**TUGAS UTS PEMROGRAMAN**

**Hendra wijaya hadi**

**TI 6C**

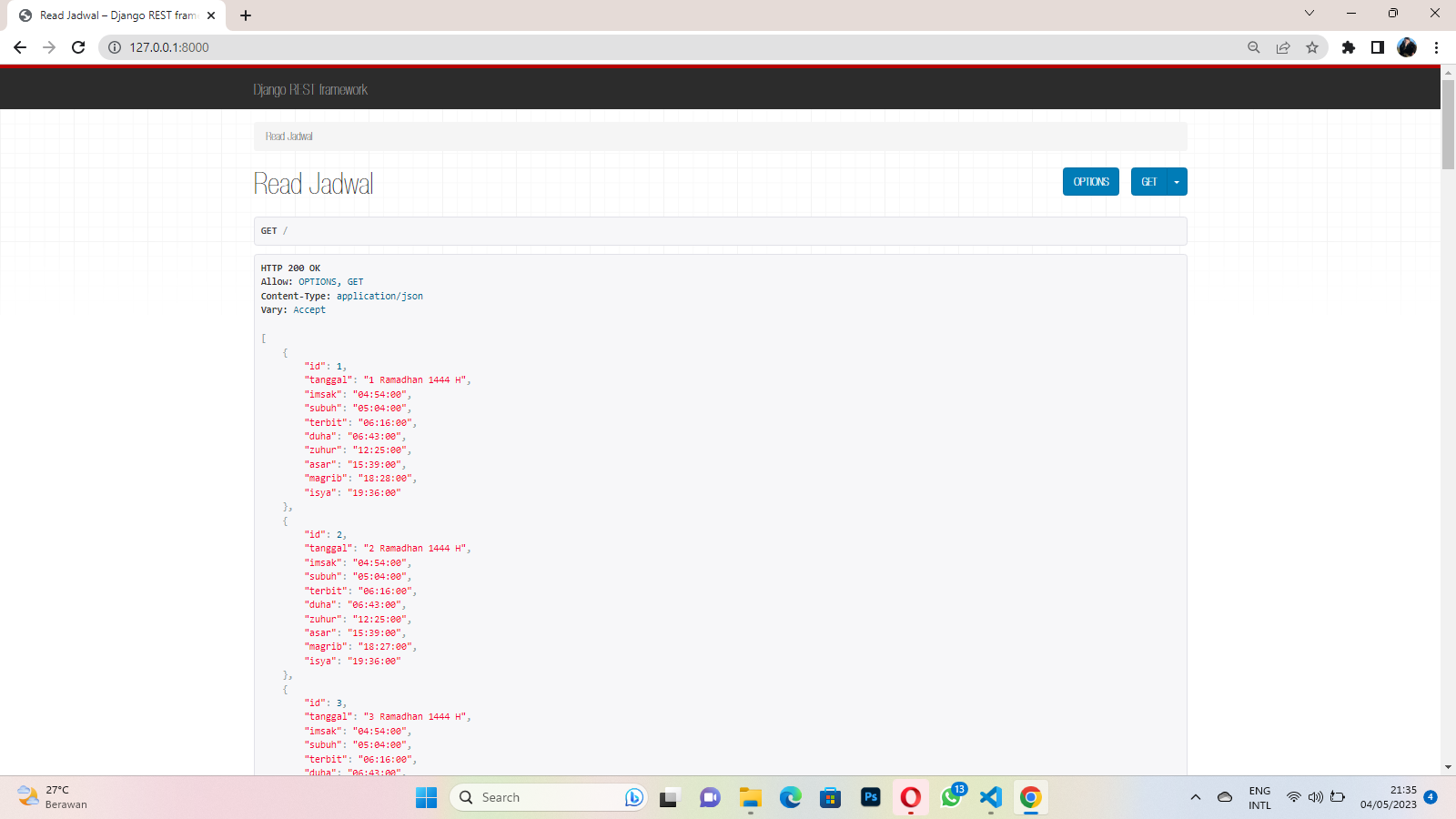
**API IMSAKIYAH 1444 H**

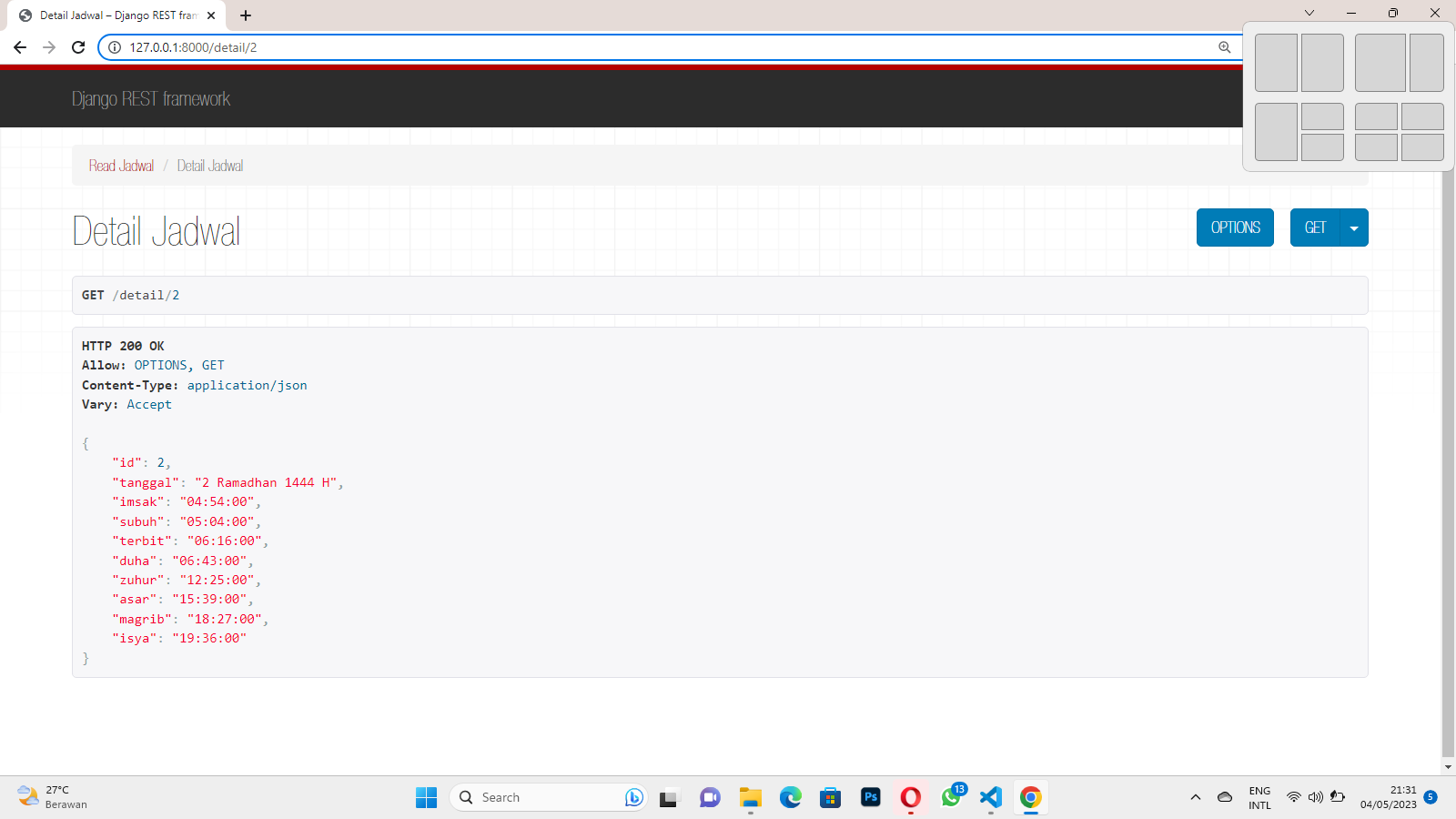
1. **Keterangan atau Penjelasan Kode**

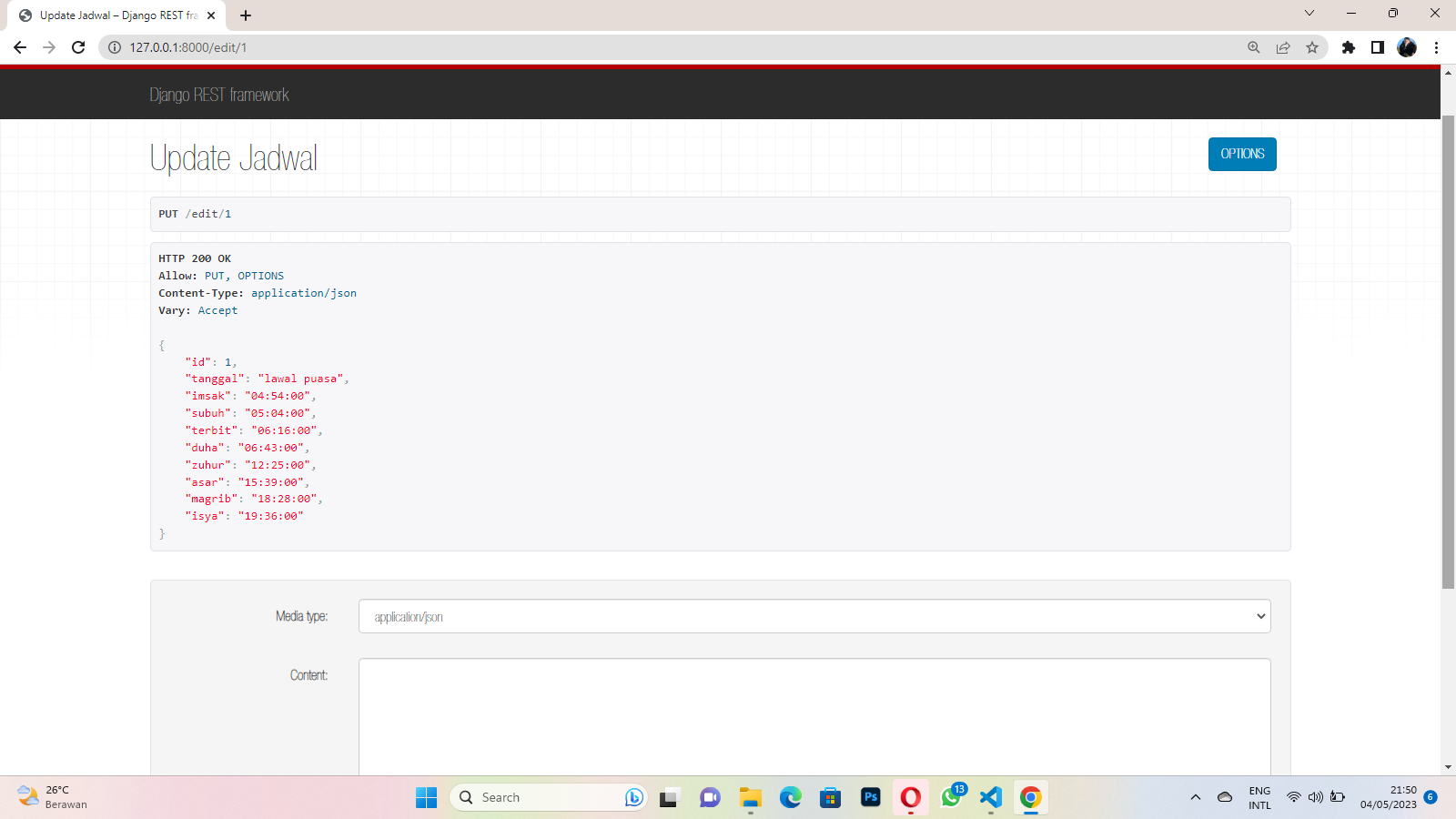
Pada kode berikut ada beberapa fungsi yang dibuat diantaranya Create, Read, Update, Delete (**CRUD**) API Imsak menggunakan Django Rest Framework.

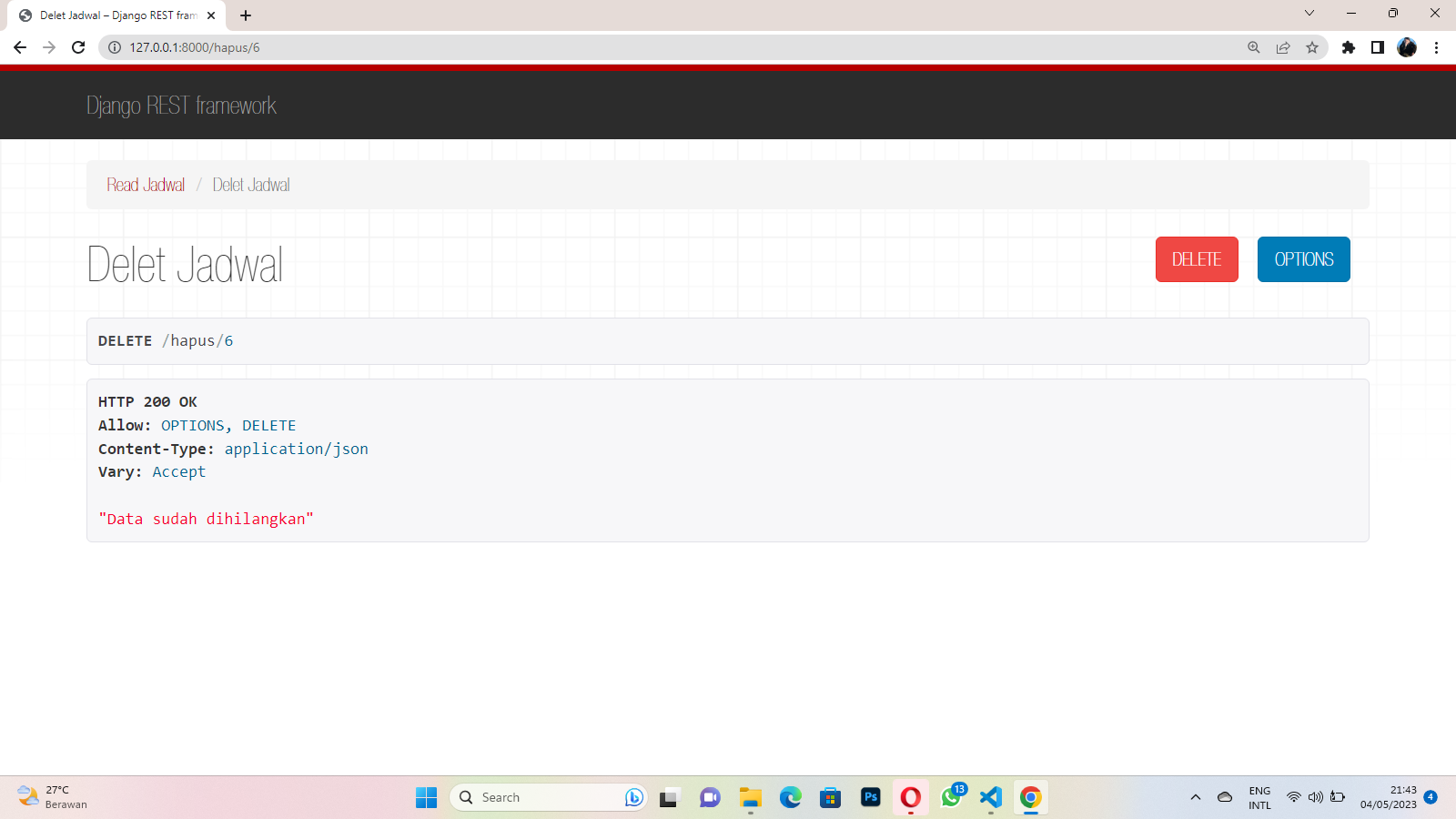
|  |
| --- |
|  |
| @api\_view(['GET'])  Decorator ini hanya dapat menerima request dengan method **HTTP GET**  Pada fungsi **readJadwal(request)**, **Jadwal.objects.all()** untuk mengambil *semua data Jadwal yang tersimpan pada database*. Kemudian, data tersebut di-serialisasi menggunakan jadwalSerializer dengan parameter **many=Tru**e, karena data yang diambil berupa *banyak* data. Selanjutnya, data yang sudah di-serialisasi dikembalikan sebagai HTTP response menggunakan Response(serializer.data). |
|  |
| @api\_view(['GET'])  Decorator ini hanya dapat menerima request dengan method **HTTP GET**  Pada fungsi **detailJadwal(request, id)**, parameter id digunakan untuk mencari data Jadwal berdasarkan id-nya yang diterima dari HTTP request. untuk mencari data Jadwal dengan id yang sama dengan id yang diterima menggunakan  **Jadwal.objects.get(pk=id)**  Setelah itu, data Jadwal yang telah ditemukan tersebut di-serialisasi menggunakan jadwalSerializer dengan parameter **many=False**, karena data yang diambil hanya berupa *satu data*. Kemudian, data yang sudah di-serialisasi dikembalikan sebagai HTTP response menggunakan **Response(serializer.data).** |
|  |
| *@api\_view(['POST'])*  Decorator ini digunakan untuk memberi tanda bahwa view function yang didekorasi adalah sebuah view yang hanya dapat menerima request dengan method **HTTP POST**  **def createJadwal(request):**  fungsi createJadwal(request), data yang diterima dari HTTP request di-serialisasi menggunakan jadwalSerializer. Data yang diterima berupa data baru yang akan ditambahkan ke database. pada saat inisialisasi jadwalSerializer, digunakan data=request.data untuk mengambil data baru yang dikirimkan oleh client melalui HTTP request.  Setelah itu, dilakukan pengecekan apakah data yang diterima oleh **jadwalSerializer** adalah *valid atau tidak*, menggunakan method **is\_valid().** Jika valid, data tersebut disimpan ke dalam database menggunakan method **save(**), yang secara otomatis akan memanggil create() pada jadwalSerializer.  Setelah data tersimpan kemudian akan dikirimkan kembali sebagai HTTP response menggunakan Response(serializer.data) |
|  |
| @api\_view(['PUT'])  Decorator ini hanya dapat menerima request dengan method **HTTP PUT**  Pada fungsi **updateJadwal(request, id),** data yang diterima dari HTTP request di-serialisasi menggunakan jadwalSerializer. pada saat inisialisasi jadwalSerializer, digunakan **instance=jadwalimsyak** untuk menentukan data yang akan **di-update**.  Data yang dikirimkan oleh client melalui HTTP request diambil menggunakan **data=request.data**. Setelah itu, dilakukan pengecekan apakah data yang diterima oleh jadwalSerializer adalah valid atau tidak. Jika valid, data tersebut disimpan ke dalam database menggunakan method save(), yang secara otomatis akan memanggil update() pada jadwalSerializer. |
|  |
| @api\_view(['DELETE'])  Decorator ini hanya dapat menerima request dengan method **HTTP DELETE**  Pada fungsi **deletJadwal(request, id),** data yang akan dihapus diambil dari database menggunakan  **Jadwal.objects.get(pk=id),** dengan pk=id sebagai primary key dari data yang akan dihapus. Kemudian, data tersebut dihapus dari database menggunakan **method delete(),** yang akan menghapus data dari database sesuai dengan primary key yang diberikan.  Setelah data berhasil dihapus, fungsi akan **mengembalikan response** dengan pesan *'Data berhasil di Hapus'*. |

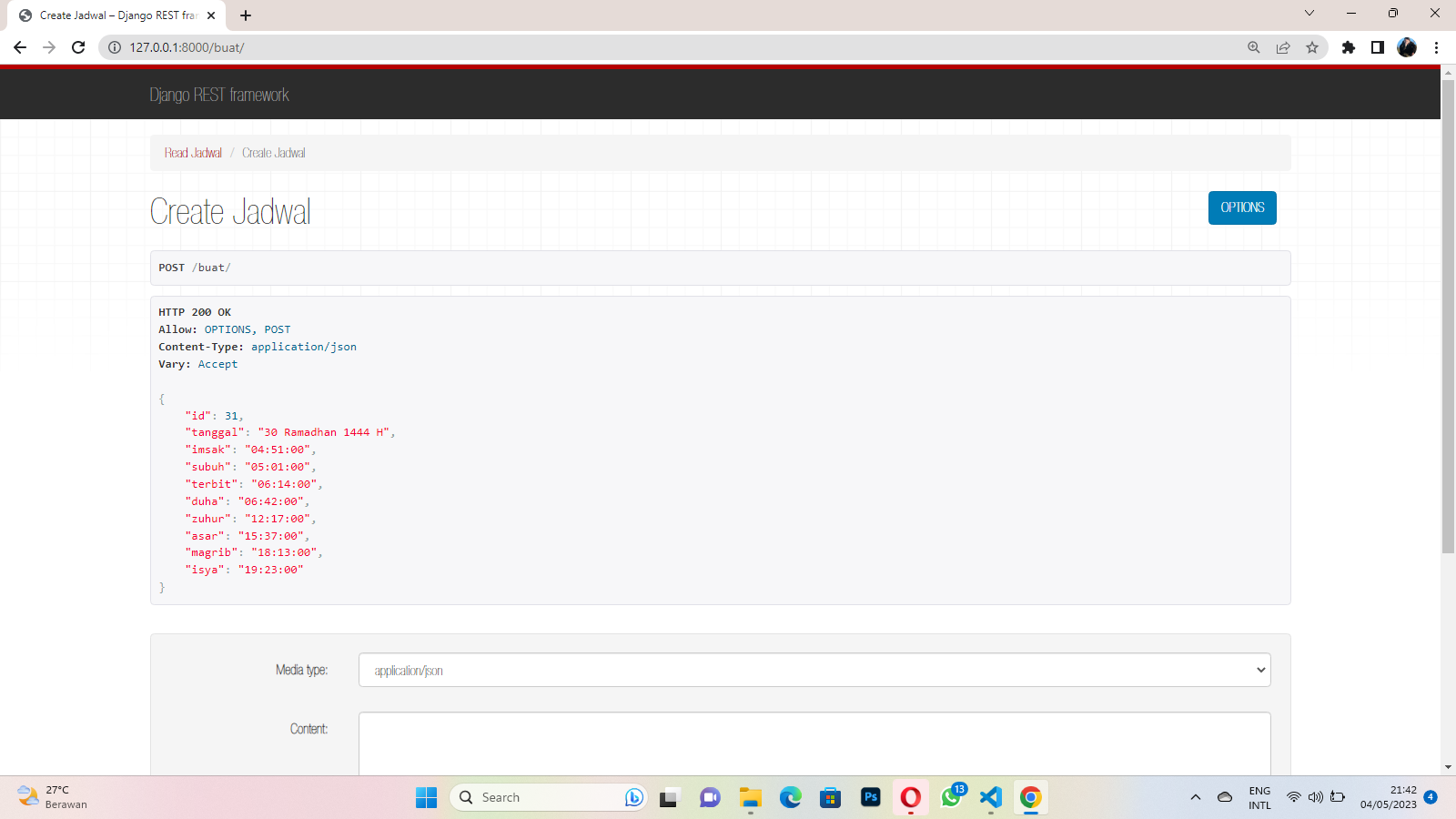
**Respon API**

****

****

****

****

****