Nama : Sabila Bintang Kusuma Dewi

NIM : 222313363

Kelas : 3SI2 No: 29

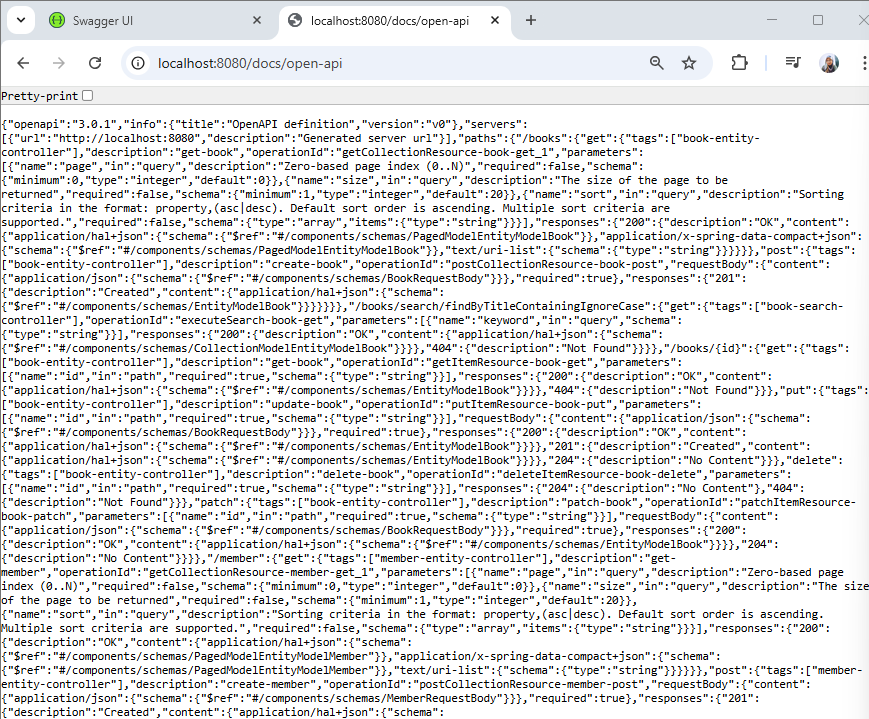
**Pemrograman Platform Khusus : Praktikum 7**

**Penugasan**

Lengkapi dokumentasi API untuk setiap endpoint pada proyek perpustakaan. Tambahkan deskripsi pada setiap endpoint, parameter, request body, responses dan bagian lainnya. Silakan pelajari lebih lanjut Springdoc pada halaman berikut https://springdoc.org/#Introduction. Selanjutnya, buat laporan pekerjaan Anda yang menampilkan halaman Swagger UI dari masing-masing endpoint dan anotasi pada kode programnya.

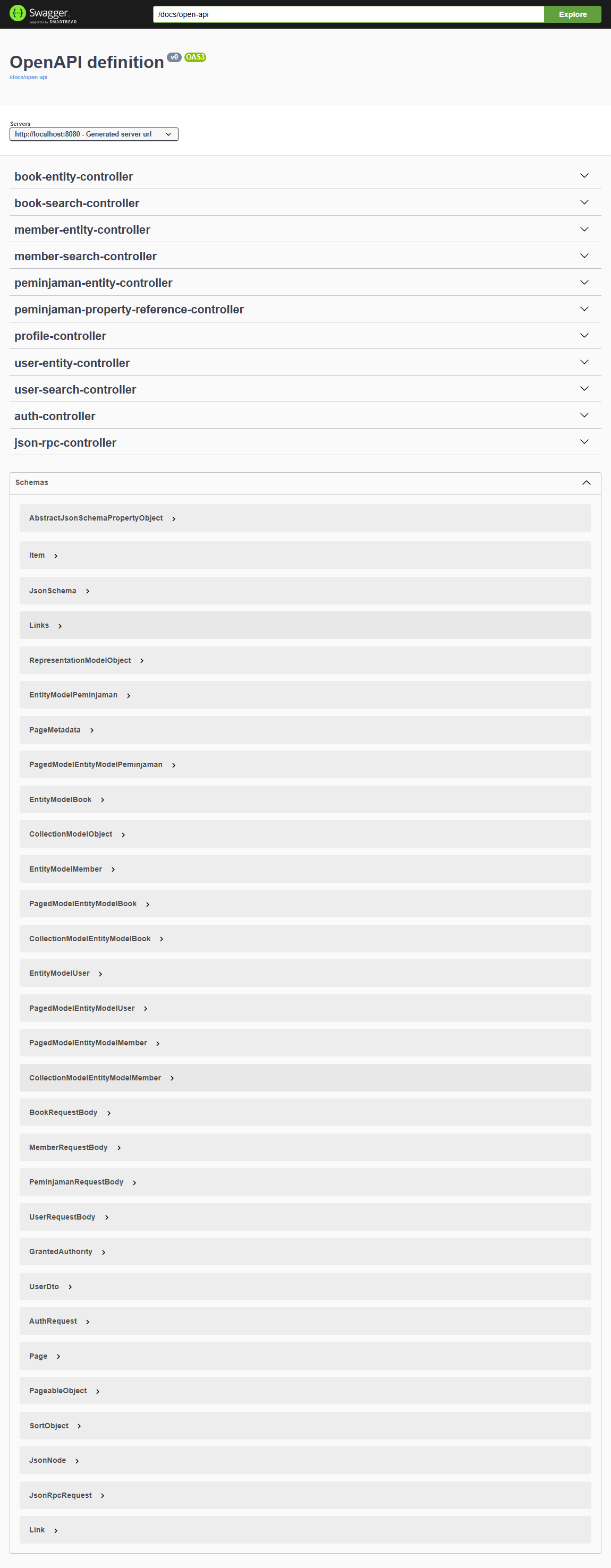
**Jawaban :**

Tampilan pada <http://localhost:8080/docs/open-api>



**LAPORAN PEKERJAAN**

Halaman utama pada swagger:



1. **Authentication (Endopoint untuk regidtrasi dan login)**
2. **POST /register**

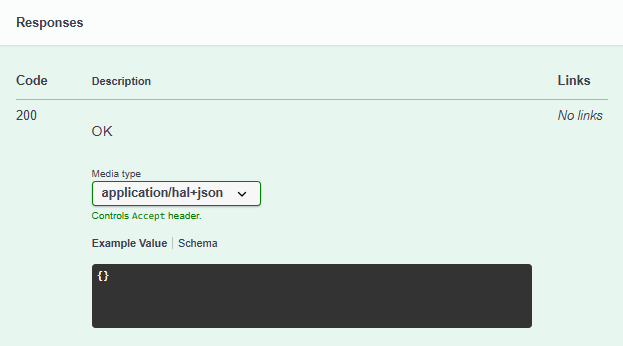
* Deskripsi: Menerima data pengguna (nama, email, password), kemudian menyimpan ke database setelah mengenkripsi password
* Request Body :
  + Name
  + Email
  + Password
* Responses
  + 200 OK: Registrasi berhasil, mengembalikan data pengguna yang telah dibuat
  + 400 Bad Request: Terjadi jika data yang dikirim tidak valid.

1. **POST/login**

* Deskripsi: Memvalidasi kredensial pengguna (email dan password). Jika valid, sistem akan memberikan acccessToken yang harus digunakan pada header Authorization untuk mengakses endpoint yang terproteksi.
* Request Body :
  + Email
  + Password

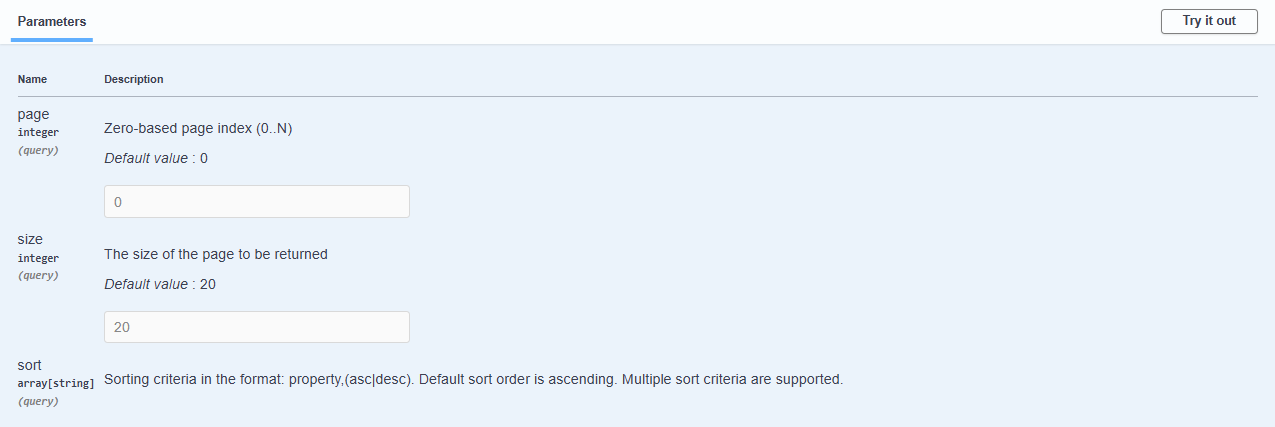


* Responses
  + 200 OK: Login berhasil, mengembalikan email dan accessToken.
  + 401 Unauthorized: Kredensial yang diberikan salah (email atau password tidak cocok).

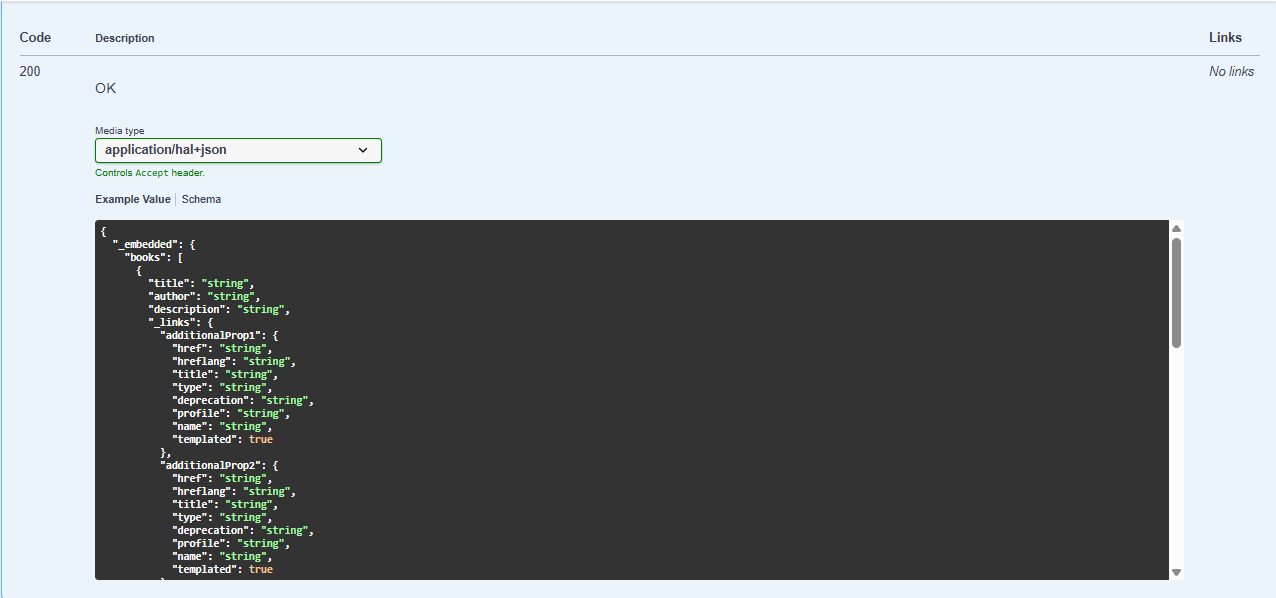
****

1. **Books (Endpoint unutk memanajemen buku)**
   1. **GET/books**

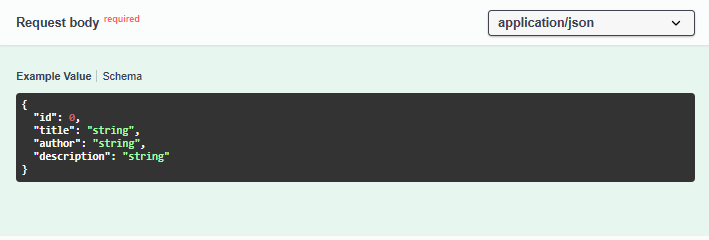
* Deskripsi: Menampilkan seluruh koleksi buku yang ada di perpustakaan..
* Parameters :
  + Page: Menampilkan jumlah halaman dalam suatu buku
  + Size: Menampilkan data yang ditampilkan per halaman (default: 20)
  + Sort: Kriteria pengurutan dalam format properti, ascrnding, atau descending. Mendukung beberapa kriteria pengurutan.



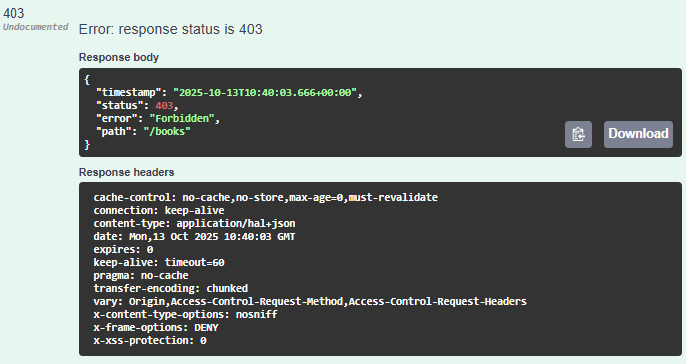
* Responses
  + 200 OK: Berhasil menampikan seluruh koleksi buku.



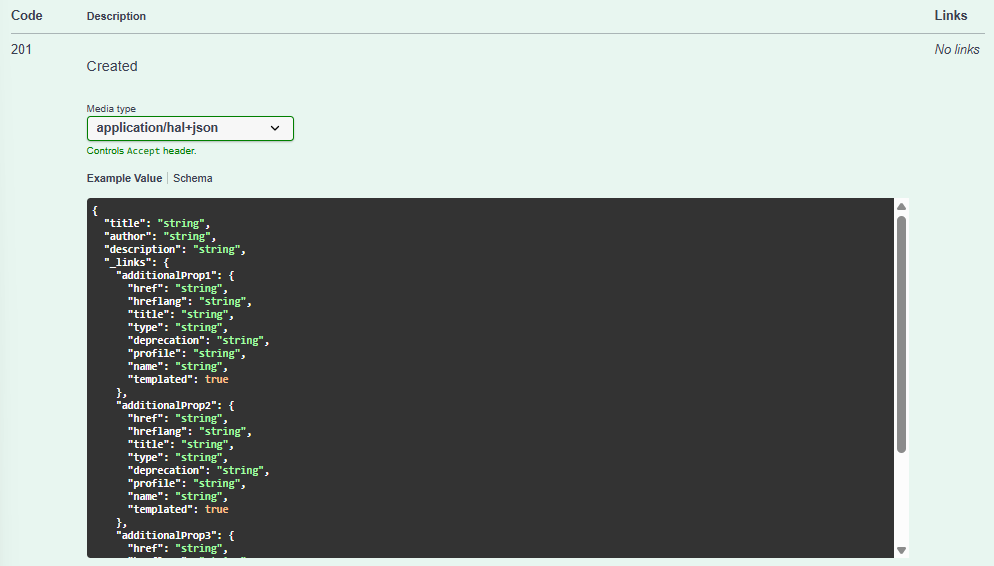
* 1. **POST/books**
* Deskripsi: Menambahkan buku kedalam database perpustakaan dengan atribut yang telah ditentukan, yaitu title, author, dan description
* Request Body :
  + title
  + author
  + description



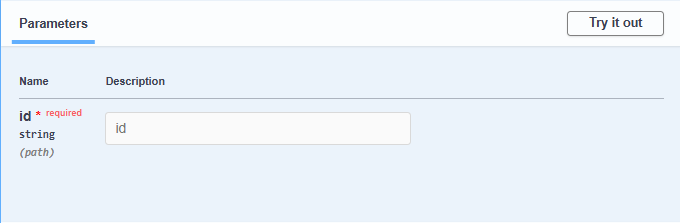
* Request URL: <http://localhost:8080/books>
* Responses
  + 403 Foebidden: Tidak bisa melakukan penambahan buku karena belum diberi akses (Belum login)



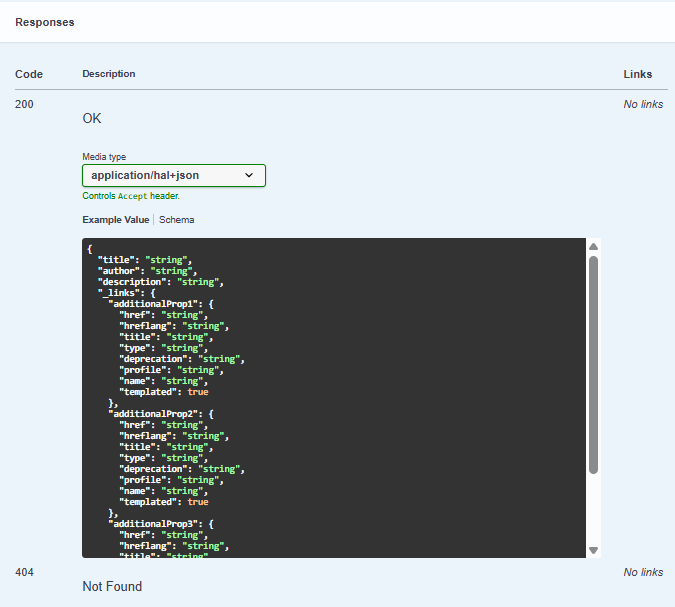
* + 201 Created: Buku telah berhasil dibuat atau ditambahkan ke database.



* 1. **GET/books/{id}**
* Deskripsi: Mengambil detail satu buku berdasarkan ID.
* Parameter: id



* Responses
  + 200 OK, mnegembalikan data buku dengan id yang dijadikan sebagai parameter
  + 404 Not Found, tidak dapat menemukan buku dengan id yang dijadikan parameter



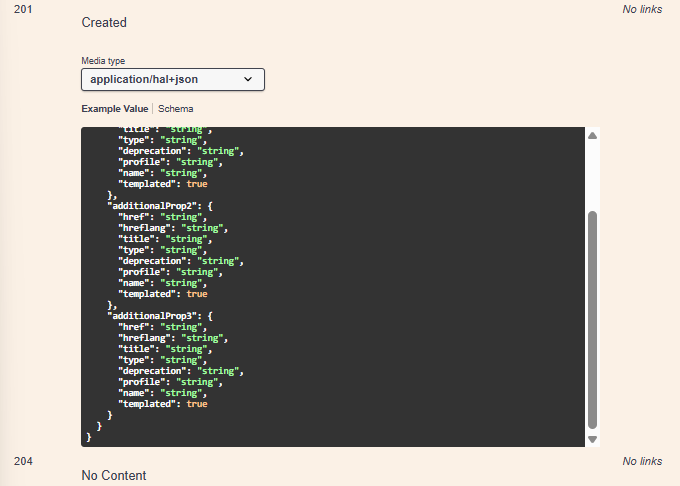
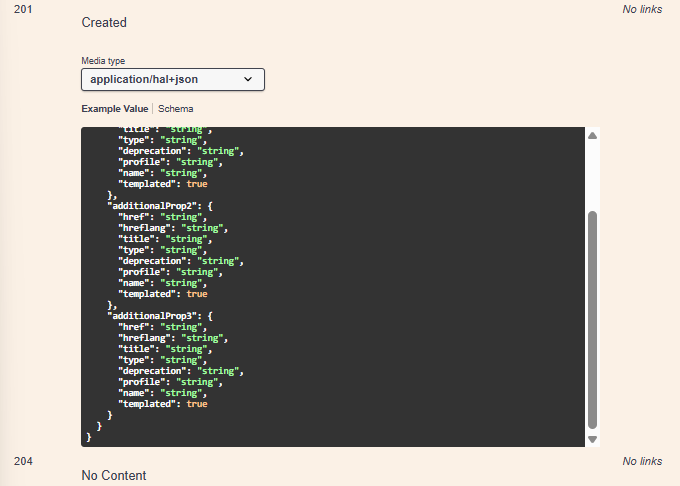
* 1. **PUT/books/{id}**
* Deskripsi: Memperbarui **seluruh** data buku. Semua field wajib diisi.
* Parameter: id



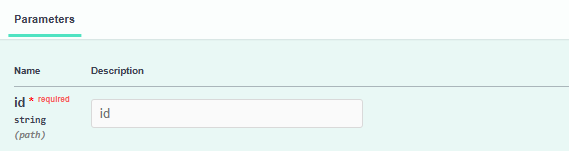
* Request Body: Book (title, author, description)



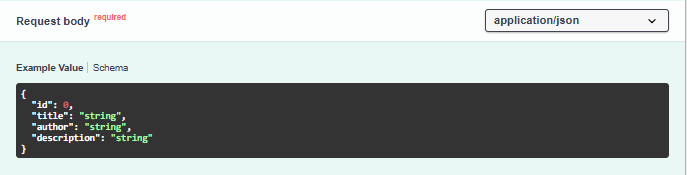
* Responses
  + 200 OK, mengembalikan data buku dengan id yang dijadikan sebagai parameter
  + 201 Created, mengembalikan data buku dengan id yang dijadikan sebagai parameter
  + 204, No content

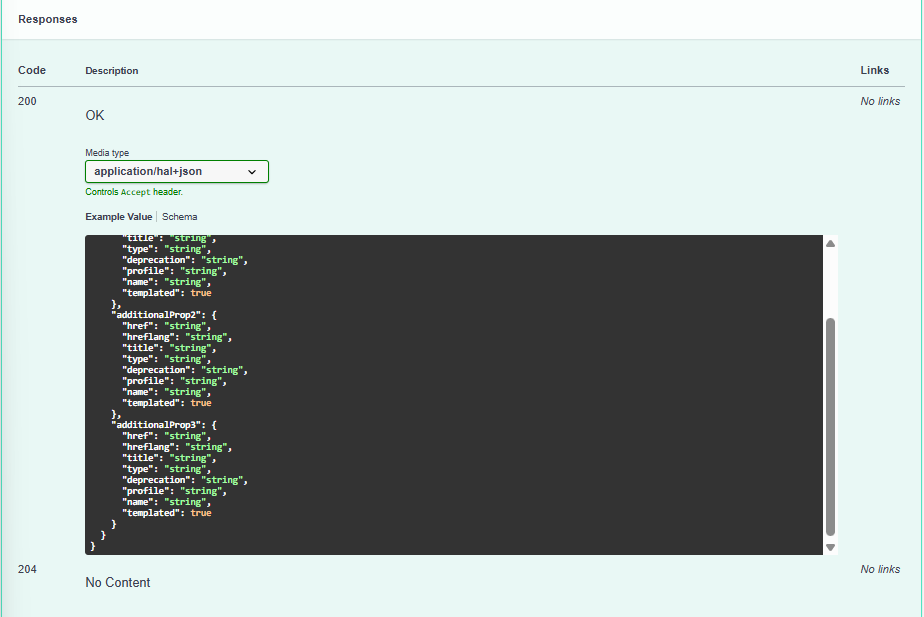
* 1. **PATCH/book/{id}**
* Deskripsi: Memperbarui **sebagian** data buku. Hanya field yang dikirim yang akan diubah.
* Parameter: id



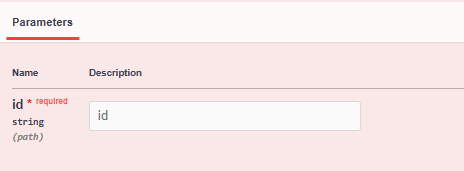
* Request Body: Book (hanya field yang ingin diubah)



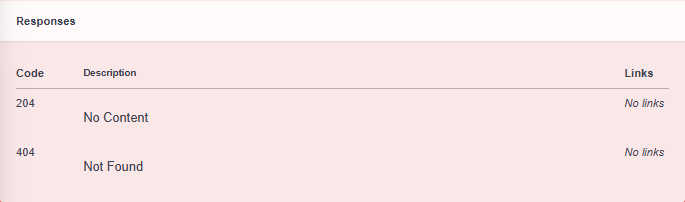
* Responses
  + 200 OK, mnegembalikan data buku dengan id yang dijadikan sebagai parameter
  + 204 No Content



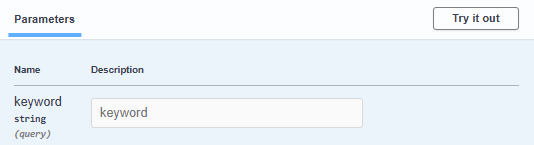
* 1. **DELETE/books/{id}**
* Deskripsi: Menghapus data buku dari sistem.
* Parameter: id



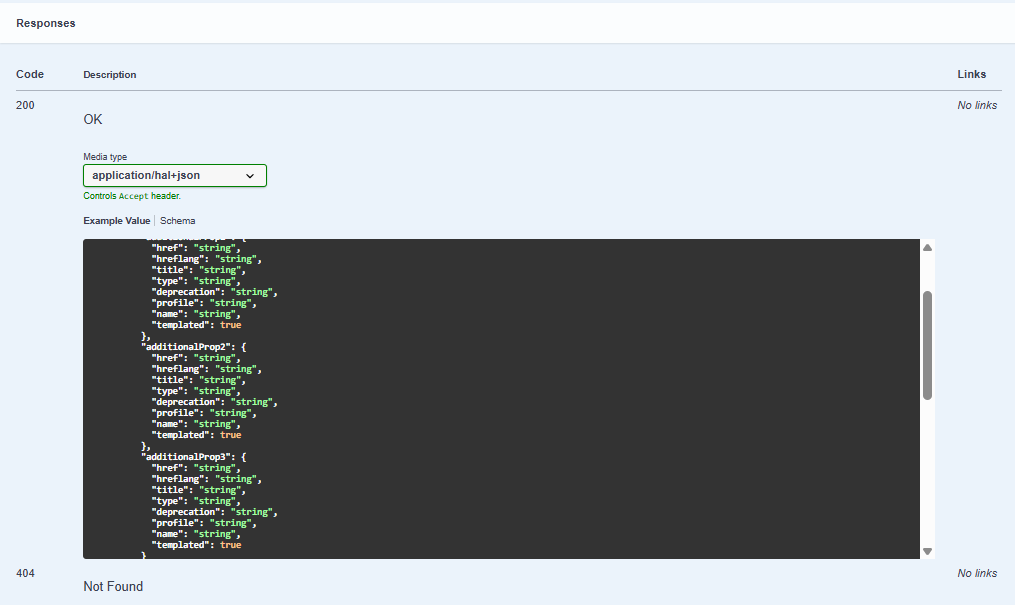
* Responses
  + 204 No Content
  + 404 Not Found, tidak dapat menemukan buku dengan id yang dijadikan parameter



* 1. **GET/books/search/findByTitleContainingIgnoreCase**
* Deskripsi: Mencari buku berdasarkan kata kunci tertentu.
* Parameter: keyword(string)



* Responses
  + 200OK, menampilkan buku yang dicari atau sesuai dengan keyword
  + 404 Not Found, tidak dapat menemukan buku dengan keyword= yang dijadikan parameter

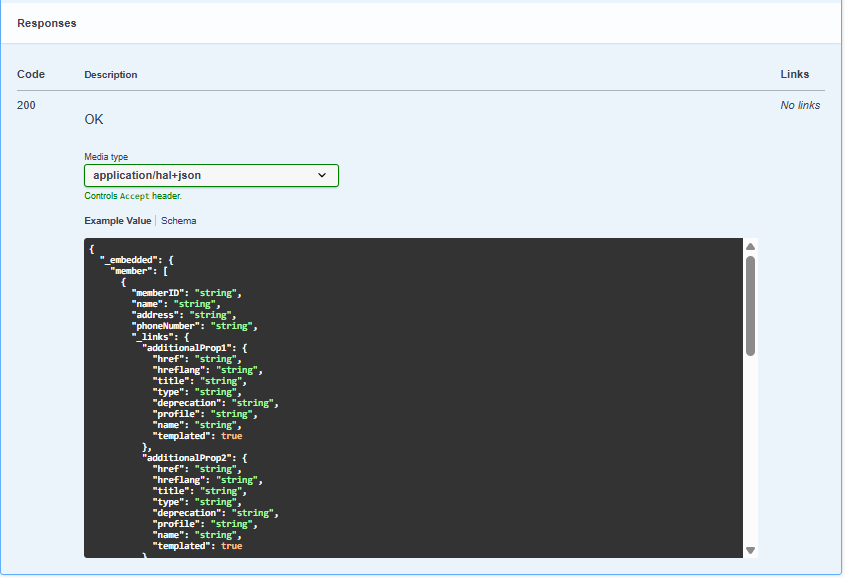


1. **Member (Endpoint untuk mengelola data anggota)**
   1. **GET /member**

**Fungsi**: Mengambil daftar semua member yang terdaftar.

**Metode HTTP**: GET

**Response Sukses**: 200 OK, dengan *body* berisi *array* dari objek Member.

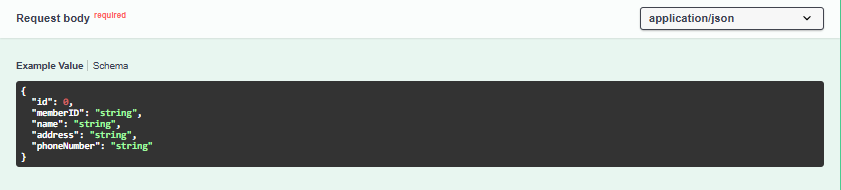


* 1. **POST /member**

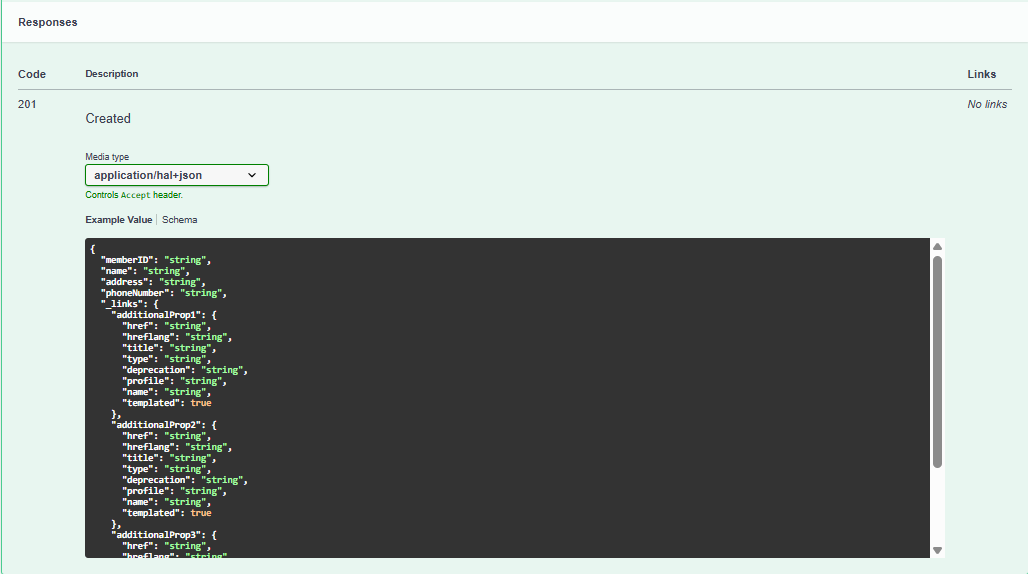
**Fungsi**: Menambahkan atau mendaftarkan seorang member baru ke dalam sistem.

**Metode HTTP**: POST

**Request Body**: Objek JSON yang merepresentasikan data member baru (misal: name, memberID, address).



**Response Sukses**: 201 Created, dengan *body* berisi data member yang baru saja dibuat, termasuk ID yang digenerate oleh sistem.

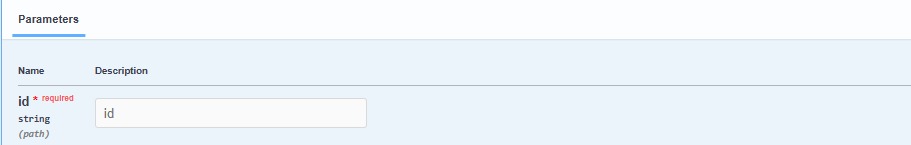


* 1. **GET /member/{id}**

**Fungsi**: Mengambil detail data seorang member spesifik berdasarkan ID uniknya.

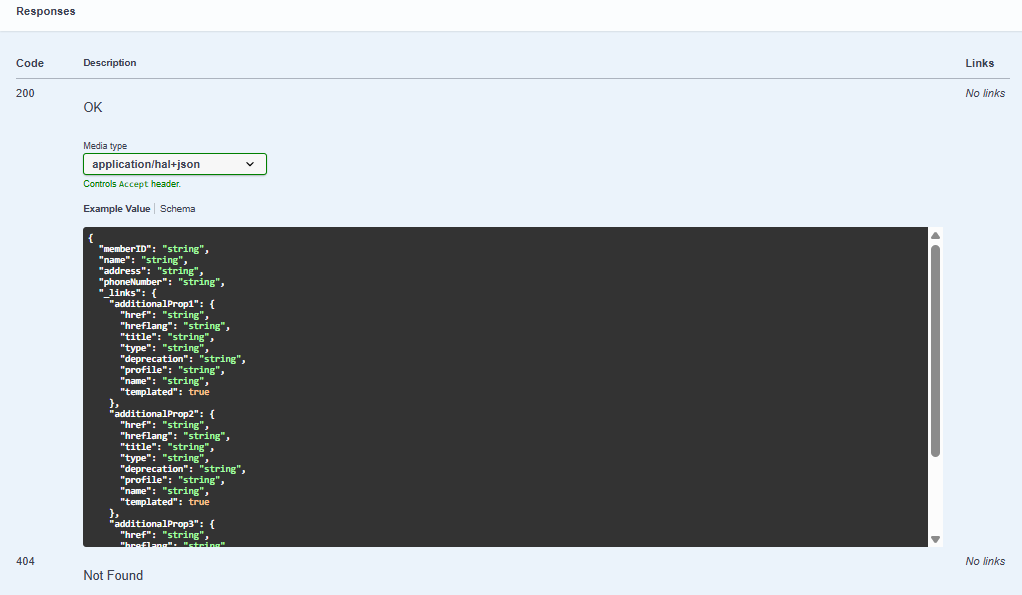
**Metode HTTP**: GET

**Parameter**: id (Path variable) - ID dari member yang ingin dicari.

****

**Response Sukses**: 200 OK, dengan *body* berisi objek Member.

**Response Gagal**: 404 Not Found jika member dengan ID tersebut tidak ditemukan.



* 1. **PUT /member/{id}**

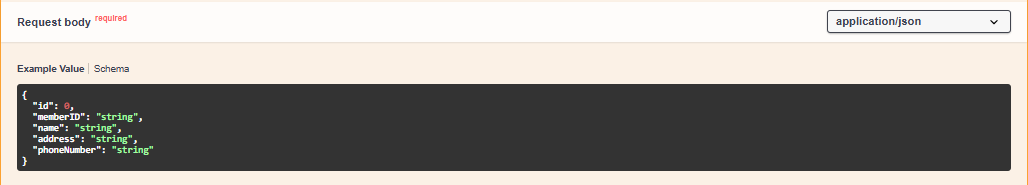
**Fungsi**: Memperbarui **seluruh** data member yang sudah ada. Metode ini mengharuskan klien mengirimkan semua atribut dari member, bahkan yang tidak berubah sekalipun.

**Metode HTTP**: PUT

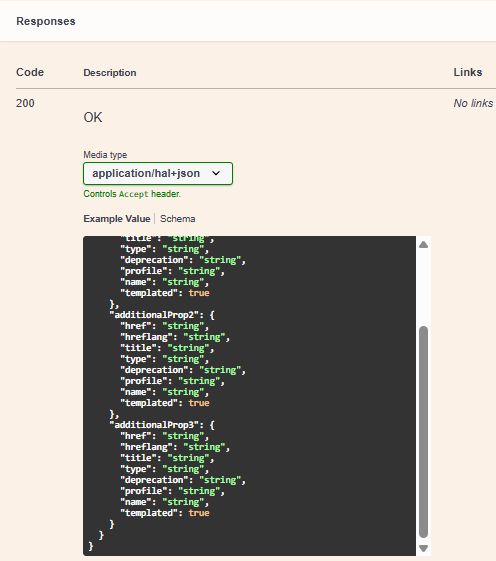
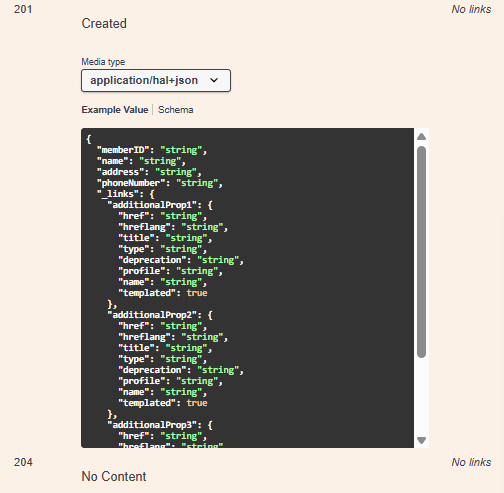
**Parameter**: id (Path variable) - ID dari member yang akan diperbarui.

****

**Request Body**: Objek JSON dengan data member yang lengkap.



**Response Sukses**: 200 OK.

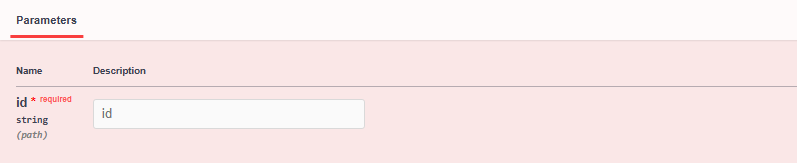
 

* 1. **DELETE /member/{id}**

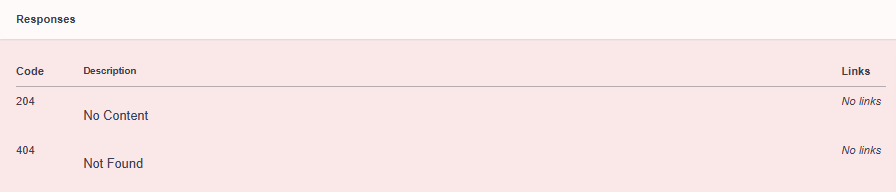
**Fungsi**: Menghapus data member dari sistem.

**Metode HTTP**: DELETE

**Parameter**: id (Path variable) - ID dari member yang akan dihapus.

****

**Response Sukses**: 204 No Content, menandakan data berhasil dihapus tanpa ada konten yang perlu dikembalikan.

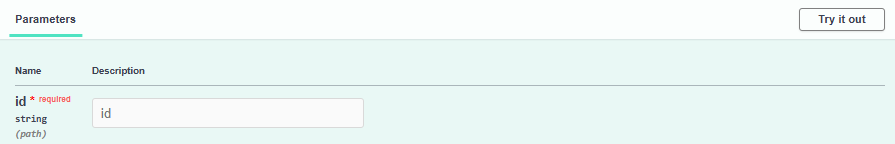


* 1. **PATCH /member/{id}**

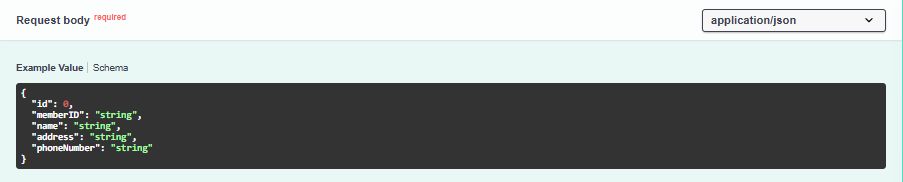
**Fungsi**: Memperbarui **sebagian** data member. Klien hanya perlu mengirimkan atribut yang ingin diubah.

**Metode HTTP**: PATCH

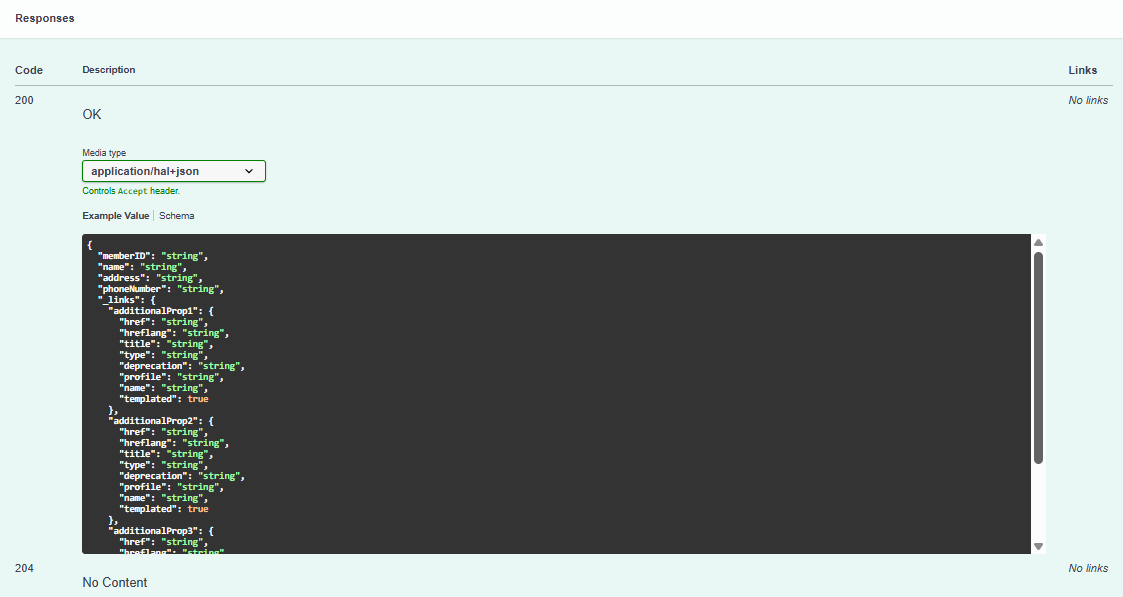
**Parameter**: id (Path variable) - ID dari member yang akan diperbarui.

****

**Request Body**: Objek JSON yang berisi hanya field yang ingin diubah.



**Response Sukses**: 200 OK.

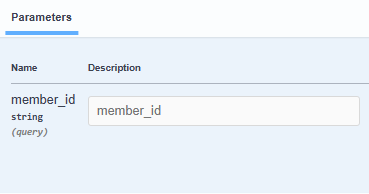
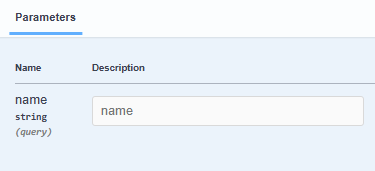


* 1. **GET /member/search/findByName dan GET /member/search/findByMemberID**

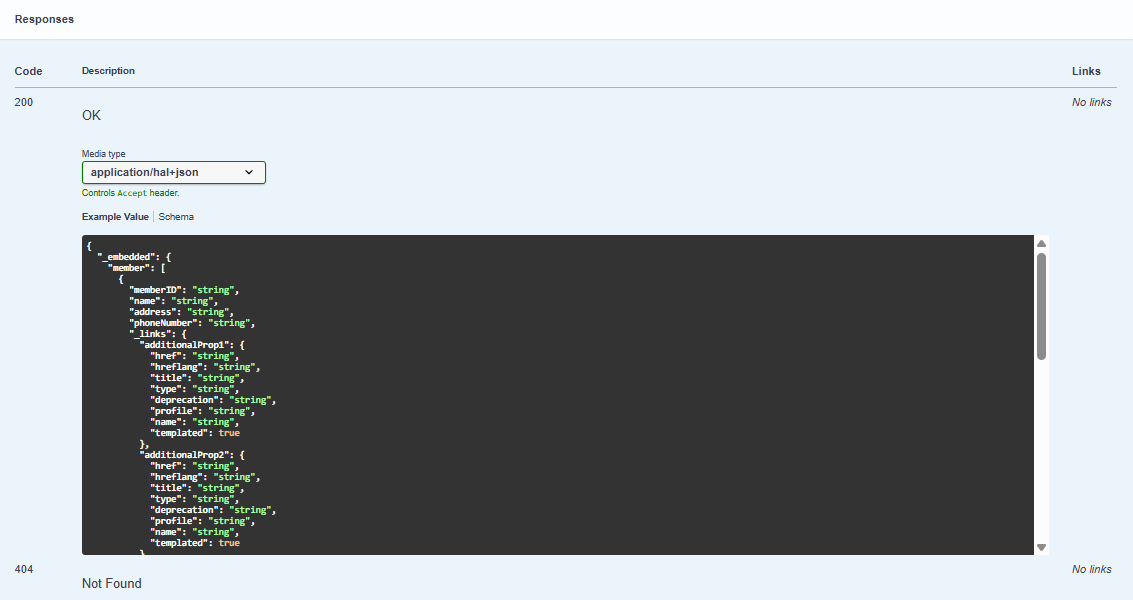
**Fungsi**: Melakukan pencarian member berdasarkan kriteria spesifik, yaitu nama atau ID Member.

**Metode HTTP**: GET

**Parameter**: name atau memberID (Request parameter).

**Response Sukses**: 200 OK, dengan *body* berisi *array* dari Member yang cocok dengan kriteria pencarian atau 404 Not Found

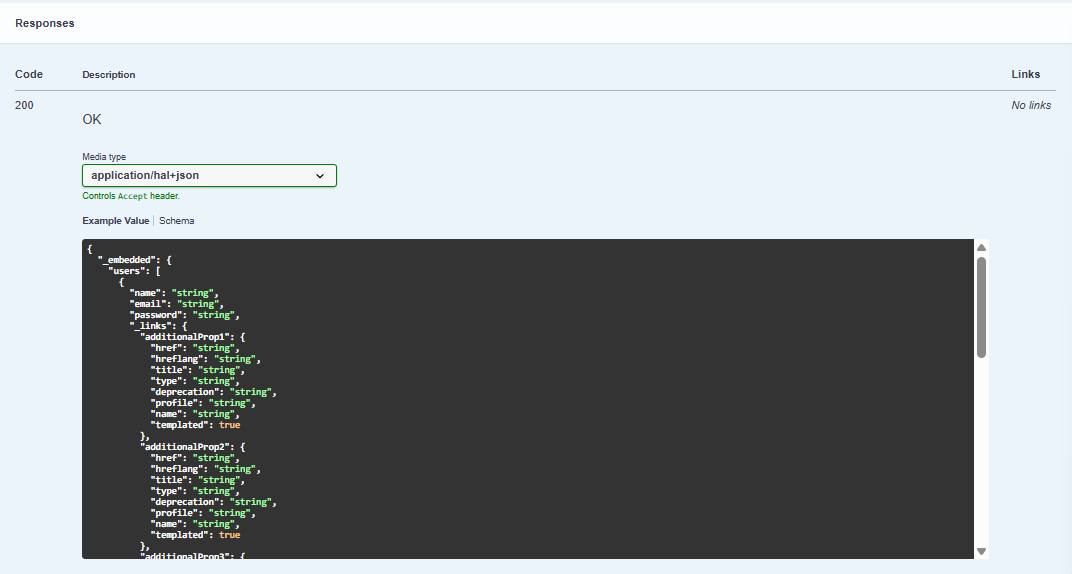


1. **User (Endpoint untuk mengelola data user)**
   1. **GET /User**

**Fungsi**: Mengambil daftar semua User yang terdaftar.

**Metode HTTP**: GET

**Response Sukses**: 200 OK, dengan *body* berisi *array* dari objek User.

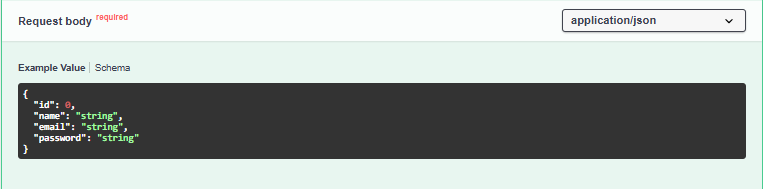


* 1. **POST /User**

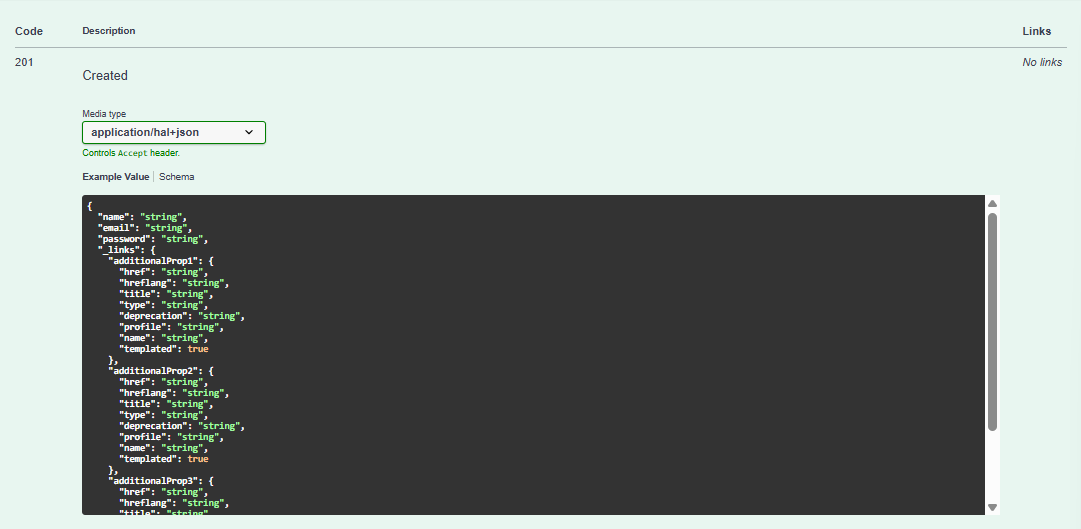
**Fungsi**: Menambahkan atau mendaftarkan seorang User baru ke dalam sistem.

**Metode HTTP**: POST

**Request Body**: Objek JSON yang merepresentasikan data User baru (misal: name, UserID, address).



**Response Sukses**: 201 Created, dengan *body* berisi data User yang baru saja dibuat, termasuk ID yang digenerate oleh sistem.

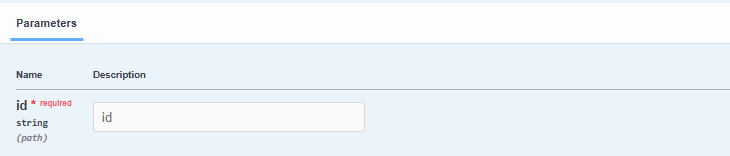


* 1. **GET /User/{id}**

**Fungsi**: Mengambil detail data seorang User spesifik berdasarkan ID uniknya.

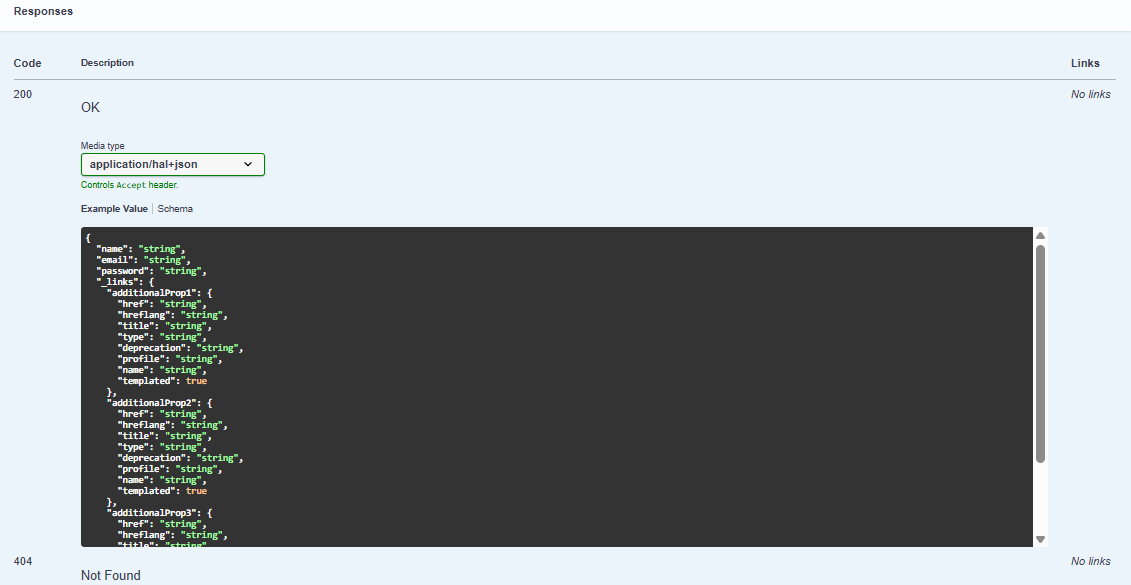
**Metode HTTP**: GET

**Parameter**: id (Path variable) - ID dari User yang ingin dicari.



**Response Sukses**: 200 OK, dengan *body* berisi objek User.

**Response Gagal**: 404 Not Found jika User dengan ID tersebut tidak ditemukan.



* 1. **PUT /User/{id}**

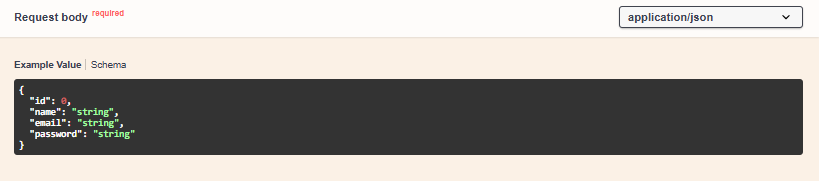
**Fungsi**: Memperbarui seluruh data User yang sudah ada. Metode ini mengharuskan klien mengirimkan semua atribut dari User, bahkan yang tidak berubah sekalipun.

**Metode HTTP**: PUT

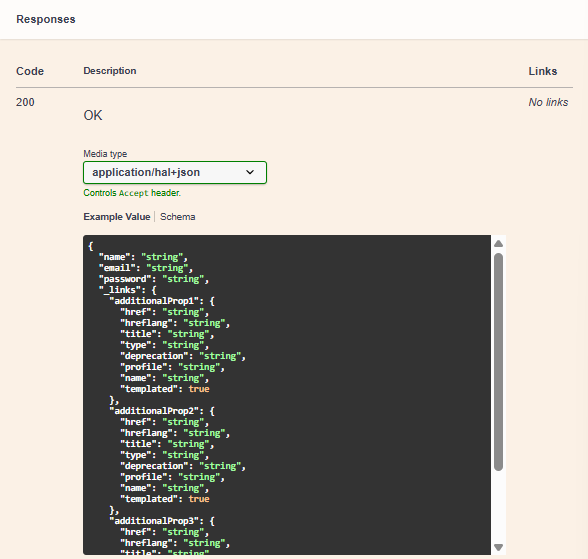
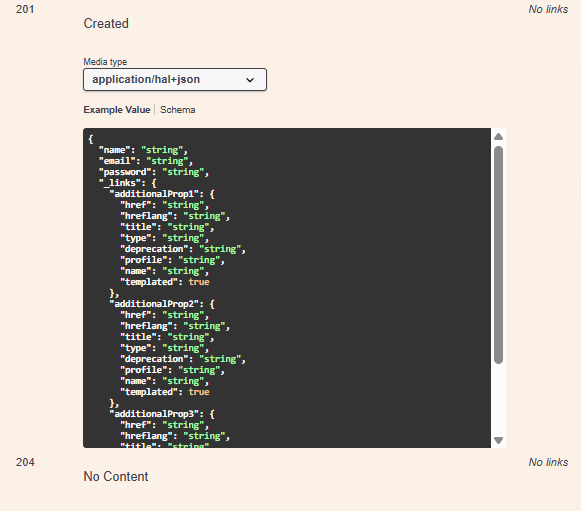
**Parameter**: id (Path variable) - ID dari User yang akan diperbarui.



**Request Body**: Objek JSON dengan data User yang lengkap.



**Response Sukses**: 200 OK, 201 Createad, 204 No Content

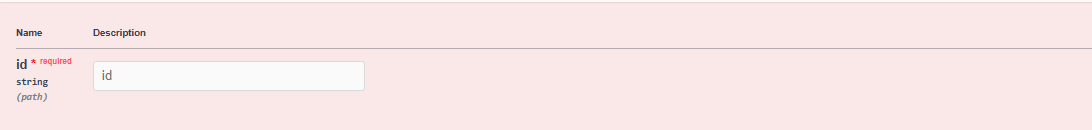
 

* 1. **DELETE /User/{id}**

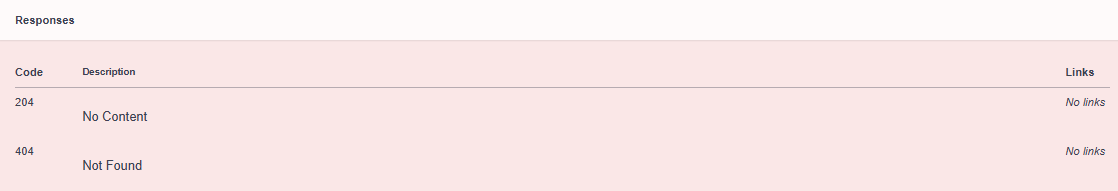
**Fungsi**: Menghapus data User dari sistem.

**Metode HTTP**: DELETE

**Parameter**: id (Path variable) - ID dari User yang akan dihapus.



**Response Sukses**: 204 No Content, menandakan data berhasil dihapus tanpa ada konten yang perlu dikembalikan.



* 1. **PATCH /User/{id}**

**Fungsi**: Memperbarui sebagian data User. Klien hanya perlu mengirimkan atribut yang ingin diubah.

**Metode HTTP**: PATCH

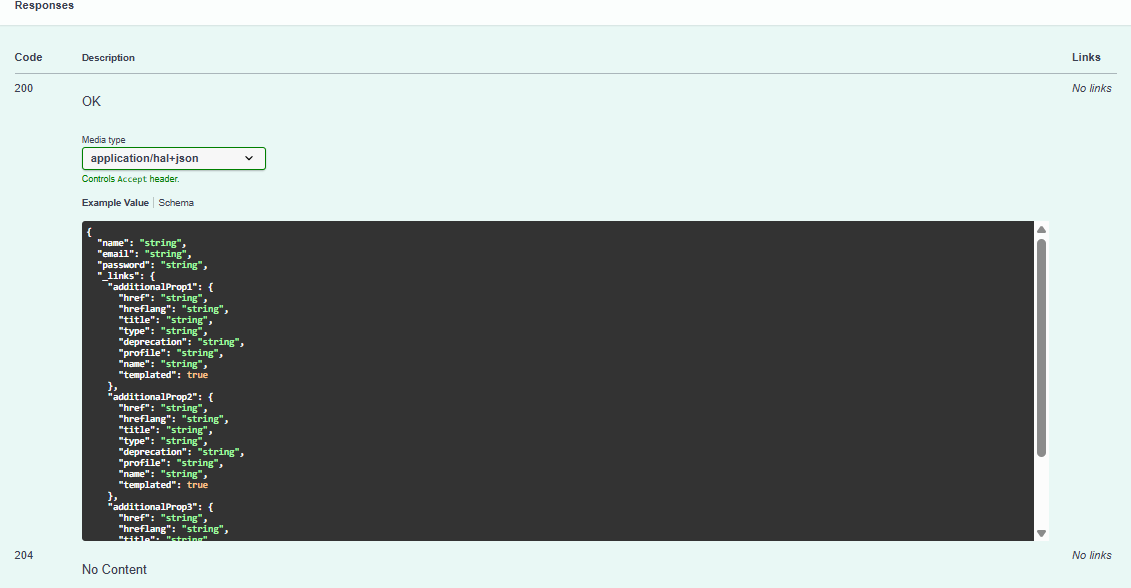
**Parameter**: id (Path variable) - ID dari User yang akan diperbarui.



**Request Body**: Objek JSON yang berisi hanya field yang ingin diubah.



**Response Sukses**: 200 OK, 204 No Content

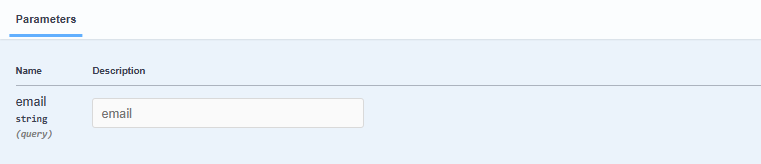


* 1. **GET /User/search/findByEmail**

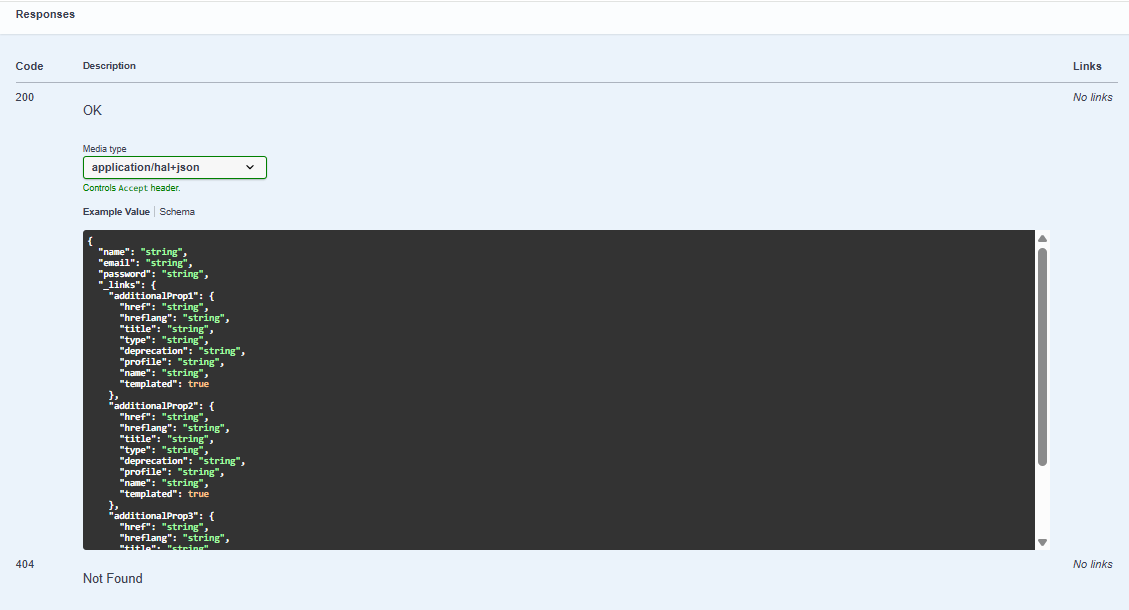
**Fungsi**: Melakukan pencarian User berdasarkan email USER

**Metode HTTP**: GET

**Parameter**: email (Request parameter).



**Response Sukses**: 200 OK, dengan *body* berisi *array* dari User yang cocok dengan kriteria pencarian.



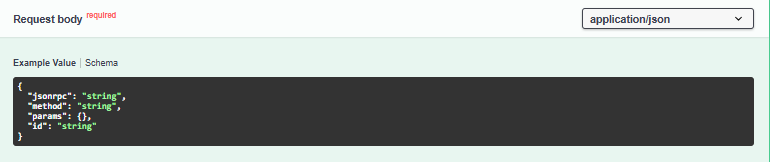
1. **JSON-RPC (Endpoint untuk JSON-RPC untuk operasi buku)**
   1. **POST/jsonrpc**

**Fungsi:** Sebagaigateaway untuk semua operasi yang berkaitan dengan manajemen buku.

**Metode HTTP:** POST

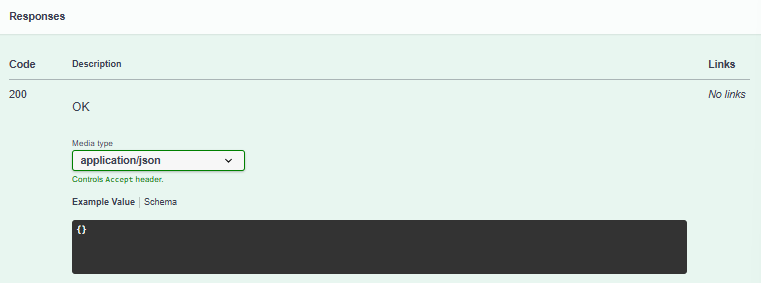
**Request Body:**

* Jsonrpc
* Method
* Params
* Id

****

**Response Sukses:** 200 OK. Body dari respons akan mengikuti format JSON-RPC, berisi result dari operasi yang dijalankan dan id yang sama dengan permintaan.

**Response Gagal:** 400 Bad Request, jika method tidak ditemukan atau jika ada kesalahan dalam memproses permintaan.

****