API Documentation

MoneyFest

Login User

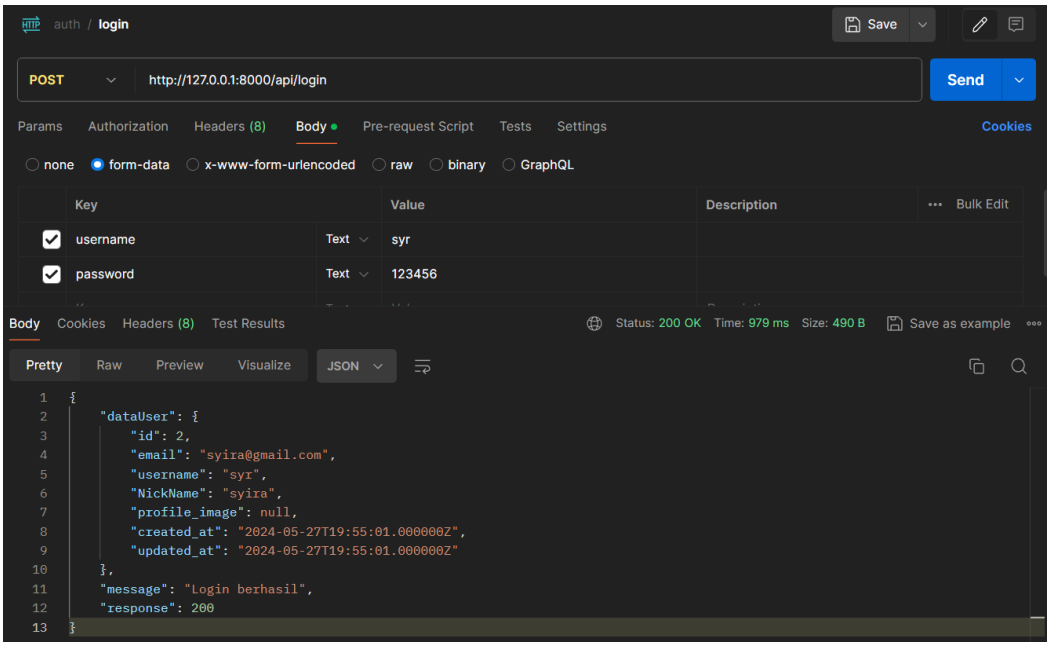
- Endpoint : **POST** '/api/login'
- Deskripsi : otentikasi user untuk masuk ke aplikasi
- Request Body :
    1. username (string, required): User's username.
    2. password (string, required): User's password.
- Response
    - success :

```
{
        "data": {
                "id": 1,
                "email": "user@example.com",
                "username": "user123",
                "NickName": "User",
                "profile_image": null,
                "created_at": "2023-06-08T10:20:30.000000Z",
                "updated_at": "2023-06-08T10:20:30.000000Z"

        },

        "message": "Login Berhasil",

        "status": 200

}
```

    - Error :

```
{
        "message" : "Username/Password salah",
        "response" : 404
}
```

- Contoh Penggunaan :



Register User

- Endpoint : **POST '**/api/register'
- Deskripsi : daftar akun untuk pengguna baru
- Request Body :
    1. email (string, required, unique): User's email address.
    2. username (string, required, unique): User's username.
    3. NickName (string, required): User's nickname.
    4. password (string, required): User's password (minimum 6 characters).
- Response
    - success :

```
{
        "data": {
                "id": 1,
                "email": "user@example.com",
                "username": "user123",
                "NickName": "User",
                "created_at": "2023-06-08T10:20:30.000000Z",
                "updated_at": "2023-06-08T10:20:30.000000Z"

        },

        "message": " User registered successfully",

        "status": 200

}
```

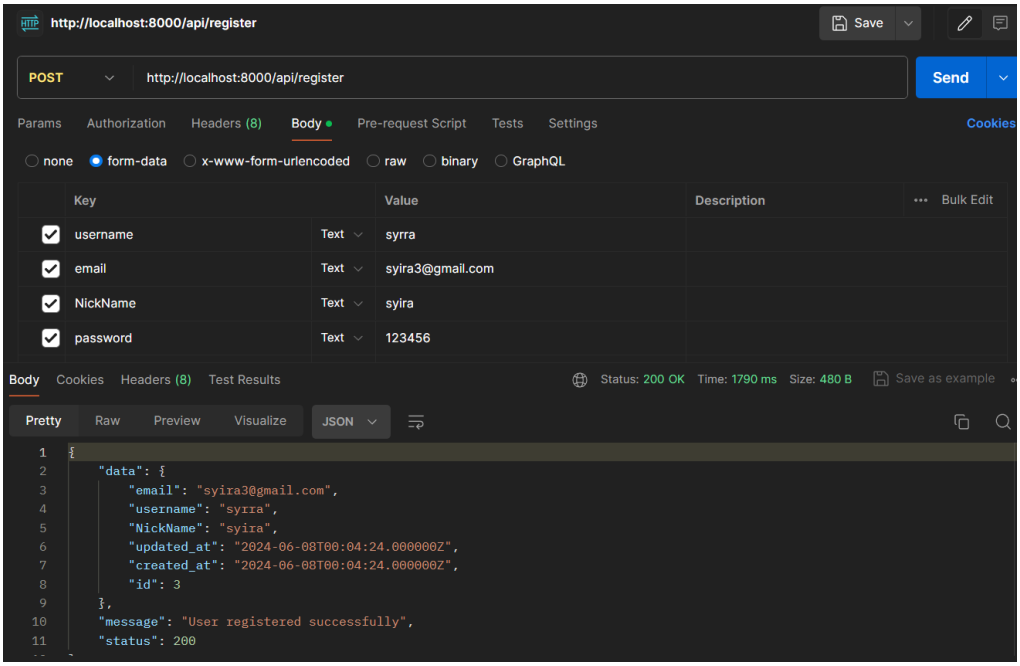
    - Error :
        - **Validation Error**
          Status Code: 422 Unprocessable Entity :

```
{
    "errors": {
        "email": ["The email has already been taken."],
        "username": ["The username has already been taken."],
        "password": ["The password must be at least 6 characters."]
    },
    "message": "Validation failed",
    "status": 422
}
```

        - **General Error**
          Status Code: 500 Internal Server Error :

```
{
    "message": "An error occurred during registration",
    "status": 500
}
```

- Contoh Penggunaan :



Mendapatkan Detail User

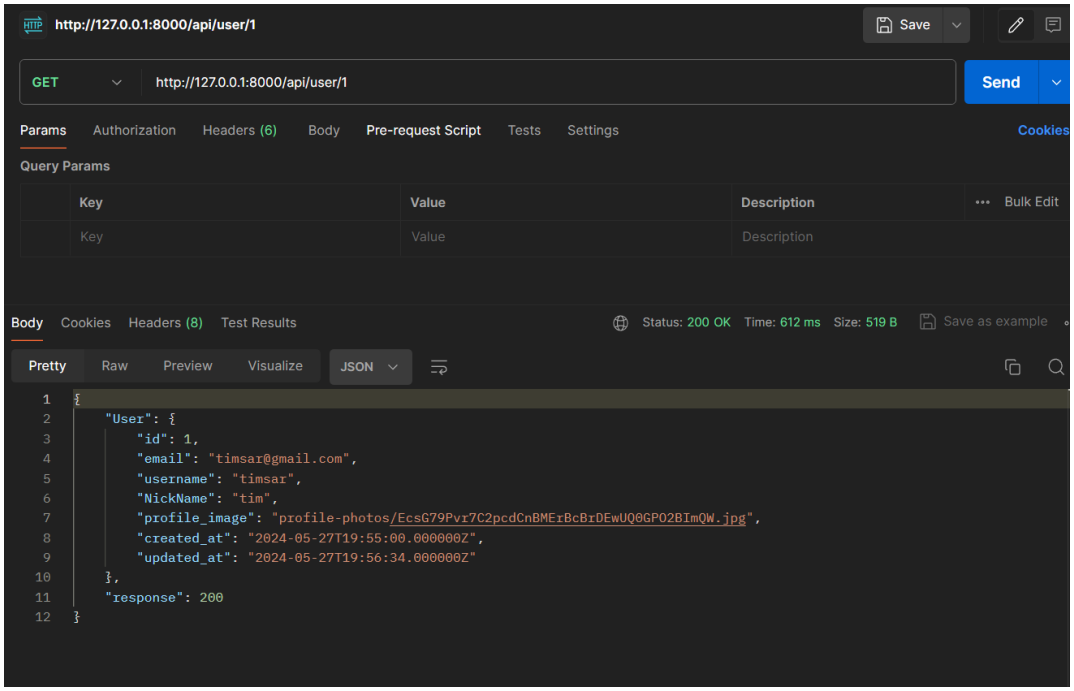
- Endpoint : **GET** '/api/user/{id}'
- Deskripsi : Mendapatkan detail user berdasarkan ID.
- Request Parameter :
    1. id (int, required): ID pengguna.username (string, required, unique): User's username.
- Response
    - success :

```
{
        "data": {
                "id": 1,
                "email": "user@example.com",
                "username": "user123",
                "NickName": "User",
                "profile_image": "path/to/profile/image.jpg",
                "created_at": "2023-06-08T10:20:30.000000Z",
                "updated_at": "2023-06-08T10:20:30.000000Z"
        },
        "response": 200
}
```

- Error :

```
{
    "message": "User not found",
    "response": 404
}
```

- Contoh Penggunaan :



Memperbarui Foto Profil User

- Endpoint : **POST** '/api/update-profile-image'
- Deskripsi : Memperbarui foto profil pengguna.
- Request Body :
    1. user_id (int, required): ID pengguna.
    2. profile_image (file, required): Gambar profil pengguna (harus berupa gambar dengan format jpeg, png, jpg, gif, dan ukuran maksimum 2MB).
- Response
    - success :

```
{
    "message": "Foto profil berhasil diperbarui",
    "photo_path": "profile-photos/new-profile-image.jpg"
}
```

    - Error :
        - Validation Error :

```
{
    "message": "The profile image must be an image."
}
```

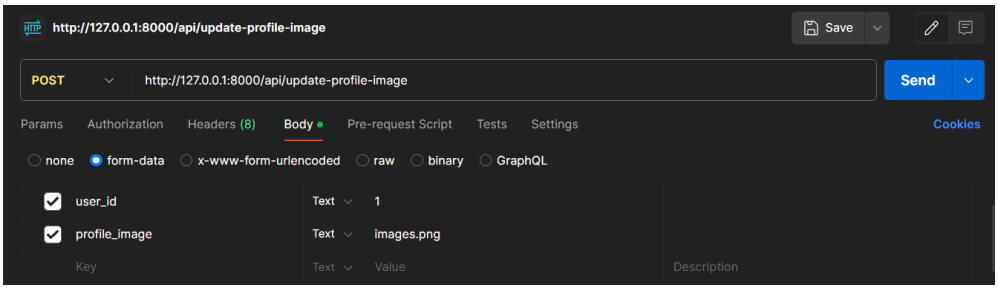
        - User Not Found :

```
{
    "message": "Pengguna tidak ditemukan"
}
```

- General Error :

```
{
    "message": "Gagal memperbarui foto profil"
}
```

- Contoh Penggunaan :



Menampilkan Detail Kategori

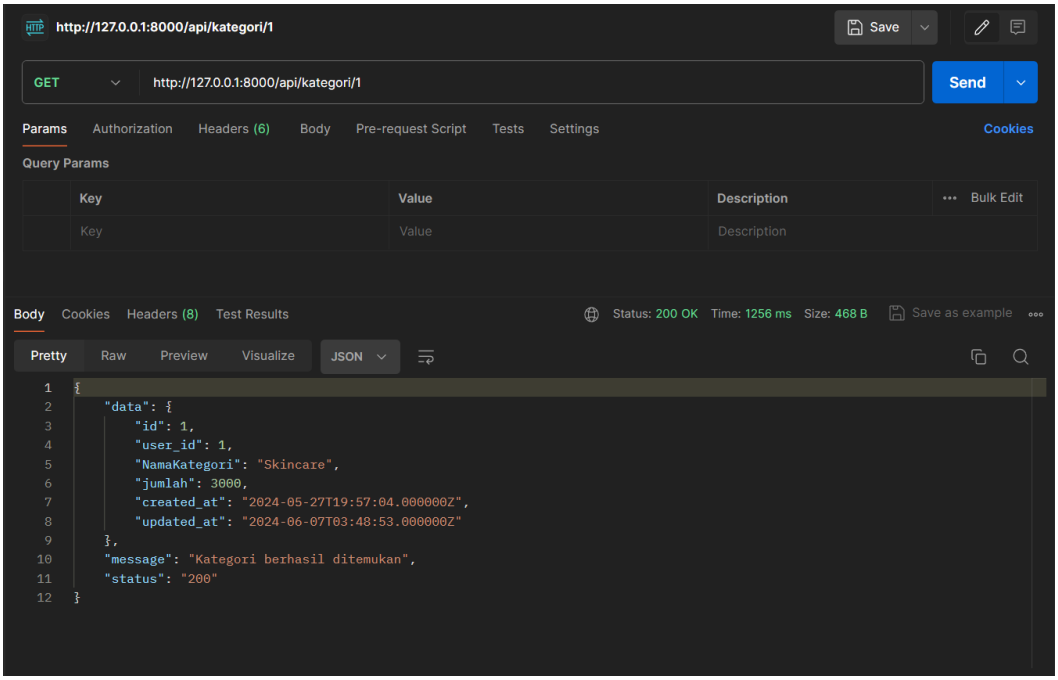
- Endpoint : **GET** 'api/kategori/{id}'
- Deskripsi : Menampilkan detail kategori berdasarkan ID.
- Request Parameter :
    1. id (int, required): ID kategori.
- Response
    - success :

```
{
    "data": {
        "id": 1,
        "user_id": 1,
        "NamaKategori": "Kategori 1",
        "created_at": "2023-06-08T10:20:30.000000Z",
        "updated_at": "2023-06-08T10:20:30.000000Z"
    },
    "message": "Kategori berhasil ditemukan",
    "status": "200"
}
```

    - Error :

```
{
    "message": "Kategori tidak ditemukan",
    "status": "404"
}
```

- Contoh Penggunaan :



Menampilkan Detail Kategori

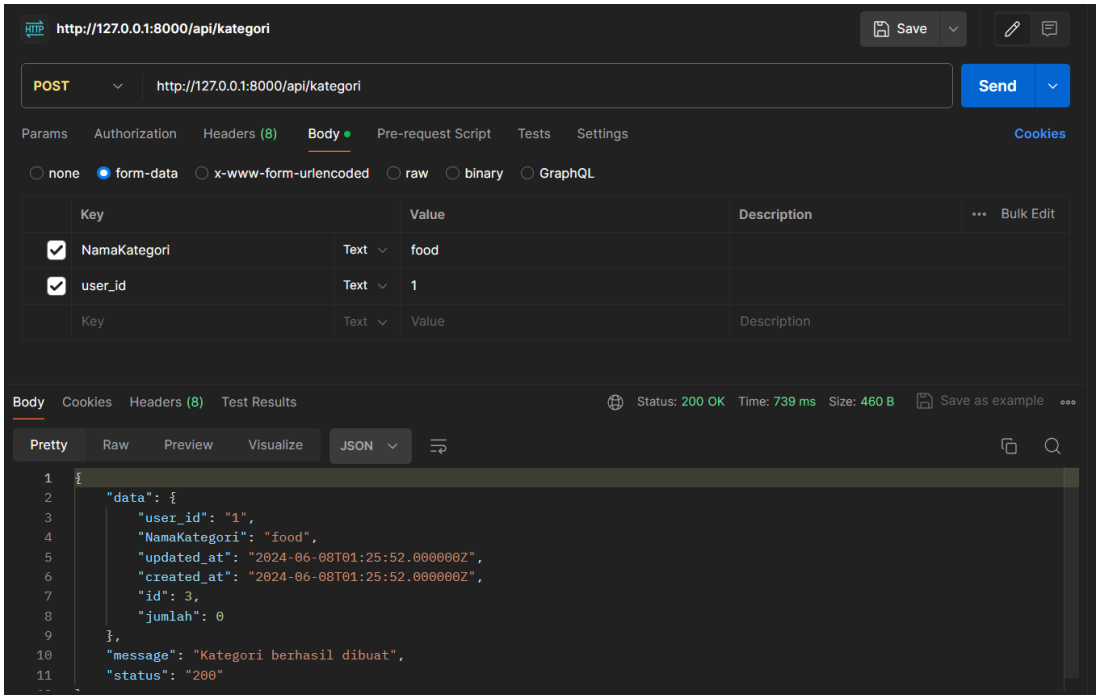
- Endpoint : **POST** 'api/kategori'
- Deskripsi : Menambahkan kategori baru..
- Request Body :
    1. user_id (int, required): ID pengguna yang membuat kategori.
    2. NamaKategori (string, required): Nama kategori yang akan ditambahkan.
- Response
    - success :

```
{
    "data": {
        "id": 1,
        "user_id": 1,
        "NamaKategori": "Kategori 1",
        "created_at": "2023-06-08T10:20:30.000000Z",
        "updated_at": "2023-06-08T10:20:30.000000Z"
    },
    "message": "Kategori berhasil dibuat",
    "status": "200"
}
```

    - Error :

```
{
    "error": {
        "user_id": ["The user id field is required."]
    },
    "message": "Gagal membuat kategori",
    "status": "400"
}
```

Contoh Penggunaan :



Mengedit Kategori

- Endpoint : **PUT** ‘api/kategori/edit/{id}’
- Deskripsi : Mengedit kategori berdasarkan ID.
- Request Parameter :
    1. **id** (int, required): ID kategori yang akan diedit.
- Request Body :
    1. **NamaKategori** (string, required): Nama kategori yang baru.
- Response
    - success :

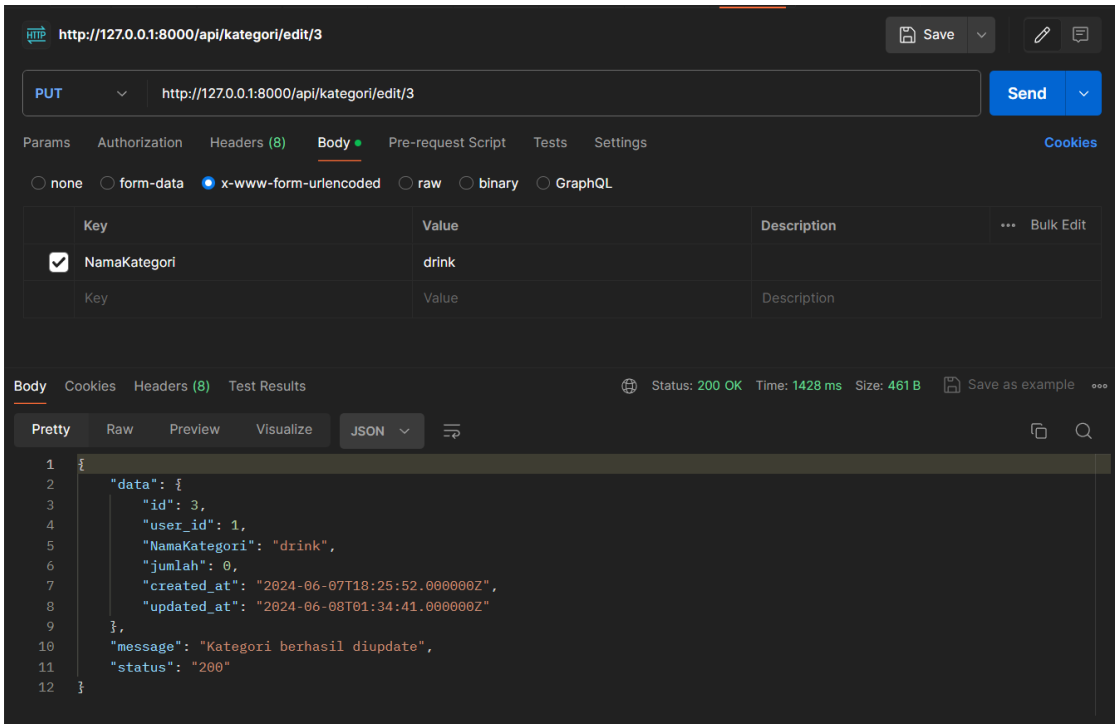
```
{
    "data": {
        "id": 1,
        "user_id": 1,
        "NamaKategori": "Kategori 1",
        "created_at": "2023-06-08T10:20:30.000000Z",
        "updated_at": "2023-06-08T10:20:30.000000Z"
    },
    "message": "Kategori berhasil diupdate",
    "status": "200"
}
```

    - Error :

```
{
    "message": "Kategori tidak ditemukan",
```

```
    "status": "404"
}
```

Contoh Penggunaan :



Menghapus Kategori

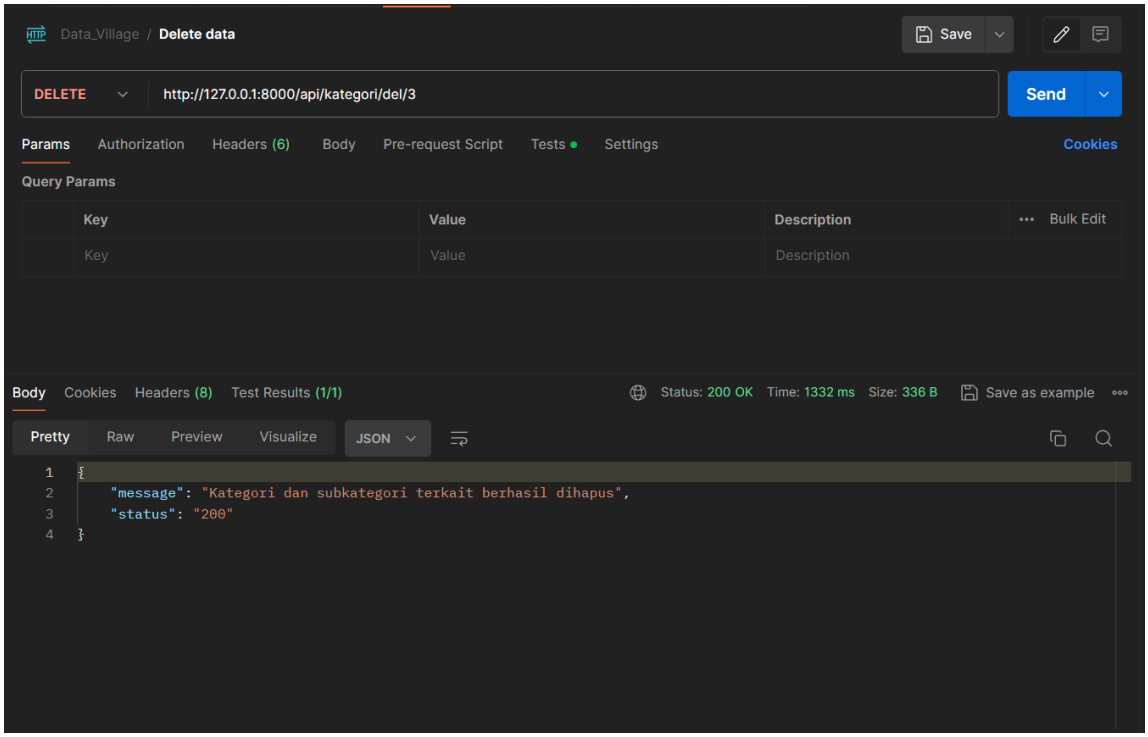
- Endpoint : **DELETE** ‘api/kategori/del/{id}’
- Deskripsi : Menghapus kategori berdasarkan ID.
- Request Parameter :
    1. **id** (int, required): ID kategori yang akan dihapus.
- Response
    - success :

```
{
    "message": "Kategori dan subkategori terkait berhasil dihapus",
    "status": 200
}
```

    - Error :

```
{
    "message": "Kategori tidak ditemukan",
    "status": 404
}
```

Contoh Penggunaan :



Mengambil Kategori Berdasarkan Pengguna (User)

- Endpoint : **GET** 'api/kategori/user/{userId}'
- Deskripsi : Mengambil semua kategori beserta subkategori yang terkait dengan pengguna (user) tertentu berdasarkan ID pengguna.
- Request Parameter :
    1. userId (int, required): ID pengguna.
- Response
    - success :

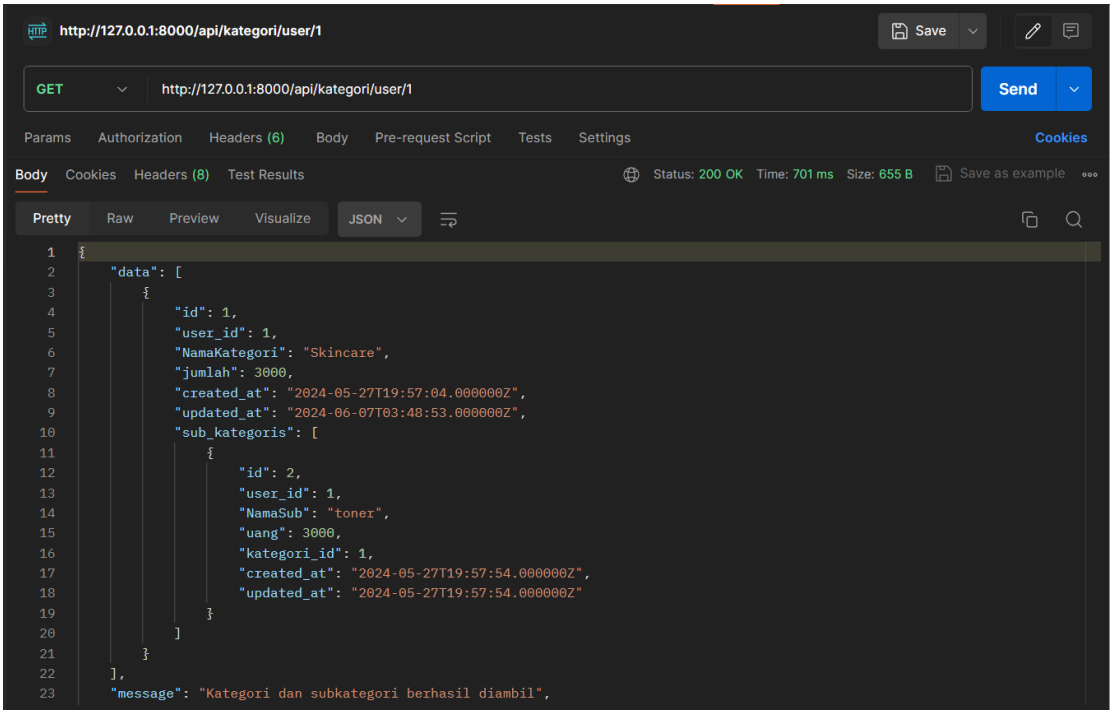
```
{
    "data": [
        {
            "id": 1,
            "user_id": 1,
            "NamaKategori": "Kategori 1",
            "created_at": "2023-06-08T10:20:30.000000Z",
            "updated_at": "2023-06-08T10:20:30.000000Z",
            "subKategoris": [
                // Subkategori details here
            ]
        },
        // Other categories
    ],
    "message": "Kategori dan subkategori berhasil diambil",
```

```
    "status": 200
}
```


- Error :


```
{
    "message": "Kategori tidak ditemukan",
    "status": 404
}
```


- Contoh Penggunaan :



Menampilkan Grafik Kategori Berdasarkan Pengguna (User) dan Bulan Kategori Dibuat

- Endpoint : **GET** ‘`api/categories/${widget.userId}/by-month/$formattedMonth`’
- Deskripsi : Menampilkan data kategori berdasarkan pengguna (user) dan bulan kategori dibuat (created-at) dalam bentuk grafik.
- Request Parameter :
    1. **userId** (int, required): ID pengguna.
- Response
    - success :

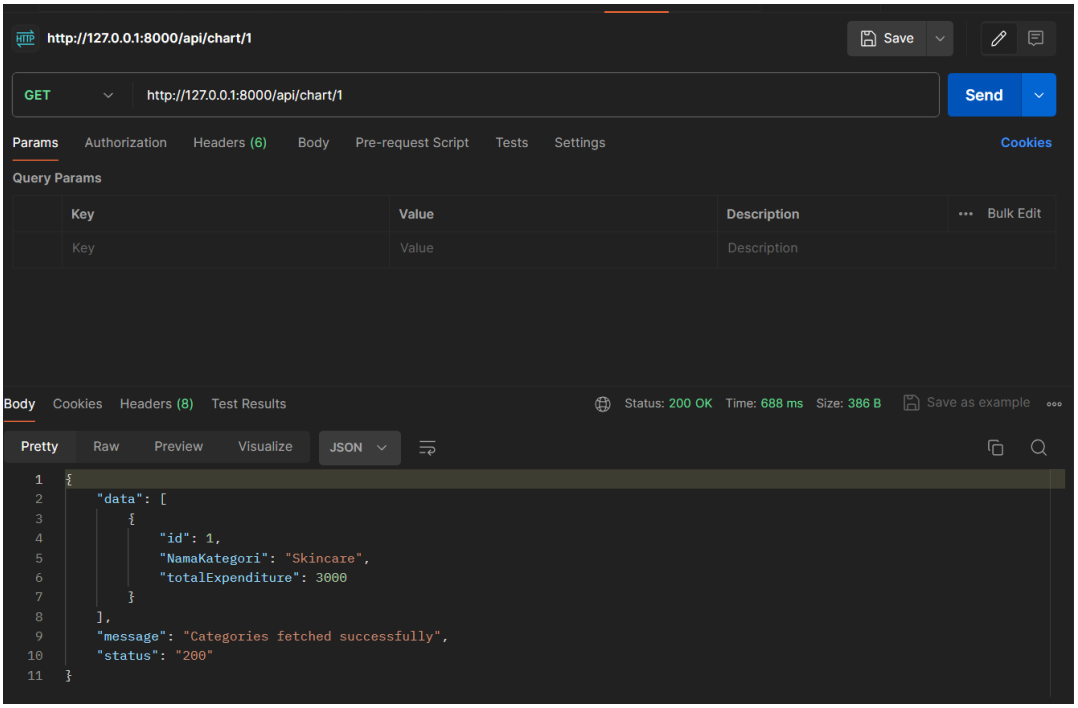

```
{
    "data": [
    {
            "id": 7,
            "user_id": 1,
            "NamaKategori": "ngetes aja",
```

```
            "jumlah": 25000,
            "created_at": "2024-06-08T15:02:37.000000Z",
            "updated_at": "2024-06-08T15:26:05.000000Z"
        },
        {
            "id": 9,
            "user_id": 1,
            "NamaKategori": "ngees 3",
            "jumlah": 0,
            "created_at": "2024-06-08T15:25:00.000000Z",
            "updated_at": "2024-06-08T15:25:00.000000Z"
        },
        {
            "id": 10,
            "user_id": 1,
            "NamaKategori": "p",
            "jumlah": 14000,
            "created_at": "2024-06-08T15:42:23.000000Z",
            "updated_at": "2024-06-08T15:43:03.000000Z"
        },
        {
            "id": 11,
            "user_id": 1,
            "NamaKategori": "tes",
            "jumlah": 25000,
            "created_at": "2024-06-08T18:48:51.000000Z",
            "updated_at": "2024-06-08T18:49:11.000000Z"
        }
    ],
    "message": "Kategori berhasil diambil berdasarkan bulan June 2024",
    "status": "200"
```


- Error :


```
{
    "message": "Kategori tidak ditemukan",
    "status": "404"
}
```


- Contoh Penggunaan :



Membuat Subkategori Baru

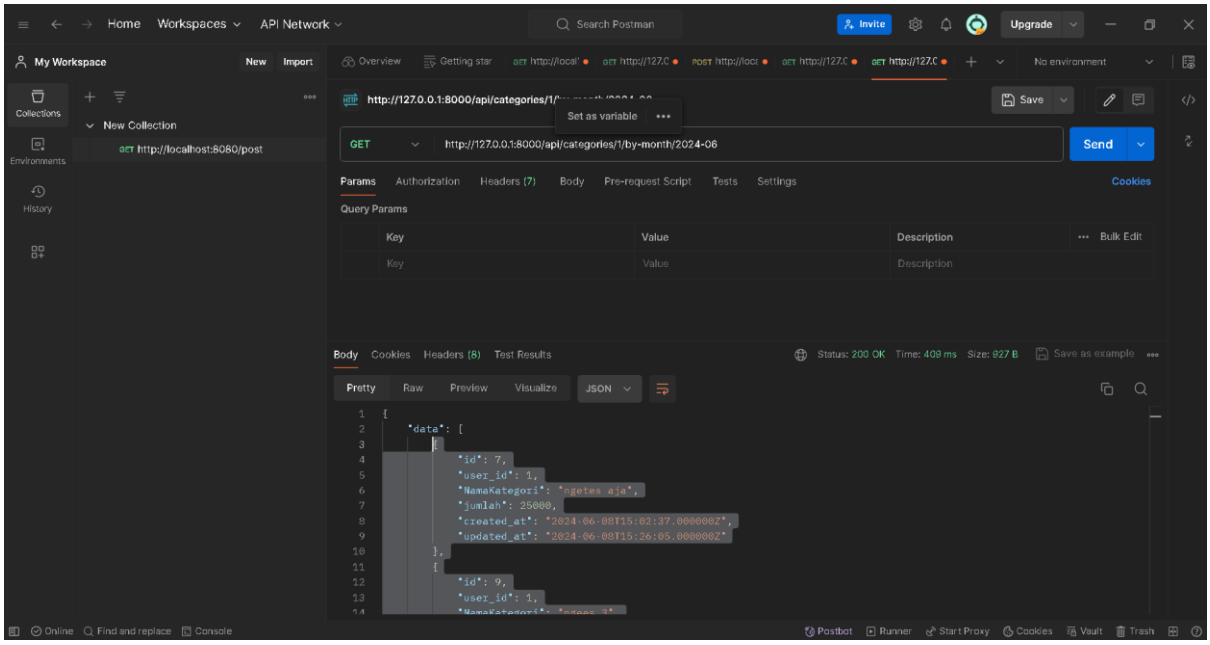
- Endpoint : **POST** 'api/subkategori'
- Deskripsi : Membuat subkategori baru untuk kategori tertentu.
- Request Body :
    1. user_id (int, required): ID pengguna.
    2. NamaSub (string, required): Nama subkategori.
    3. uang (numeric, required): Jumlah uang untuk subkategori.
    4. kategori_id (int, required): ID kategori yang terkait.
- Response
    - success :

```
{
    "data": {
        "id": 1,
        "user_id": 1,
        "NamaSub": "Subkategori 1",
        "uang": 1000,
        "kategori_id": 1,
        "created_at": "2023-06-08T10:20:30.000000Z",
        "updated_at": "2023-06-08T10:20:30.000000Z"
    },
    "message": "Subkategori berhasil dibuat",
    "status": 200
}
```

    - Error :

```
{
    "message": "Gagal membuat sub kategori",
    "errors": {
        "NamaSub": [
            "NamaSub field is required"
        ],
        "uang": [
            "The uang field is required."
        ],
        "kategori_id": [
            "The kategori id field is required."
        ]
    },
    "status": "400"
}
```

Contoh Penggunaan :



]

Mengedit Subkategori

- Endpoint : **PUT** ‘api/subkategori/edit/{id}’
- Deskripsi : Mengedit subkategori yang sudah ada berdasarkan ID.
- Request Parameter :
    1. **id** (int, required): ID subkategori yang akan diubah.
- Response
    - success :

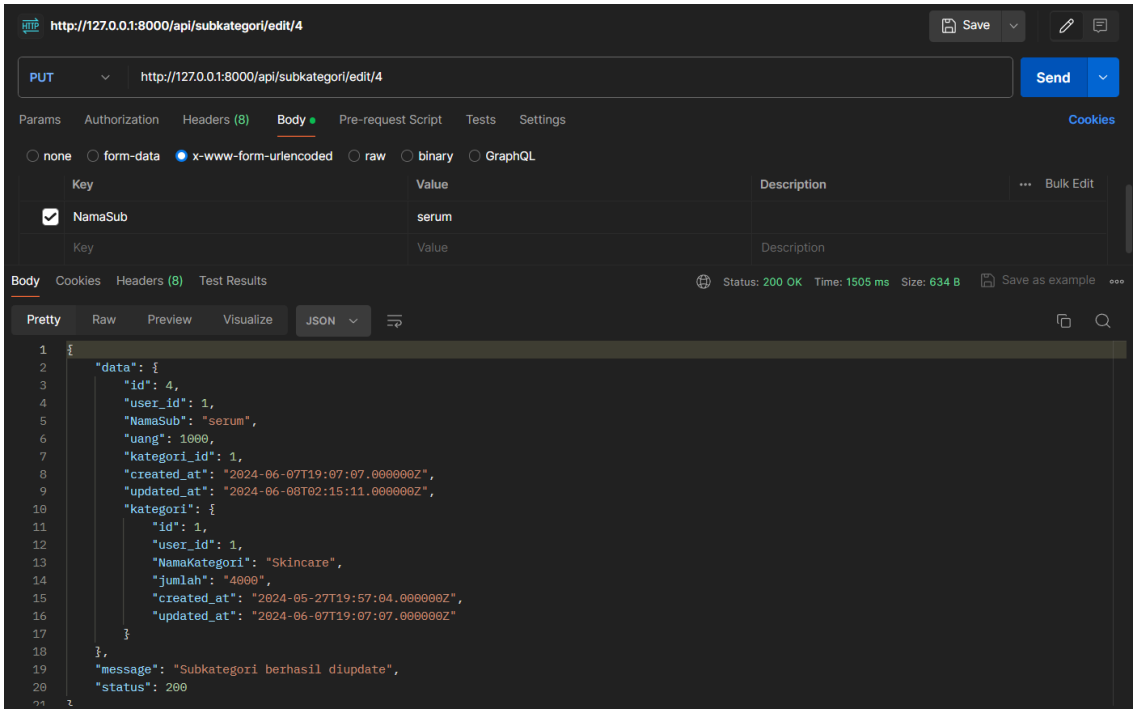
```
{
    "data": {
        "id": 1,
        "user_id": 1,
        "NamaSub": "Subkategori 1 (Updated)",
        "uang": 2000,
        "kategori_id": 1,
```

```
            "created_at": "2023-06-08T10:20:30.000000Z",
            "updated_at": "2023-06-08T10:22:30.000000Z"
        },
        "message": "Subkategori berhasil diupdate",
        "status": 200
    }
```


- Error :


```
{
    "message": "Sub kategori tidak ditemukan",
    "status": "404"
}
```


Contoh Penggunaan :



Menghapus Subkategori

- Endpoint : **DELETE** ‘api/subkategori/del/{id}’
- Deskripsi : Menghapus subkategori yang sudah ada berdasarkan ID.
- Request Parameter :
    1. id (int, required): ID subkategori yang akan dihapus.
- Response
    - success :

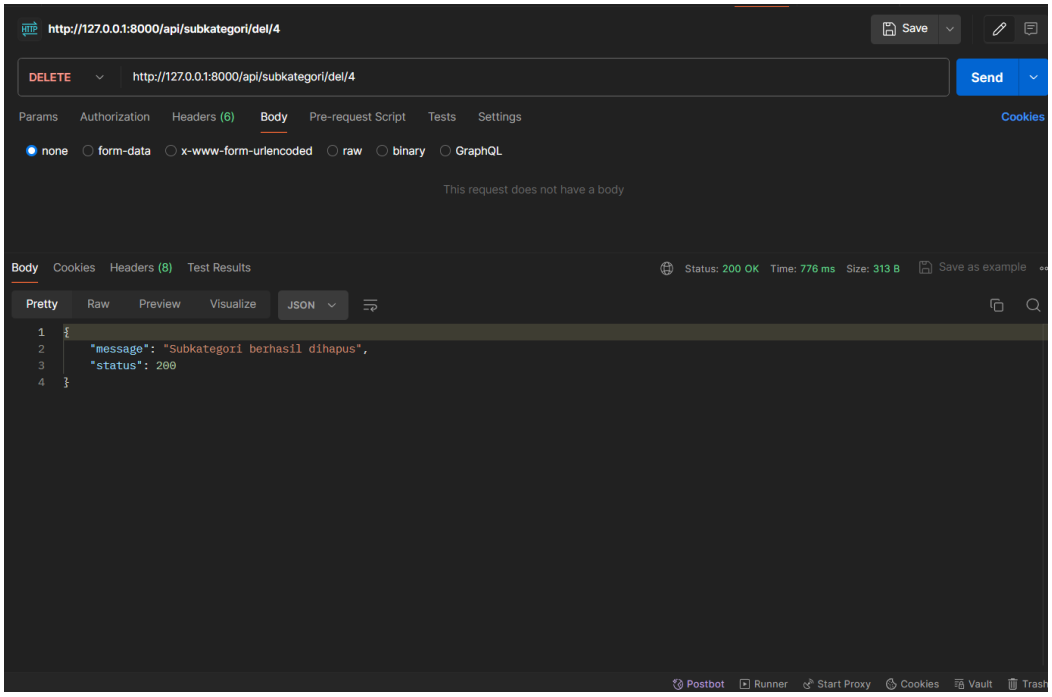

```
{
    "message": "Subkategori berhasil dihapus",
    "status": 200
}
```


    - Error :


```
{
```

```
    "message": "Sub kategori tidak ditemukan",
    "status": "404"
}
```

Contoh Penggunaan :



Mengambil Subkategori Berdasarkan Pengguna dan Kategori

- Endpoint : **GET** ‘api/subkategori/user/{userId}/{kategoriId}’
- Deskripsi : Mengambil semua subkategori yang dimiliki oleh pengguna dan terkait dengan kategori tertentu.
- Request Parameter :
    1. userId (int, required): ID pengguna.
    2. kategoriId (int, required): ID kategori.
- Response
    - success :

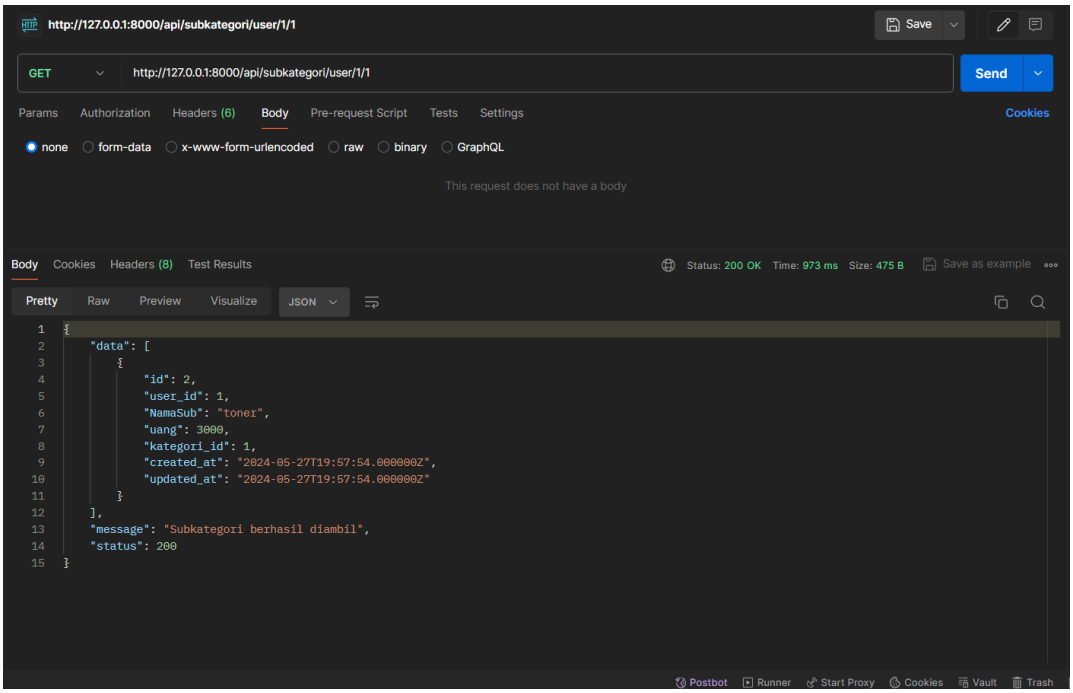
```
{
    "data": [
        {
            "id": 1,
            "user_id": 1,
            "NamaSub": "Subkategori 1",
            "uang": 1000,
            "kategori_id": 1,
            "created_at": "2023-06-08T10:20:30.000000Z",
            "updated_at": "2023-06-08T10:20:30.000000Z"
        },
        // Other subcategories
```

```
    ],
    "message": "Subkategori berhasil diambil",
    "status": 200
}
```


- Error :


```
{
    "data": [],
    "message": "Tidak ada subkategori yang ditemukan",
    "status": 404
}
```


Contoh Penggunaan :



Menampilkan Detail Instalment

- Endpoint : **GET** ‘api/instalments/{id}’
- Deskripsi : Mengambil detail instalment berdasarkan ID.
- Request Parameter :
    1. Id (int, required): ID pengguna.
- Response
    - success :

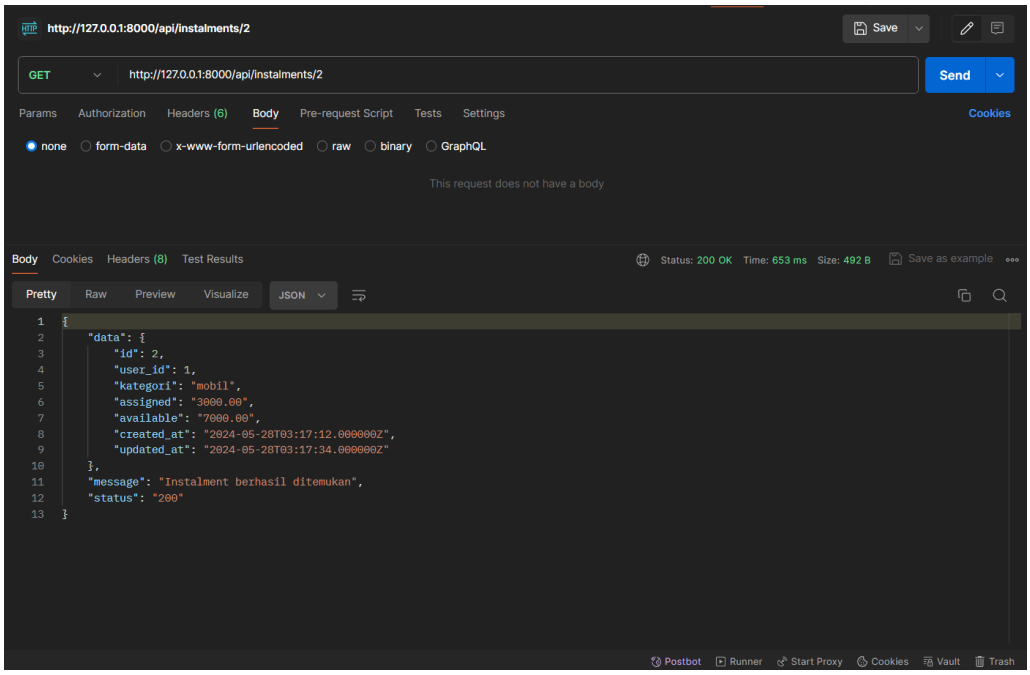

```
{
    "data": {
        "id": 1,
        "user_id": 1,
        "kategori": "Hiburan",
        "available": 500,
        "assigned": 500,
```

```
            "created_at": "2023-06-08T10:20:30.000000Z",
            "updated_at": "2023-06-08T10:20:30.000000Z"
        },
        "message": "Instalment berhasil ditemukan",
        "status": 200
    }
```


- Error :


```
{
    "message": "Instalment tidak ditemukan",
    "status": "404"
}
```


Contoh Penggunaan :



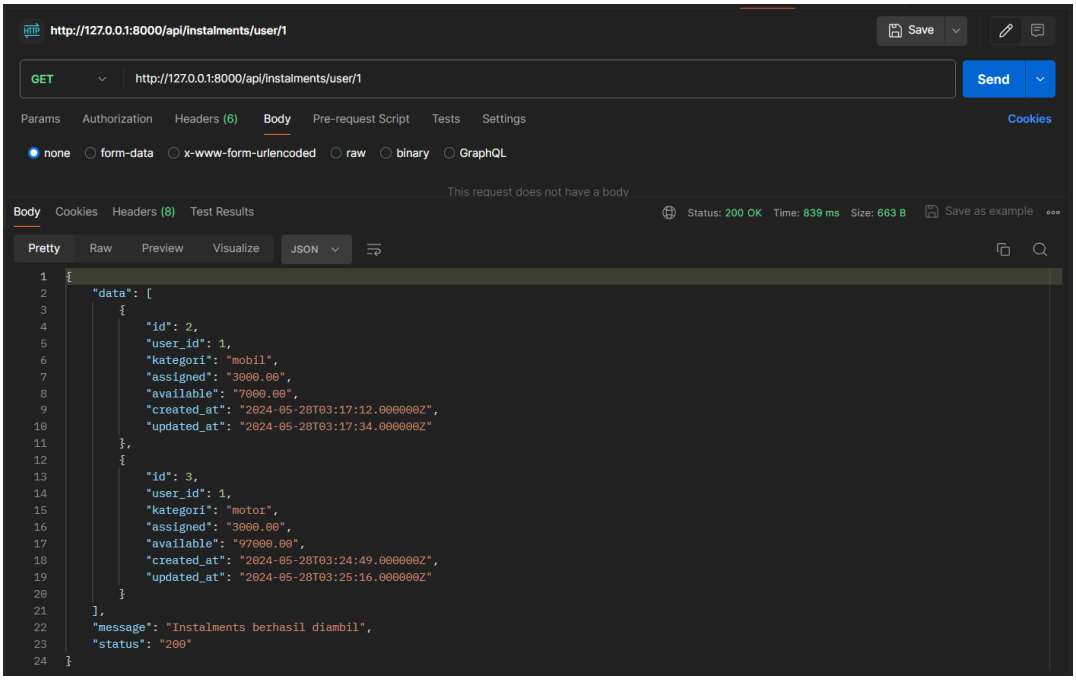
Menampilkan Daftar Instalment Pengguna

- Endpoint : **GET** ‘api/instalments/user/{userId}’
- Deskripsi : Mengambil semua instalment yang dimiliki oleh pengguna..
- Request Parameter :
    1. **userId** (int, required): ID pengguna.
- Response
    - success :


```
{
    "data": [
        {
            "id": 1,
            "user_id": 1,
            "kategori": "Hiburan",
```

```
                "available": 500,
                "assigned": 500,
                "created_at": "2023-06-08T10:20:30.000000Z",
                "updated_at": "2023-06-08T10:20:30.000000Z"
            },
            // Other instalments
        ],
        "message": "Instalments berhasil diambil",
        "status": "200"
    }
```


Contoh Penggunaan :



Menambahkan Instalment Baru

- Endpoint : **POST** ‘api/create/instalments’
- Deskripsi : Membuat instalment baru.
- Request Body :
    1. user_id (int, required): ID pengguna.
    2. kategori (string, required): Nama kategori instalment.
    3. available (numeric, required): Jumlah uang yang tersedia untuk instalment.
- Response
    - success :

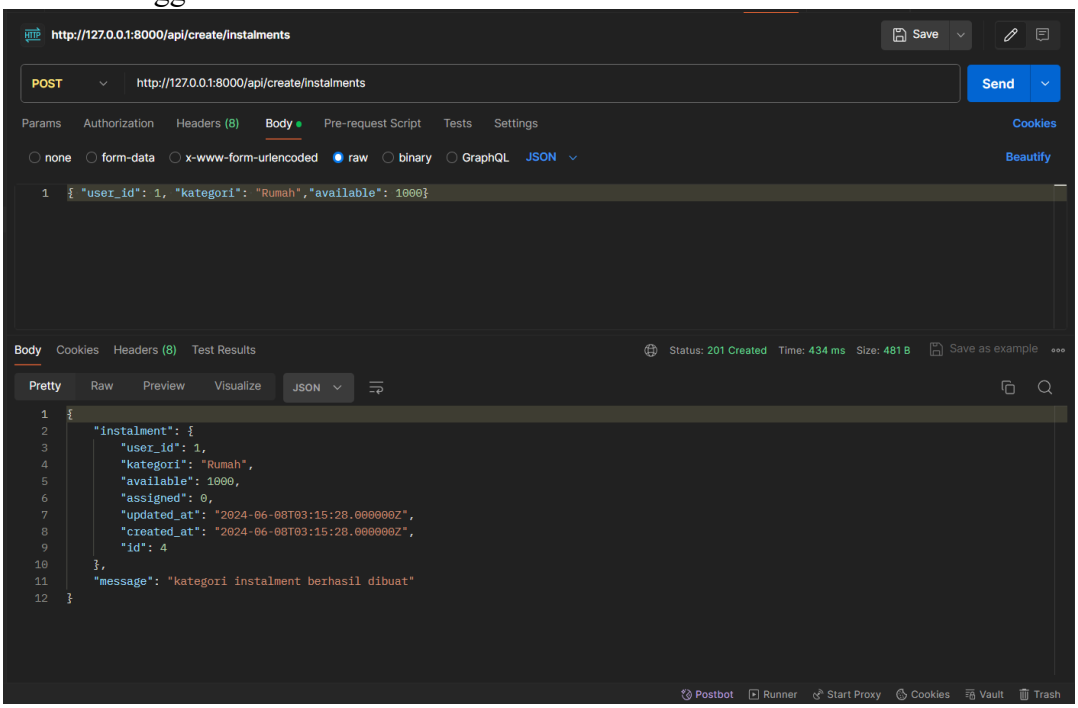

```
    {
        "instalment": {
            "id": 1,
```

```
            "user_id": 1,
            "kategori": "Hiburan",
            "available": 500,
            "assigned": 0,
            "created_at": "2023-06-08T10:20:30.000000Z",
            "updated_at": "2023-06-08T10:20:30.000000Z"
        },
        "message": "kategori instalment berhasil dibuat"
    }
```


- Error :


```
{
    "message": "The given data was invalid.",
    "errors": {
        "user_id": ["The user id field is required."],
        "kategori": ["The kategori field is required."],
        "available": ["The available field is required."]
    }
}
```


Contoh Penggunaan :



AddAmount Instalment

- Endpoint : **PUT** ‘api/instalments/{id}’
- Deskripsi : Memasukkan dan menambah uang di instalment.
- Request Parameter :
    1. id (int, required): ID instalment yang akan diperbarui.
- Requset Body

1. **assigned** (numeric, required): Jumlah uang yang dialokasikan ke instalment.

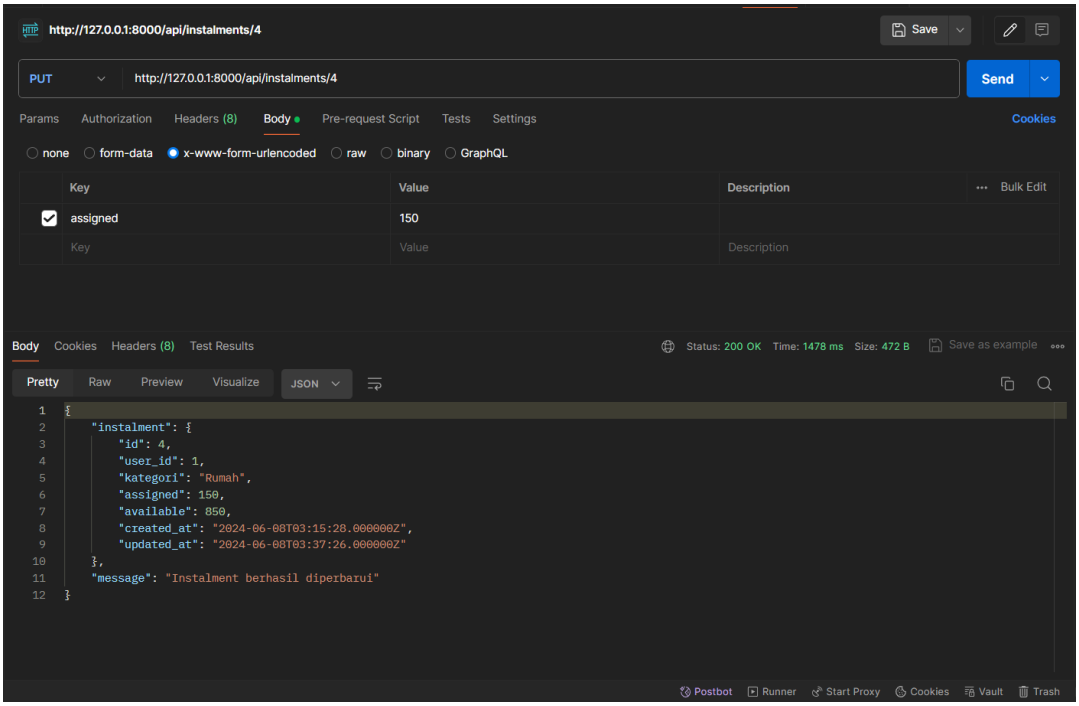
- Response
  - success :

```
{
    "instalment": {
        "id": 1,
        "user_id": 1,
        "kategori": "Hiburan",
        "available": 400,
        "assigned": 100,
        "created_at": "2023-06-08T10:20:30.000000Z",
        "updated_at": "2023-06-08T10:30:45.000000Z"
    },
    "message": "Instalment berhasil diperbarui"
}
```

  - Error :

```
{
    "message": "Input jumlah uang dengan sesuai"
}
```

Contoh Penggunaan :



Menghapus Instalment

- Endpoint : **DELETE** ‘api/instalments/del/{id}’
- Deskripsi : Menghapus instalment berdasarkan ID.

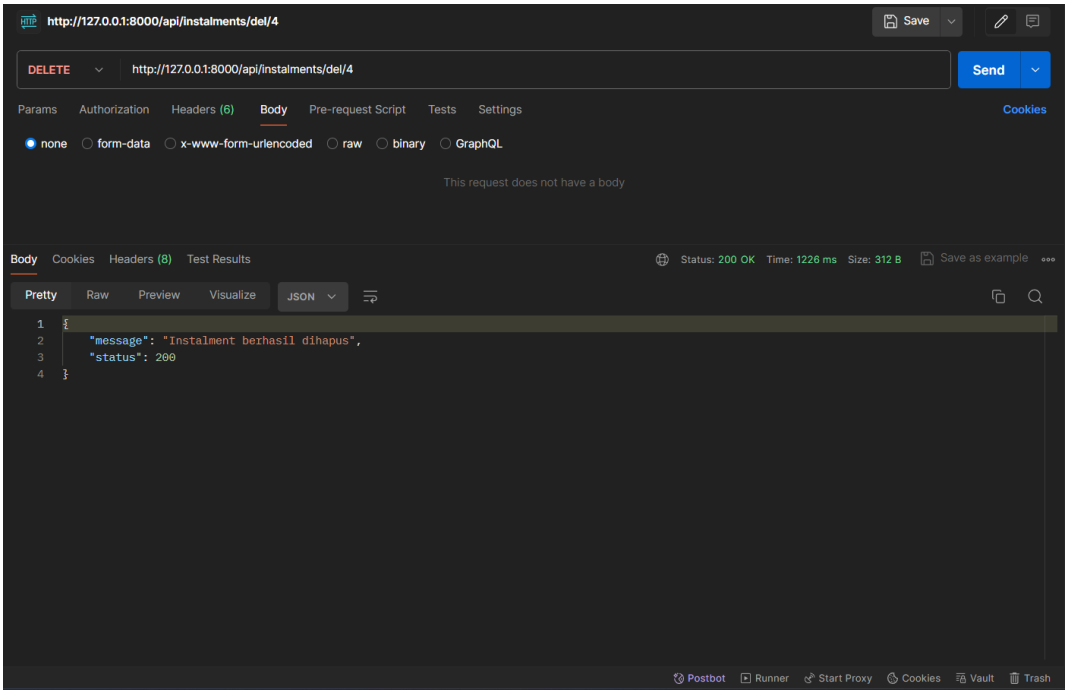
- Request Parameter :
    1. **id** (int, required): ID instalment yang akan dihapus.
- Response
    - success :

```
{
    "message": "Instalment berhasil dihapus"
}
```

    - Error :

```
{
    "message": "Instalment tidak ditemukan",
    "status": "404"
}
```

Contoh Penggunaan :



Mendapatkan Instalments Berdasarkan Pengguna

- Endpoint : **GET** ‘api/instalments/user/{userId}’
- Deskripsi : Mengambil daftar instalments berdasarkan ID pengguna.
- Request Parameter :
    1. userId (int, required): ID pengguna untuk mencari instalments.
- Response
    - success :

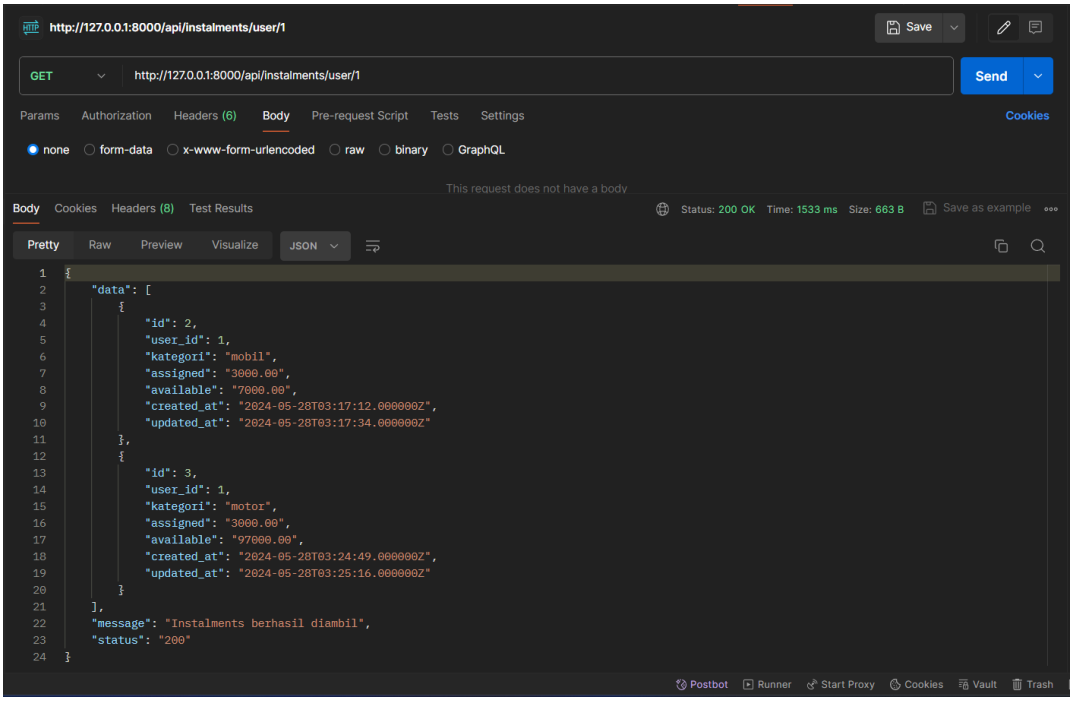
```
{
    "data": [
        {
            "id": 1,
```

```
            "user_id": 1,
            "kategori": "Hiburan",
            "available": 500,
            "assigned": 0,
            "created_at": "2023-06-08T10:20:30.000000Z",
            "updated_at": "2023-06-08T10:20:30.000000Z"
        },
        {
            "id": 2,
            "user_id": 1,
            "kategori": "Belanja",
            "available": 200,
            "assigned": 100,
            "created_at": "2023-06-08T10:25:45.000000Z",
            "updated_at": "2023-06-08T10:30:15.000000Z"
        }
    ],
    "message": "Instalments berhasil diambil",
    "status": "200"
}
```


- Error :


```
{
    "data": [],
    "message": "Tidak ada instalment yang ditemukan",
    "status": "404"
}
```


Contoh Penggunaan :

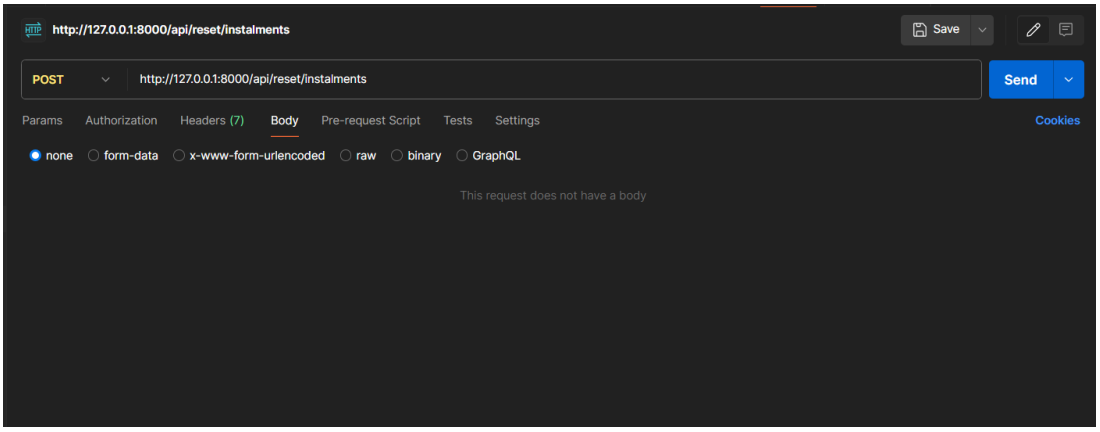


Mengatur Ulang Instalments

- Endpoint : **POST** ‘api/reset/instalments’
- Deskripsi : Menghapus semua data instalments dari database.
- Response
  - success :

```
{
    "message": "Semua data instalment berhasil dihapus"
}
```

Contoh Penggunaan :



Menambahkan atau Memperbarui Saldo

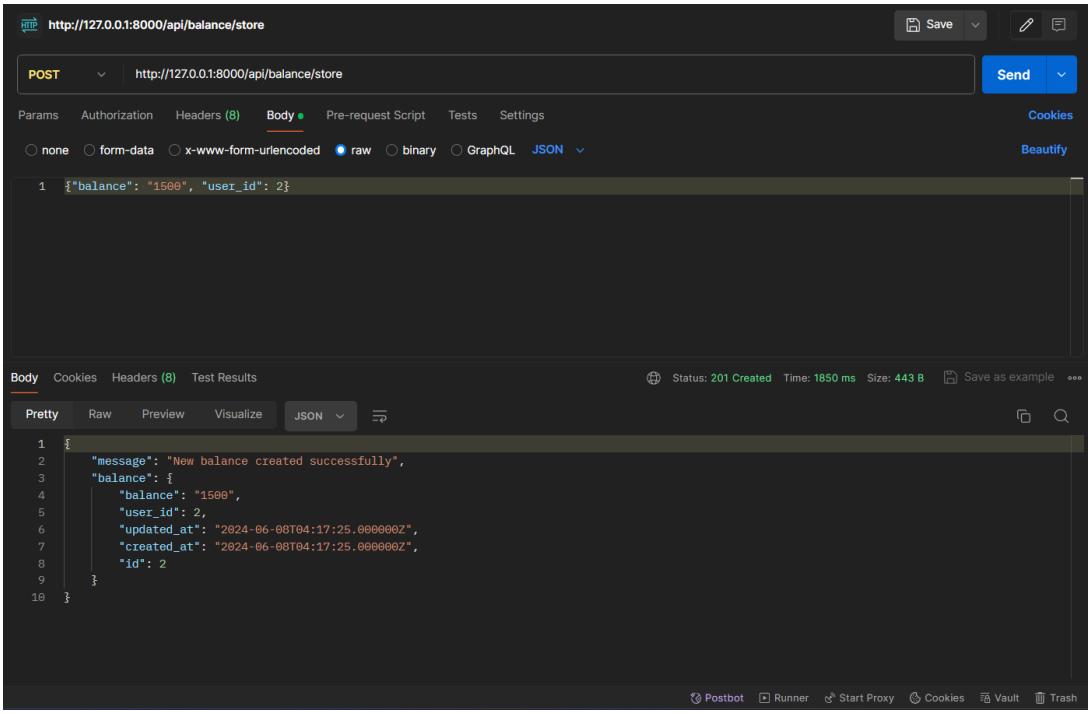
- Endpoint : **POST** ‘api/balance/store’
- Deskripsi : Menambahkan saldo baru atau memperbarui saldo yang ada untuk pengguna.
- Request Body:
    1. balance (string, required): Jumlah saldo yang akan ditambahkan atau diperbarui.
    2. user_id (int, required): ID pengguna yang saldonya akan ditambahkan atau diperbarui.
- Response
    - success :

```
{
    "message": "Balance updated successfully",
    "balance": {
        "id": 1,
        "user_id": 1,
        "balance": 2000
    }
}
{
    "message": "New balance created successfully",
    "balance": {
        "id": 2,
        "user_id": 2,
        "balance": 1500
    }
}
```

    - Error :

```
{
    "message": "Validation error",
    "errors": {
        "balance": ["The balance field is required."],
        "user_id": ["The user_id field is required."]
    }
}
```

Contoh Penggunaan :



Mengambil Saldo Berdasarkan User ID

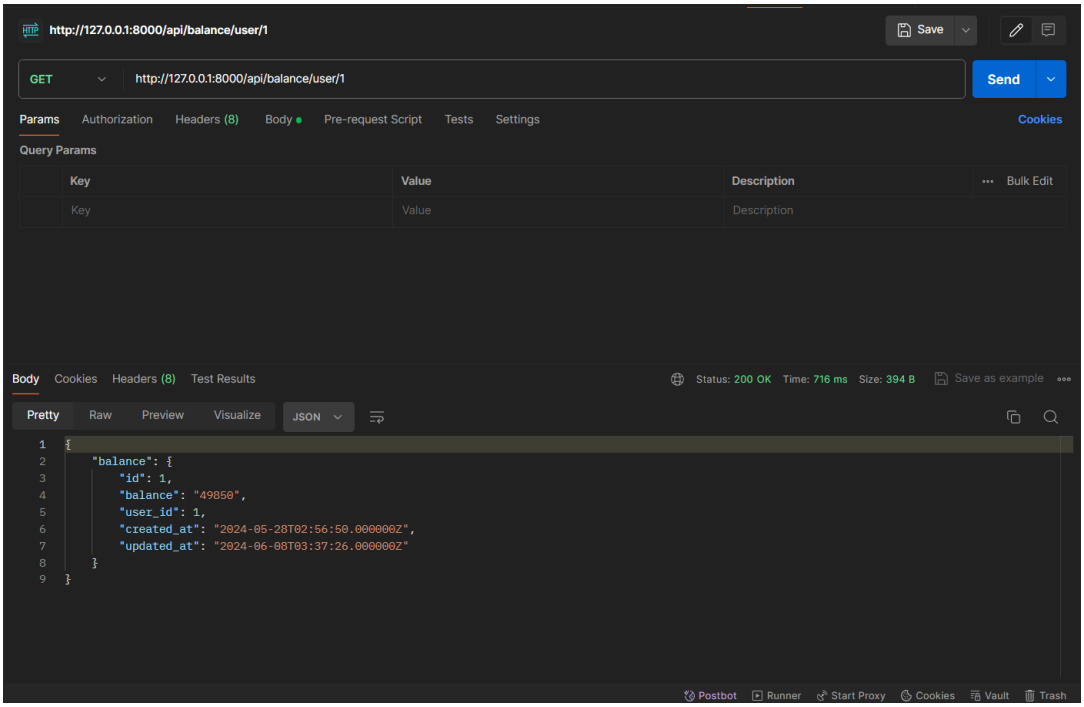
- Endpoint : **GET** ‘api/balance/user/{userId}’
- Deskripsi : Mengambil saldo berdasarkan ID pengguna.
- Request Parameter :
    1. **userId** (int, required): ID pengguna yang saldonya ingin diambil.
- Response
    - success :

```
{
    "balance": {
        "id": 1,
        "user_id": 1,
        "balance": 2000
    }
}
```

    - Error :

```
{
    "message": "Balance not found"
}
```

Contoh Penggunaan :

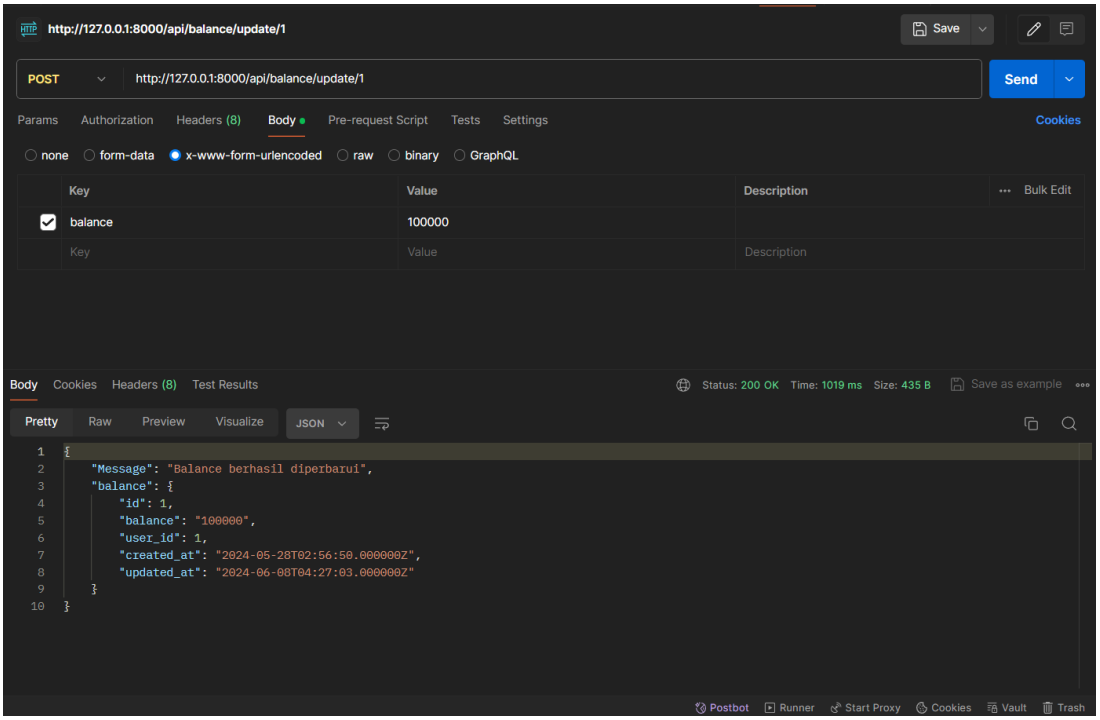


Memperbarui Saldo

- Endpoint : **POST** 'api/balance/update/{userId}'
- Deskripsi : Memperbarui saldo untuk pengguna.
- Request Parameter :
    1. **userId** (int, required): ID pengguna yang saldonya akan diperbarui.
- Respon Body :
    1. **balance** (string, required): Jumlah saldo yang baru.
- Response
    - success :

```
{
    "Message": "Balance berhasil diperbarui",
    "balance": {
        "id": 1,
        "user_id": 1,
        "balance": 2500
    }
}
```

Contoh Penggunaan :

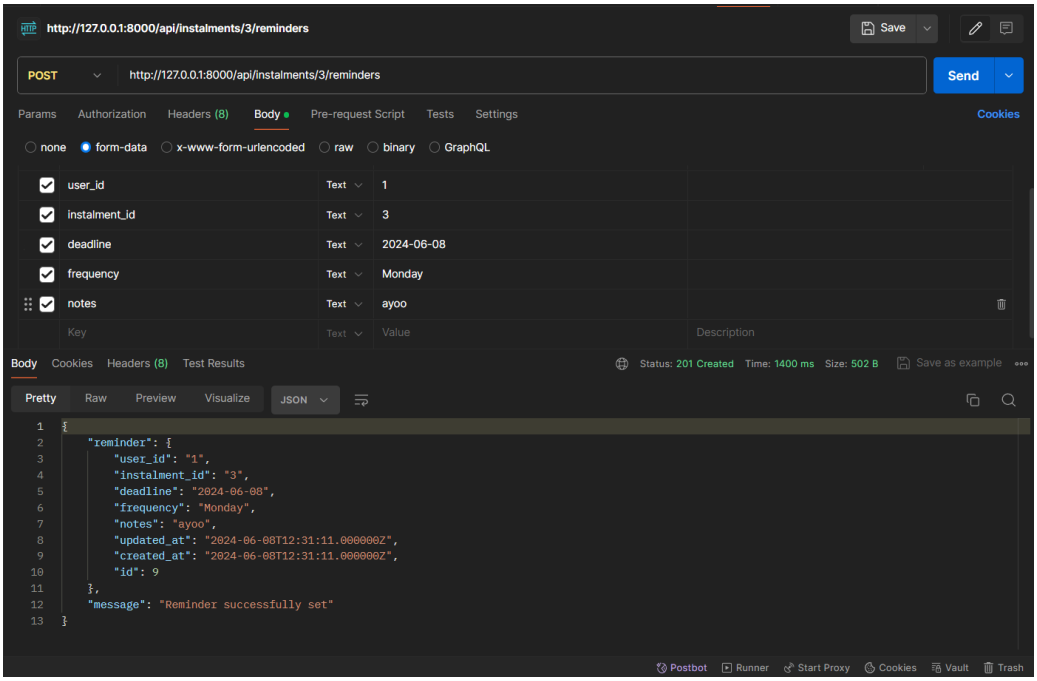


Menambahkan atau Memperbarui Saldo

- Endpoint : **POST** 'api/instalments/{id}/reminders'
- Deskripsi : Menambahkan reminders untuk instalments
- Request parameter :
    1. **Instalment_Id** (int, required): ID instalment yang ingin diambil.
- Request Body:
    1. user_id (int, required): ID pengguna yang akan dicantumkan pada table reminders.
- Response
    - success :

```
{
    "reminder": {
        "user_id": 1,
        "instalment_id": 1,
        "deadline": 8/8/2024,
       "frequency": Monday,
       "notes": -
    }
"message": "Reminder successfully set",
}
```

Contoh Penggunaan :


http://127.0.0.1:8000/api/instalments/3/reminders
Save
POST
http://127.0.0.1:8000/api/instalments/3/reminders
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
user_id
Text
1
instalment_id
Text
3
deadline
Text
2024-06-08
frequency
Text
Monday
notes
Text
ayoo
Key
Value
Description
Body
Cookies
Headers (8)
Test Results
Status: 201 Created
Time: 1400 ms
Size: 502 B
Save as example
Pretty
Raw
Preview
Visualize
JSON
{
"reminder": {
"user_id": "1",
"instalment_id": "3",
"deadline": "2024-06-08",
"frequency": "Monday",
"notes": "ayoo",
"updated_at": "2024-06-08T12:31:11.000000Z",
"created_at": "2024-06-08T12:31:11.000000Z",
"id": 9
},
"message": "Reminder successfully set"
}
Postbot
Runner
Start Proxy
Cookies
Vault
Trash