TUGAS UTS PEMROGRAMAN

API IMSAKIYAH 1444 H

A. Sekilas tentang Api Imsakiyah

Data waktu Imsak sangat penting bagi umat Muslim yang sedang menjalankan ibadah puasa selama bulan Ramadhan. Rest API Imsakiyah Ramadhan umumnya menyediakan informasi dalam format JSON yang dapa diakses melalui HTTP request. Informasi yang biasanya tersedia dalam Rest API Imsakiyah Ramadhan meliputi Tanggal, waktu Imsak, Subuh, Terbit, Duha, waktu Dzuhur, waktu Ashar, waktu Maghrib, dan waktu Isya' untuk setiap hari selama bulan Ramadhan. Pada pembuatan Rest Api ini menggunakan bahasa pemrograman Python dengan Framework Django.

B. Keterangan atau Penjelasan Kode

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

```python
# menampilkan semua data
@api_view(['GET'])
def readJadwal(request):
    jadwalimsyak = JadwalModels.objects.all()
    serializer = jadwalSerializer(jadwalimsyak, many=True)
    return Response(serializer.data)
```

@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).

```python
# menampilkan 1 data atau detail
@api_view(['GET'])
def detailJadwal(request,id):
    jadwalimsyak = JadwalModels.objects.get(pk=id)
    serializer = jadwalSerializer(jadwalimsyak, many=False)
    return 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 denganidyang 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).**

```
# menambah/input data
@api_view(['POST'])
def createJadwal(request):
    serializer = jadwalSerializer(data=request.data)
    if serializer.is_valid():
        serializer.save()
    return 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-serialisasimenggunakan jadwalSerializer. Data yang diterima berupa data baru yang akan ditambahkan ke database. pada saat inisialisasi jadwalSerializer, digunakan data=request.data untuk mengambildatabaru 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)

```
# mengedit/update data
@api_view(['PUT'])
def updateJadwal(request,id):
    jadwalimsyak = JadwalModels.objects.get(pk=id)
    serializer=jadwalSerializer(instance=jadwalimsyak, data=request.data)
    if serializer.is_valid():
        serializer.save()
    return Response(serializer.data)
```

@api_view(['PUT'])
Decorator ini hanya dapat menerima request dengan method **HTTP PUT**

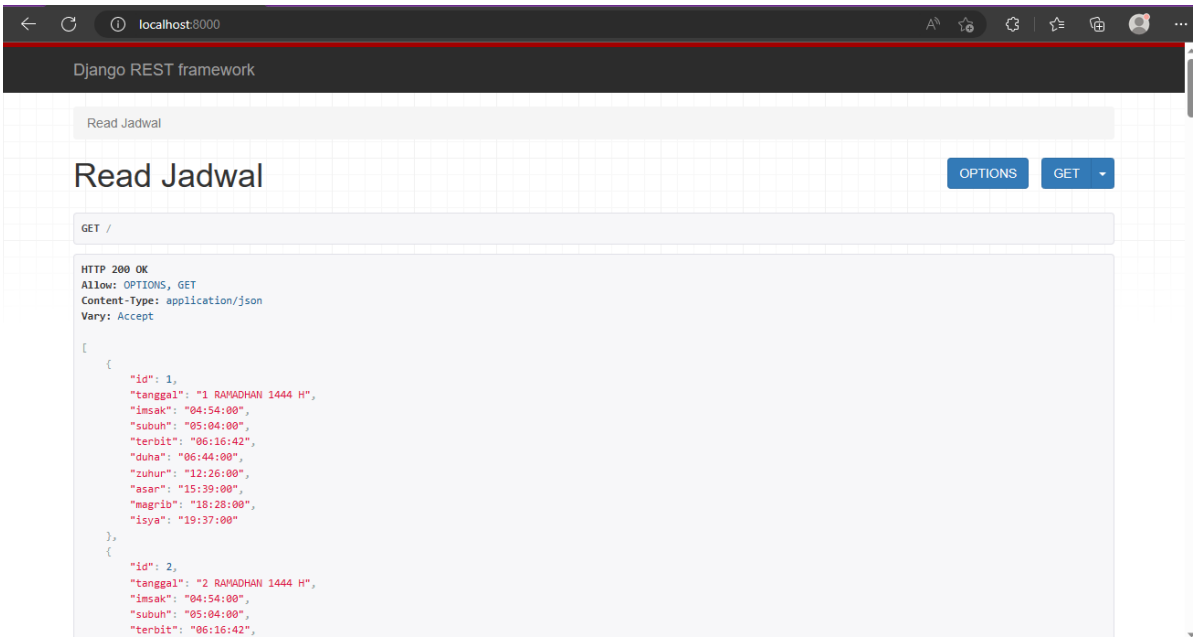
Pada fungsi **updateJadwal(request, id),** data yang diterima dari HTTP request di-serialisasi menggunakan jadwalSerializer. pada saat inisialisasi jadwalSerializer, digunakan **instance=jadwalimsyak** untuk menentukan data yang akan **di-update**.
Data yang dikirimkan oleh client melalui HTTP request 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.

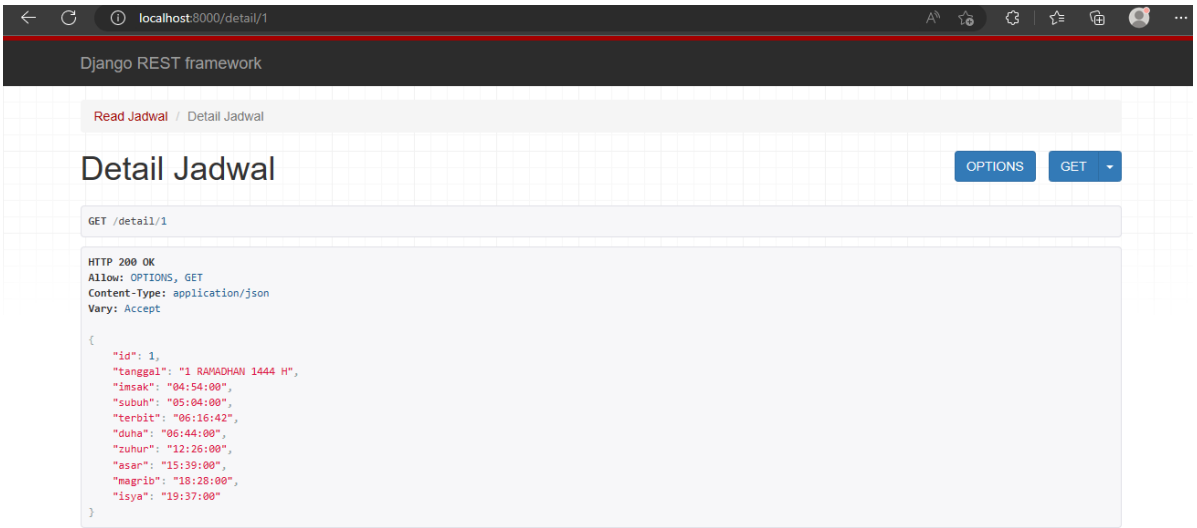
```
# meghapus/delet
@api_view(['DELETE'])
def deletJadwal(request,id):
    jadwalimsyak = JadwalModels.objects.get(pk=id)
    jadwalimsyak.delete()

    return Response('Data sudah dihilangkan')
```

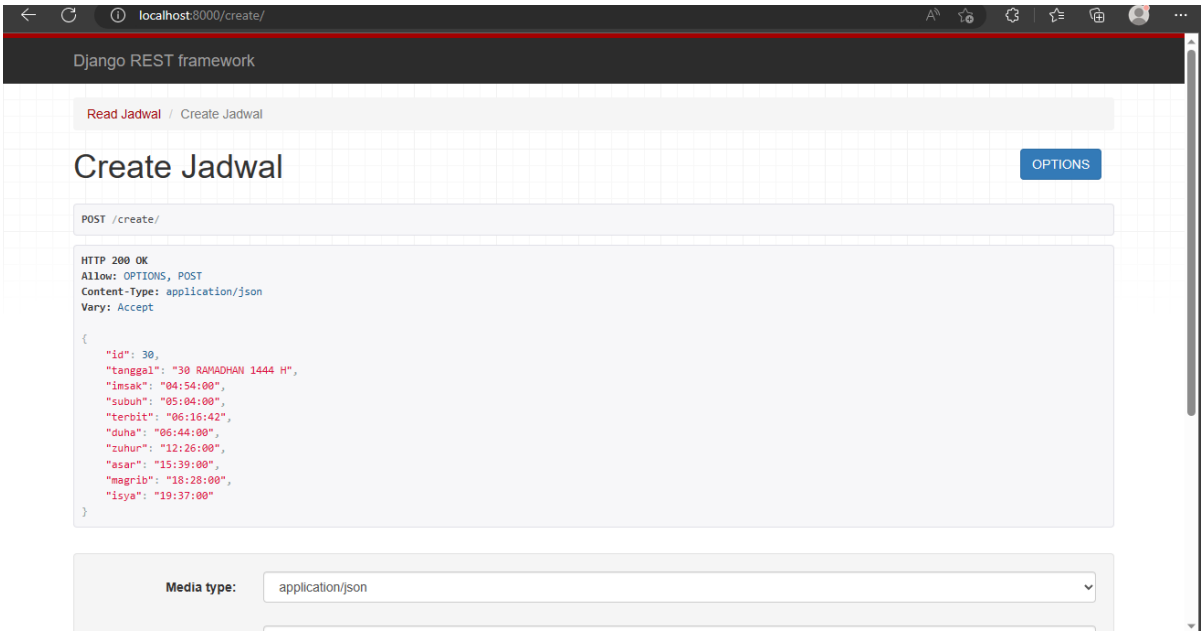
@api_view(['DELETE'])
Decorator ini hanya dapat menerima request dengan method **HTTP DELETE**

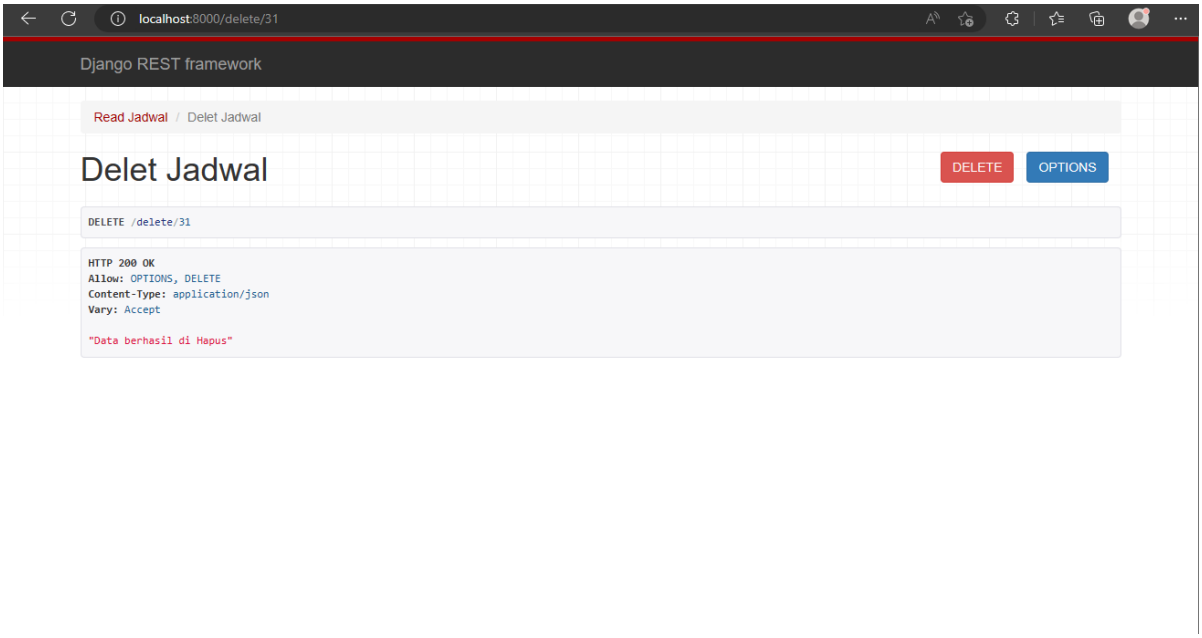
Pada fungsi **deletJadwal(request, id),** data yang akan dihapus diambil dari database menggunakan
Jadwal.objects.get(pk=id), dengan pk=id sebagai primary key dari data yang akan dihapus. Kemudian, data tersebut dihapus dari database menggunakan **method delete(),** yang akan menghapus data dari database sesuai dengan primary key yang diberikan.
Setelah data berhasil dihapus, fungsi akan **mengembalikan response** dengan pesan *'Data berhasil di Hapus'*.

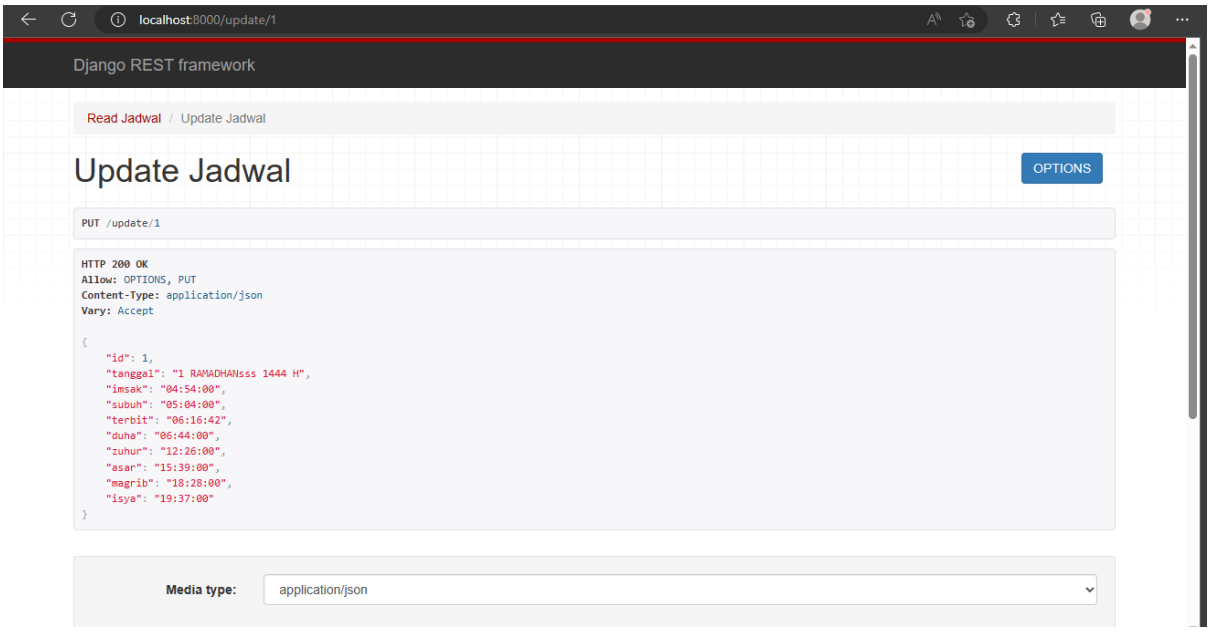
Respon API


localhost:8000
Django REST framework
Read Jadwal
Read Jadwal
OPTIONS
GET
GET /
HTTP 200 OK
Allow: OPTIONS, GET
Content-Type: application/json
Vary: Accept
[
{
"id": 1,
"tanggal": "1 RAMADHAN 1444 H",
"imsak": "04:54:00",
"subuh": "05:04:00",
"terbit": "06:16:42",
"duha": "06:44:00",
"zuhur": "12:26:00",
"asar": "15:39:00",
"magrib": "18:28:00",
"isya": "19:37:00"
},
{
"id": 2,
"tanggal": "2 RAMADHAN 1444 H",
"imsak": "04:54:00",
"subuh": "05:04:00",
"terbit": "06:16:42",


localhost:8000/detail/1
Django REST framework
Read Jadwal / Detail Jadwal
Detail Jadwal
OPTIONS
GET
GET /detail/1
HTTP 200 OK
Allow: OPTIONS, GET
Content-Type: application/json
Vary: Accept
{
"id": 1,
"tanggal": "1 RAMADHAN 1444 H",
"imsak": "04:54:00",
"subuh": "05:04:00",
"terbit": "06:16:42",
"duha": "06:44:00",
"zuhur": "12:26:00",
"asar": "15:39:00",
"magrib": "18:28:00",
"isya": "19:37:00"
}


localhost:8000/create/
Django REST framework
Read Jadwal / Create Jadwal
Create Jadwal
OPTIONS
POST /create/
HTTP 200 OK
Allow: OPTIONS, POST
Content-Type: application/json
Vary: Accept
{
"id": 30,
"tanggal": "30 RAMADHAN 1444 H",
"imsak": "04:54:00",
"subuh": "05:04:00",
"terbit": "06:16:42",
"duha": "06:44:00",
"zuhur": "12:26:00",
"asar": "15:39:00",
"magrib": "18:28:00",
"isya": "19:37:00"
}
Media type:
application/json


localhost:8000/delete/31
Django REST framework
Read Jadwal / Delet Jadwal
Delet Jadwal
DELETE
OPTIONS
DELETE /delete/31
HTTP 200 OK
Allow: OPTIONS, DELETE
Content-Type: application/json
Vary: Accept
"Data berhasil di Hapus"


localhost:8000/update/1
Django REST framework
Read Jadwal / Update Jadwal
Update Jadwal
OPTIONS
PUT /update/1
HTTP 200 OK
Allow: OPTIONS, PUT
Content-Type: application/json
Vary: Accept
{
"id": 1,
"tanggal": "1 RAMADHANsss 1444 H",
"imsak": "04:54:00",
"subuh": "05:04:00",
"terbit": "06:16:42",
"duha": "06:44:00",
"zuhur": "12:26:00",
"asar": "15:39:00",
"magrib": "18:28:00",
"isya": "19:37:00"
}
Media type:
application/json