

# Laporan Pengerjaan

## Ujian Tengah Semester

Backend Programming - Semester Genap 2023/2024

Nama : Hengky Laurencio  
NIM : 535230168  
Jurusan : Teknik Informatika  
Kelas : A  
Semester : 2  
Mata kuliah : Backed Programing  
Dosen : Janson Hendryli S. Kom. M.Kom.

Link Pengumpulan: <https://github.com/HengkyLaurencio/Midtest-Backend-Programming>

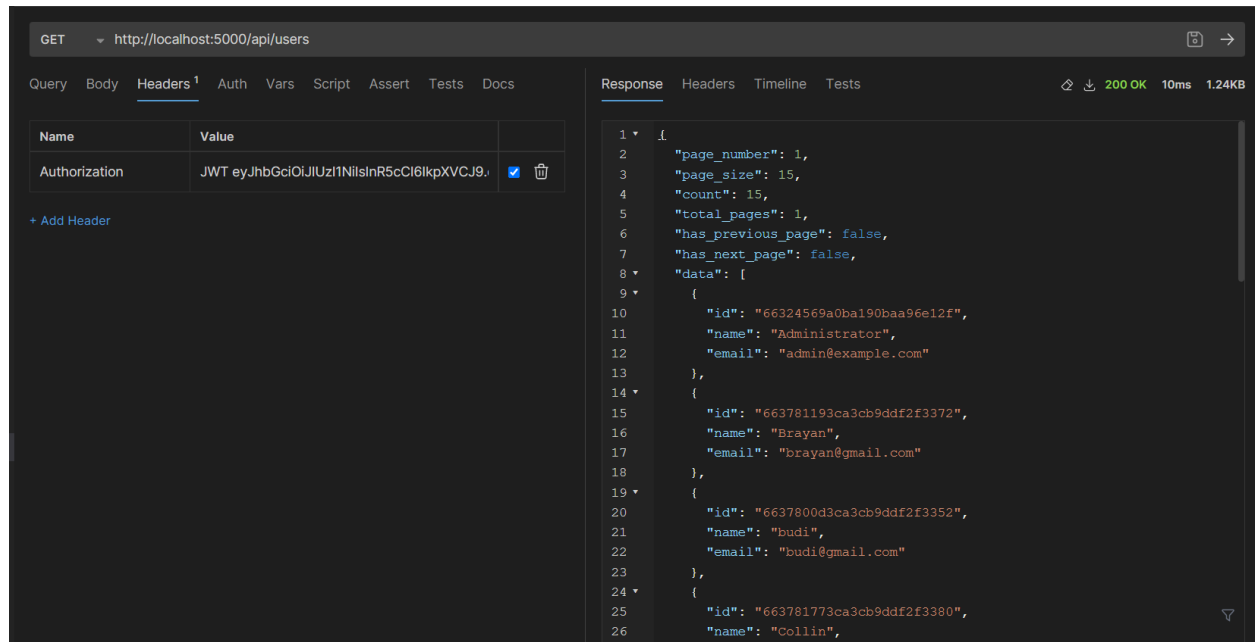
### Daftar ISI

<b>Daftar ISI.....</b>	<b>1</b>
<b>Soal 1 - Pagination dan Filter.....</b>	<b>2</b>
Pagination.....	2
Sorting.....	3
Filtering/Searching.....	3
Contoh.....	4
<b>Soal 2 - Login Attempts Limit.....</b>	<b>4</b>
<b>Soal 3 - Banking.....</b>	<b>6</b>
GET /banking.....	6
POST /banking.....	7
DELETE /banking/:account_number.....	7
POST /banking/login.....	8
Transactions.....	8
POST /banking/transactions/deposit.....	8
POST /banking/transactions/transfer.....	9
POST /banking/transactions/withdraw.....	9
GET /banking/transactions/history.....	10
Admin.....	10
GET /banking/admin/accounts.....	11
GET /banking/admin/transactions.....	11
Hal Yang Dapat Dikembangkan.....	11

## Soal 1 - Pagination dan Filter

GET <http://localhost:5000/api/users>

Pada endpoint ini digunakan untuk mendapatkan semua users, yang di sort berdasarkan email-ascending secara default.



The screenshot shows a REST client interface with the following details:

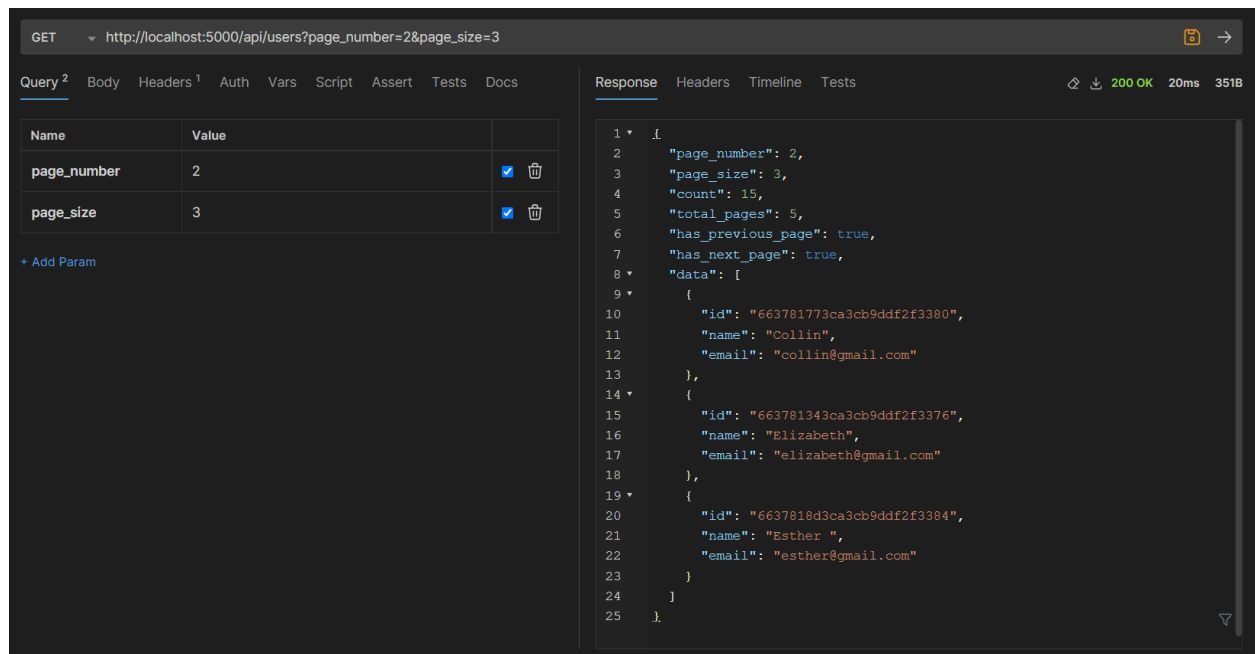
- Query:** GET <http://localhost:5000/api/users>
- Headers:** Authorization: JWT eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9. (checked)
- Response:** 200 OK, 10ms, 1.24KB
- Response Body:**

```
1 {
2   "page_number": 1,
3   "page_size": 15,
4   "count": 15,
5   "total_pages": 1,
6   "has_previous_page": false,
7   "has_next_page": false,
8   "data": [
9     {
10      "id": "66324569a0ba190baa96e12f",
11      "name": "Administrator",
12      "email": "admin@example.com"
13    },
14    {
15      "id": "663781193ca3cb9ddf2f3372",
16      "name": "Brayan",
17      "email": "brayan@gmail.com"
18    },
19    {
20      "id": "6637800d3ca3cb9ddf2f3352",
21      "name": "budi",
22      "email": "budi@gmail.com"
23    },
24    {
25      "id": "663781773ca3cb9ddf2f3380",
26      "name": "Collin",
```

## Pagination

GET [http://localhost:5000/api/users?page\\_number=2&page\\_size=3](http://localhost:5000/api/users?page_number=2&page_size=3)

Query 'page\_number' digunakan untuk menentukan halaman mana yang akan dibuka, sedangkan 'page\_size' menentukan jumlah users yang akan ditampilkan dalam satu halaman.



The screenshot shows a REST client interface with the following details:

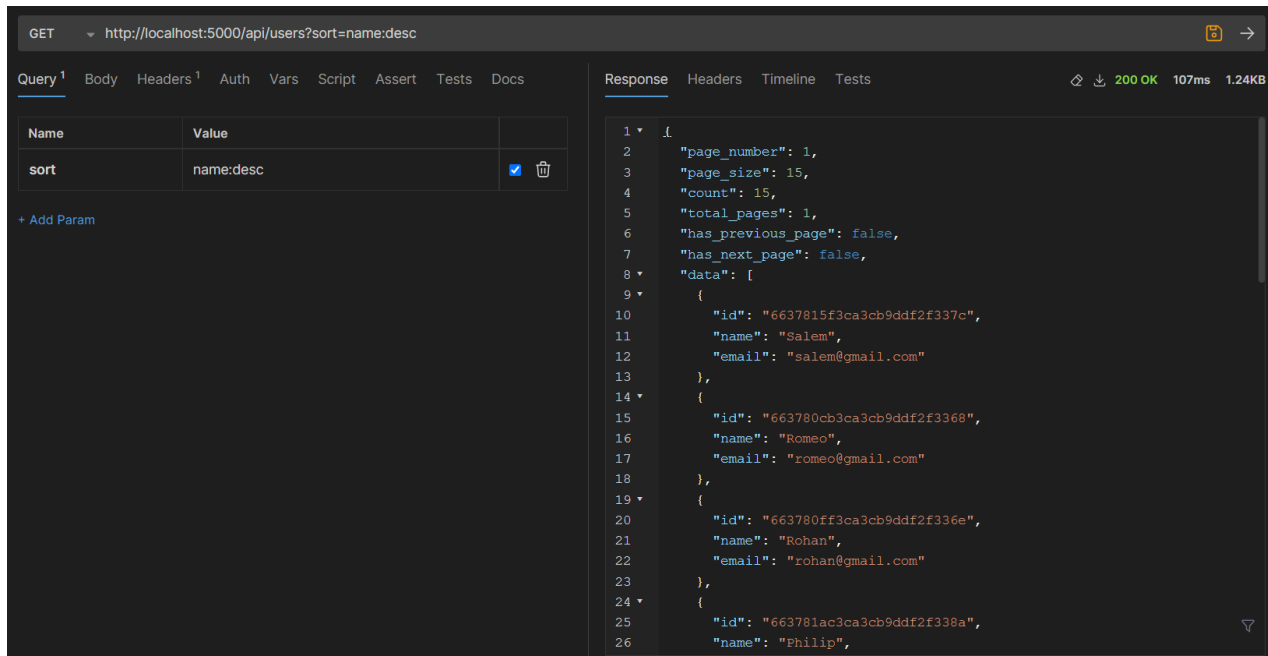
- Query:** GET [http://localhost:5000/api/users?page\\_number=2&page\\_size=3](http://localhost:5000/api/users?page_number=2&page_size=3)
- Params:** page\_number: 2, page\_size: 3 (checked)
- Response:** 200 OK, 20ms, 351B
- Response Body:**

```
1 {
2   "page_number": 2,
3   "page_size": 3,
4   "count": 15,
5   "total_pages": 5,
6   "has_previous_page": true,
7   "has_next_page": true,
8   "data": [
9     {
10      "id": "663781773ca3cb9ddf2f3380",
11      "name": "Collin",
12      "email": "collin@gmail.com"
13    },
14    {
15      "id": "663781343ca3cb9ddf2f3376",
16      "name": "Elizabeth",
17      "email": "elizabeth@gmail.com"
18    },
19    {
20      "id": "6637818d3ca3cb9ddf2f3384",
21      "name": "Esther ",
22      "email": "esther@gmail.com"
23    }
24   ]
25 }
```

## Sorting

GET <http://localhost:5000/api/users?sort=name:desc>

Query 'sort' untuk mengurutkan hasil users memiliki format: <field name>:<sort order>. Pada contoh ini Users diurutkan dengan field: name secara descending.



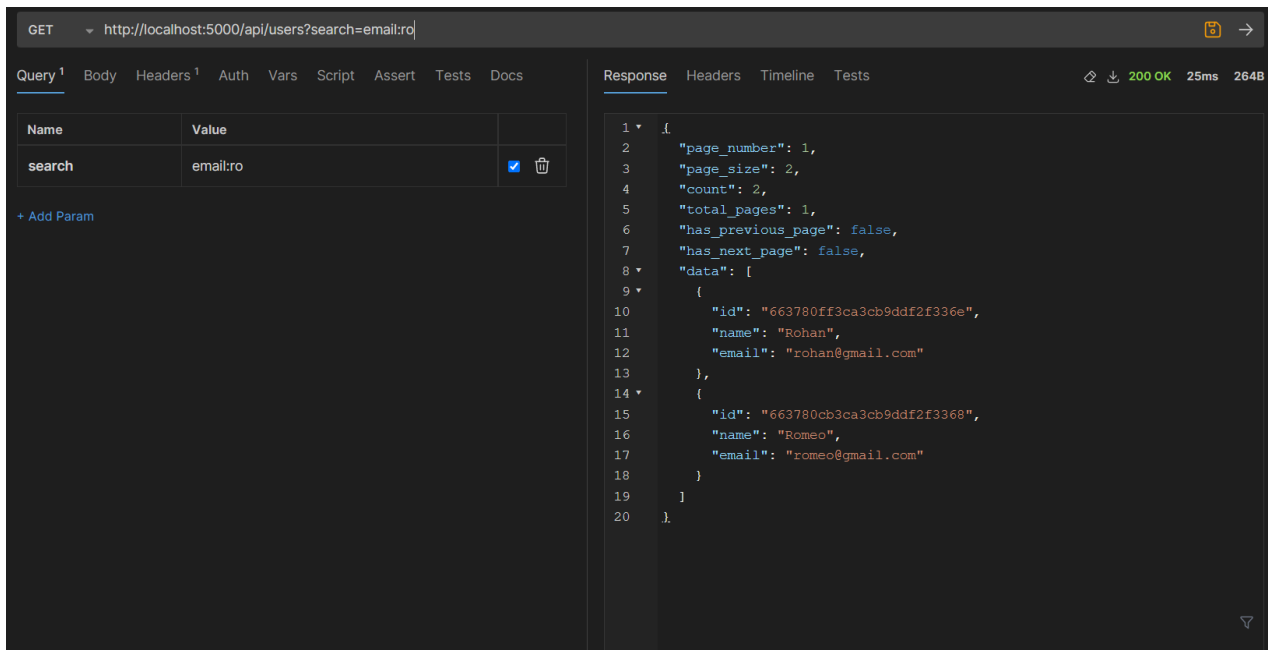
The screenshot shows a REST client interface with a GET request to `http://localhost:5000/api/users?sort=name:desc`. The response is a JSON object with the following structure:

```
1 {
2   "page_number": 1,
3   "page_size": 15,
4   "count": 15,
5   "total_pages": 1,
6   "has_previous_page": false,
7   "has_next_page": false,
8   "data": [
9     {
10      "id": "6637815f3ca3cb9ddf2f337c",
11      "name": "Salem",
12      "email": "salem@gmail.com"
13    },
14    {
15      "id": "663780cb3ca3cb9ddf2f3368",
16      "name": "Romeo",
17      "email": "romeo@gmail.com"
18    },
19    {
20      "id": "663780ff3ca3cb9ddf2f336e",
21      "name": "Rohan",
22      "email": "rohan@gmail.com"
23    },
24    {
25      "id": "663781ac3ca3cb9ddf2f338a",
26      "name": "Philip",
```

## Filtering/Searching

GET <http://localhost:5000/api/users?search=email:ro>

Query 'search' memiliki format: <field name>:<sort order>. Digunakan untuk mencari substring pada field name. Pada contoh ini kita mencari email yang memiliki substring ro.



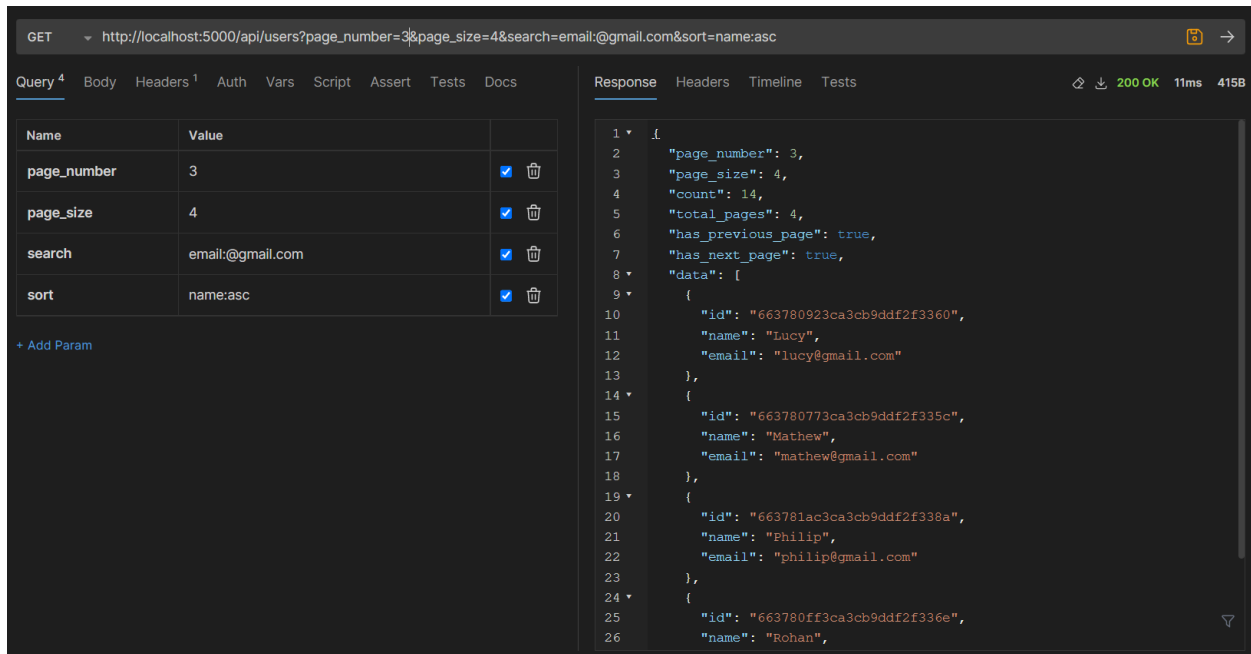
The screenshot shows a REST client interface with a GET request to `http://localhost:5000/api/users?search=email:ro`. The response is a JSON object with the following structure:

```
1 {
2   "page_number": 1,
3   "page_size": 2,
4   "count": 2,
5   "total_pages": 1,
6   "has_previous_page": false,
7   "has_next_page": false,
8   "data": [
9     {
10      "id": "663780ff3ca3cb9ddf2f336e",
11      "name": "Rohan",
12      "email": "rohan@gmail.com"
13    },
14    {
15      "id": "663780cb3ca3cb9ddf2f3368",
16      "name": "Romeo",
17      "email": "romeo@gmail.com"
18    }
19  ]
20 }
```

## Contoh

GET /users?page\_number=3&page\_size=4&search=email:@gmail.com&sort=name:asc

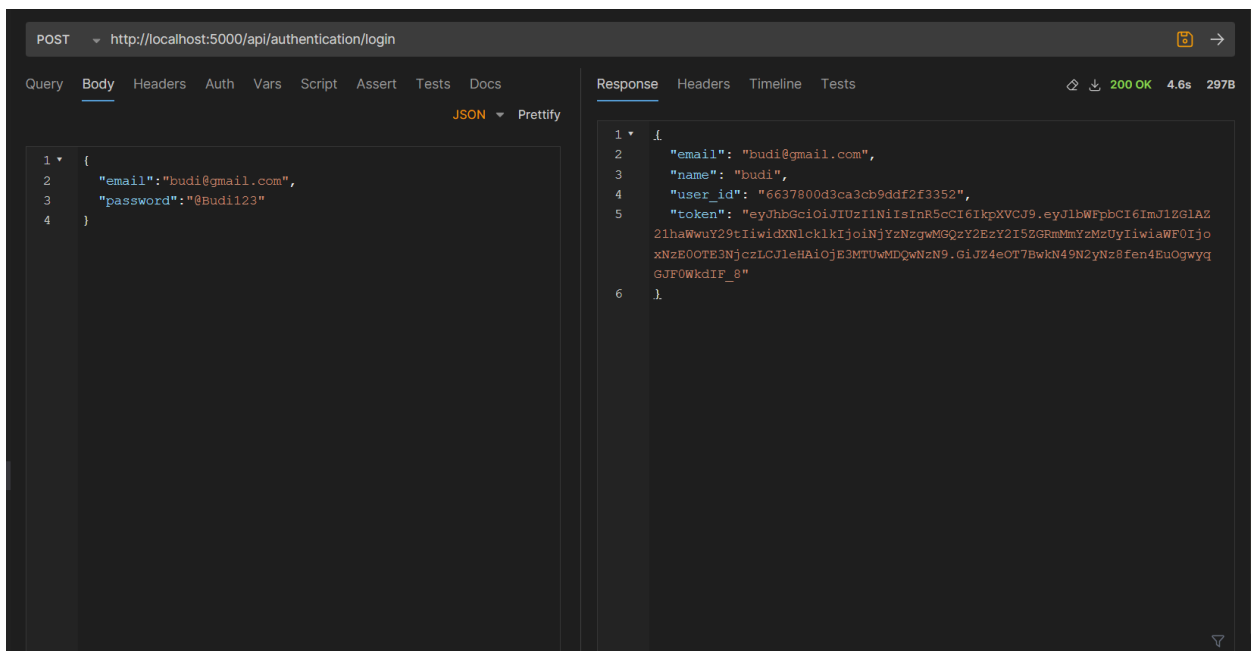
Halaman 3, dengan jumlah 4 users, cari email yang memiliki @gmail.com, dan sort berdasarkan name secara ascending.



## Soal 2 - Login Attempts Limit

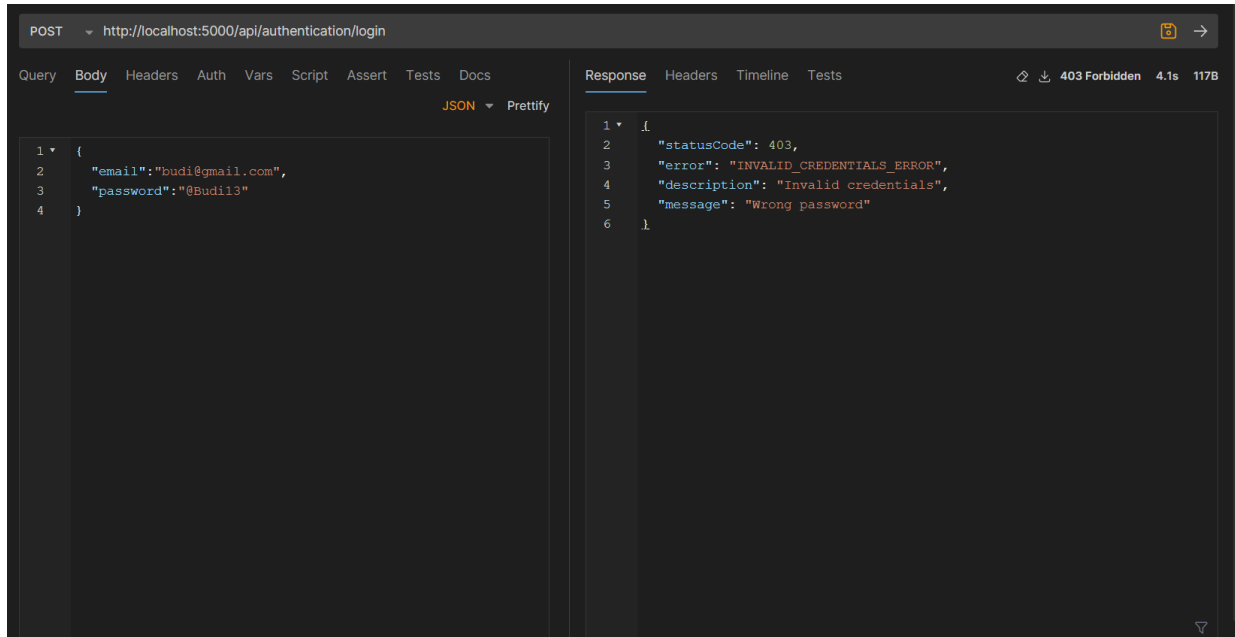
## POST/authentication/login

Endpoint ini digunakan untuk melakukan login pada user menggunakan email dan password. Jika user berhasil login, user akan mendapatkan token yang akan digunakan untuk authentication.



Jika password yang dimasukan salah sebanyak 5 kali, user tidak akan bisa login selama 30 menit. Setelah 30 menit attempt akan di reset menjadi 0. Jika user gagal login lalu berhasil pada percobaan selanjutnya attempt akan di reset menjadi 0. Contoh ada user yang login tetapi gagal sebanyak 2 kali, tetapi pada percobaan ke 3 user berhasil login, sehingga attempt akan di reset menjadi 0. Contoh lain jika ada user yang melakukan login tetapi gagal sebanyak 3 kali, kemudian user tersebut tidak login selama 30 menit, attempt juga akan di reset.

Hasil juga user gagal login



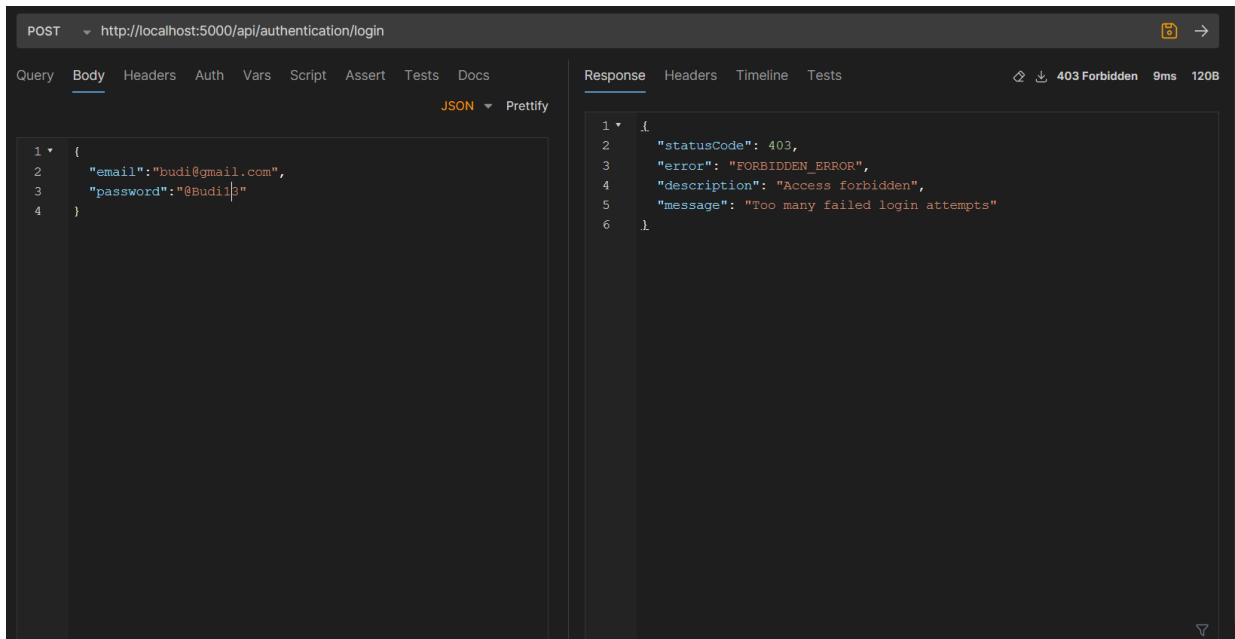
The screenshot shows a REST client interface with a POST request to `http://localhost:5000/api/authentication/login`. The request body is a JSON object: `{ "email": "budi@gmail.com", "password": "@Budi13" }`. The response is a JSON object: `{ "statusCode": 403, "error": "INVALID_CREDENTIALS_ERROR", "description": "Invalid credentials", "message": "Wrong password" }`. The status bar indicates a 403 Forbidden response with a 4.1s duration and 117B body size.

```
POST http://localhost:5000/api/authentication/login

{
  "email": "budi@gmail.com",
  "password": "@Budi13"
}
```

```
{
  "statusCode": 403,
  "error": "INVALID_CREDENTIALS_ERROR",
  "description": "Invalid credentials",
  "message": "Wrong password"
}
```

Jika user gagal login sebanyak 5 kali lalu mencoba lagi pada percobaan ke 6



The screenshot shows a REST client interface with a POST request to `http://localhost:5000/api/authentication/login`. The request body is a JSON object: `{ "email": "budi@gmail.com", "password": "@Budi13" }`. The response is a JSON object: `{ "statusCode": 403, "error": "FORBIDDEN_ERROR", "description": "Access forbidden", "message": "Too many failed login attempts" }`. The status bar indicates a 403 Forbidden response with a 9ms duration and 120B body size.

```
POST http://localhost:5000/api/authentication/login

{
  "email": "budi@gmail.com",
  "password": "@Budi13"
}
```

```
{
  "statusCode": 403,
  "error": "FORBIDDEN_ERROR",
  "description": "Access forbidden",
  "message": "Too many failed login attempts"
}
```

### Soal 3 - Banking

Banking system ini terhubung dengan users, di mana tiap user dapat membuat 1 atau lebih banking account. Dalam account ini user dapat melakukan transaksi seperti deposit, transfer, dan withdraw. User juga dapat menghapus account miliknya. Tetapi user tidak dapat menghapus account milih user lain.

Hal ini dapat dilakukan melalui token saat login user, token tersebut memiliki sebuah payload yang dapat digunakan untuk mengidentifikasi user tersebut.

#### GET /banking

Endpoint ini digunakan untuk mendapatkan informasi mengenai user dan banking account miliknya. Contoh di bawah di mana tidak ada account karena user belum membuatnya. Pastikan token yang digunakan adalah token milik user yang diinginkan.

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:5000/api/banking/
- Headers:**

Name	Value
Authorization	jwt eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.e3...
- Response:**

```
1 {
2   "userId": "6634ded53bbd1c5e561f5eea",
3   "name": "hengky",
4   "email": "hengky@gmail.com",
5   "accounts": {
6     "message": "This user does not have an account yet"
7   }
8 }
```

Berikut jika user memiliki beberapa banking account

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:5000/api/banking/
- Headers:**

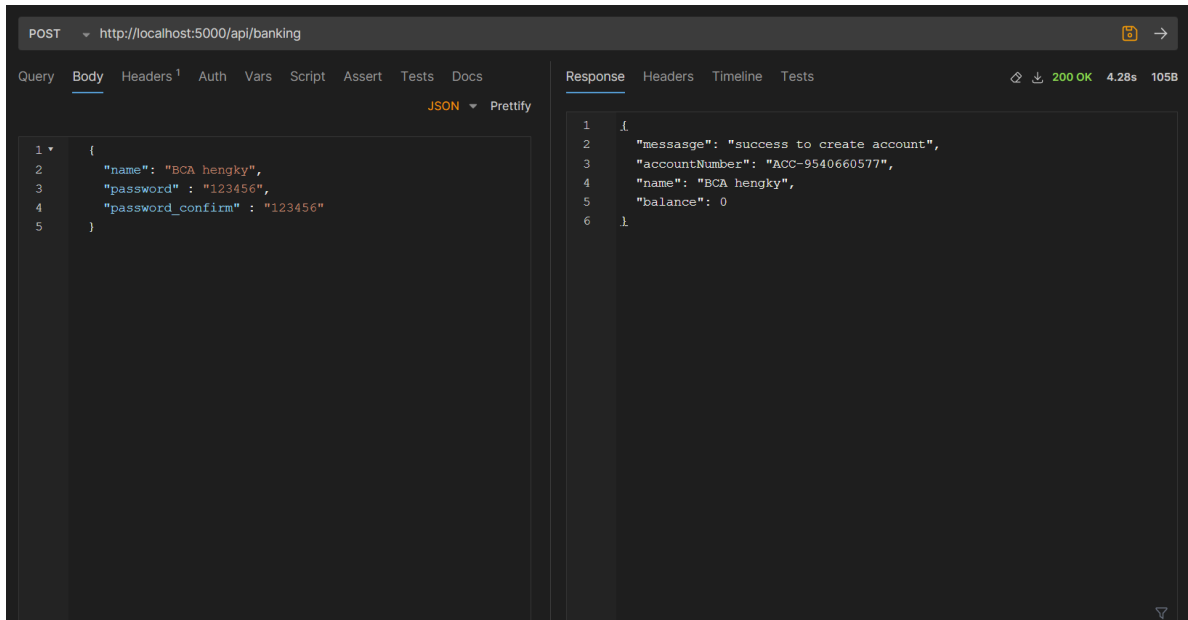
Name	Value
Authorization	jwt eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.e3...
- Response:**

```
1 {
2   "userId": "6634ded53bbd1c5e561f5eea",
3   "name": "hengky",
4   "email": "hengky@gmail.com",
5   "accounts": [
6     {
7       "accountNumber": "ACC-9540660577",
8       "name": "BCA hengky",
9       "balance": 0
10    },
11    {
12       "accountNumber": "ACC-5416823026",
13       "name": "MANDIRI hengky",
14       "balance": 0
15    },
16    {
17       "accountNumber": "ACC-2035184253",
18       "name": "BNI hengky",
19       "balance": 0
20    }
21  ]
22 }
```

## POST /banking

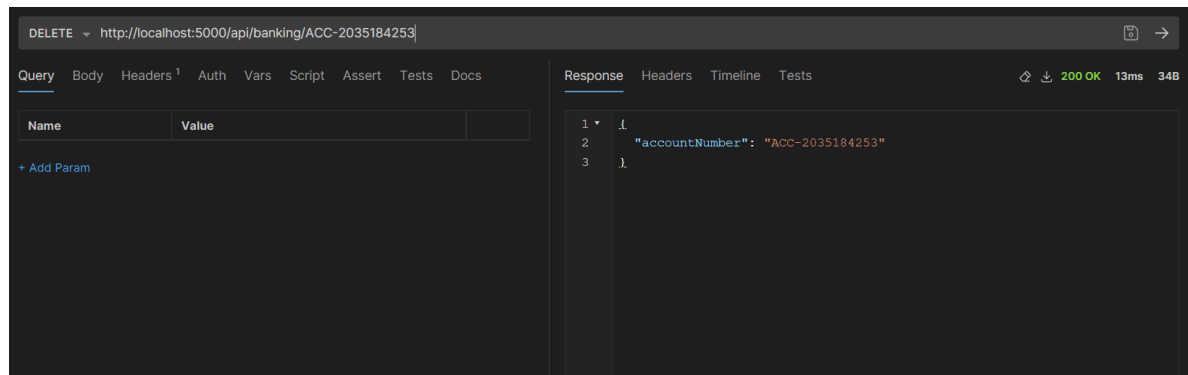
Endpoint ini digunakan untuk membuat banking account. Terdapat 3 field pada body yang harus diisi name, password, dan password\_confirm.

Pastikan lagi token yang digunakan adalah token dari user yang diinginkan

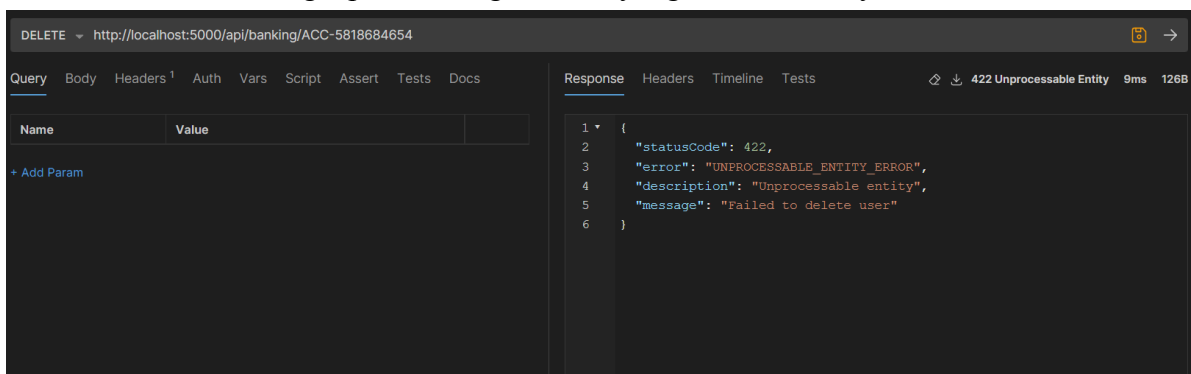


## DELETE /banking/:account\_number

Endpoint ini digunakan untuk menghapus banking account. Perlu diperhatikan bahwa User hanya bisa menghapus banking account yang ia miliki, tidak bisa menghapus banking account milik user lain.

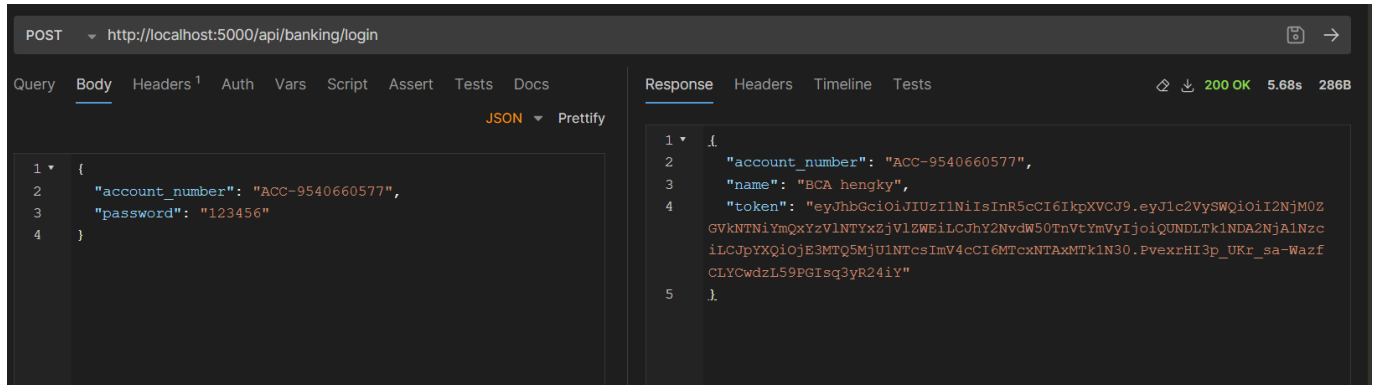


Jika user mencoba untuk menghapus banking account yang bukan miliknya



## POST /banking/login

Endpoint ini digunakan untuk user agar dapat login banking account miliknya, sehingga account tersebut dapat melakukan transaksi.



Ini akan mendapatkan token, yang digunakan untuk melakukan transaksi

## Transactions

Terdapat 4 endpoint pada transactions yakni:

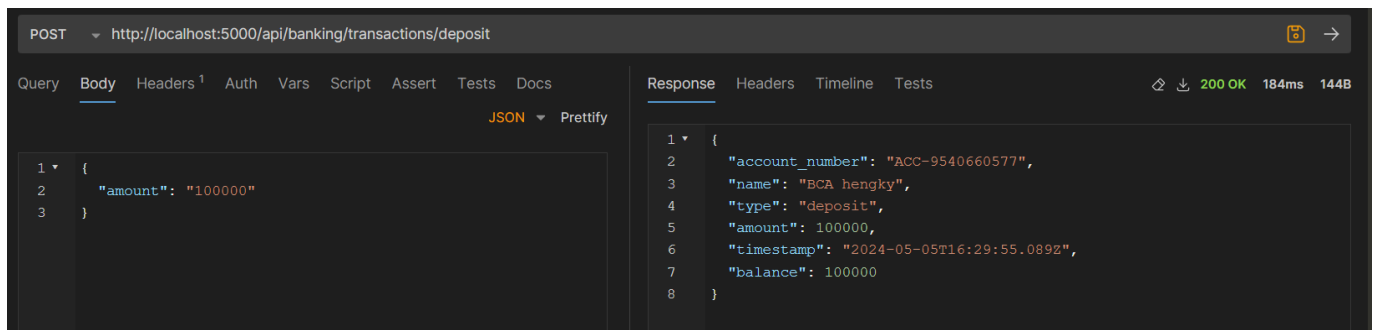
- POST /banking/transactions/deposit
- POST /banking/transactions/transfer
- POST /banking/transactions/withdraw
- GET /banking/transactions/history

Sebelum melakukan transaksi ada yang yang perlu diperhatikan pada header Authorization, dikarenakan memiliki format “bank <account token>”

Name	Value	
Authorization	bank eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.	<input checked="" type="checkbox"/>

## POST /banking/transactions/deposit

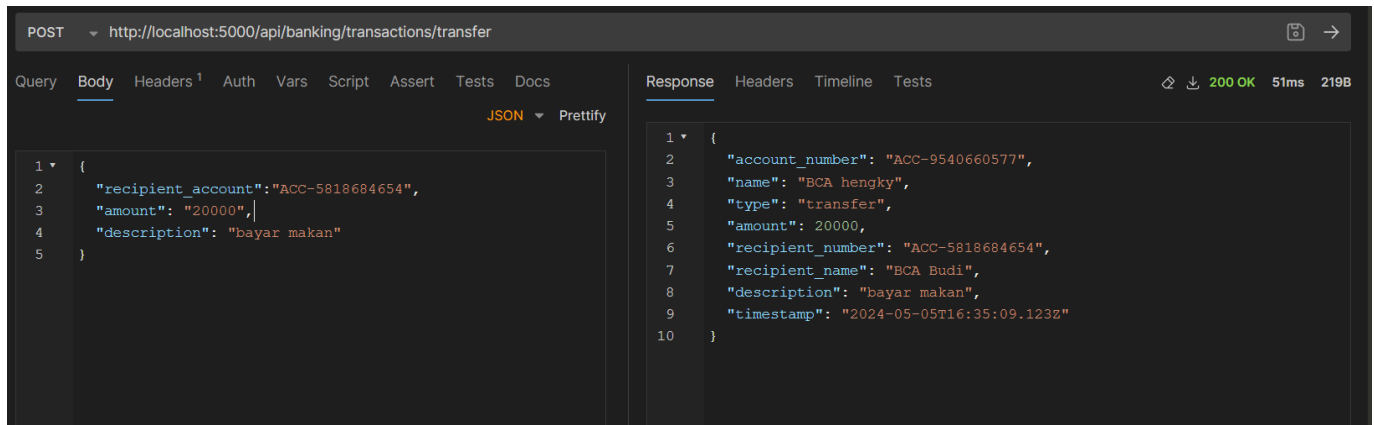
Endpoint ini digunakan untuk melakukan deposit pada banking account. Terdapat 1 field pada body yakni amount.



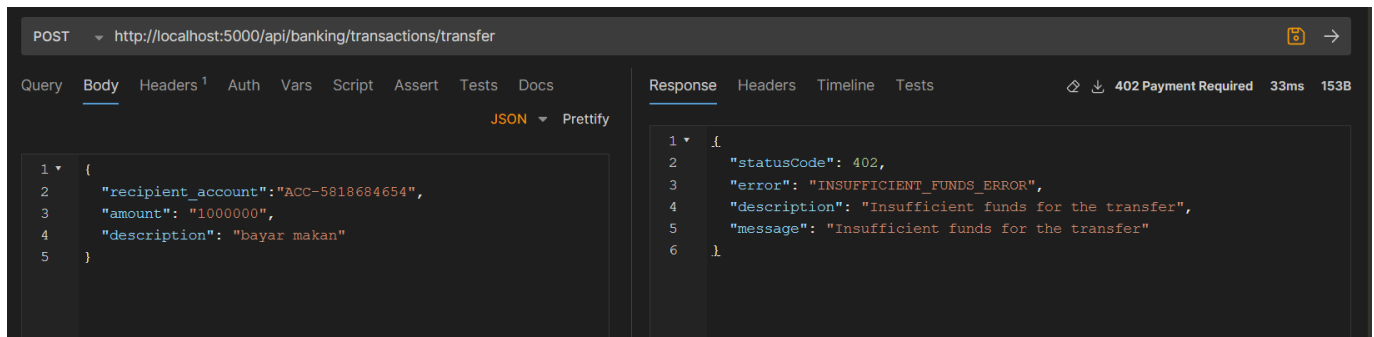


## POST /banking/transactions/transfer

Endpoint ini digunakan untuk melakukan transfer kepada banking account lain. Ketika uang yang akan di transfer lebih besar dari pada saldo yang dimiliki oleh account, maka transfer tidak dapat dilakukan.

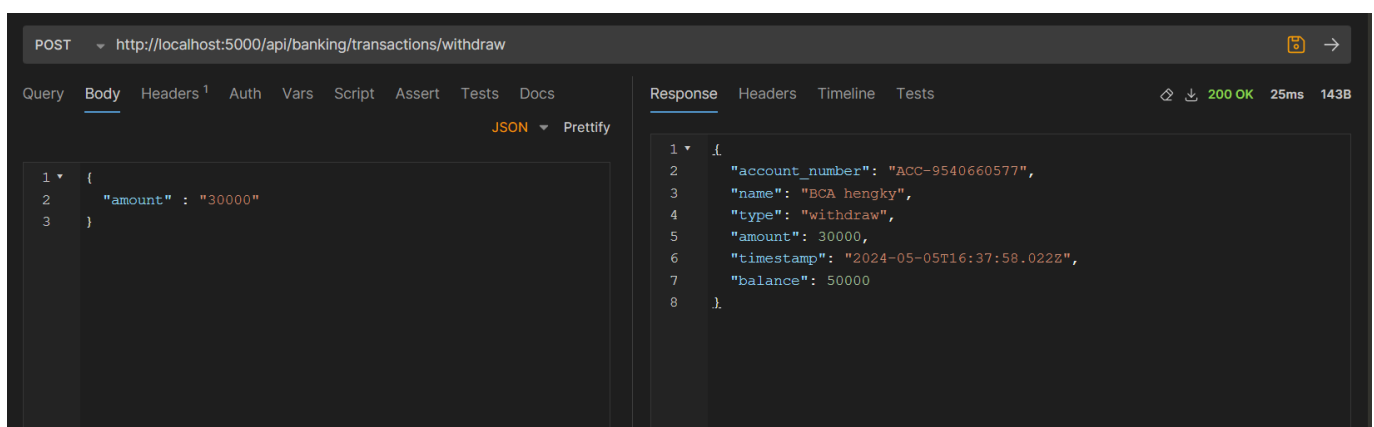


Jika uang yang ditransfer lebih besar dari saldo yang dimiliki.



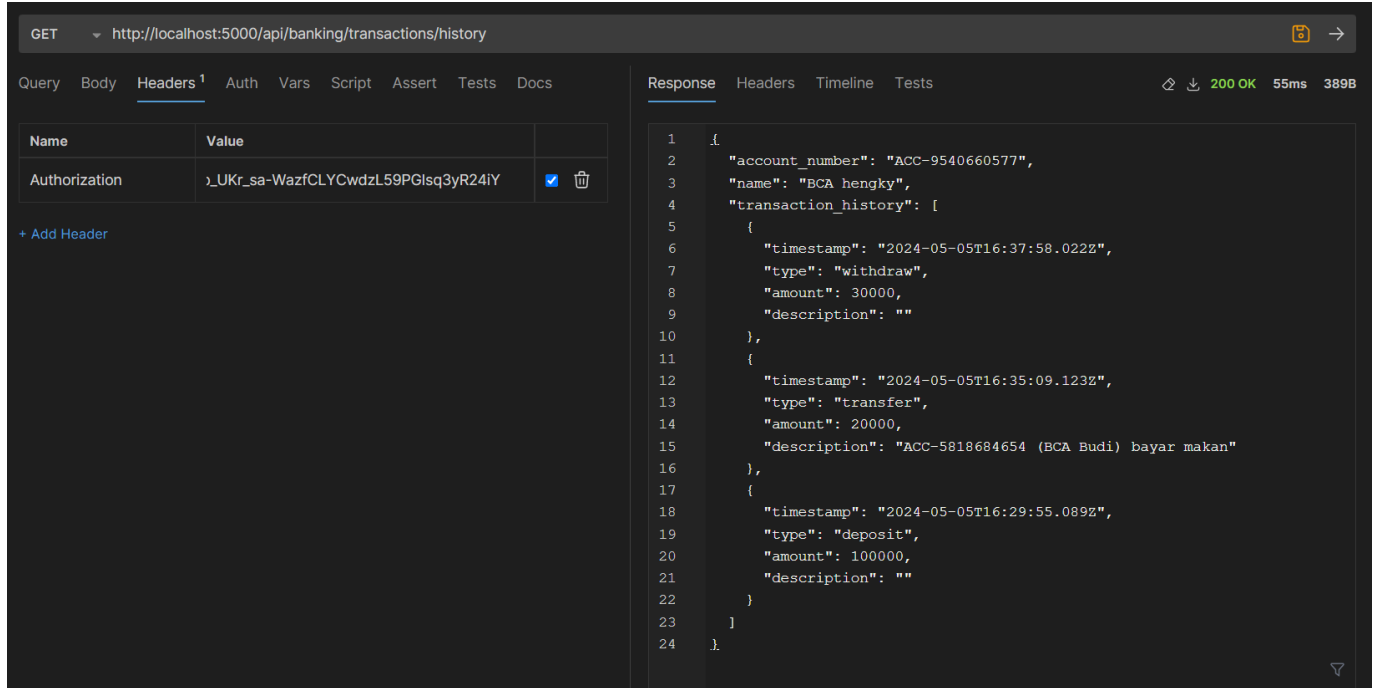
## POST /banking/transactions/withdraw

Endpoint ini digunakan untuk melakukan withdraw uang yang ada di dalam banking account. Ketika uang yang akan di withdraw lebih besar dari pada saldo yang dimiliki oleh account, maka withdraw tidak dapat dilakukan.



## GET /banking/transactions/history

Endpoint ini digunakan untuk melihat semua transaksi yang sudah dilakukan oleh accounts tersebut, transaksi yang didapatkan sudah disort berdasarkan waktu.



Query: GET http://localhost:5000/api/banking/transactions/history

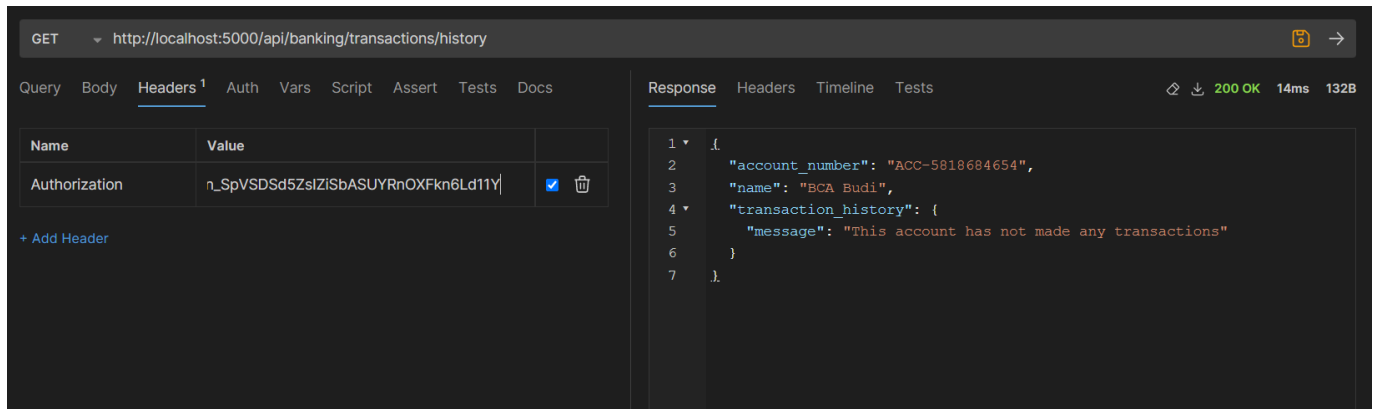
Headers:

Name	Value
Authorization	eyJUKr_sa-WazfCLYCwdzL59PGlsq3yR24lY

Response (200 OK, 55ms, 389B):

```
1 {
2   "account_number": "ACC-9540660577",
3   "name": "BCA hengky",
4   "transaction_history": [
5     {
6       "timestamp": "2024-05-05T16:37:58.022Z",
7       "type": "withdraw",
8       "amount": 30000,
9       "description": ""
10    },
11    {
12      "timestamp": "2024-05-05T16:35:09.123Z",
13      "type": "transfer",
14      "amount": 20000,
15      "description": "ACC-5818684654 (BCA Budi) bayar makan"
16    },
17    {
18      "timestamp": "2024-05-05T16:29:55.089Z",
19      "type": "deposit",
20      "amount": 100000,
21      "description": ""
22    }
23  ]
24 }
```

jika account belum melakukan transaksi



Query: GET http://localhost:5000/api/banking/transactions/history

Headers:

Name	Value
Authorization	n_SpVSDSd5ZsIZIsbASUYRnOXFkn6Ld11Y

Response (200 OK, 14ms, 132B):

```
1 {
2   "account_number": "ACC-5818684654",
3   "name": "BCA Budi",
4   "transaction_history": {
5     "message": "This account has not made any transactions"
6   }
7 }
```

## Admin

Terdapat 2 endpoint pada admin yakni:

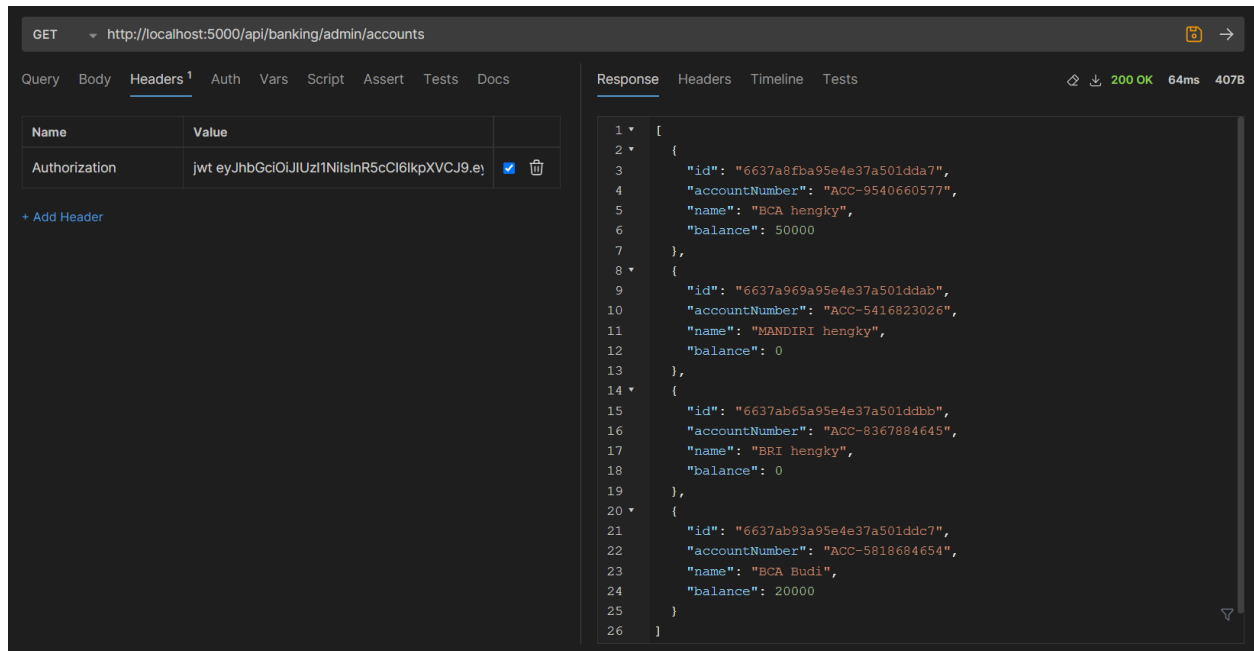
- GET /banking/admin/accounts
- GET /banking/admin/transactions

Authorization pada header kembali menggunakan format “jwt <user token>”

Pada route ini hanya bisa diakses oleh admin yang memiliki email “admin@example.com”, dikarenakan pada route dapat mengakses semua accounts dan transaction

## GET /banking/admin/accounts

Endpoint ini digunakan oleh admin untuk mendapatkan semua account yang terdaftar.



Query GET http://localhost:5000/api/banking/admin/accounts

Query Body Headers<sup>1</sup> Auth Vars Script Assert Tests Docs

Name	Value
Authorization	jwt eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ej

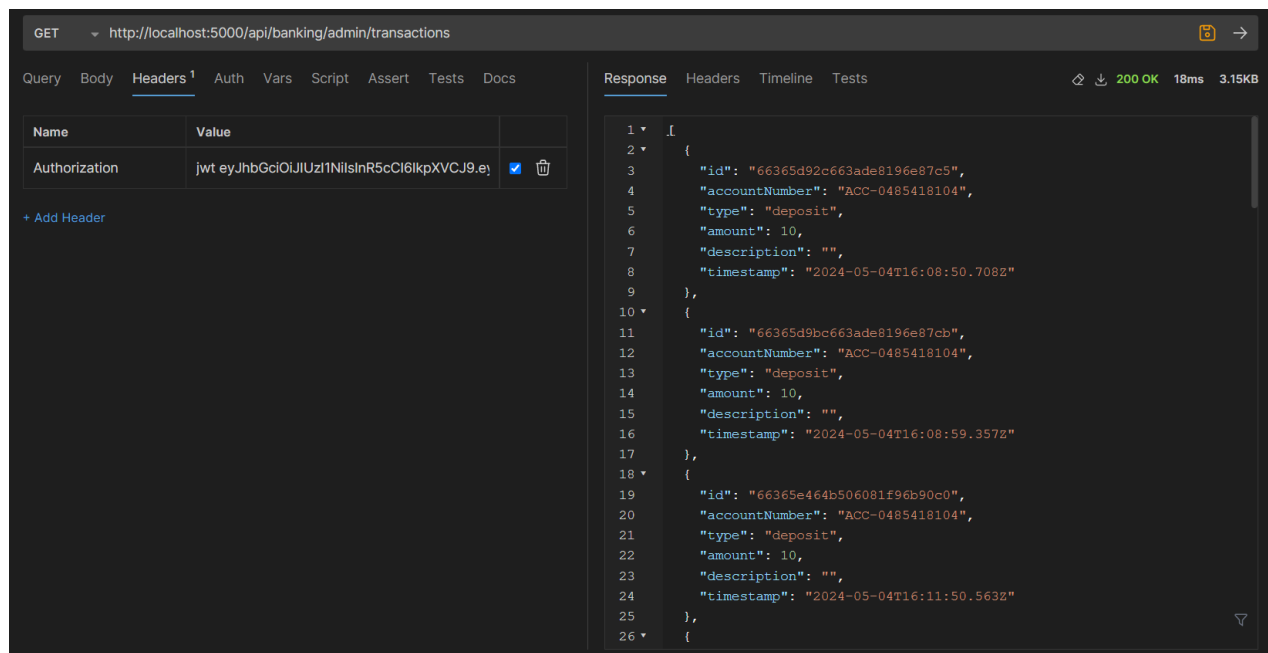
+ Add Header

Response Headers Timeline Tests 200 OK 64ms 407B

```
1 [
2 {
3   "id": "6637a8fba95e4e37a501dda7",
4   "accountNumber": "ACC-9540660577",
5   "name": "BCA hengky",
6   "balance": 50000
7 },
8 {
9   "id": "6637a969a95e4e37a501ddab",
10  "accountNumber": "ACC-5416823026",
11  "name": "MANDIRI hengky",
12  "balance": 0
13 },
14 {
15  "id": "6637ab65a95e4e37a501ddbb",
16  "accountNumber": "ACC-8367884645",
17  "name": "BRI hengky",
18  "balance": 0
19 },
20 {
21  "id": "6637ab93a95e4e37a501ddc7",
22  "accountNumber": "ACC-5818684654",
23  "name": "BCA Budi",
24  "balance": 20000
25 }
26 ]
```

## GET /banking/admin/transactions

Endpoint ini digunakan oleh admin untuk mendapatkan semua transaksi yang sudah terjadi.



Query GET http://localhost:5000/api/banking/admin/transactions

Query Body Headers<sup>1</sup> Auth Vars Script Assert Tests Docs

Name	Value
Authorization	jwt eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ej

+ Add Header

Response Headers Timeline Tests 200 OK 18ms 3.15KB

```
1 [
2 {
3   "id": "66365d92c663ade8196e87c5",
4   "accountNumber": "ACC-0485418104",
5   "type": "deposit",
6   "amount": 10,
7   "description": "",
8   "timestamp": "2024-05-04T16:08:50.708Z"
9 },
10 {
11  "id": "66365d9bc663ade8196e87cb",
12  "accountNumber": "ACC-0485418104",
13  "type": "deposit",
14  "amount": 10,
15  "description": "",
16  "timestamp": "2024-05-04T16:08:59.357Z"
17 },
18 {
19  "id": "66365e464b506081f96b90c0",
20  "accountNumber": "ACC-0485418104",
21  "type": "deposit",
22  "amount": 10,
23  "description": "",
24  "timestamp": "2024-05-04T16:11:50.563Z"
25 },
26 {
```

## Hal Yang Dapat Dikembangkan

- Menambahkan pagination dan filter pada GET /banking/transactions/history, GET /banking/admin/accounts, dan GET /banking/admin/transactions.
- Menambahkan status pada accountSchema sehingga account dapat diaktifkan atau dibekukan
- Menambahkan hak akses pada admin
  - Menghapus banking account
  - Mengaktifkan atau Membekukan banking account