PDF EXPLANATION

Anggota Kelompok 06:

1) Kukuh Agus Hermawan (24/533395/PA/22573)
2) Aulia Fathus Tsani (24/534388/PA/22661)
3) Alifa Batrisyia Nariswari (24/535256/PA/22707)

CRUD Data Menggunakan Postman

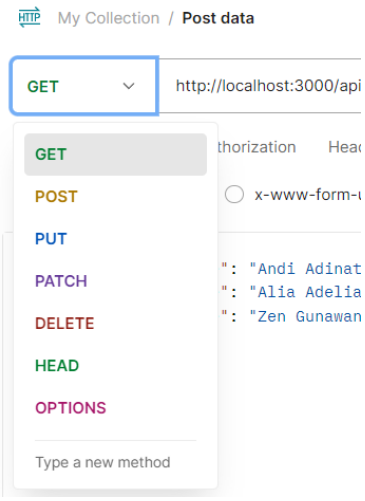
Pada tugas week 3 ini, seluruh proses pengelolaan data (CRUD: *Create, Read, Update, Delete*) diuji menggunakan Postman. Postman merupakan salah satu tools standar industri yang digunakan untuk mengirim request HTTP ke server, sehingga memudahkan proses verifikasi API sebelum membangun antarmuka frontend. Penggunaan Postman pada tahap ini bertujuan untuk memastikan bahwa backend telah bekerja dengan benar, termasuk validasi input, manipulasi data pada database, serta pengembalian respons sesuai standar API.

Cara Menggunakan Postman untuk Mengakses API InfiniRead

Berikut langkah-langkah umum untuk setiap operasi CRUD:

1. **Memilih Metode HTTP**

   Pada sisi kiri kolom URL, pilih method sesuai operasi:



Operasi	Method
Create	**POST**
Read	**GET**
Update	**PUT / PATCH**
Delete	**DELETE**

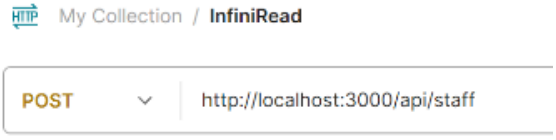
2. **Mengisi URL Endpoint**

   Setiap request diarahkan ke server backend di:

   http://localhost:3000/api/...

   Contoh:

   - Tambah staff:
   - POST http://localhost:3000/api/staff



3. **Menambahkan Body JSON (Untuk POST, PUT, PATCH, DELETE)**

   Masuk ke:

   Body → raw → JSON

   Lalu isi dengan JSON sesuai yang ditentukan di dokumentasi.

   Contoh POST (insert):

   {"Staff_Name": "Andi Adinata", "Position": "Librarian"}

4. **Mengirim Request**

   Klik tombol:

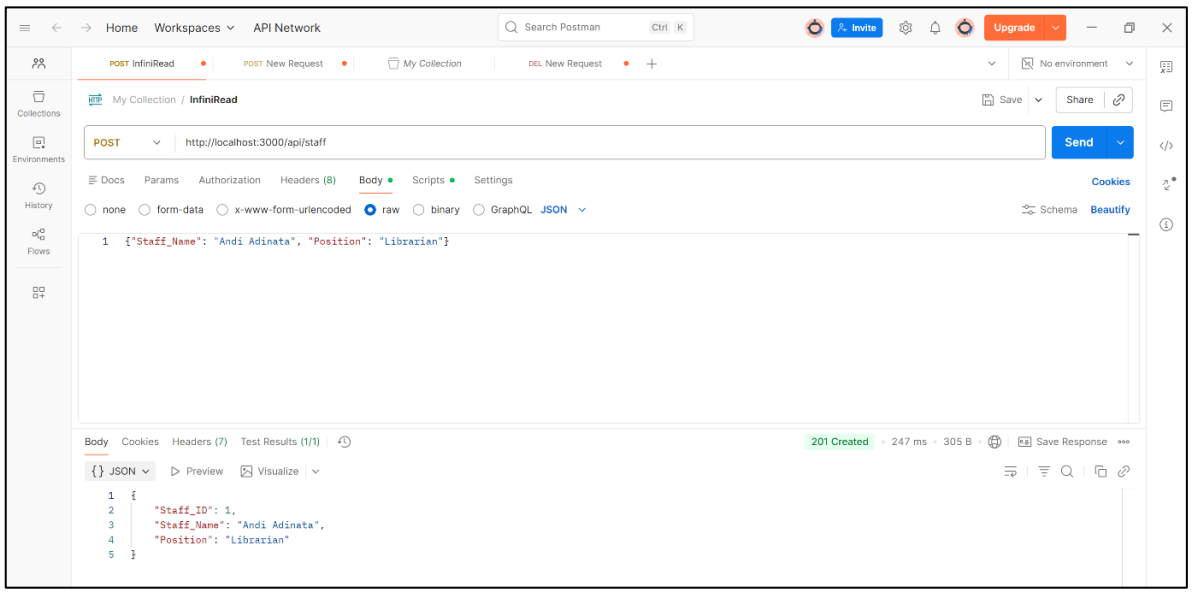
   Send

   Kemudian lihat hasil inspeksi pada bagian response:

   - status code (200, 201, 400, 404)

Code	**Arti**	**Makna**
200	OK	Request berhasil
201	Created	Data baru berhasil dibuat
400	Bad Request	Request salah / format keliru
404	Not Found	Data / endpoint tidak ditemukan

   - body JSON hasil server
   - pesan error jika ada



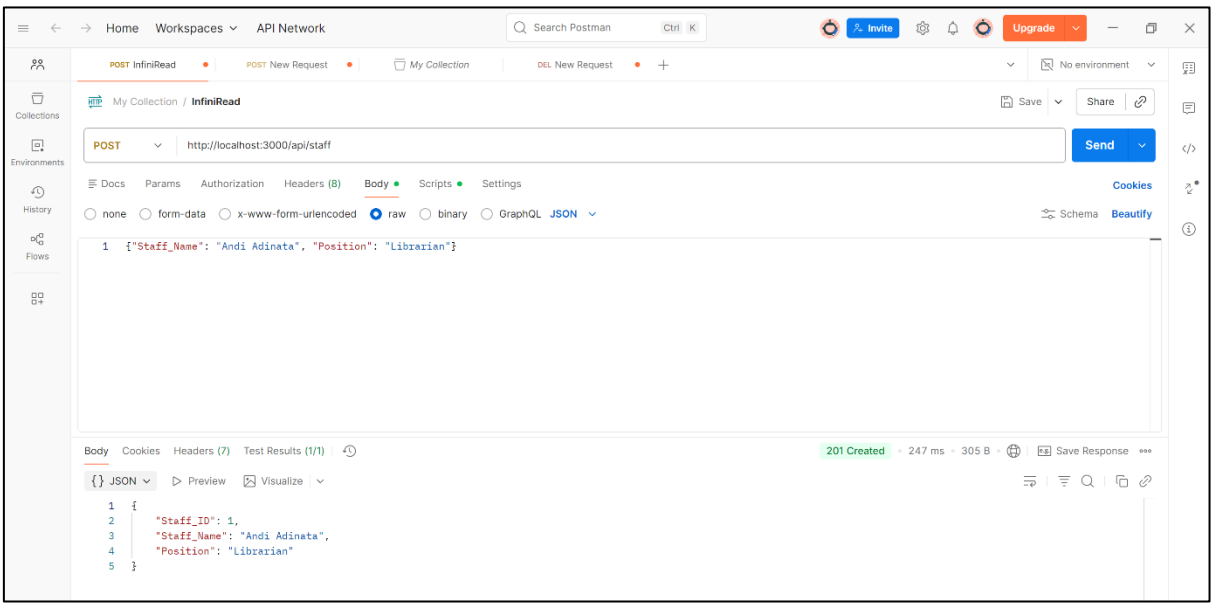
OPERASI CRUD PADA DATABASE

1. Create Data (POST)

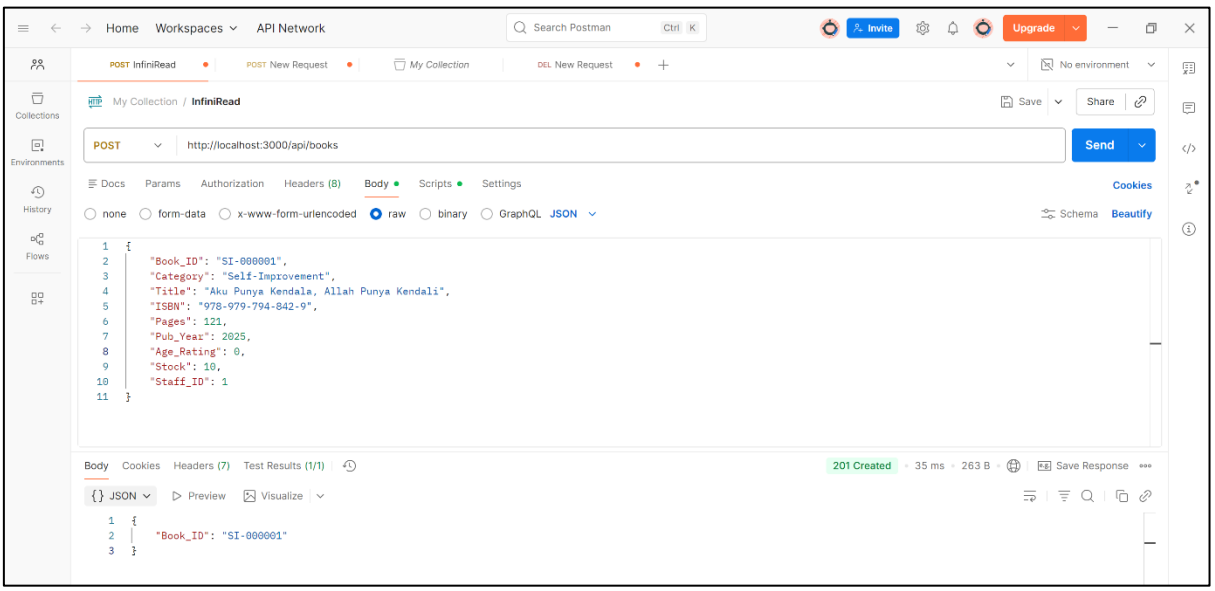
Untuk menambah data baru, digunakan metode POST. Setelah memilih POST dan memasukkan URL endpoint, pengguna membuka *Body* → *raw* → *JSON* lalu menuliskan data sesuai format, misalnya data member atau staff baru. Ketika dikirim, server akan menampilkan respons seperti *"Created"* jika input valid.

A. Contoh Request Single Insert

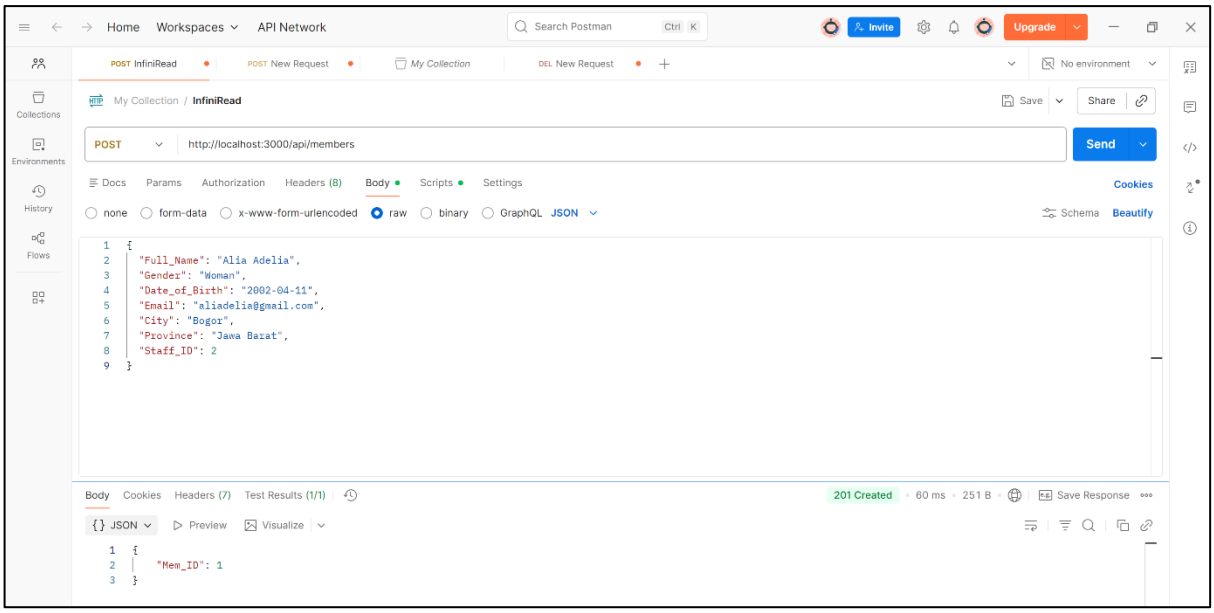
- **Staff**



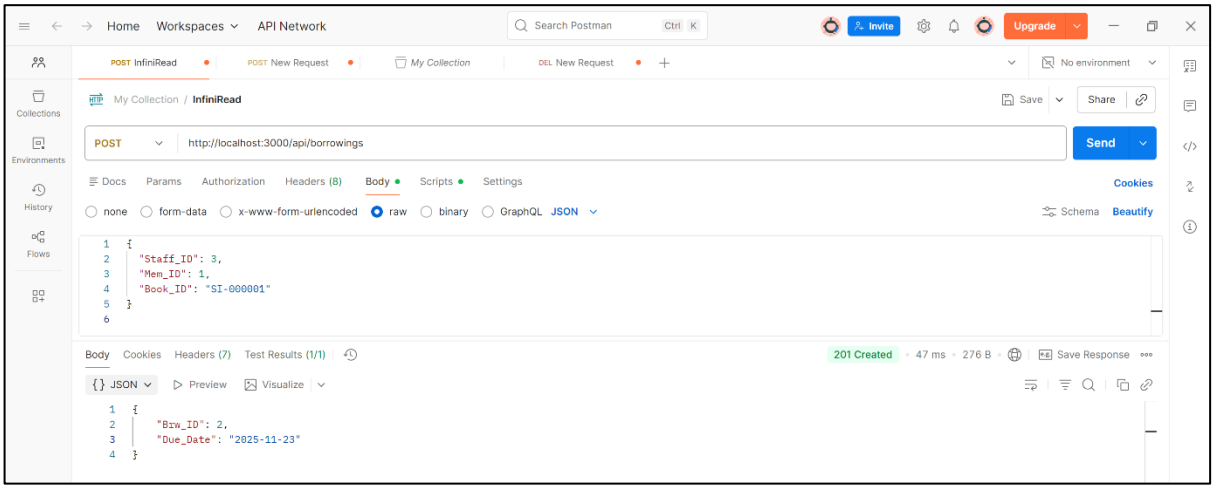
- **Books**



- **Members**



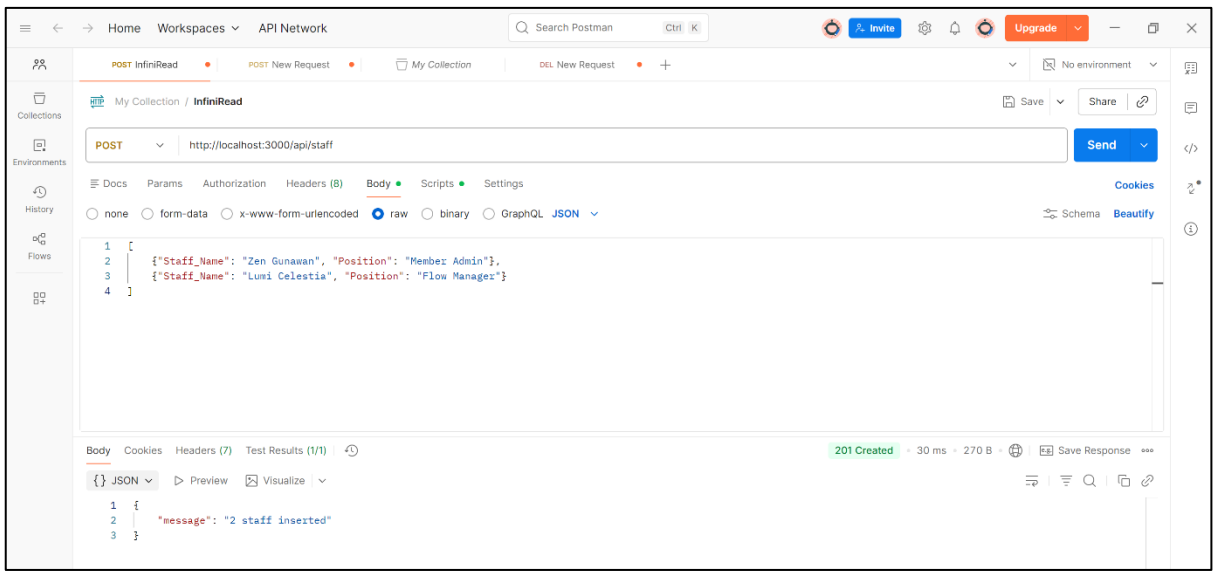
- **Borrowings**



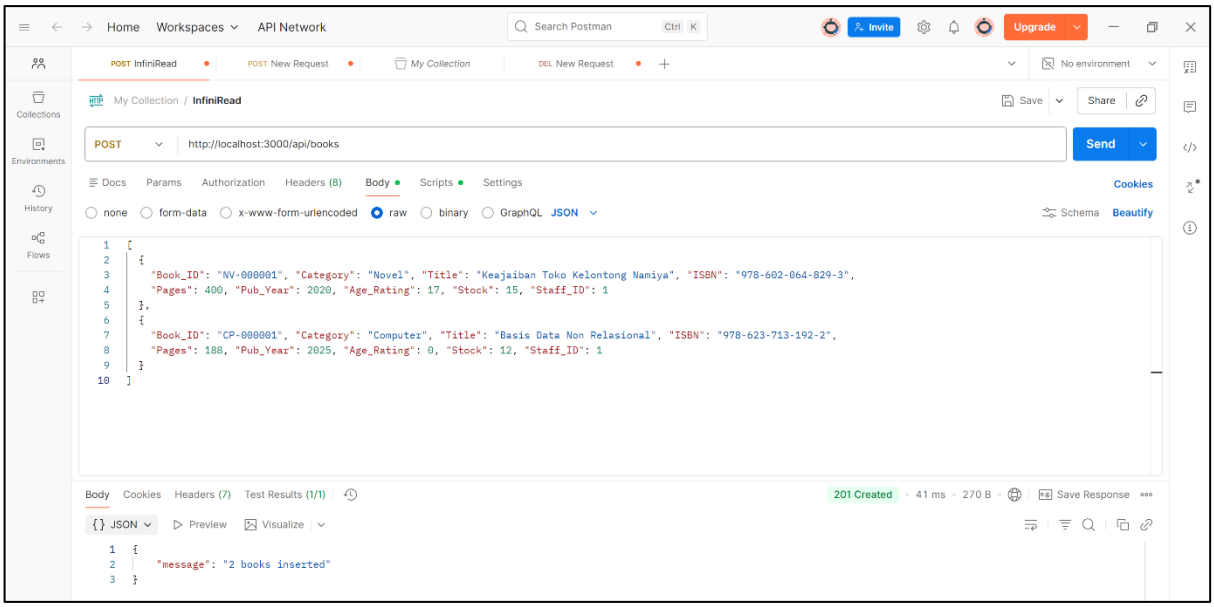
B. Contoh Request Multiple Insert

Sistem mendukung input banyak data dalam satu kali permintaan.

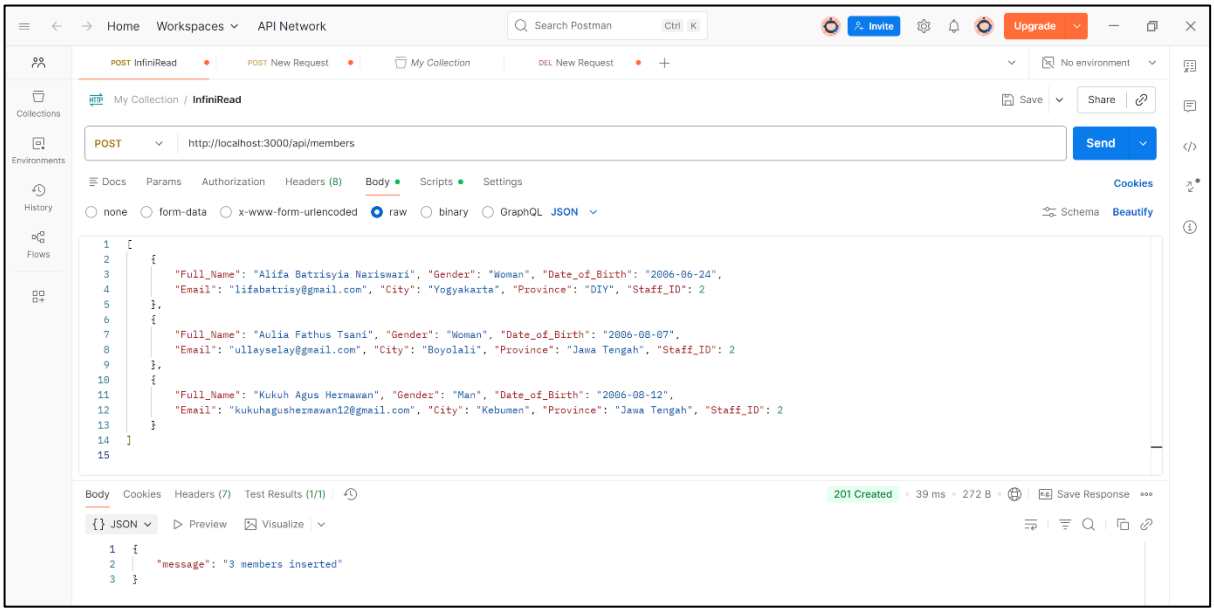
- **Staff**



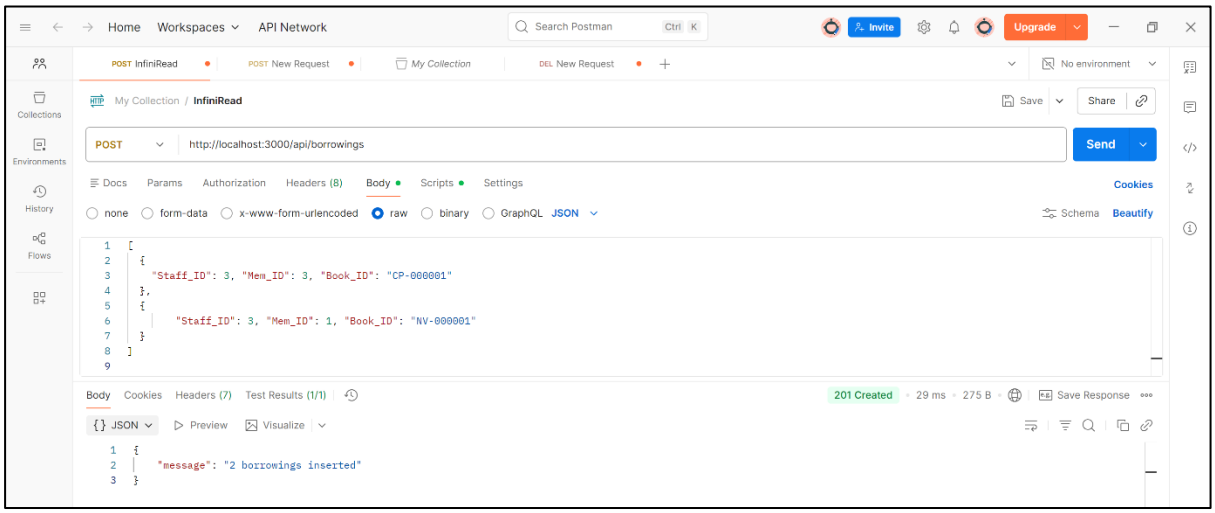
- **Books**



- **Members**

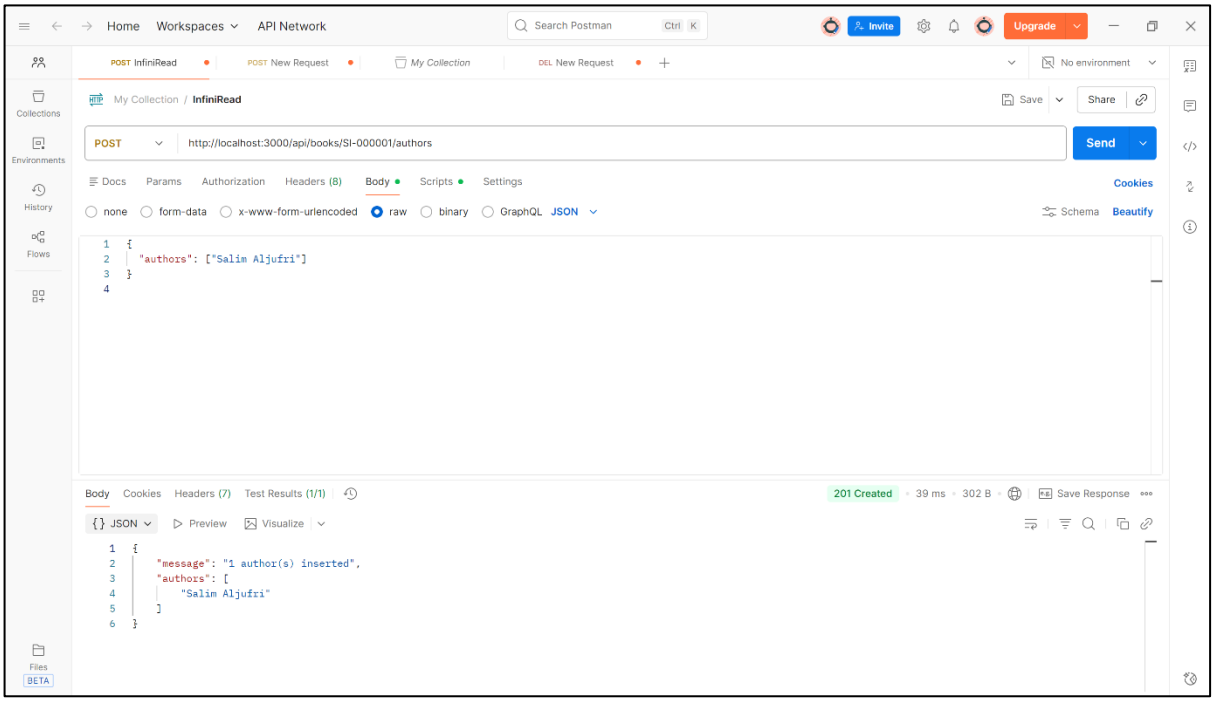


- **Borrowings**

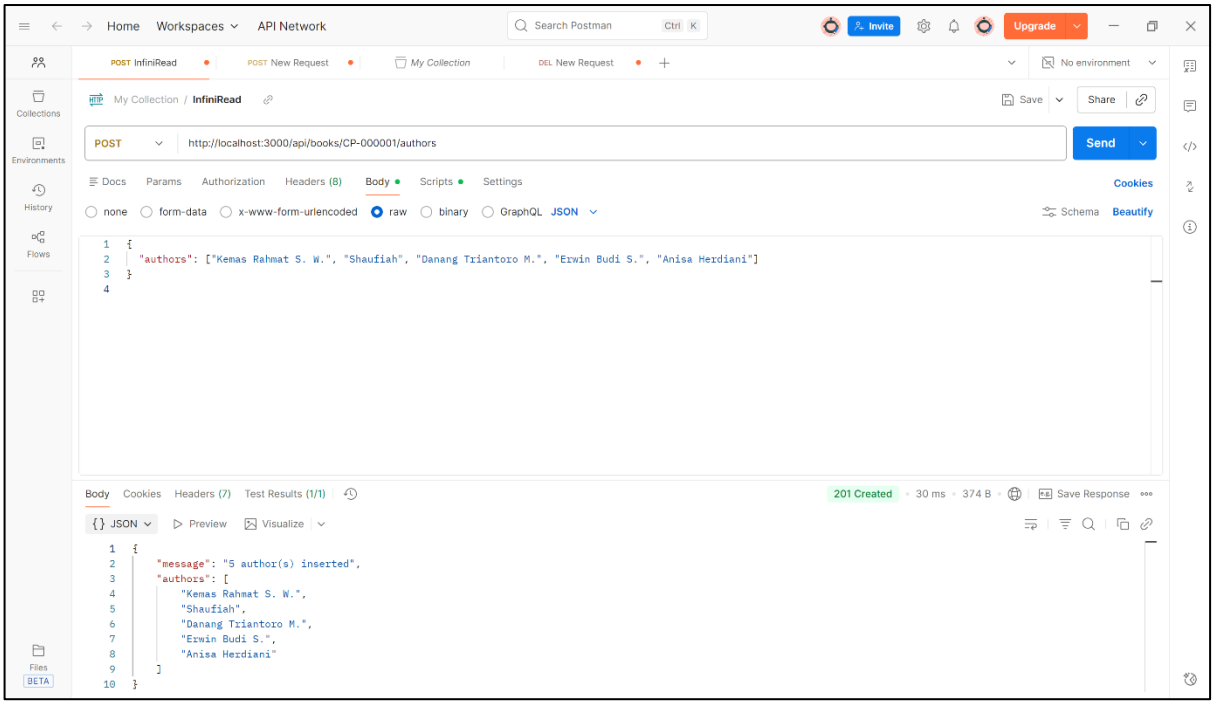


C. Request Insert Authors

- **Satu author**



- **Beberapa author**

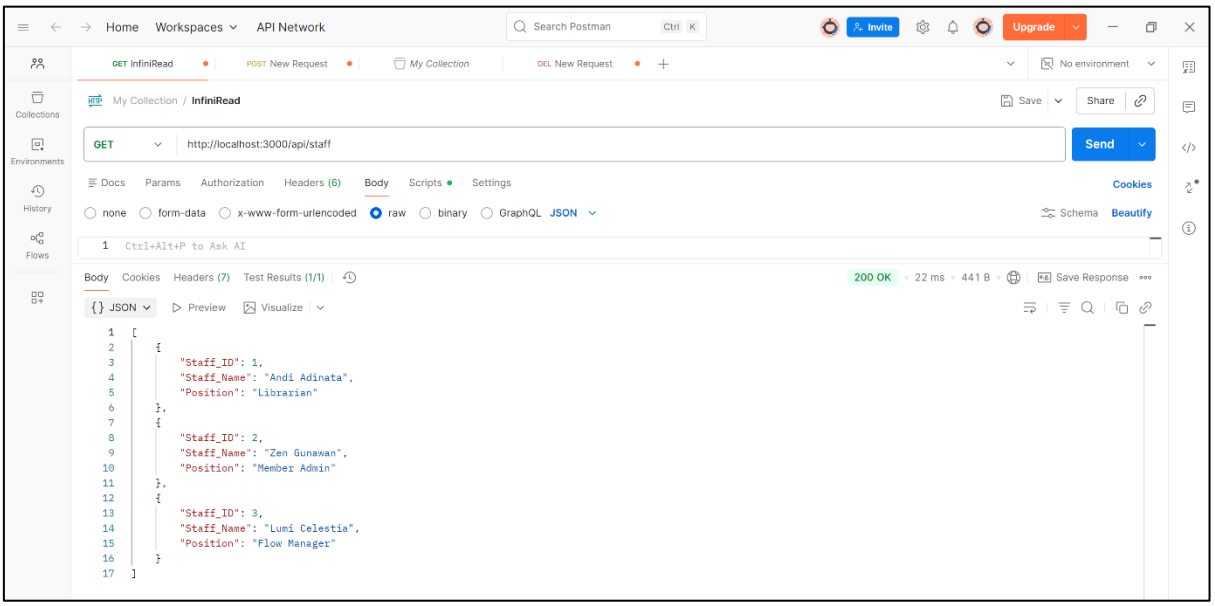


2. Read Data (GET)

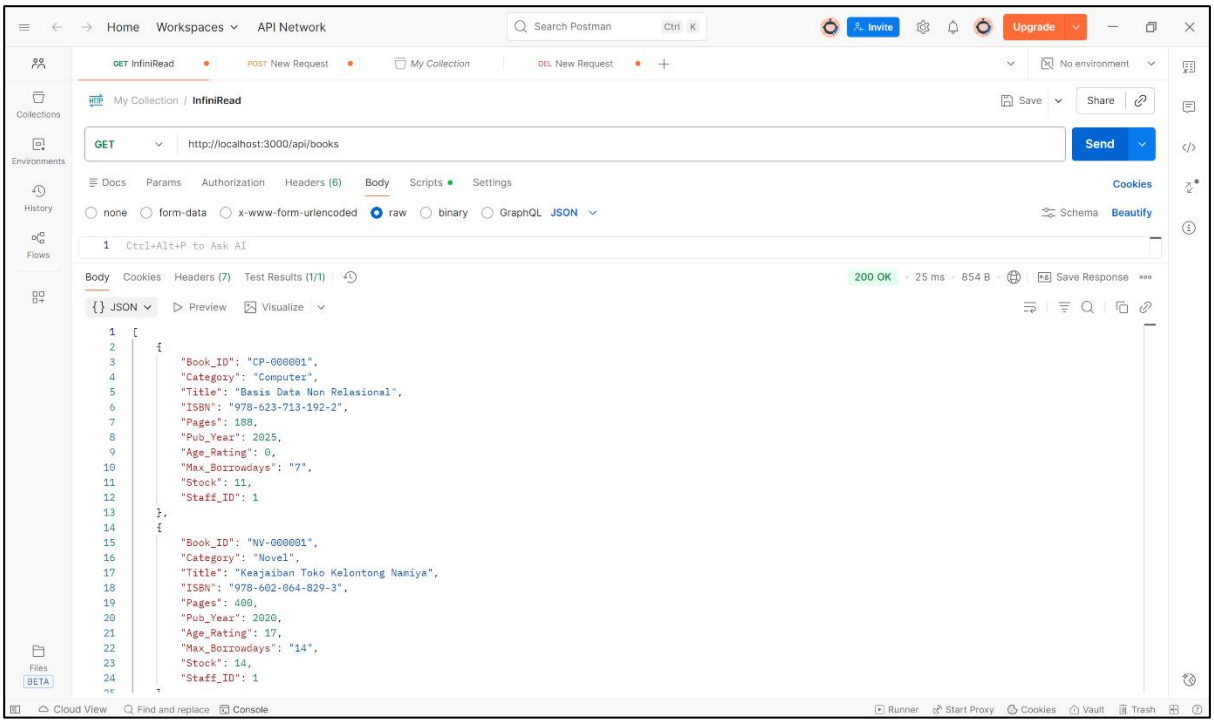
Operasi GET digunakan untuk menampilkan data dari server. Pada Postman, cukup memilih metode GET, memasukkan URL endpoint seperti http://localhost:3000/api/books, lalu menekan Send. Respons akan ditampilkan dalam bentuk JSON berisi seluruh data pada tabel tersebut.

A. Read All

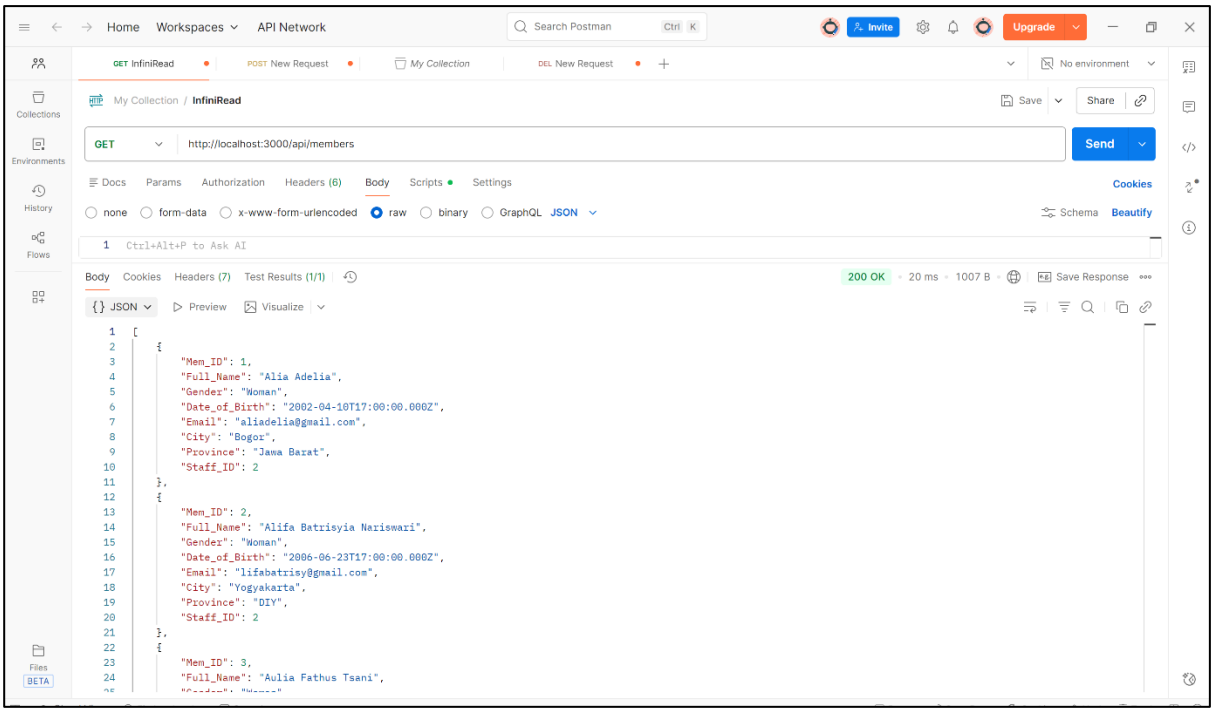
- **Staff**



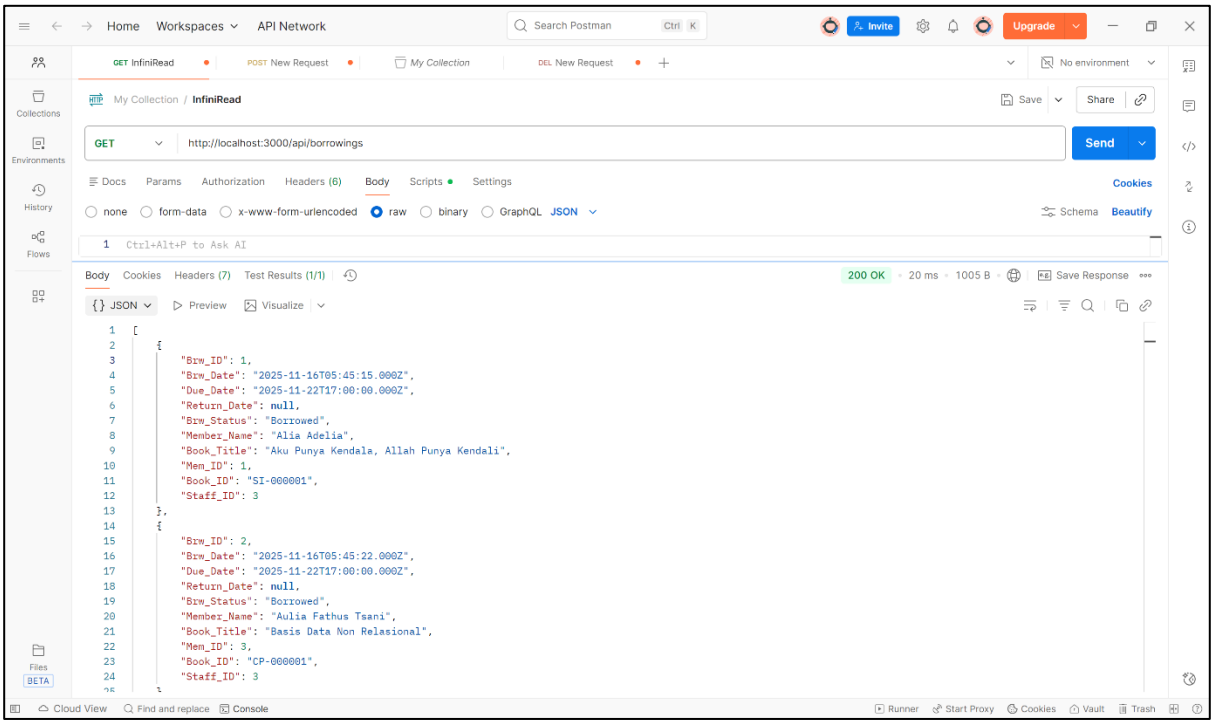
- **Books**



- **Members**



- **Borrowings**

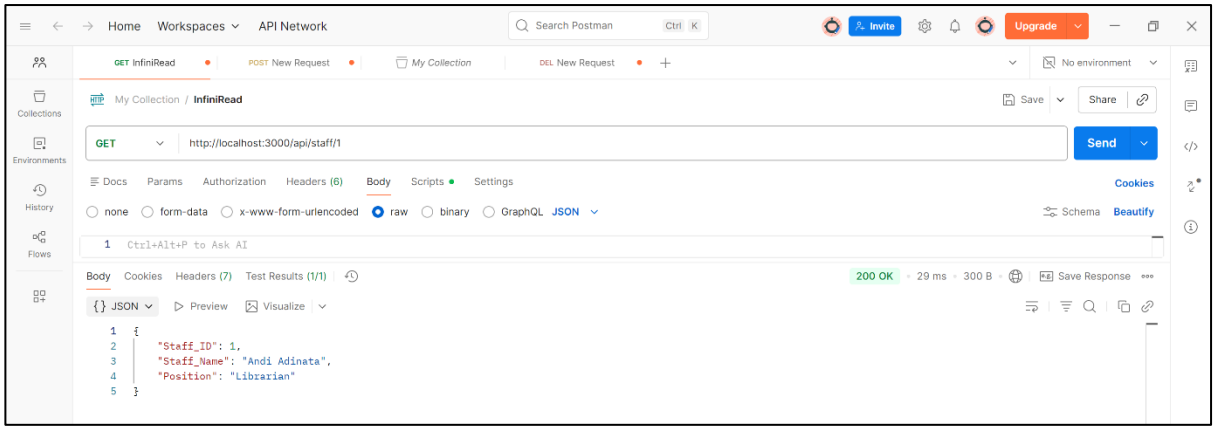


Format waktu seperti *2025-11-15T17:00:00.000Z* muncul karena ketika data bertipe DATE diambil dari MySQL, Node.js secara otomatis mengubahnya menjadi objek Date JavaScript dan menampilkannya dalam format standar internasional ISO 8601, yang selalu menyertakan jam dan zona waktu meskipun di database hanya tersimpan

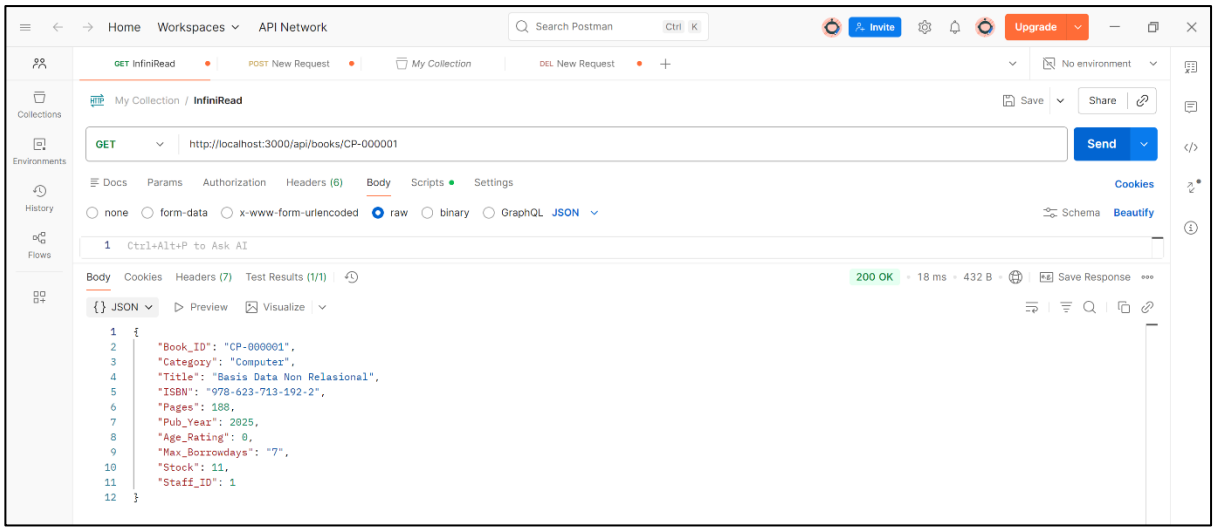
tanggalnya saja. Akibat konversi ini, data DATE yang sederhana berubah menjadi format lengkap dengan waktu bawaan (biasanya 00:00:00 diubah ke UTC).

B. Read by ID

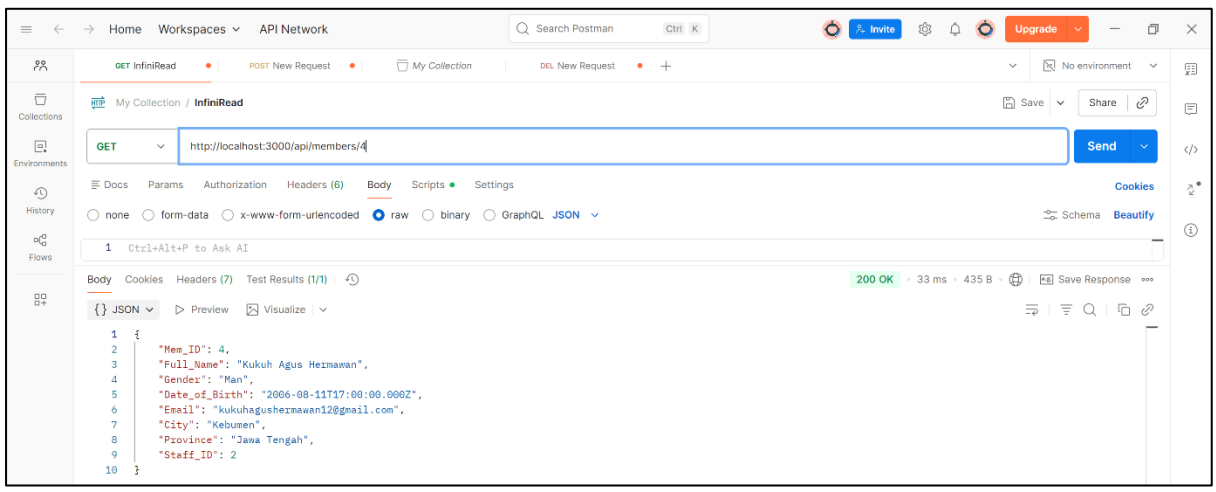
- **Staff**



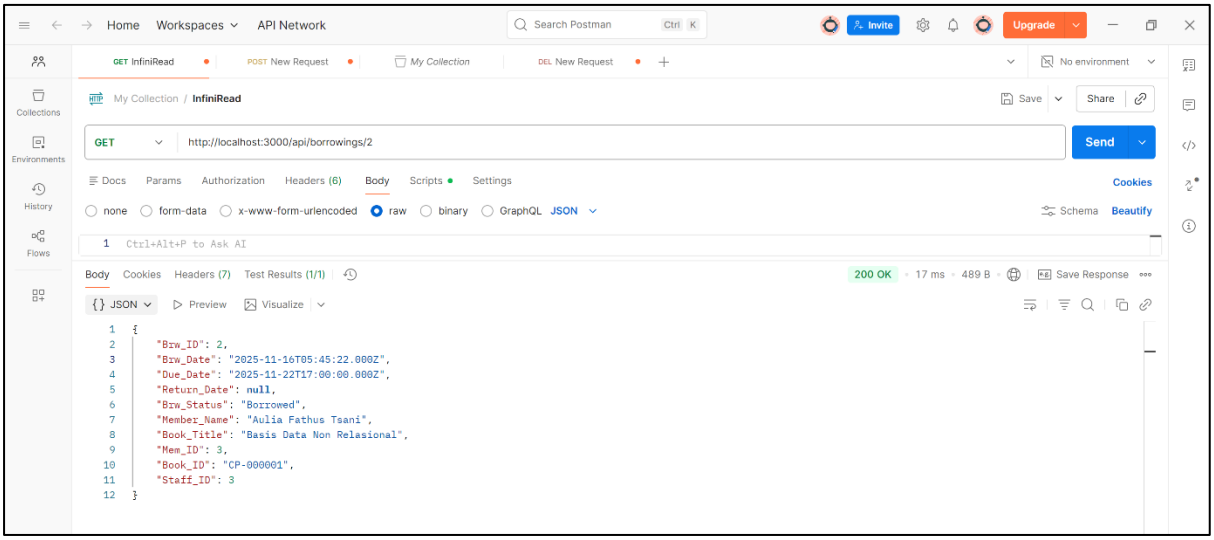
- **Books**



- **Members**



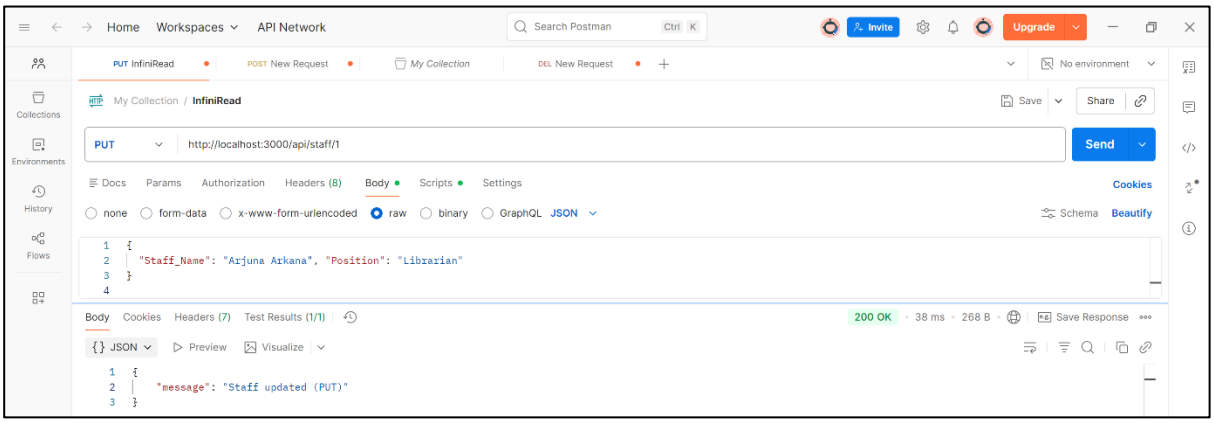
- **Borrowings**



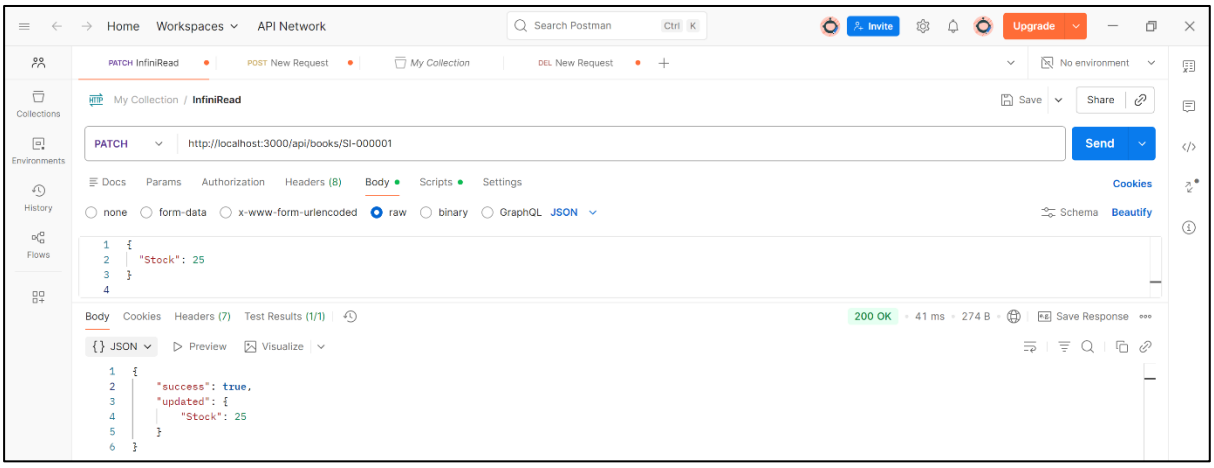
3. Update Data (PUT/PATCH)

Operasi Update digunakan untuk mengubah data yang sudah ada. (menggunakan PUT [harus input keseluruhan data] atau PATCH [cukup data tertentu yang mau di edit]). Pengguna menuliskan URL endpoint yang sama, misalnya /api/books, kemudian mengisi *Body → JSON* dengan data yang ingin diperbarui, termasuk ID sebagai acuan. Setelah menekan Send, Postman akan menampilkan respons apakah data berhasil diperbarui atau terjadi kesalahan input.

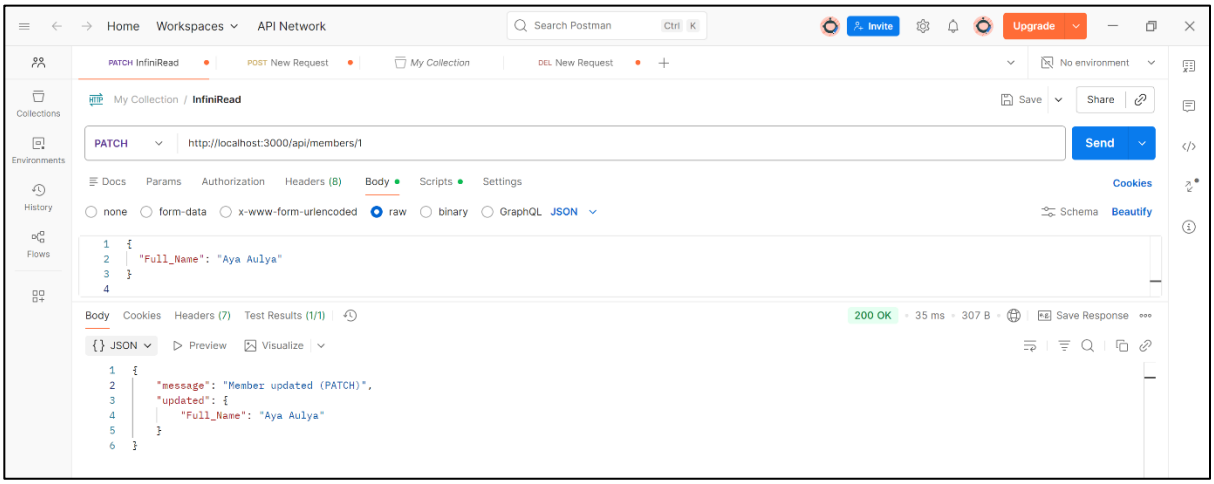
- **Staff**



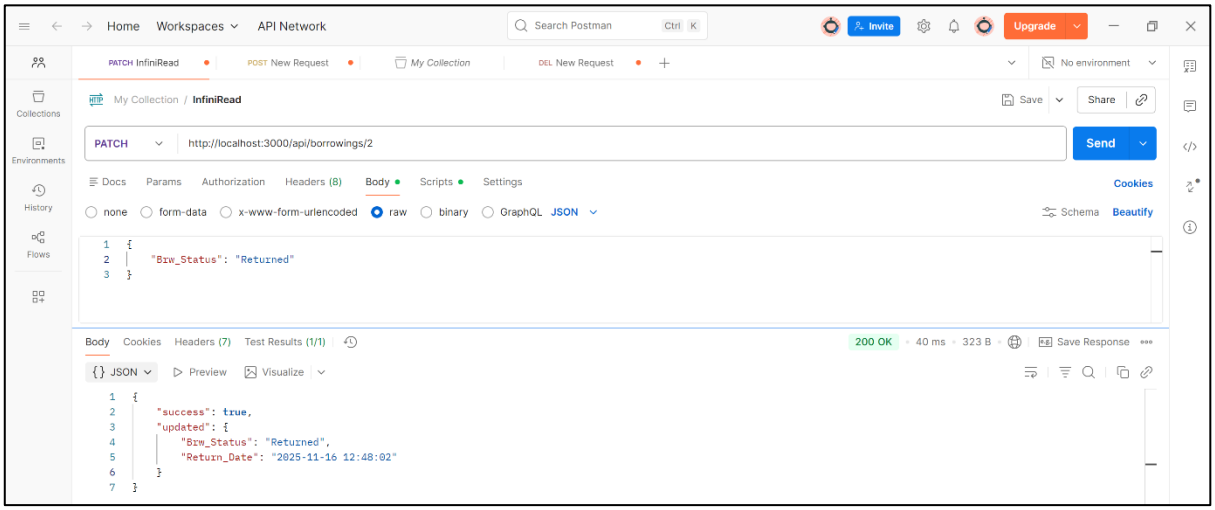
- **Books**



- **Members**

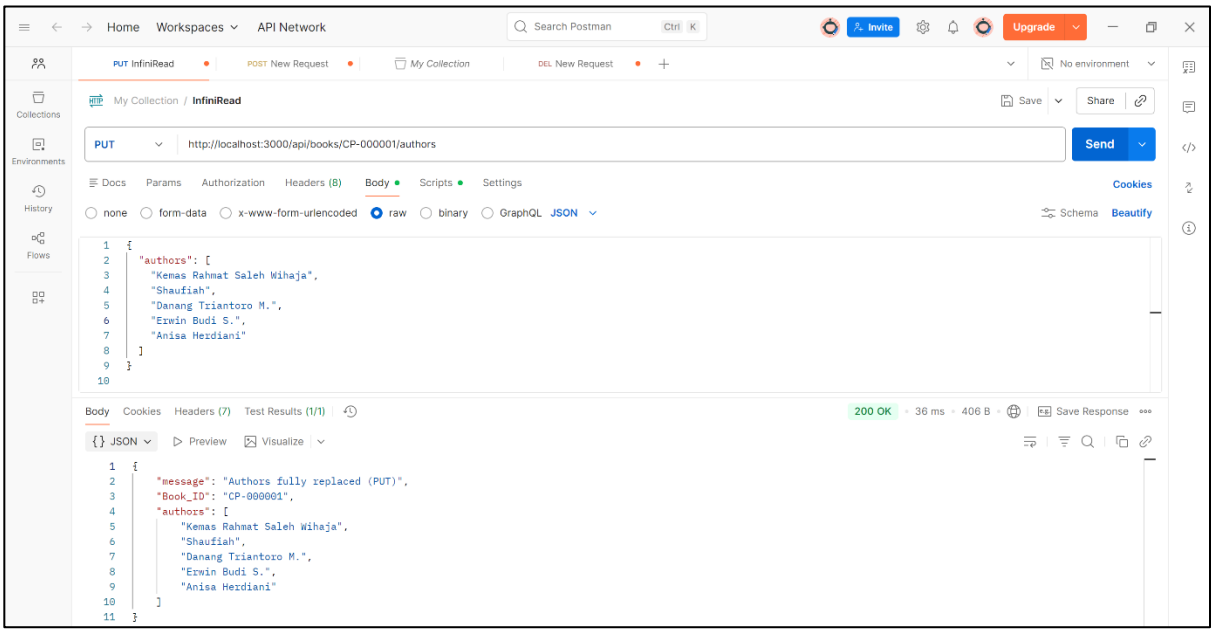


- **Borrowings**

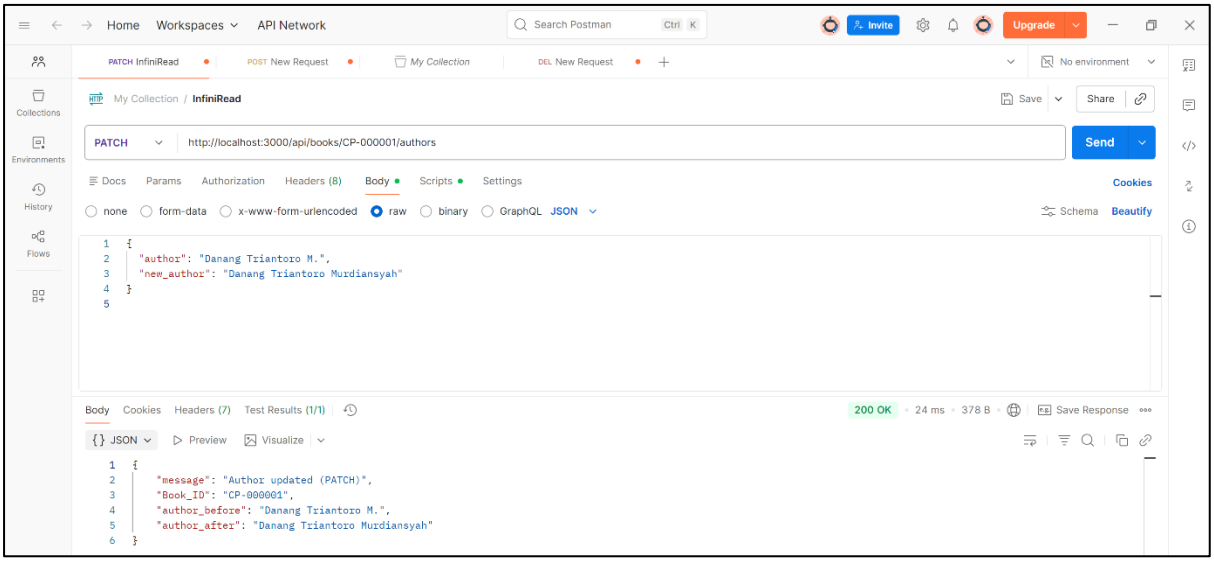


Ubah data author

- Put: langsung mengedit keseluruhan



- Patch: mengedit author tertentu satu persatu

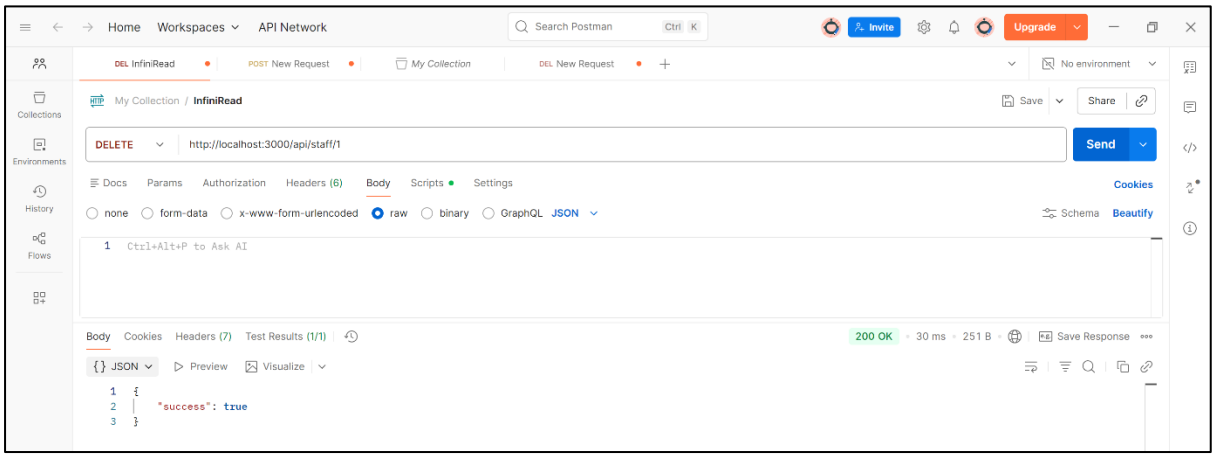


4. Delete Data (DELETE)

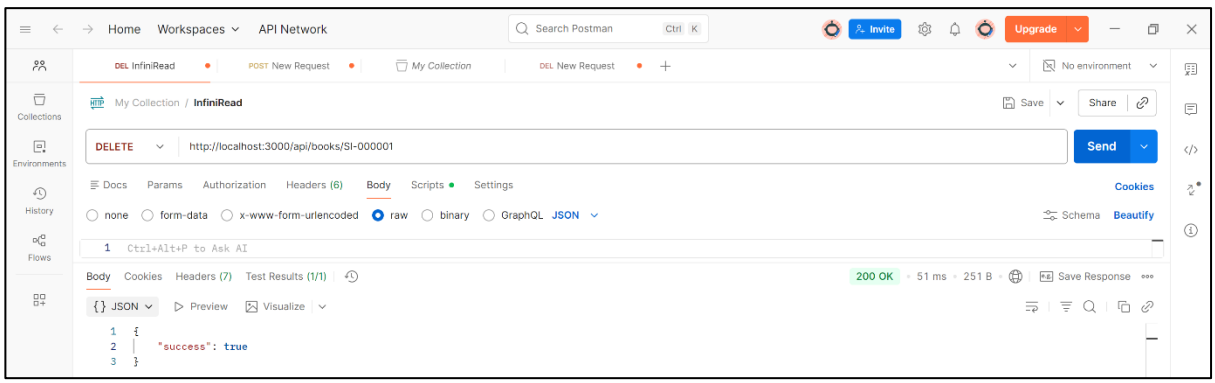
Operasi DELETE digunakan untuk menghapus data berdasarkan ID. Pada Postman, pilih metode DELETE, masukkan URL endpoint, lalu pada Body → raw → JSON isi format {"ids":[ID]} untuk menghapus satu atau beberapa data sekaligus. Setelah menekan Send, server akan menghapus ID yang valid dan menampilkan respons sukses, sementara ID yang tidak ditemukan atau format yang salah akan menghasilkan pesan error.

A. Delete by ID

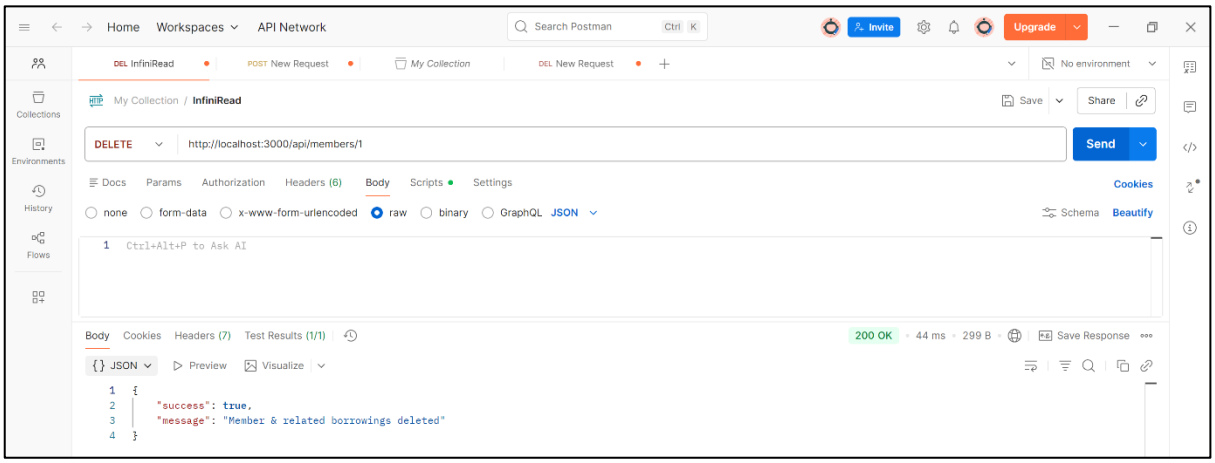
- **Staff**



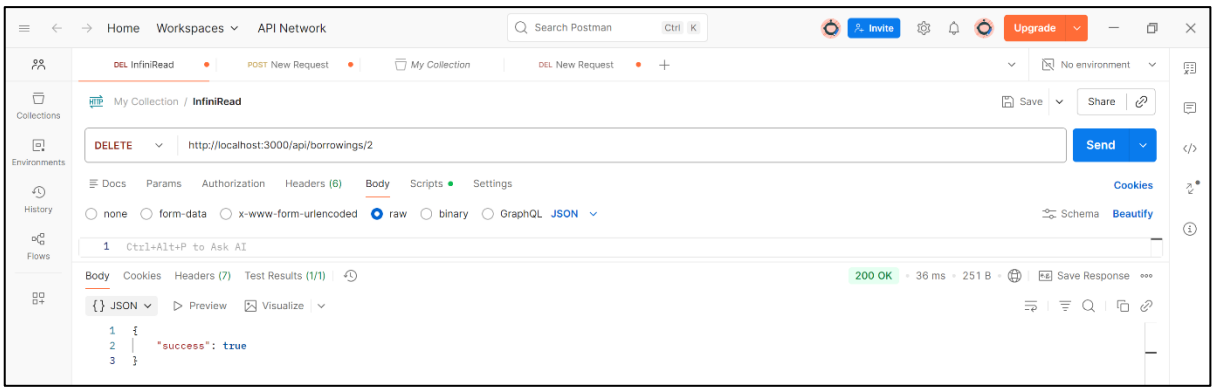
- **Books**



- **Members**



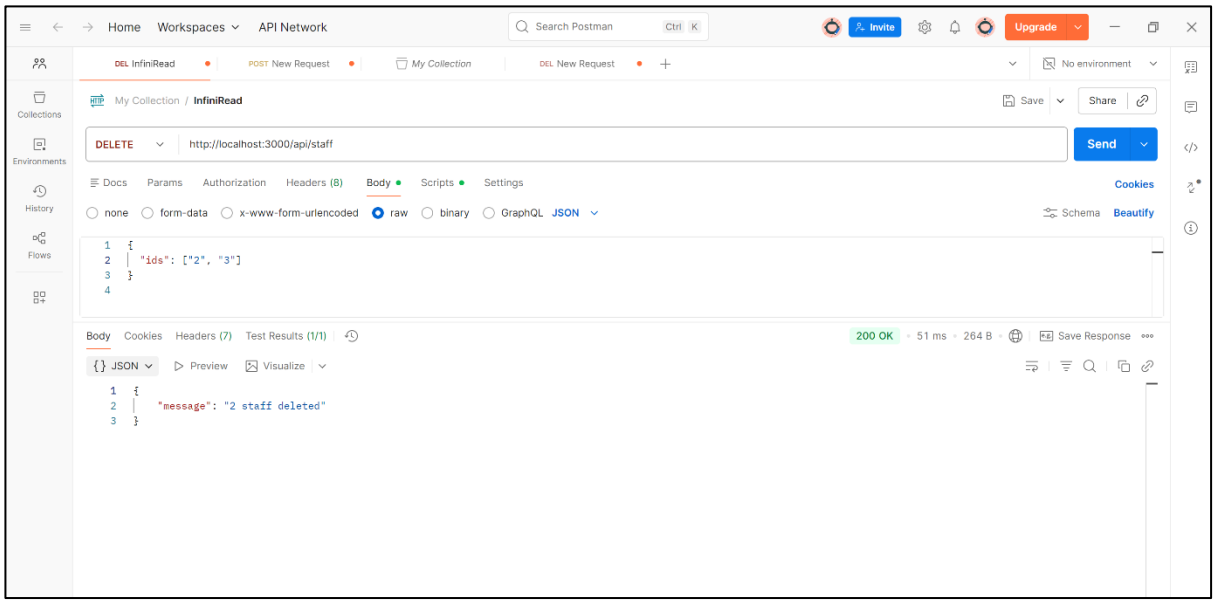
- **Borrowings**



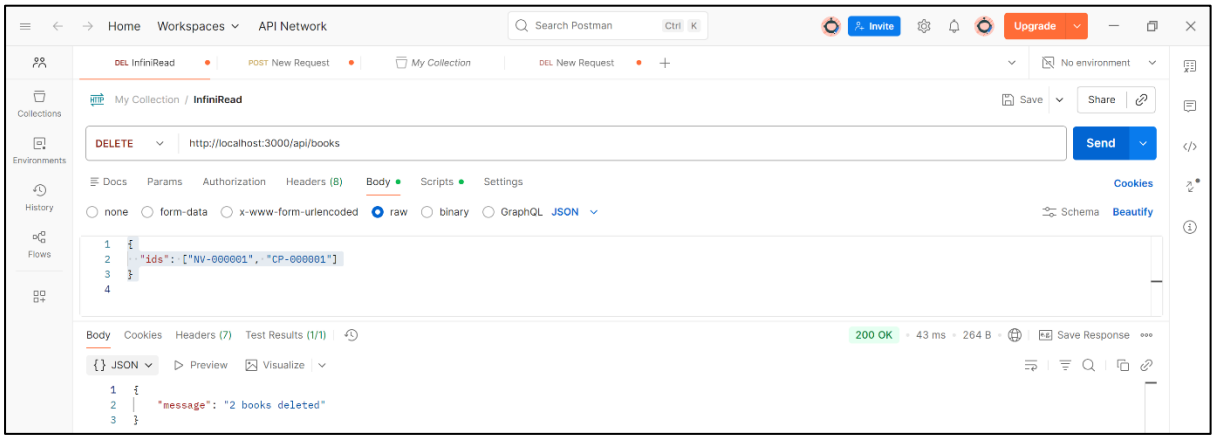
B. Delete Multiple Books (Bulk Delete)

Sistem juga mendukung penghapusan banyak data sekaligus.

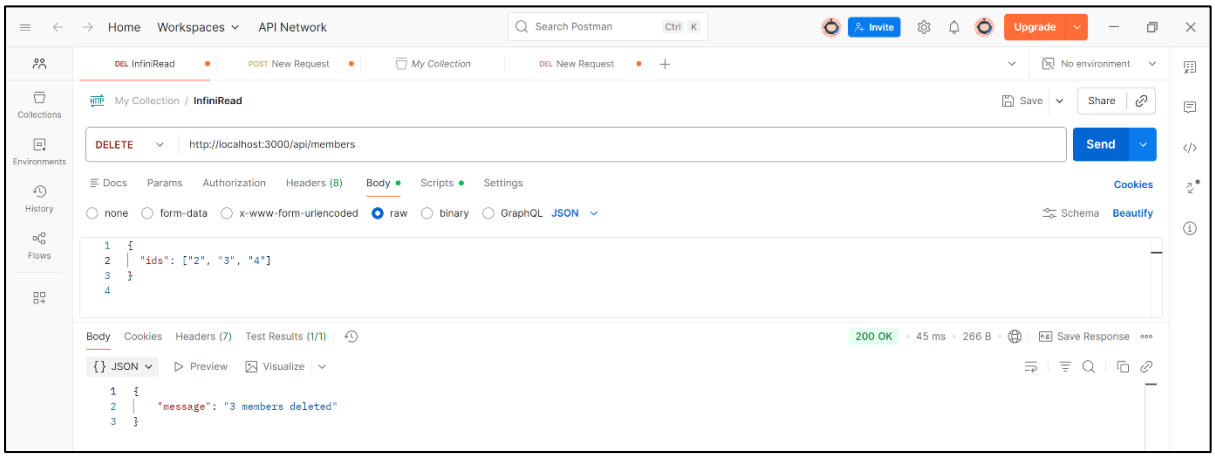
- **Staff**



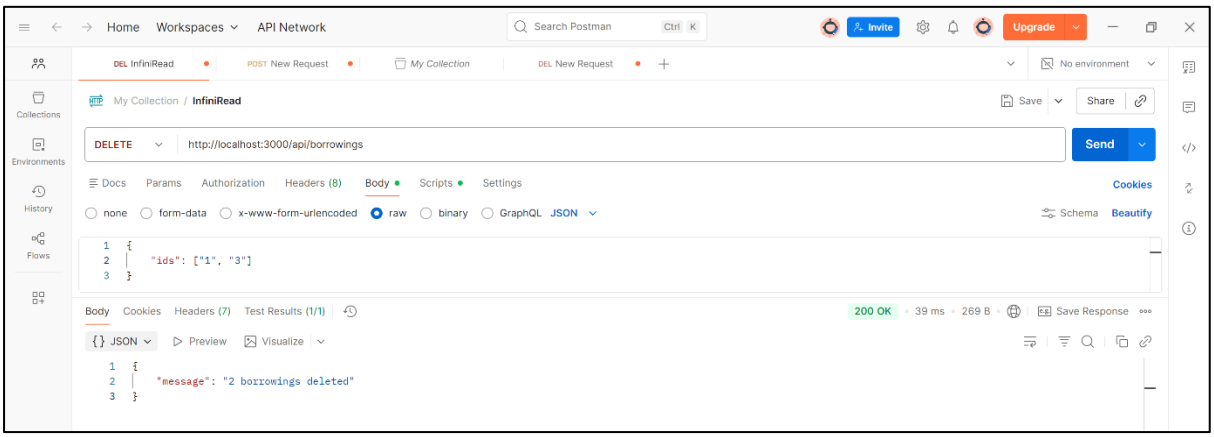
- **Books**



- **Members**



- **Borrowings**



Kesimpulan:

Pengujian fitur CRUD pada sistem *InfiniRead* melalui Postman menunjukkan bahwa seluruh operasi Create, Read, Update, dan Delete dapat dijalankan dengan baik pada semua tabel, termasuk Staff, Members, Books, Borrowings, dan Book_Author. Setiap request berhasil diproses oleh server, mulai dari penambahan data baru, pengambilan data, pembaruan data, hingga penghapusan data tunggal maupun batch. Seluruh respons API dikembalikan dalam

format JSON. Validasi input dan pesan kesalahan juga berjalan sesuai aturan sehingga menjaga integritas data. Secara keseluruhan, pengujian ini membuktikan bahwa backend InfiniRead telah berfungsi stabil dan siap untuk tahap integrasi dengan user interface.