

PROJEK AKHIR PASTI

**11322005 - MARIA E. SIBARANI
11322015 - RIVAE L H. MANURUNG
11322031 - DANIEL P. MANALU
11322044 - KRISTINA SITORUS**

- Kelompok 15 -

Contents

1. Detail Proyek
2. Deskripsi Umum
3. Microservice dan Monolith
4. Fungsi Aplikasi
5. Diagram
6. Pengujian Service
7. Demo

Detail Proyek

1. Setiap kasus harus menerapkan Microservice.
2. Bahasa pemrograman minimal 2 jenis dan wajib menggunakan bahasa Go (cth: Go & Java / Go & PHP / dll).
3. Banyak service menyesuaikan minimum table atau minimum fitur atau minimum requirement PA2.
4. Messaging protocol yang digunakan adalah REST API.
5. Untuk app consumer, dibebaskan menggunakan web-based.

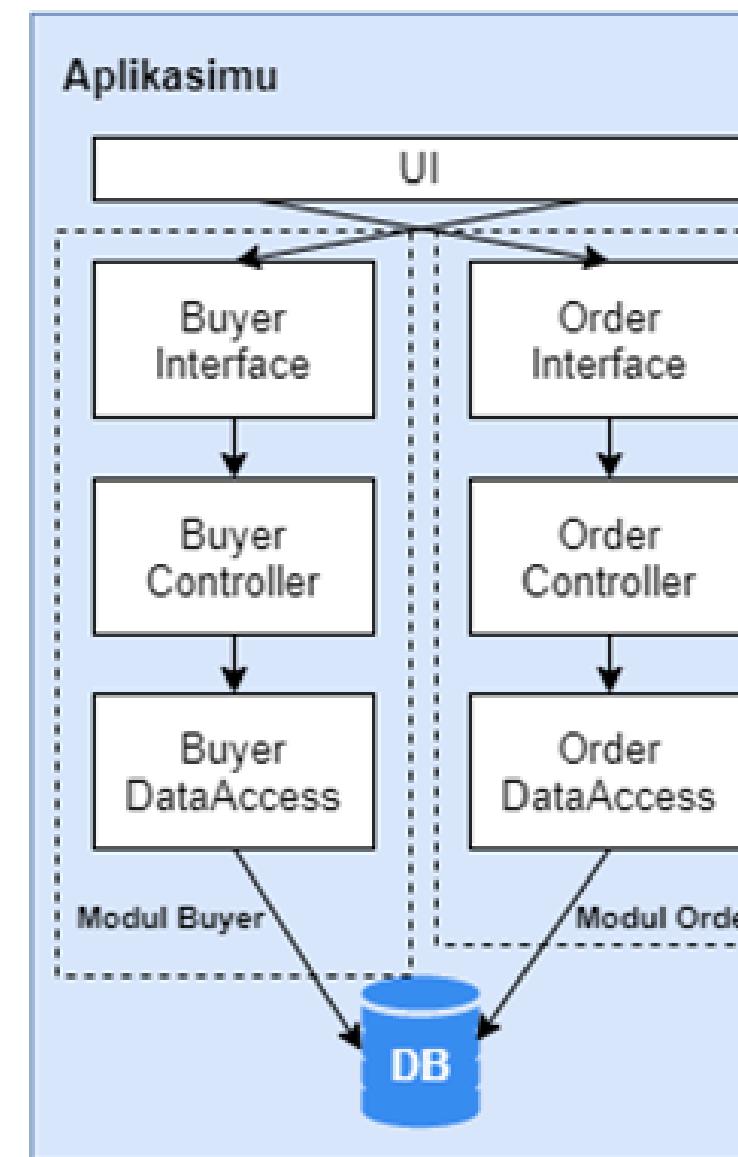
DESKRIPSI UMUM APLIKASI

Aplikasi mobile ini dirancang untuk manajemen dan reservasi online, serta dapat digunakan oleh admin dan customer untuk memasarkan dan membeli produk. Dalam aplikasi ini, admin mengelola produk yang akan dijual, sedangkan pelanggan dapat melakukan pemesanan produk. Aplikasi ini dibangun menggunakan tiga bahasa pemrograman, dengan bagian back-end dan front-end menggunakan bahasa yang berbeda. Back-end dibangun dengan bahasa GO, sementara front-end menggunakan PHP (untuk admin) dan DART (untuk customer). Selain itu, aplikasi ini juga menggunakan teknologi Flutter, GoFiber, Apache, dan Laravel. Berikut adalah penjelasan lebih detail mengenai teknologi yang digunakan:

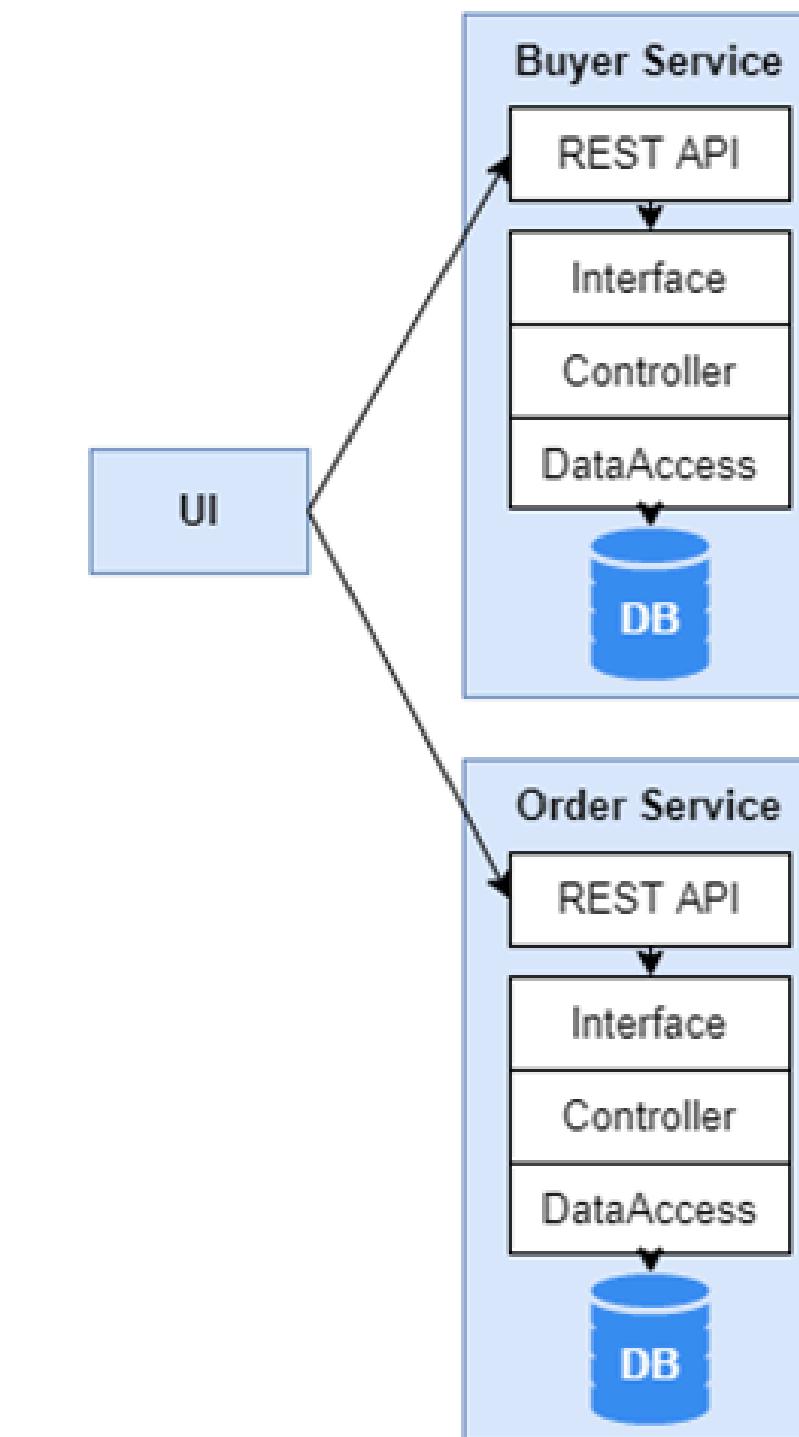
1. Flutter: Framework yang digunakan untuk membangun aplikasi menggunakan bahasa pemrograman DART.
2. GoFiber: Framework yang digunakan untuk membangun aplikasi menggunakan bahasa pemrograman GO atau Golang.
3. Apache: Web server yang digunakan untuk melayani permintaan HTTP dan mengirimkan konten web kepada pengguna berdasarkan permintaan.
4. Laravel: Framework yang digunakan untuk membangun aplikasi menggunakan bahasa pemrograman PHP.

PERBANDINGAN MONOLITH DENGAN MICROSERVICES

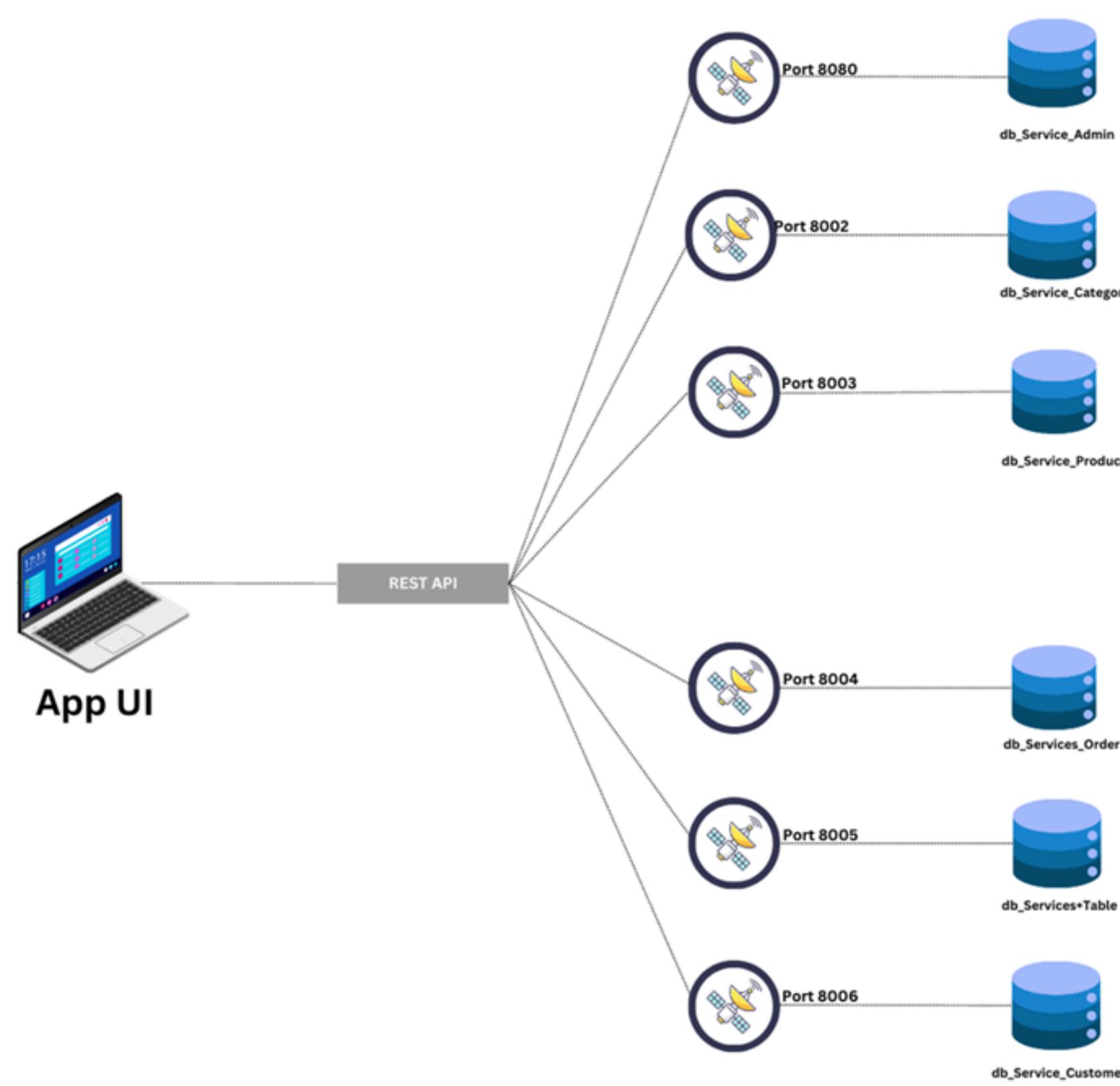
MONOLITH



MICROSERVICES



ARSITEKTUR MICROSERVICES

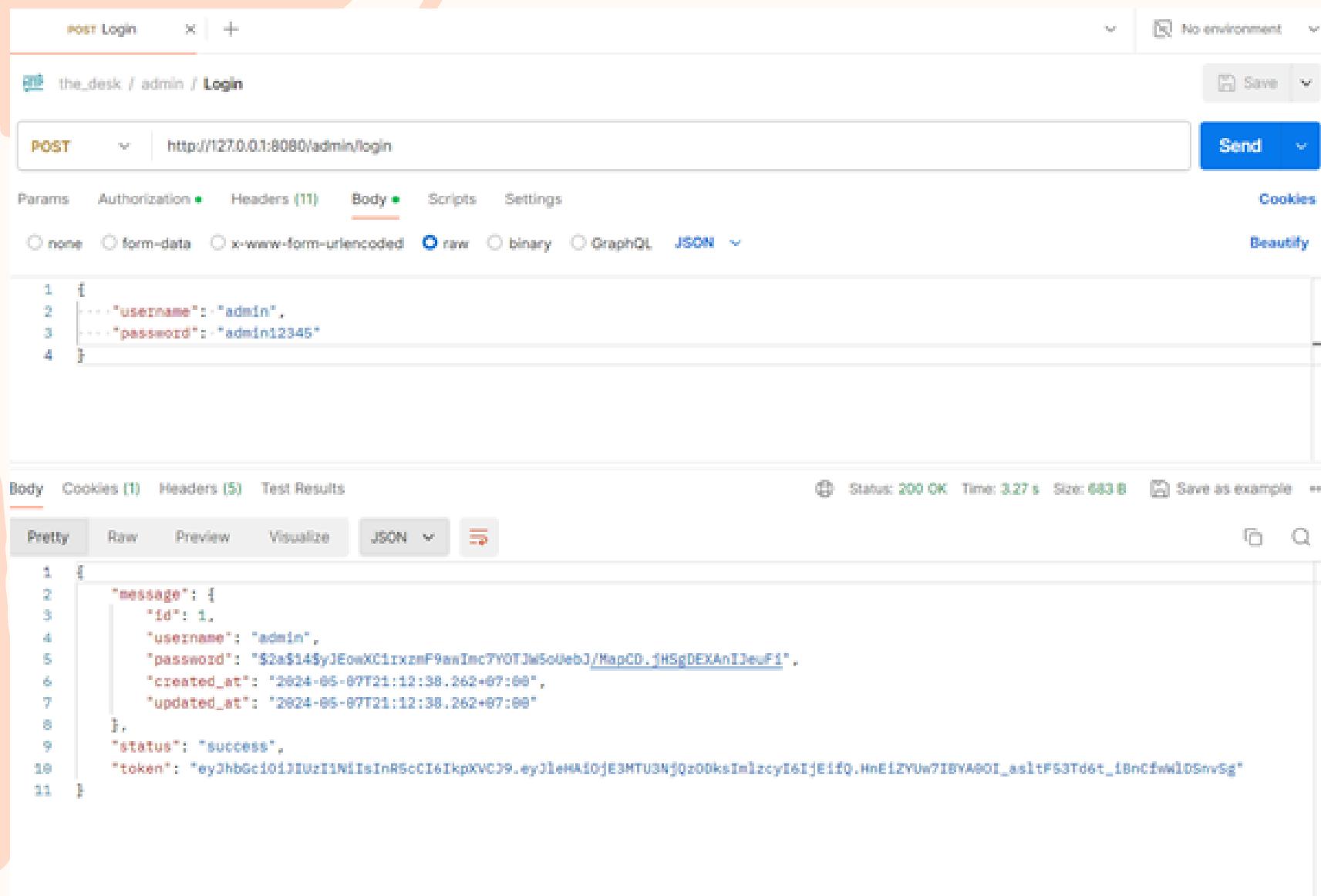


Fungsi Aplikasi

1. Fungsi Autentikasi
2. Fungsi Mengelola Product bagi Admin
3. Fungsi Mengelola Category Product bagi Admin
4. Fungsi Mengelola Table bagi Admin
5. Fungsi Melihat Order bagi Admin
6. Fungsi Melihat Product Customer
7. Fungsi Melihat Category Customer

Pengujian Service

POST LOGIN



Post Login | POST http://127.0.0.1:8080/admin/login

Params Authorization Headers (11) Body Scripts Settings

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

```
1: {  
2:   ... "username": "admin",  
3:   ... "password": "admin12345"  
4: }
```

Body Cookies (1) Headers (5) Test Results

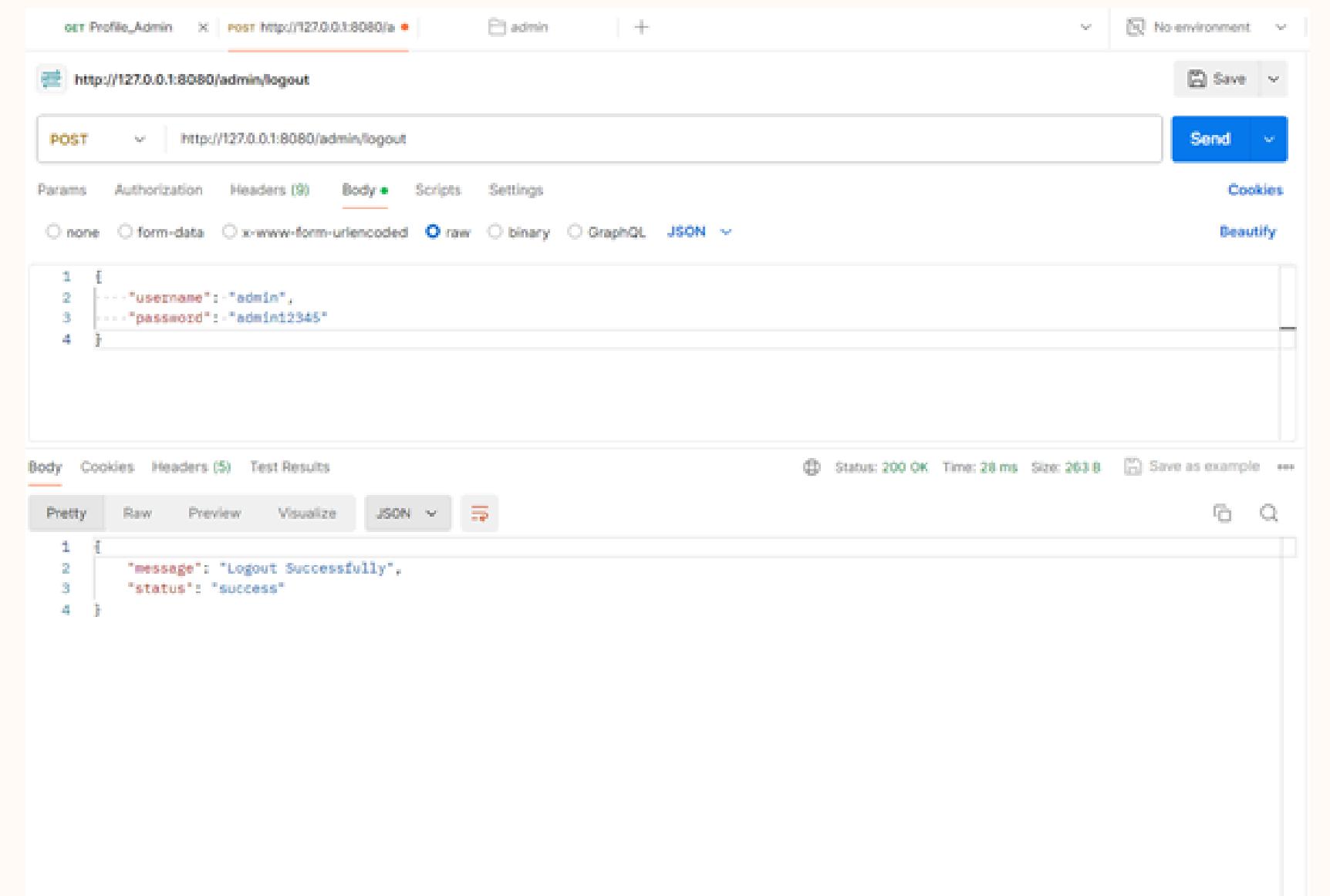
Status: 200 OK Time: 3.27 s Size: 643 B Save as example

Pretty Raw Preview Visualize JSON

```
1: {  
2:   "message": {  
3:     "id": 1,  
4:     "username": "admin",  
5:     "password": "S2a$14$yJEowXCinxzxF9enImc7YOTJwSoUebJ/MaP0D.jHSgDEXAnI3euF1",  
6:     "created_at": "2024-05-07T21:12:38.262+07:00",  
7:     "updated_at": "2024-05-07T21:12:38.262+07:00"  
8:   },  
9:   "status": "success",  
10:  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVC99.yJleHA10jE3HTU3NjQz00ksImLzcyI6IjEif0.HnElZYUw7IBYA801_as1tFS3Td6t_iBnCfwWlDSmVsg"  
11: }
```

AUTENTIKASI ADMIN

POST LOGOUT



get Profile_Admin | POST http://127.0.0.1:8080/a | admin

http://127.0.0.1:8080/admin/logout

POST http://127.0.0.1:8080/admin/logout

Params Authorization Headers (9) Body Scripts Settings

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

```
1: {  
2:   ... "username": "admin",  
3:   ... "password": "admin12345"  
4: }
```

Body Cookies Headers (5) Test Results

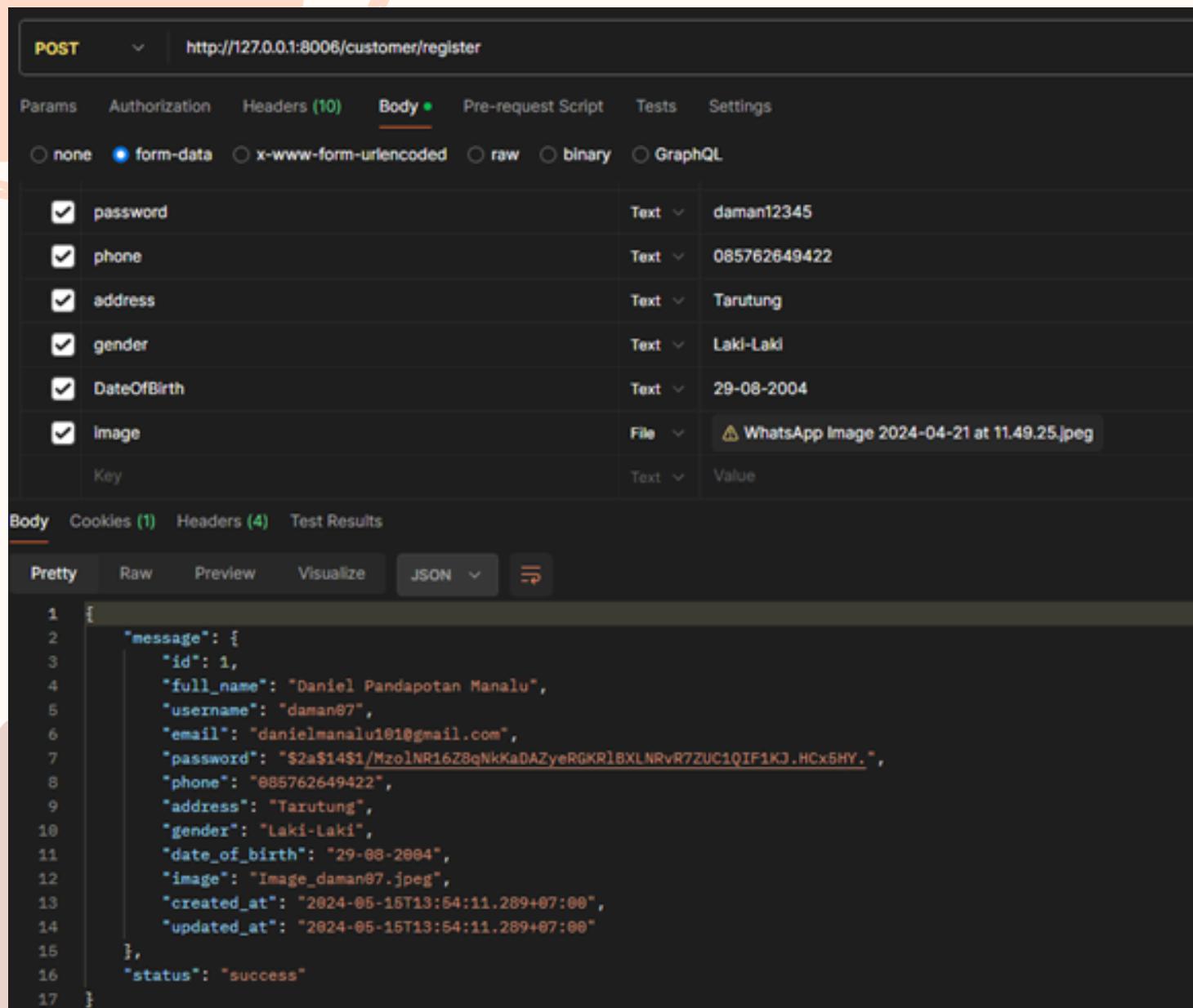
Status: 200 OK Time: 28 ms Size: 263 B Save as example

Pretty Raw Preview Visualize JSON

```
1: {  
2:   "message": "Logout Successfully",  
3:   "status": "success"  
4: }
```

Pengujian Service

POST REGISTRASI



POST http://127.0.0.1:8006/customer/register

Body (form-data)

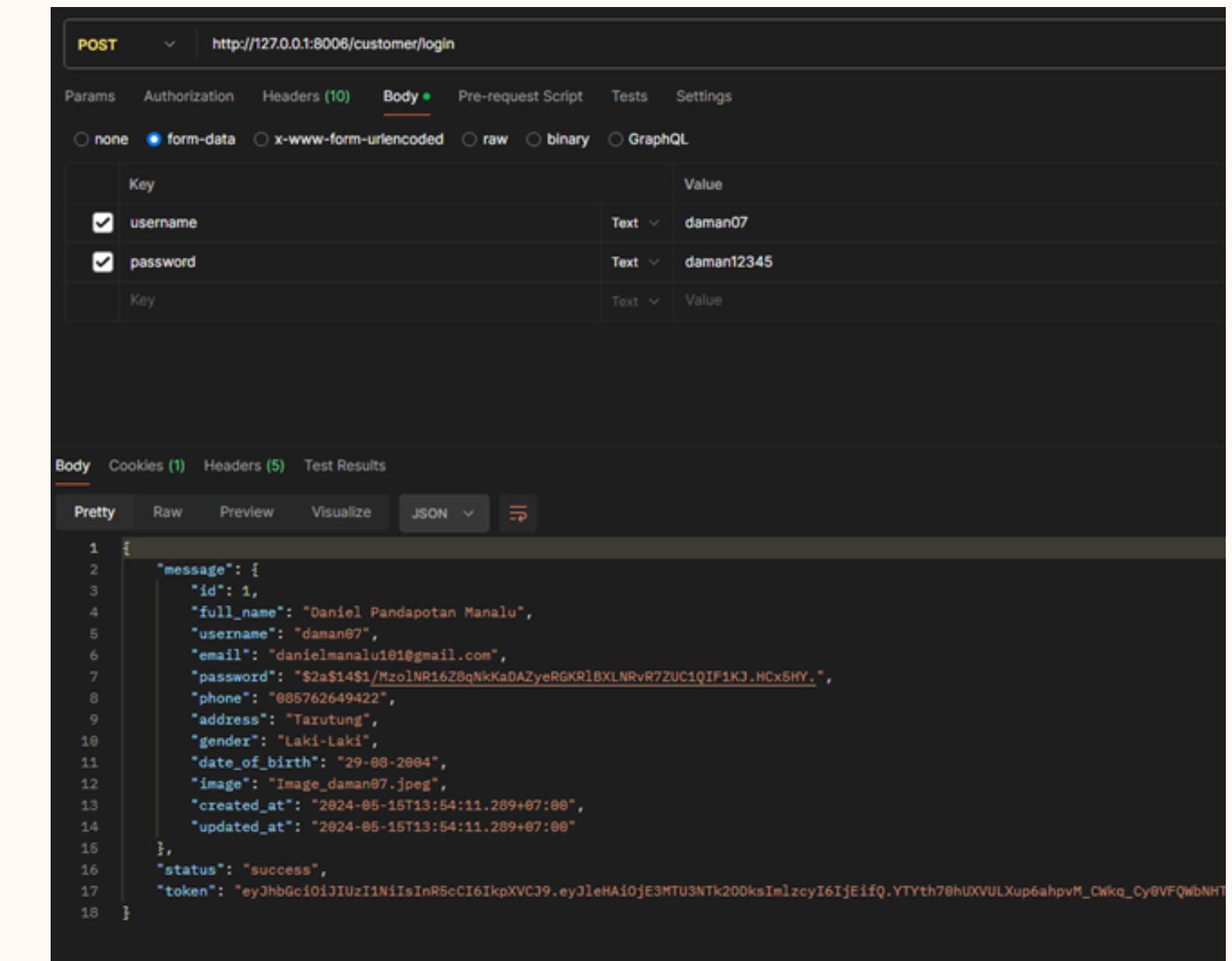
password	Text	daman12345
phone	Text	085762649422
address	Text	Tarutung
gender	Text	Laki-Laki
DateOfBirth	Text	29-08-2004
image	File	WhatsApp Image 2024-04-21 at 11.49.25.jpeg

Body (Pretty)

```
1 {
2   "message": {
3     "id": 1,
4     "full_name": "Daniel Pandapotan Manalu",
5     "username": "daman07",
6     "email": "danielmanalu101@gmail.com",
7     "password": "$2a$14$1/MzolNR16Z8qNkKaDAZyeRGKR1BXLNrvR7ZUC1QIF1KJ.HCx5HY.",
8     "phone": "085762649422",
9     "address": "Tarutung",
10    "gender": "Laki-Laki",
11    "date_of_birth": "29-08-2004",
12    "image": "Image_daman07.jpeg",
13    "created_at": "2024-05-15T13:54:11.289+07:00",
14    "updated_at": "2024-05-15T13:54:11.289+07:00"
15  },
16  "status": "success"
17}
```

AUTENTIKASI CUSTOMER

POST LOGIN



POST http://127.0.0.1:8006/customer/login

Body (form-data)

username	Text	daman07
password	Text	daman12345

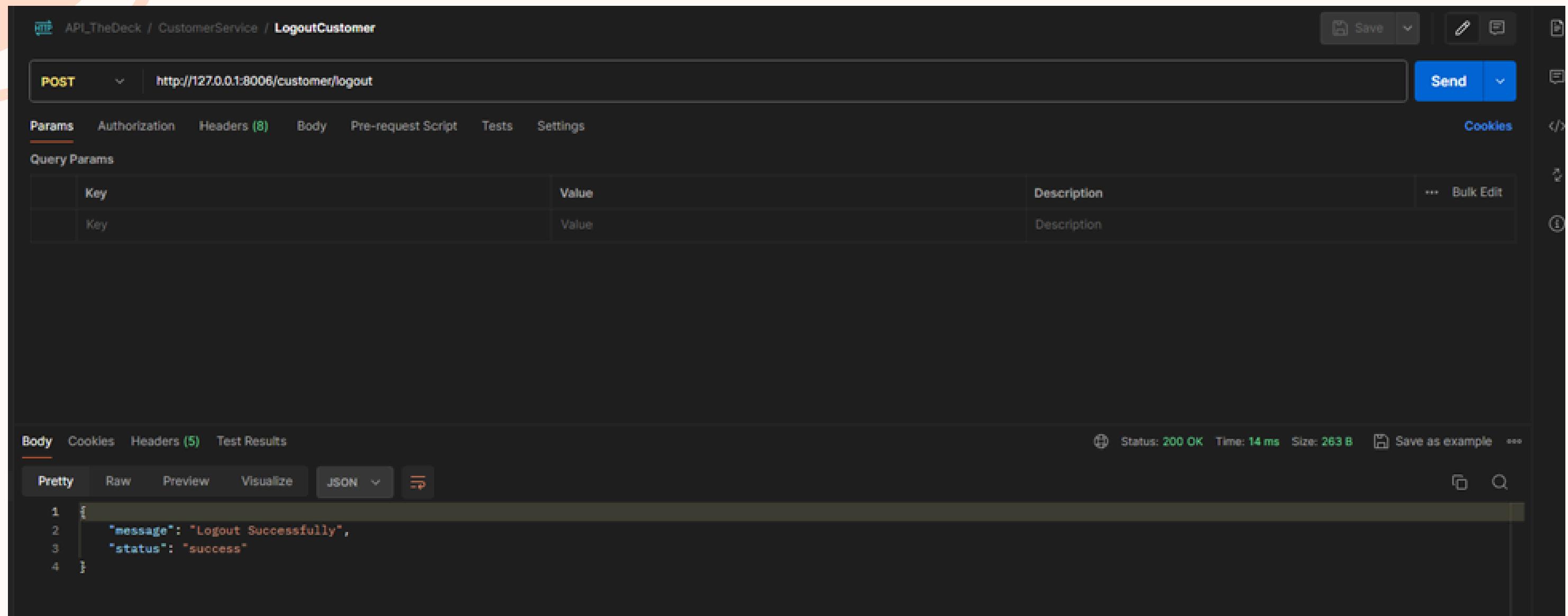
Body (Pretty)

```
1 {
2   "message": {
3     "id": 1,
4     "full_name": "Daniel Pandapotan Manalu",
5     "username": "daman07",
6     "email": "danielmanalu101@gmail.com",
7     "password": "$2a$14$1/MzolNR16Z8qNkKaDAZyeRGKR1BXLNrvR7ZUC1QIF1KJ.HCx5HY.",
8     "phone": "085762649422",
9     "address": "Tarutung",
10    "gender": "Laki-Laki",
11    "date_of_birth": "29-08-2004",
12    "image": "Image_daman07.jpeg",
13    "created_at": "2024-05-15T13:54:11.289+07:00",
14    "updated_at": "2024-05-15T13:54:11.289+07:00"
15  },
16  "status": "success",
17  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOiJE3MTU3NTk200ksImlzcyI6IjEifQ.YTYth70hUXVULXup6ahpvM_Cwkg_Cy0VFQWbNHT"
18}
```

Pengujian Service

AUTENTIKASI
CUSTOMER

POST LOGOUT



The screenshot shows a POST request to the `LogoutCustomer` endpoint. The URL is `http://127.0.0.1:8006/customer/logout`. The response status is 200 OK, with a time of 14 ms and a size of 263 B. The response body is:

```
1: {  
2:   "message": "Logout Successfully",  
3:   "status": "success"  
4: }
```

Pengujian Service

CREATE

POST Login | POST Create_Product | + | No environment | Save | POST Create_Product

the_desk / product / Create_Product

POST http://127.0.0.1:8003/product/create

Send

Params Authorization Headers (11) Body Scripts Settings Cookies

None form-data x-www-form-urlencoded raw binary GraphQL

description Text ini minuman

price Text 12000

category_id Text 3

image File

Description

Body Cookies (1) Headers (4) Test Results

Status: 200 OK Time: 63 ms Size: 351 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": [
3     {
4       "id": 3,
5       "name": "aqua",
6       "description": "ini minuman",
7       "image": "Image_aqua.png",
8       "price": "12000",
9       "category_id": 3,
10      "created_at": "2024-06-15T14:37:30.794+07:00",
11      "updated_at": "2024-06-15T14:37:30.794+07:00"
12    },
13    {
14      "status": "success"
15    }
16  ]
17}
```

POST Login | POST Create_Product | GET Get_Product | GET Get_ProductById | + | No environment | Save | GET All

the_desk / product / Get_Product

GET http://127.0.0.1:8003/product/

Send

Params Authorization Headers (0) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies (1) Headers (4) Test Results

Status: 200 OK Time: 8 ms Size: 765 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": [
3     {
4       "id": 1,
5       "name": "Bakso",
6       "description": "ini bakso",
7       "image": "Image_Bakso.png",
8       "price": "12000",
9       "category_id": 3,
10      "created_at": "2024-06-11T21:02:28.697+07:00",
11      "updated_at": "2024-06-11T21:02:28.697+07:00"
12    },
13    {
14      "id": 2,
15      "name": "Ikan bakar",
16      "description": "ini ikan bakar",
17      "image": "Image_Ikan_bakar.jpeg",
18      "price": "45000"
19    }
20  ]
21}
```

F8 Postman F7 Runner ⌘ Start Prev ⌘ Cookies ⌘ Vault ⌘ Trash ⌘

Pengujian Service

GET BY ID

The screenshot shows the Postman interface with a GET request to `http://127.0.0.1:8003/product/1`. The response status is 200 OK, time is 7 ms, and size is 351 B. The response body is:

```
1 {  
2   "message": {  
3     "id": 1,  
4     "name": "Bakso",  
5     "description": "ini bakso",  
6     "image": "Image_Bakso.png",  
7     "price": "12000",  
8     "category_id": 2,  
9     "created_at": "2024-05-11T21:02:28.697+07:00",  
10    "updated_at": "2024-05-11T21:02:28.697+07:00"  
11  },  
12  "status": "success"  
13 }
```

PRODUK

UPDATE

The screenshot shows the Postman interface with a PUT request to `http://127.0.0.1:8003/product/3/edit`. The response status is 200 OK, time is 83 ms, and size is 365 B. The response body is:

```
1 {  
2   "message": {  
3     "id": 3,  
4     "name": "aqua",  
5     "description": "ini merupakan minuman aqua",  
6     "image": "Image_aqua.png",  
7     "price": "7000",  
8     "category_id": 3,  
9     "created_at": "2024-05-15T14:37:30.794+07:00",  
10    "updated_at": "2024-05-15T14:45:44.198+07:00"  
11  },  
12  "status": "success"  
13 }
```

Pengujian Service

PRODUK

DELETE

The screenshot shows a Postman collection named "DEL Delete_Product". The request URL is "the_desk / product / Delete_Product" and the method is "DELETE" directed at "http://127.0.0.1:8003/product/3/delete". The "Params" tab is selected, showing a single query parameter "Key". The response status is 200 OK, with a response body containing the JSON object: { "message": "Product deleted successfully!", "status": "success" }.

DEL Delete_Product

the_desk / product / Delete_Product

DELETE http://127.0.0.1:8003/product/3/delete

Params Authorization Headers (8) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	...

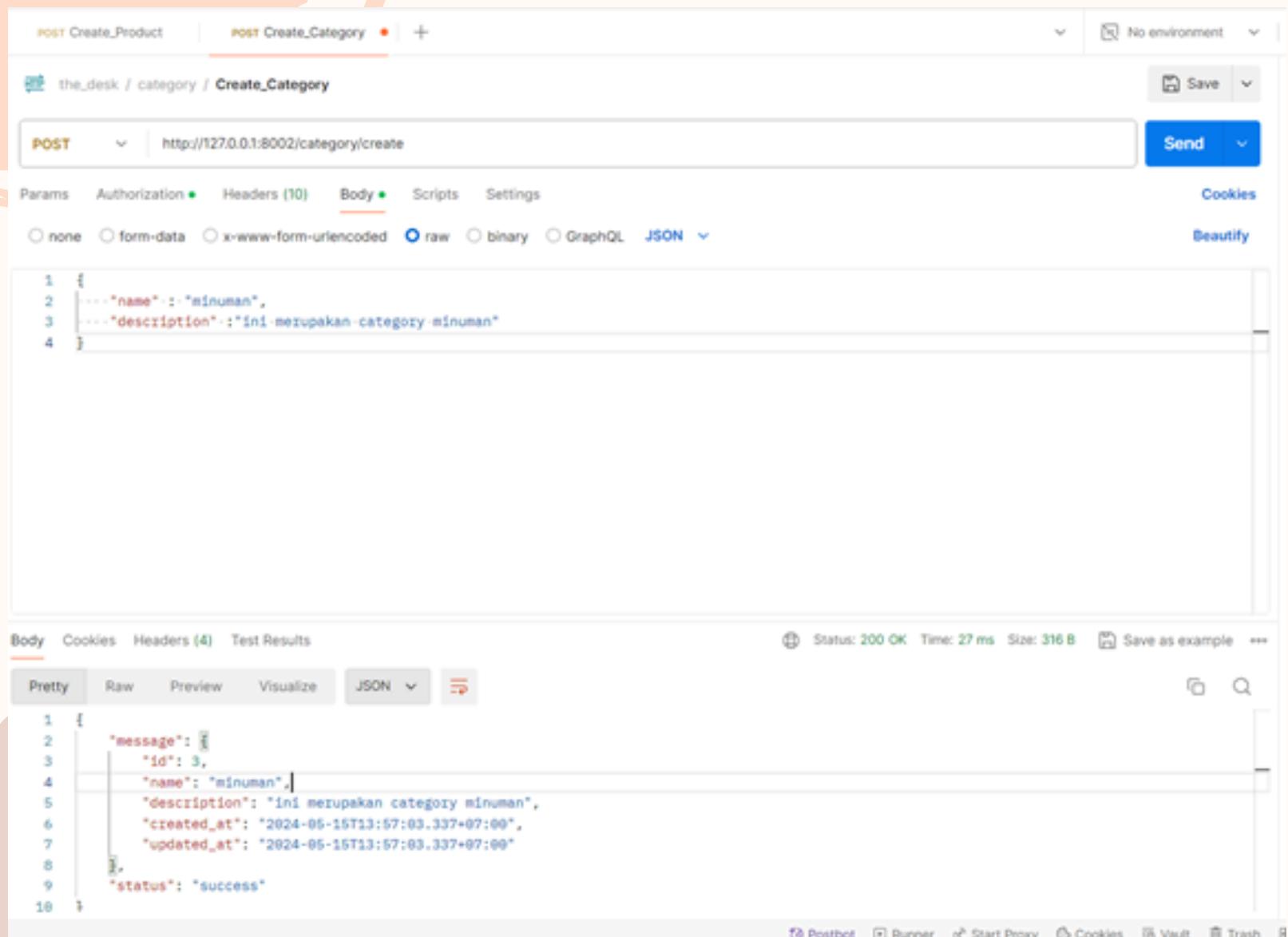
Body Cookies (1) Headers (4) Test Results Status: 200 OK Time: 11 ms Size: 184 B Save as example

Pretty Raw Preview Visualize JSON

```
1: {  
2:   "message": "Product deleted successfully!",  
3:   "status": "success"  
4: }
```

Pengujian Service

CREATE



The screenshot shows the Postman interface with a POST request to 'the_desk / category / Create_Category'. The URL is `http://127.0.0.1:8002/category/create`. The Body tab contains the following JSON:

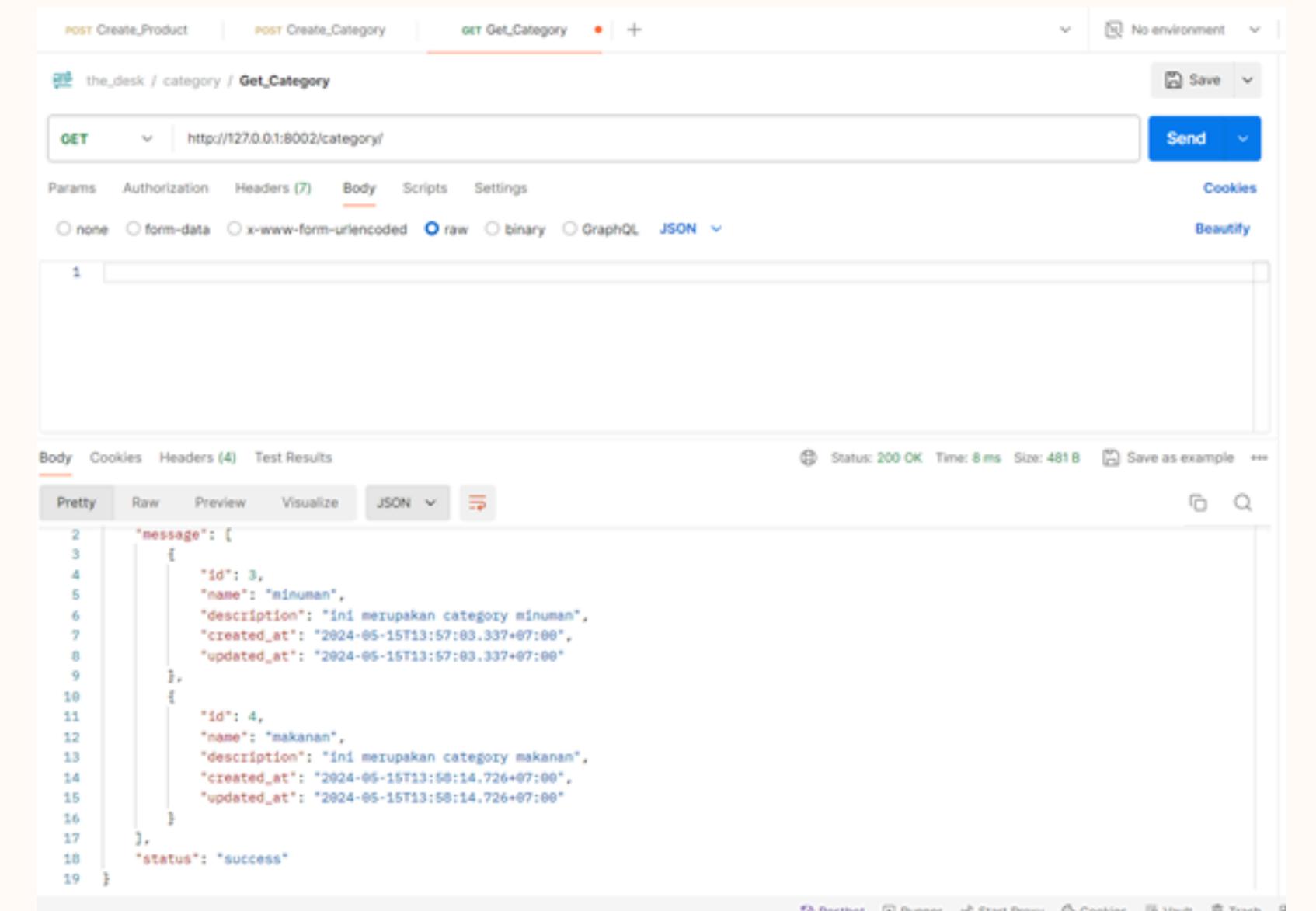
```
1 {
2   "name": "minuman",
3   "description": "ini merupakan category minuman"
4 }
```

The response status is 200 OK, time 27 ms, size 316 B. The JSON response is:

```
1 {
2   "message": [
3     {
4       "id": 3,
5       "name": "minuman",
6       "description": "ini merupakan category minuman",
7       "created_at": "2024-05-15T13:57:03.337+07:00",
8       "updated_at": "2024-05-15T13:57:03.337+07:00"
9     }
10 ],
11 "status": "success"
12 }
```

CATEGORY ADMIN

GET ALL

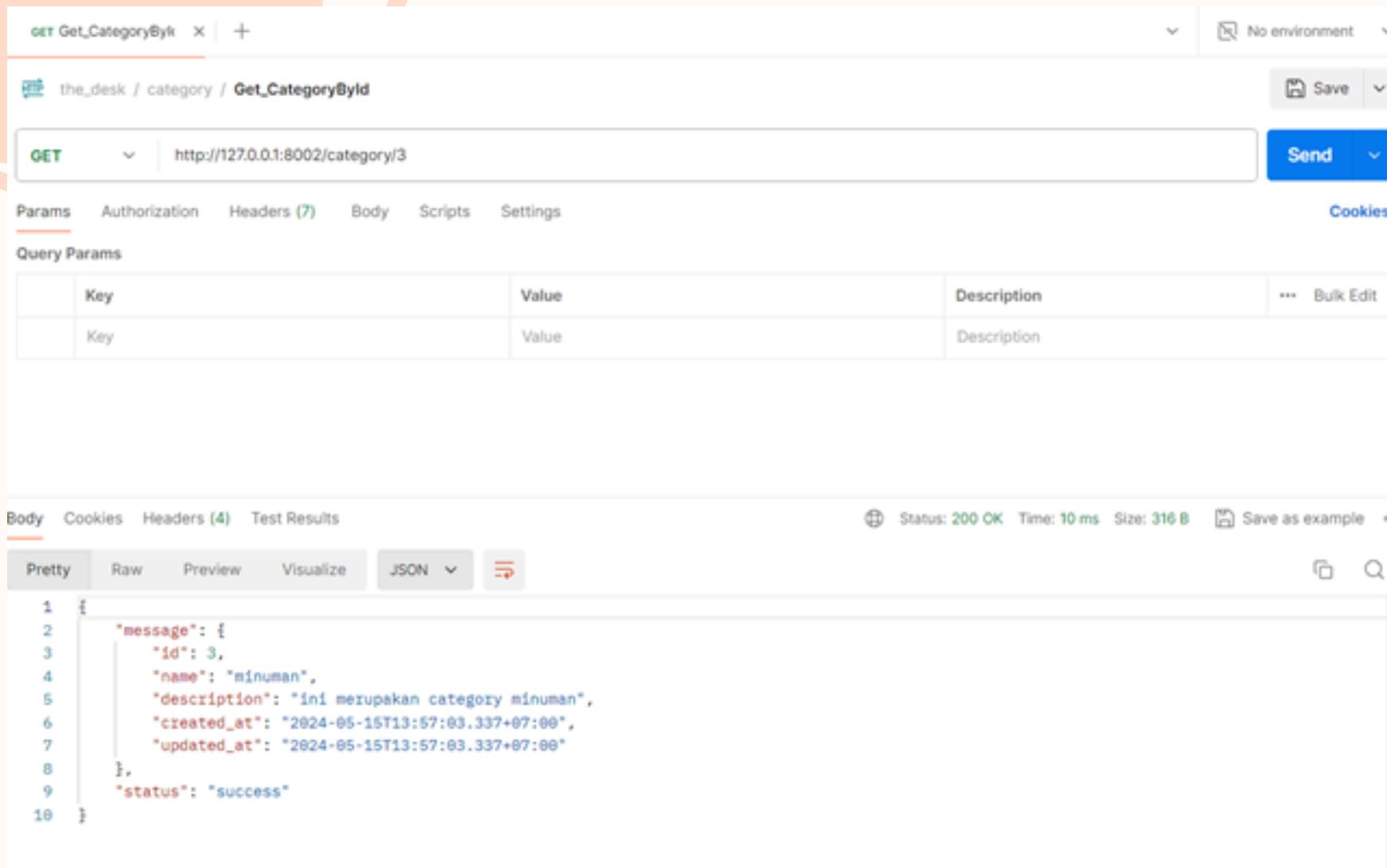


The screenshot shows the Postman interface with a GET request to 'the_desk / category / Get_Category'. The URL is `http://127.0.0.1:8002/category/`. The response status is 200 OK, time 8 ms, size 481 B. The JSON response is:

```
1 {
2   "message": [
3     {
4       "id": 3,
5       "name": "minuman",
6       "description": "ini merupakan category minuman",
7       "created_at": "2024-05-15T13:57:03.337+07:00",
8       "updated_at": "2024-05-15T13:57:03.337+07:00"
9     },
10     {
11       "id": 4,
12       "name": "makanan",
13       "description": "ini merupakan category makanan",
14       "created_at": "2024-05-15T13:58:14.726+07:00",
15       "updated_at": "2024-05-15T13:58:14.726+07:00"
16     }
17   ],
18   "status": "success"
19 }
```

Pengujian Service

GET BY ID



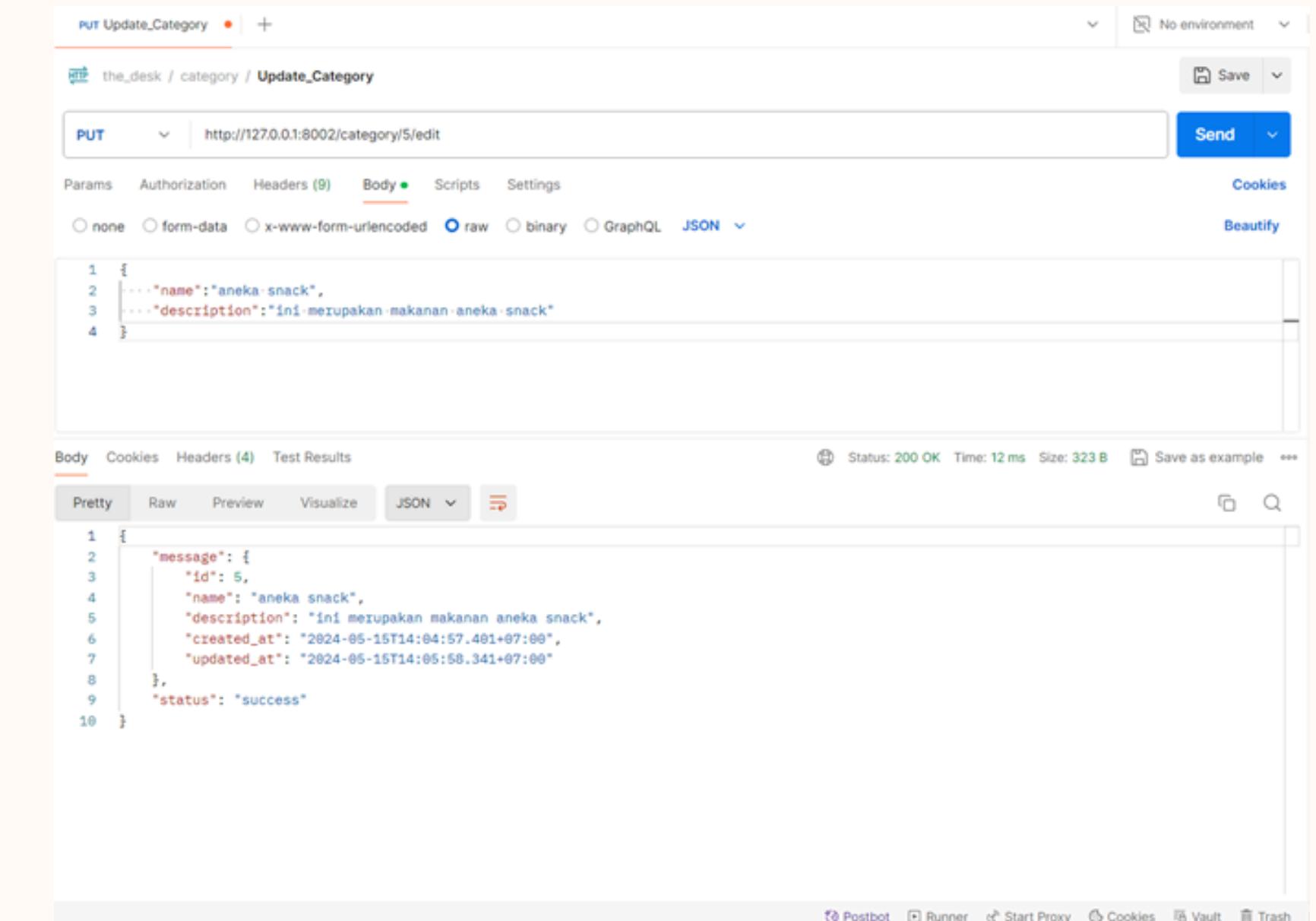
The screenshot shows the Postman interface with a request labeled "GET Get_CategoryById". The URL is set to "http://127.0.0.1:8002/category/3". The "Body" tab is selected, showing the following JSON payload:

```
1 {
2   "message": {
3     "id": 3,
4     "name": "minuman",
5     "description": "ini merupakan category minuman",
6     "created_at": "2024-05-15T13:57:03.337+07:00",
7     "updated_at": "2024-05-15T13:57:03.337+07:00"
8   },
9   "status": "success"
10 }
```

The response status is 200 OK, with a response body containing the category details.

CATEGORY ADMIN

UPDATE



The screenshot shows the Postman interface with a request labeled "PUT Update_Category". The URL is set to "http://127.0.0.1:8002/category/5/edit". The "Body" tab is selected, showing the following JSON payload:

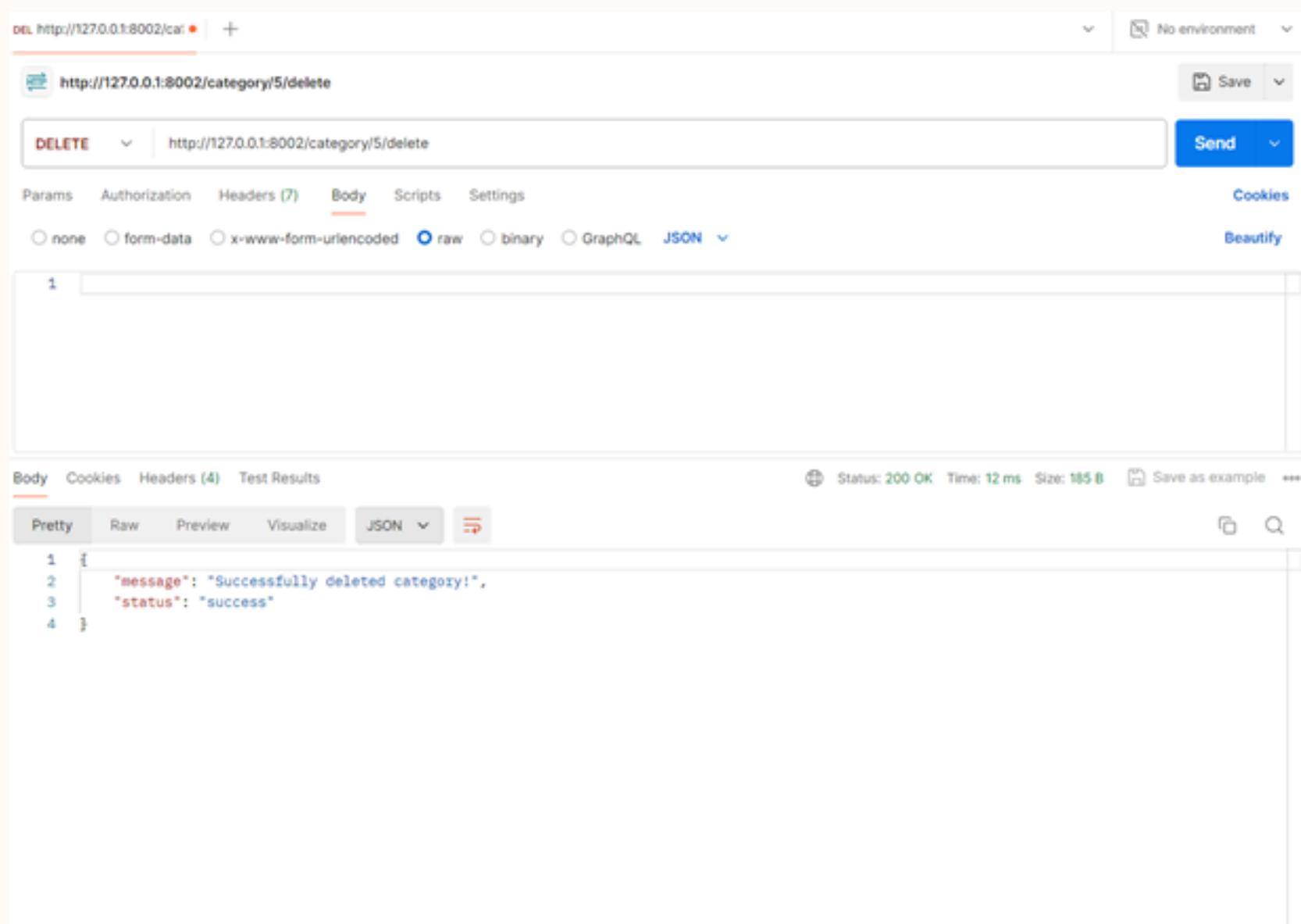
```
1 {
2   ...
3   "name": "aneka snack",
4   ...
5   "description": "ini merupakan makanan aneka snack"
6 }
```

The response status is 200 OK, with a response body indicating success and updated category details.

Pengujian Service

CATEGORY ADMIN

DELETE



The screenshot shows a POSTMAN interface with the following details:

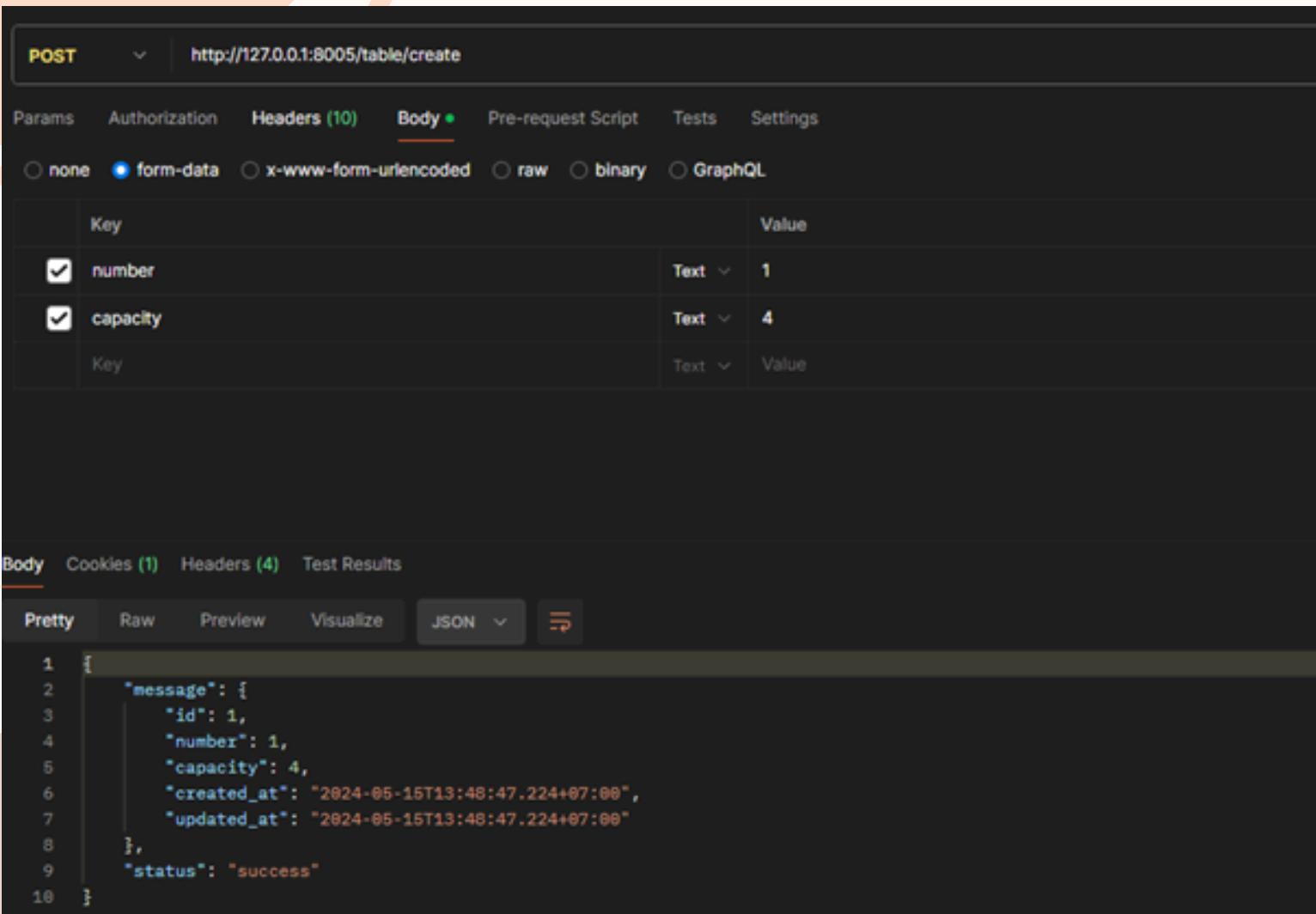
- Method: DELETE
- URL: <http://127.0.0.1:8002/category/5/delete>
- Body tab selected
- Body content:

```
1
```
- Headers tab (7 items):
 - Content-Type: application/json
 - Accept: application/json
 - Content-Length: 185
 - Host: 127.0.0.1:8002
 - Connection: keep-alive
 - User-Agent: PostmanRuntime/7.29.2
 - Accept-Encoding: gzip, deflate
- Response status: 200 OK
- Response body:

```
1 {
2   "message": "Successfully deleted category!",
3   "status": "success"
4 }
```

Pengujian Service

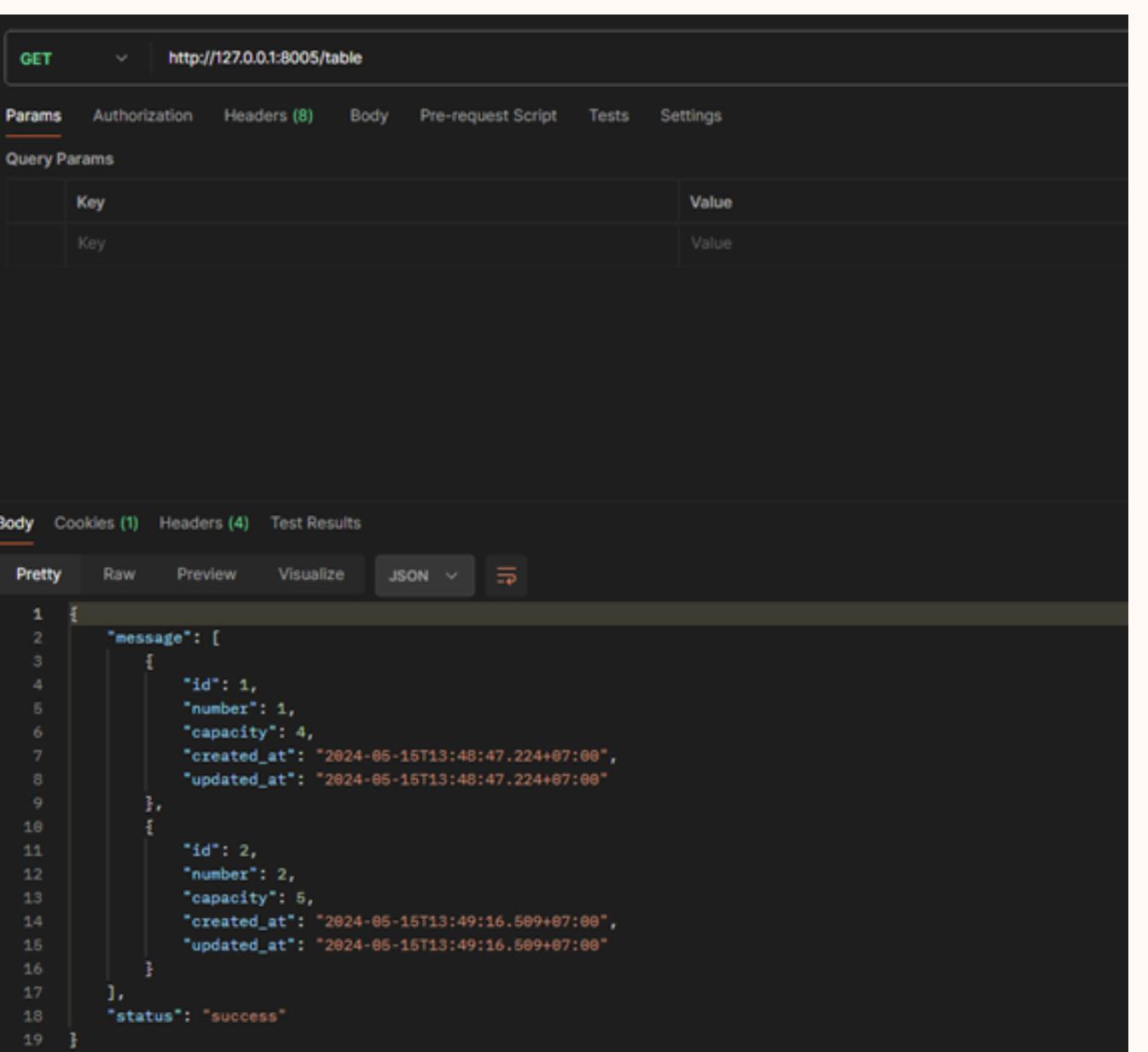
CREATE



POST <http://127.0.0.1:8005/table/create>

Body (1)

```
1 {
  "message": [
    {
      "id": 1,
      "number": 1,
      "capacity": 4,
      "created_at": "2024-05-15T13:48:47.224+07:00",
      "updated_at": "2024-05-15T13:48:47.224+07:00"
    },
    {
      "id": 2,
      "number": 2,
      "capacity": 5,
      "created_at": "2024-05-15T13:49:16.509+07:00",
      "updated_at": "2024-05-15T13:49:16.509+07:00"
    }
  ],
  "status": "success"
}
```



GET <http://127.0.0.1:8005/table>

Body (1)

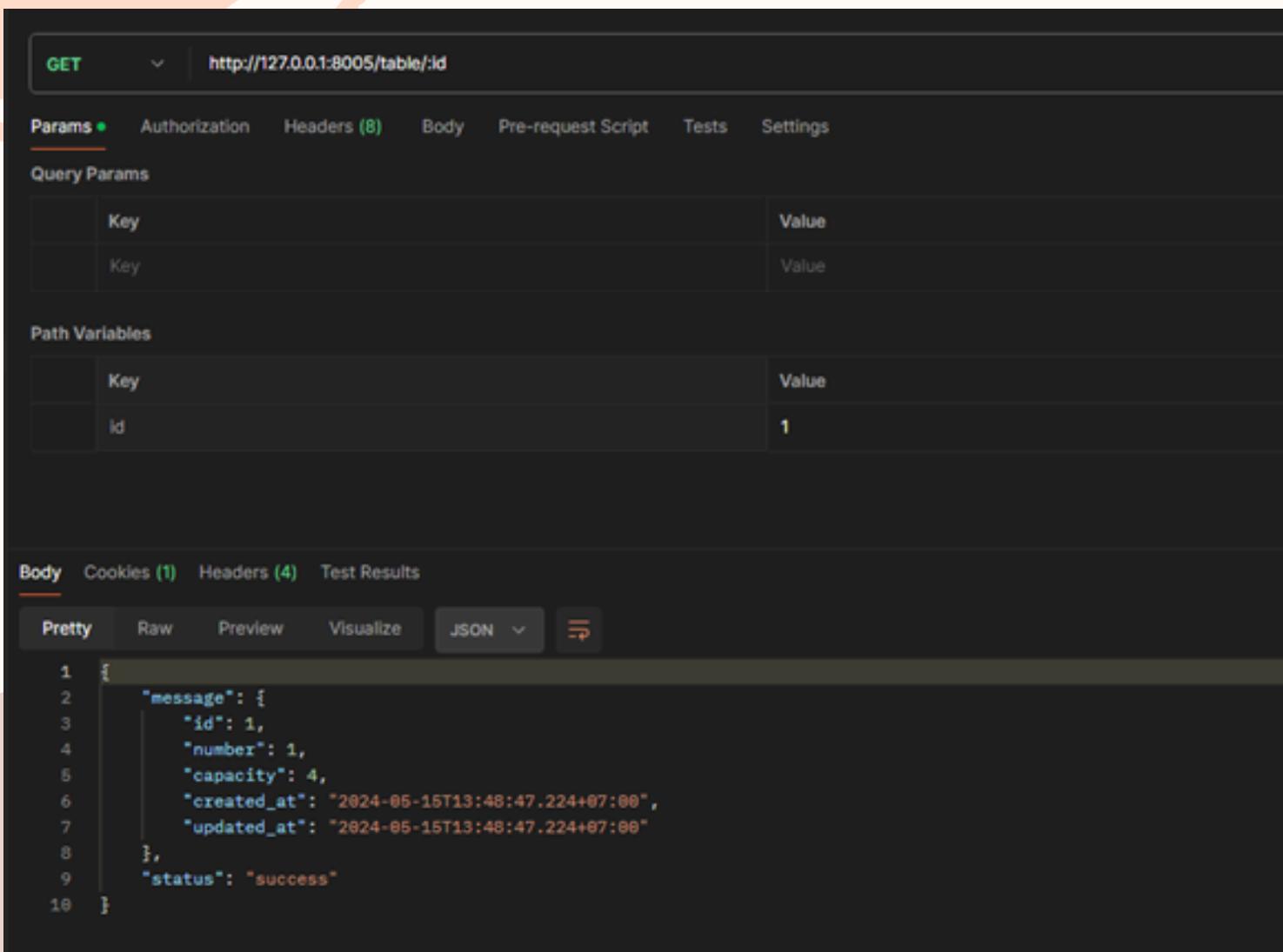
```
1 {
  "message": [
    {
      "id": 1,
      "number": 1,
      "capacity": 4,
      "created_at": "2024-05-15T13:48:47.224+07:00",
      "updated_at": "2024-05-15T13:48:47.224+07:00"
    },
    {
      "id": 2,
      "number": 2,
      "capacity": 5,
      "created_at": "2024-05-15T13:49:16.509+07:00",
      "updated_at": "2024-05-15T13:49:16.509+07:00"
    }
  ],
  "status": "success"
}
```

TABLE ADMIN

GET ALL

Pengujian Service

GET BY ID



GET http://127.0.0.1:8005/table/:id

Params

Key	Value

Query Params

Key	Value

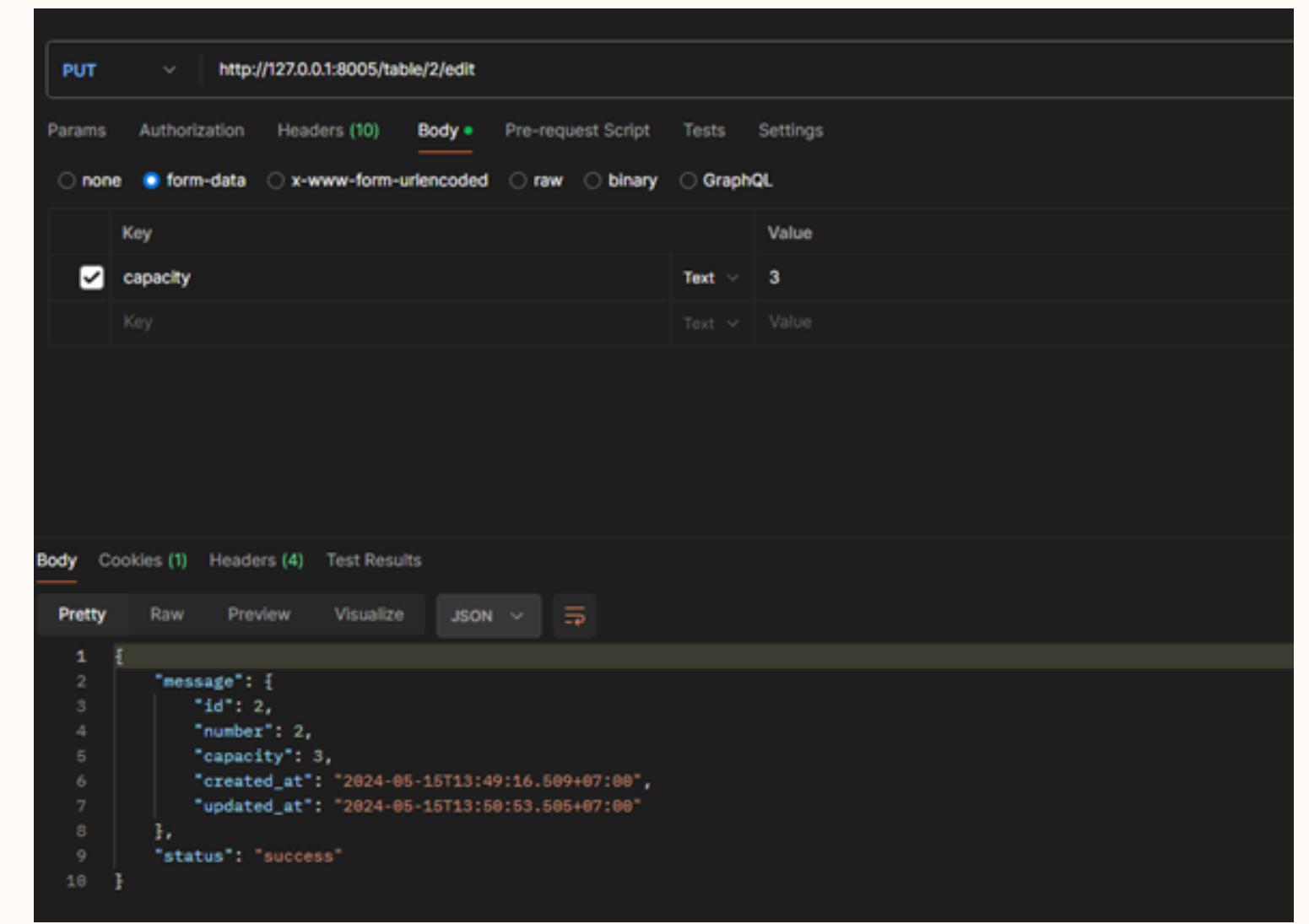
Path Variables

Key	Value
id	1

Body

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": {
3     "id": 1,
4     "number": 1,
5     "capacity": 4,
6     "created_at": "2024-05-15T13:48:47.224+07:00",
7     "updated_at": "2024-05-15T13:48:47.224+07:00"
8   },
9   "status": "success"
10 }
```



PUT http://127.0.0.1:8005/table/2/edit

Params

Authorization

Headers

Body

Pre-request Script

Tests

Settings

Body

None form-data x-www-form-urlencoded raw binary GraphQL

Key	Value
capacity	3

Body

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": {
3     "id": 2,
4     "number": 2,
5     "capacity": 3,
6     "created_at": "2024-05-15T13:49:16.509+07:00",
7     "updated_at": "2024-05-15T13:50:53.505+07:00"
8   },
9   "status": "success"
10 }
```

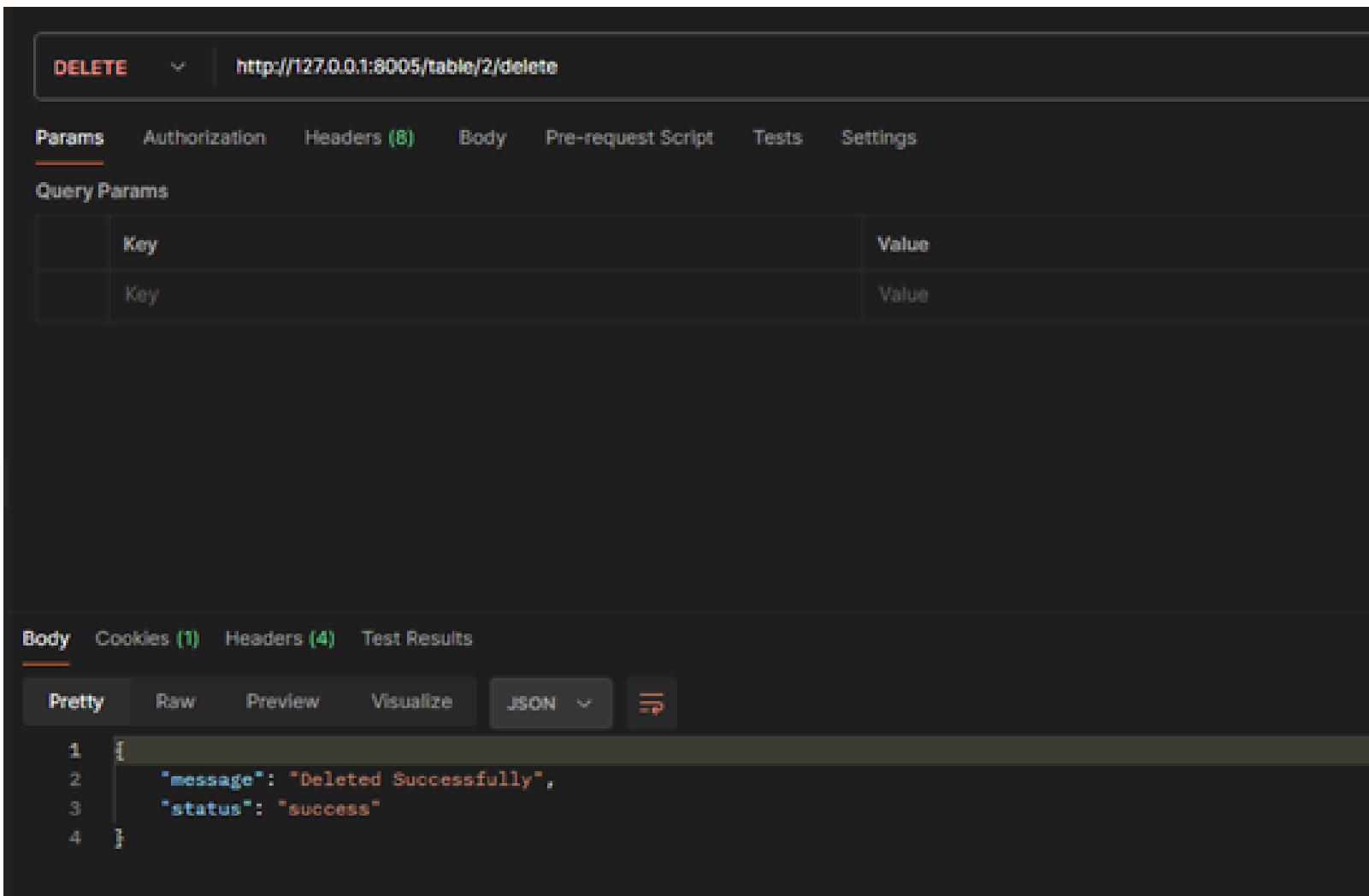
TABLE ADMIN

UPDATE

Pengujian Service

TABLE ADMIN

DELETE



DELETE <http://127.0.0.1:8005/table/2/delete>

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

Query Params

Key	Value
Key	Value

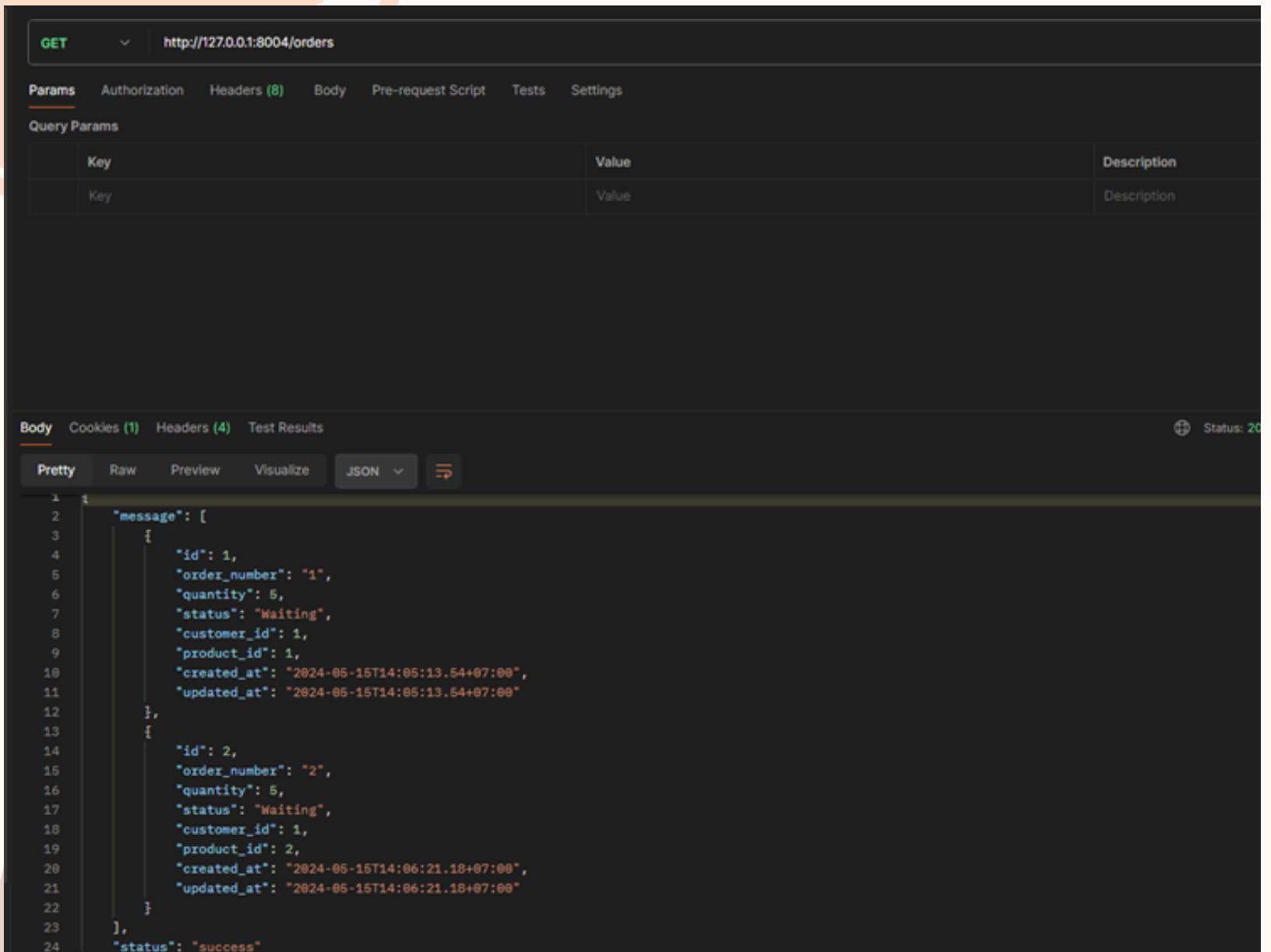
Body Cookies (1) Headers (4) Test Results

Pretty Raw Preview Visualize JSON

```
1 [ { "message": "Deleted Successfully", "status": "success" } ]
```

Pengujian Service

GET ALL



GET http://127.0.0.1:8004/orders

Params

Key	Value	Description

Query Params

Key	Value	Description

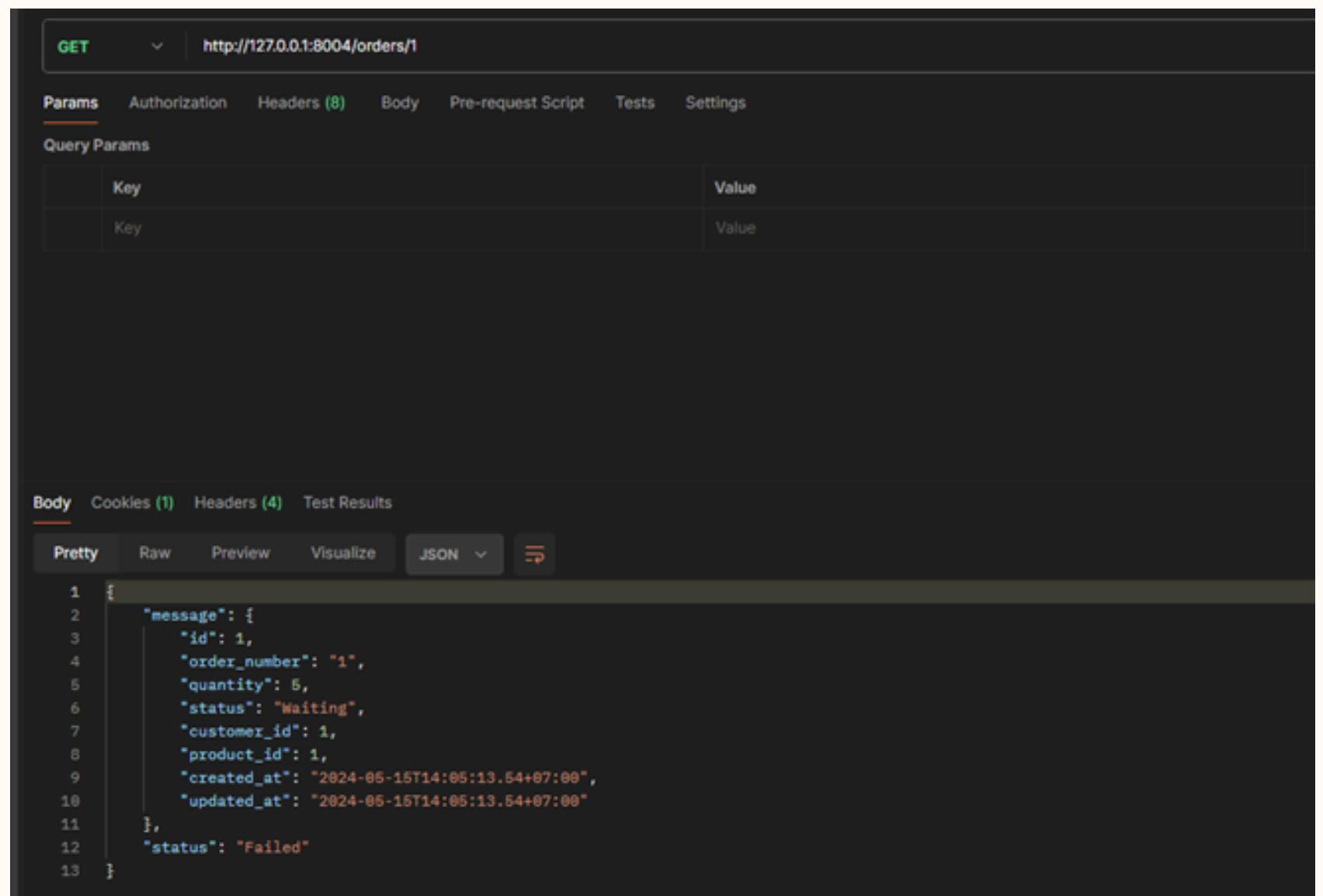
Body

Pretty Raw Preview Visualize JSON

```
1: {
2:   "message": [
3:     {
4:       "id": 1,
5:       "order_number": "1",
6:       "quantity": 5,
7:       "status": "Waiting",
8:       "customer_id": 1,
9:       "product_id": 1,
10:      "created_at": "2024-05-15T14:06:13.54+07:00",
11:      "updated_at": "2024-05-15T14:06:13.54+07:00"
12:    },
13:    {
14:      "id": 2,
15:      "order_number": "2",
16:      "quantity": 5,
17:      "status": "Waiting",
18:      "customer_id": 1,
19:      "product_id": 2,
20:      "created_at": "2024-05-15T14:06:21.18+07:00",
21:      "updated_at": "2024-05-15T14:06:21.18+07:00"
22:    }
23:  ],
24:  "status": "success"
25: }
```

MELIHAT ORDER ADMIN

GET BY ID



GET http://127.0.0.1:8004/orders/1

Params

Key	Value	Description

Query Params

Key	Value	Description

Body

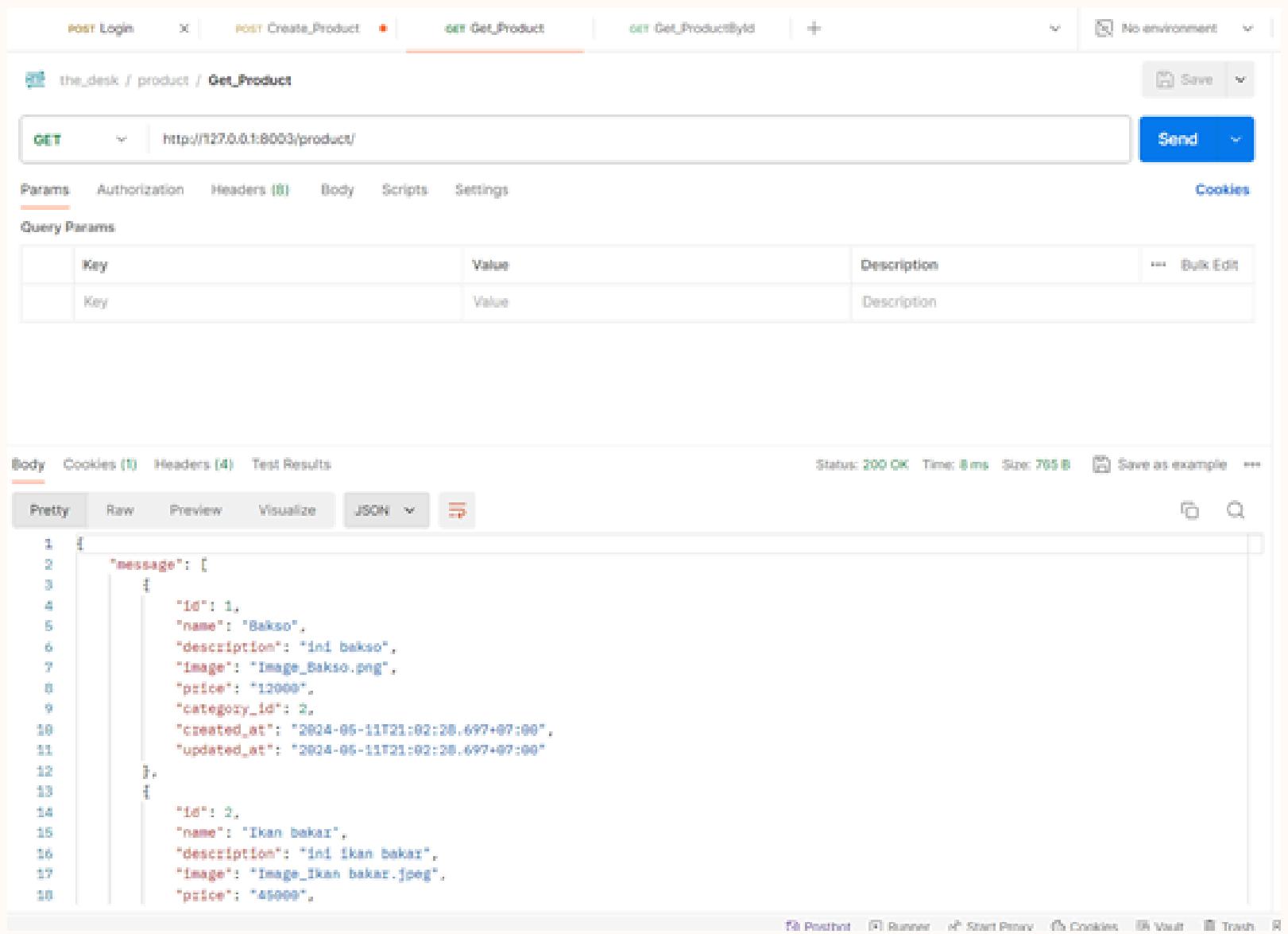
Pretty Raw Preview Visualize JSON

```
1: {
2:   "message": {
3:     "id": 1,
4:     "order_number": "1",
5:     "quantity": 5,
6:     "status": "Waiting",
7:     "customer_id": 1,
8:     "product_id": 1,
9:     "created_at": "2024-05-15T14:06:13.54+07:00",
10:    "updated_at": "2024-05-15T14:06:13.54+07:00"
11:  },
12:  "status": "Failed"
13: }
```

Pengujian Service

MELIHAT PRODUCT
CUSTOMER

GET



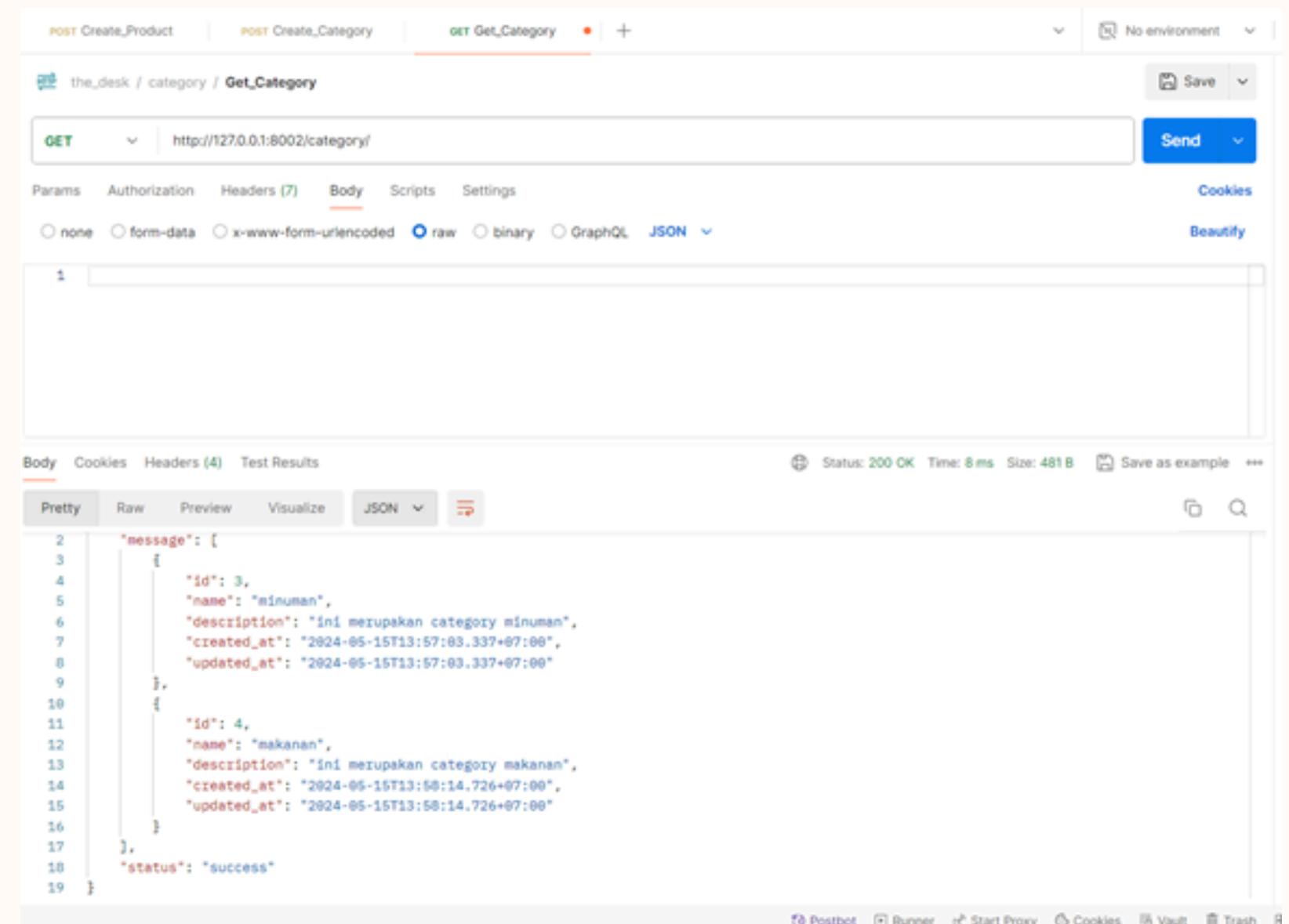
The screenshot shows the Postman application interface. At the top, there are tabs for 'Post Login', 'Post Create_Product', 'Get_Product' (which is highlighted in red), and 'Get_ProductById'. Below the tabs, the URL 'http://127.0.0.1:8000/products/' is entered into the 'Send' button. The 'Params' tab is selected, showing a table with two rows: one for 'Key' and one for 'Value'. In the 'Body' section, the 'Pretty' tab is selected, displaying a JSON response. The response is as follows:

```
1  [
2     "message": [
3         {
4             "id": 1,
5             "name": "Bakso",
6             "description": "ini bakso",
7             "image": "Image_Bakso.png",
8             "price": "12000",
9             "category_id": 2,
10            "created_at": "2024-05-11T21:02:28.697+07:00",
11            "updated_at": "2024-05-11T21:02:28.697+07:00"
12        },
13        {
14            "id": 2,
15            "name": "Ikan bakar",
16            "description": "ini ikan bakar",
17            "image": "Image_Ikan_bakar.jpeg",
18            "price": "45000",
19        }
20    ]
21 ]
```

At the bottom of the interface, there are buttons for 'Postman', 'Runner', 'Start Proxy', 'Cookies', 'Vault', 'Trash', and 'Help'.

Pengujian Service

GET

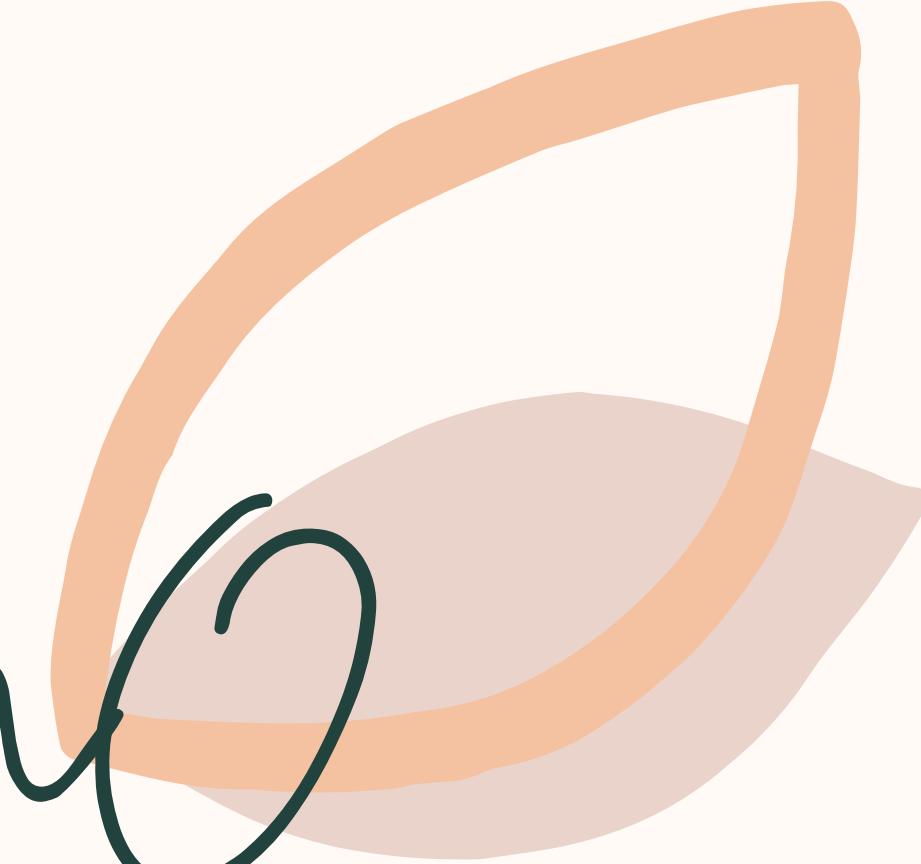


The screenshot shows the Postman application interface. A GET request is being made to `http://127.0.0.1:8002/category/`. The response is a JSON object with the following structure:

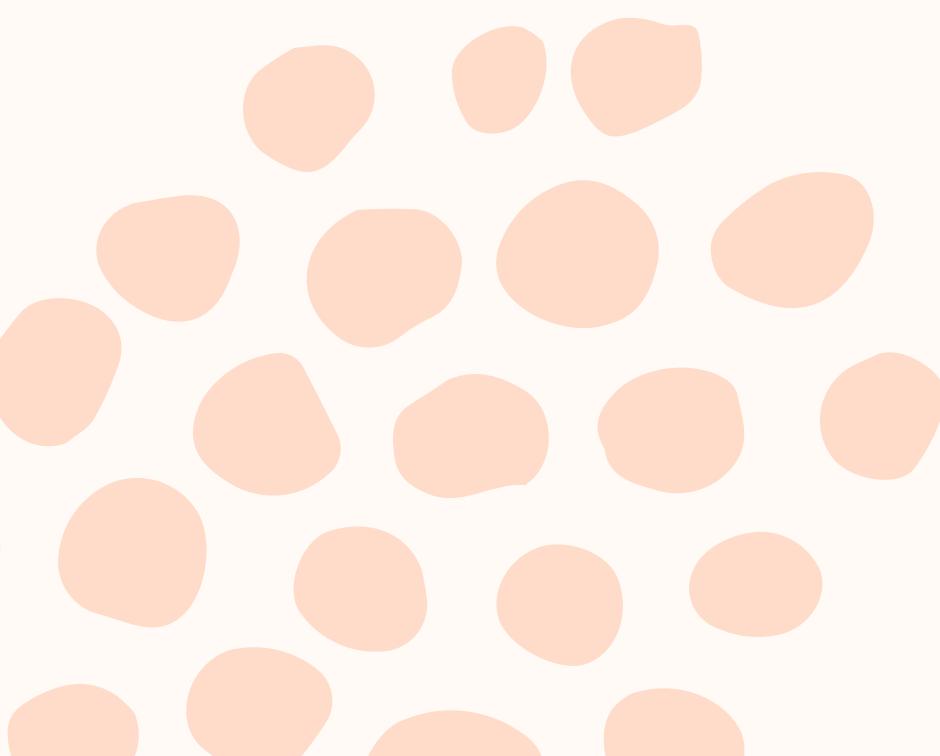
```
2 "message": [3   {4     "id": 3,5       "name": "minuman",6       "description": "ini merupakan category minuman",7       "created_at": "2024-05-15T13:57:03.337+07:00",8       "updated_at": "2024-05-15T13:57:03.337+07:00"9   },10   {11     "id": 4,12       "name": "makanan",13       "description": "ini merupakan category makanan",14       "created_at": "2024-05-15T13:58:14.726+07:00",15       "updated_at": "2024-05-15T13:58:14.726+07:00"16   },17 ],18 "status": "success"19 }
```

The status bar at the bottom indicates a `200 OK` response with a size of `481 B`.

MELIHAT CATEGORY
CUSTOMER



Deno





Thank
you