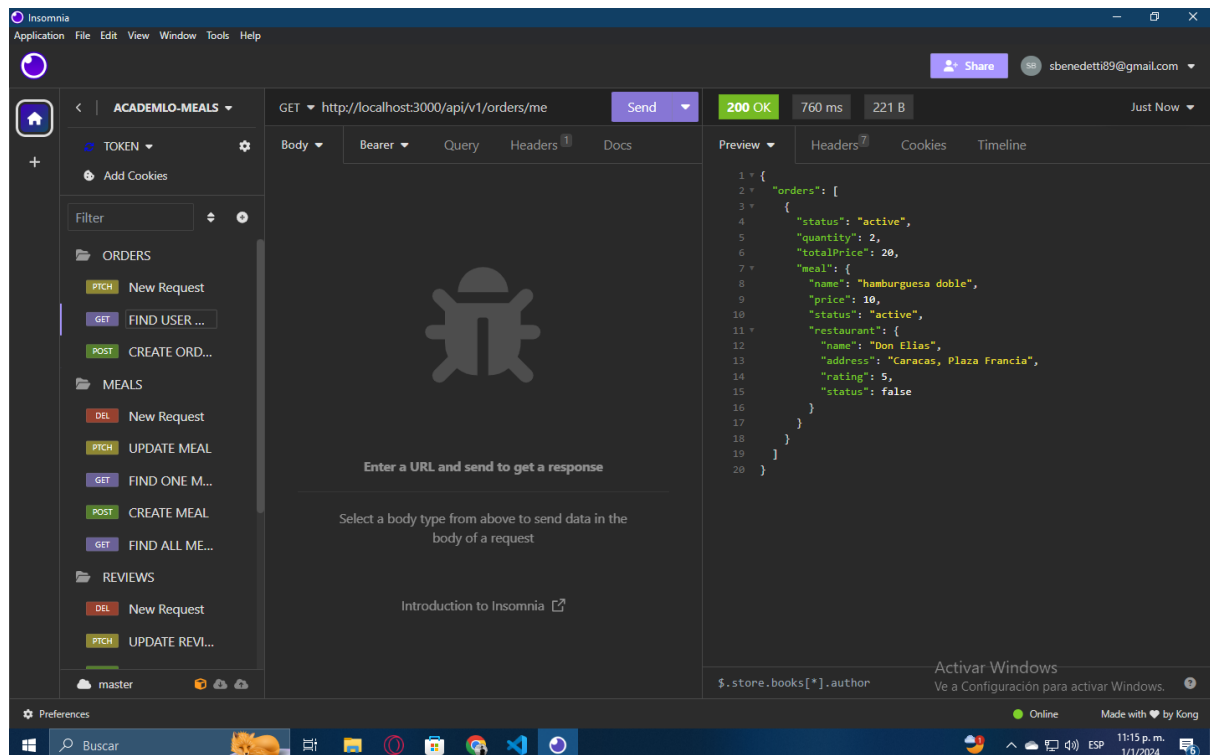


# ORDERS

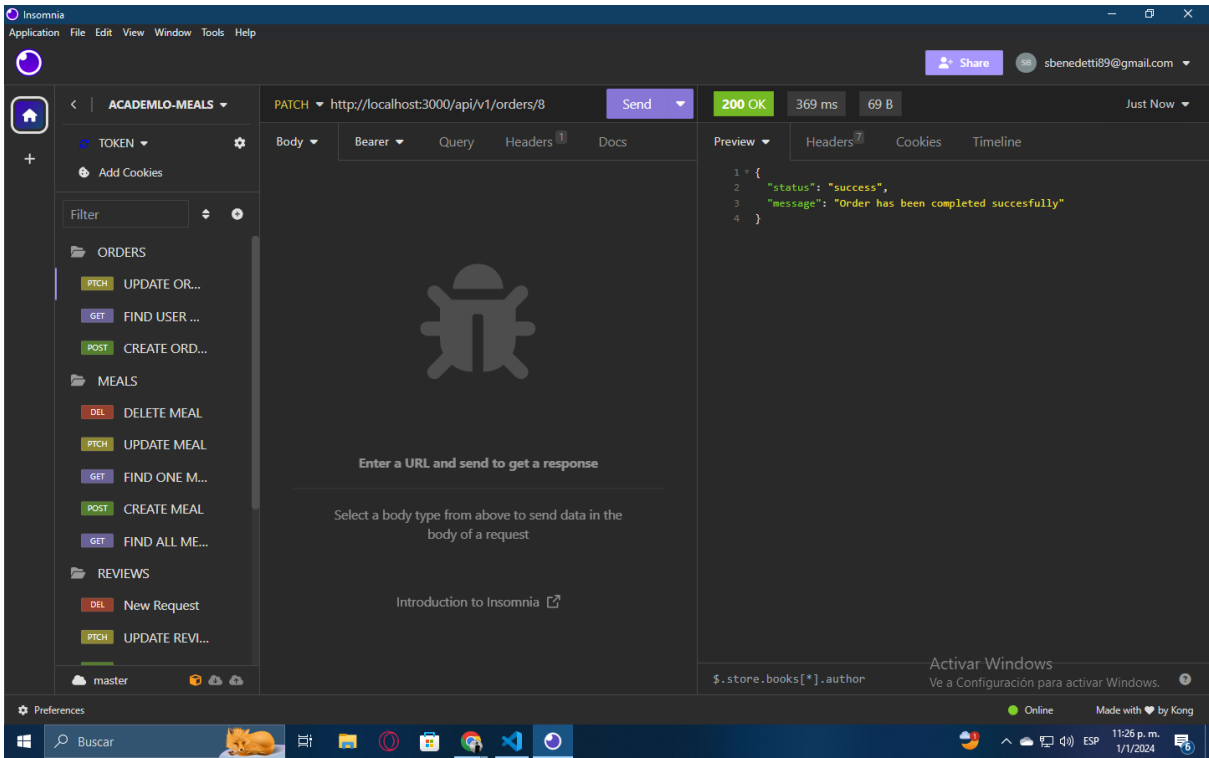
## CREATE ORDER



## FIND SESSION USER ORDER



# UPDATE ORDER



# DELETE ORDER

