

TUGAS UTS PEMROGRAMAN

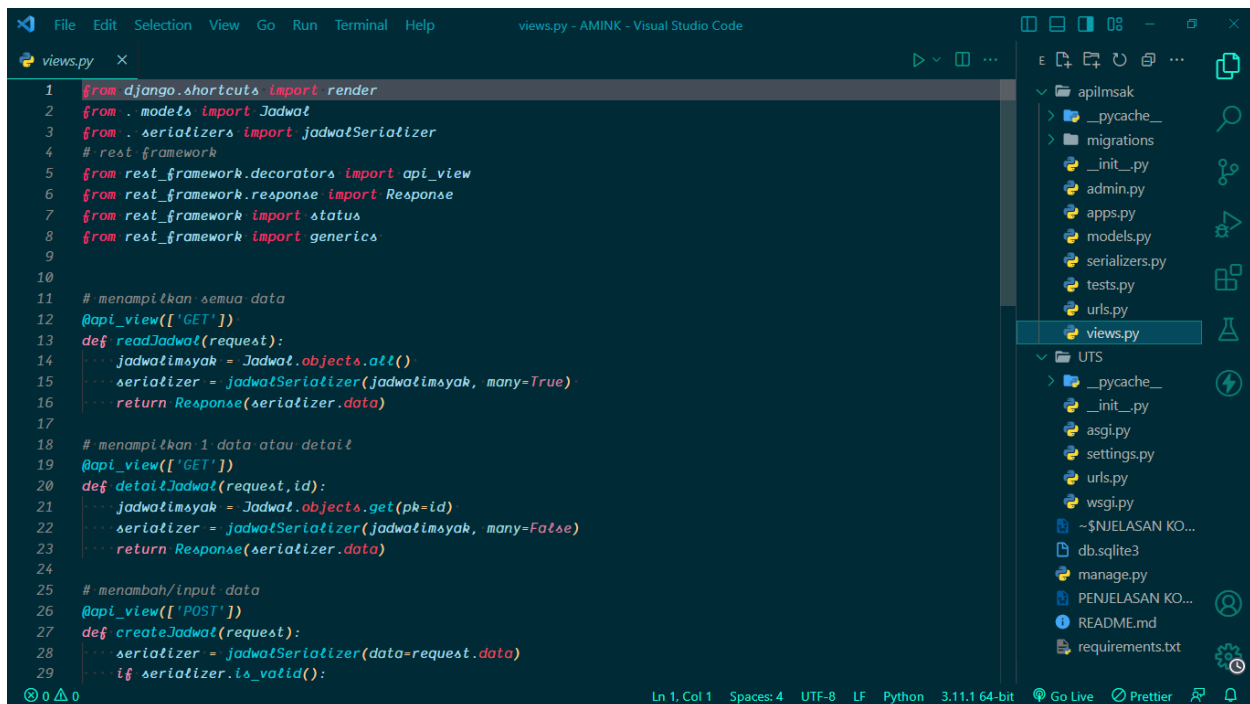
Nama : M. HARIS SUHARMIN

Kelas : TI C

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 dapat 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.

A. Keterangan atau Penjelasan Kode

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



```
1 from django.shortcuts import render
2 from .models import Jadwal
3 from .serializers import JadwalSerializer
4 # rest framework
5 from rest_framework.decorators import api_view
6 from rest_framework.response import Response
7 from rest_framework import status
8 from rest_framework import generics
9
10
11 # menampilkan semua data
12 @api_view(['GET'])
13 def readJadwal(request):
14     jadwalimsyak = Jadwal.objects.all()
15     serializer = JadwalSerializer(jadwalimsyak, many=True)
16     return Response(serializer.data)
17
18 # menampilkan 1 data atau detail
19 @api_view(['GET'])
20 def detailJadwal(request, id):
21     jadwalimsyak = Jadwal.objects.get(pk=id)
22     serializer = JadwalSerializer(jadwalimsyak, many=False)
23     return Response(serializer.data)
24
25 # menambah/input data
26 @api_view(['POST'])
27 def createJadwal(request):
28     serializer = JadwalSerializer(data=request.data)
29     if serializer.is_valid():
```

@api_view(['GET'])

Decorator ini digunakan untuk memberi tanda bahwa view function yang didekorasi adalah sebuah view yang 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=True**, karena data yang diambil berupa *banyak data*. Selanjutnya, data yang sudah di-serialisasi dikembalikan sebagai HTTP response menggunakan **Response(serializer.data)**.

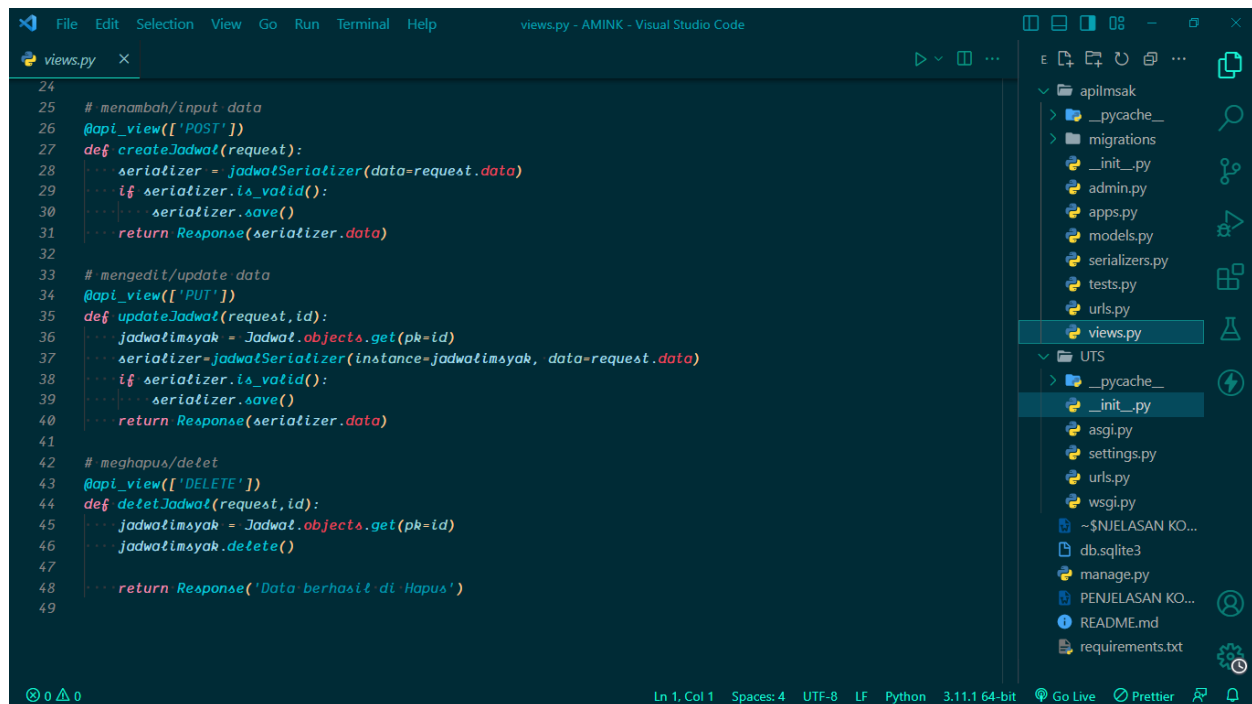
@api_view(['GET'])

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)**.



```
24
25 # menambah/input data
26 @api_view(['POST'])
27 def createJadwal(request):
28     serializer = JadwalSerializer(data=request.data)
29     if serializer.is_valid():
30         serializer.save()
31     return Response(serializer.data)
32
33 # mengedit/update data
34 @api_view(['PUT'])
35 def updateJadwal(request, id):
36     jadwalimsyak = Jadwal.objects.get(pk=id)
37     serializer = JadwalSerializer(instance=jadwalimsyak, data=request.data)
38     if serializer.is_valid():
39         serializer.save()
40     return Response(serializer.data)
41
42 # menghapus/delet
43 @api_view(['DELETE'])
44 def deleteJadwal(request, id):
45     jadwalimsyak = Jadwal.objects.get(pk=id)
46     jadwalimsyak.delete()
47
48     return Response('Data berhasil di Hapus')
49
```

@api_view(['POST'])

Decorator ini 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'])

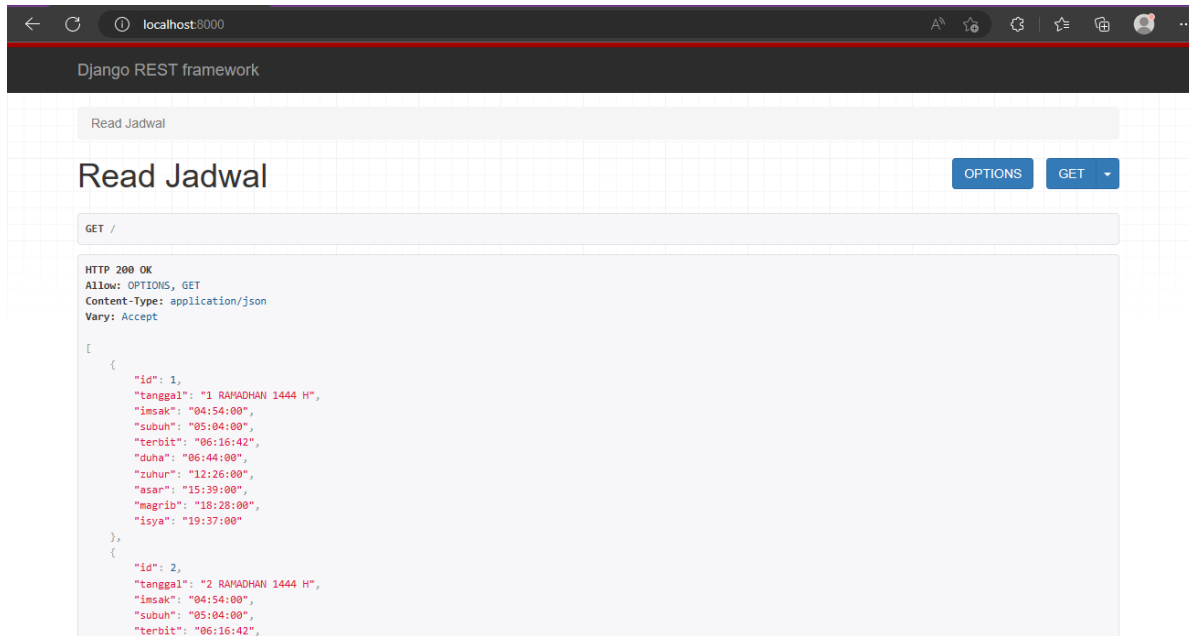
Decorator ini hanya dapat menerima request dengan method **HTTP DELETE**

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.

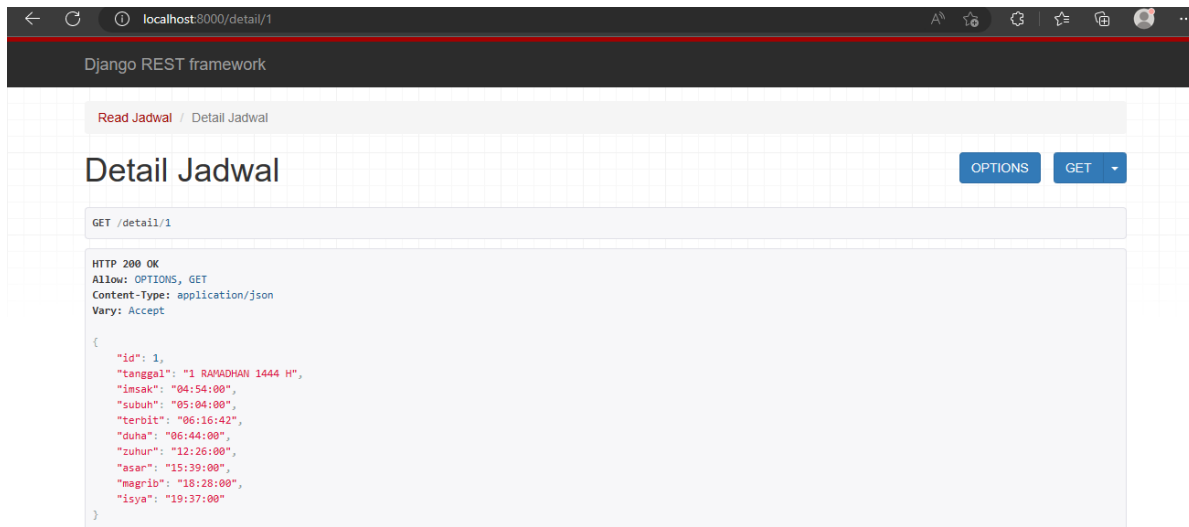
Setelah data berhasil dihapus, fungsi akan **mengembalikan response** dengan pesan *'Data berhasil di Hapus'*.

Respon API

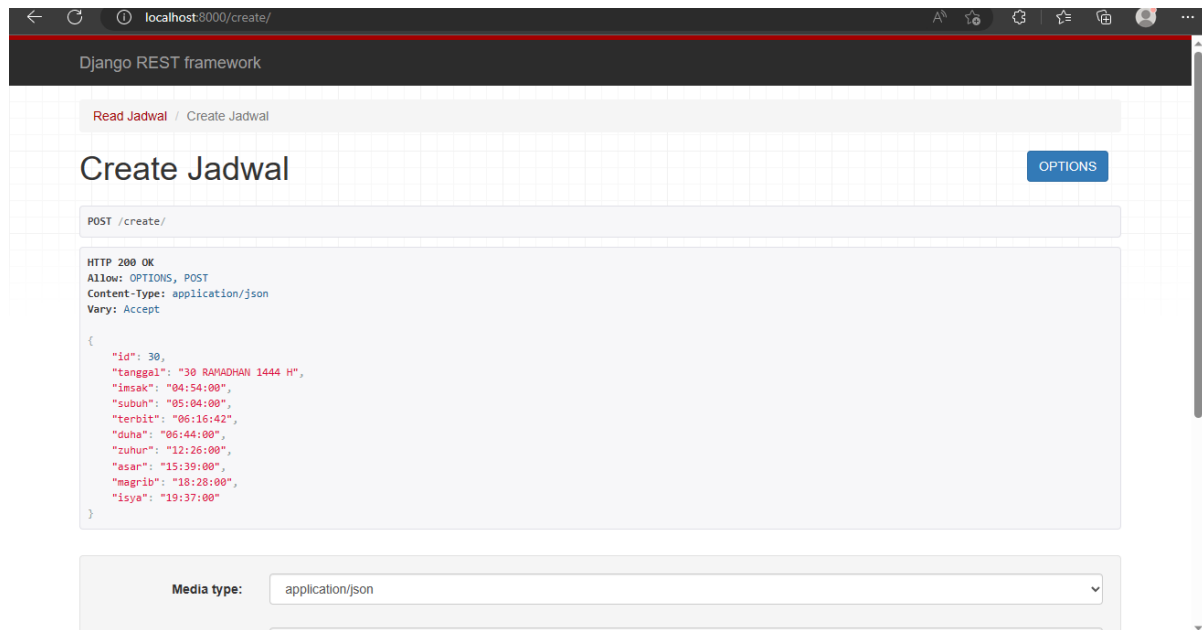
1. Melihat semua data



2. Melihat satu data atau data tertentu



3. Membuat data baru



Read Jadwal / Create Jadwal

Create Jadwal

OPTIONS

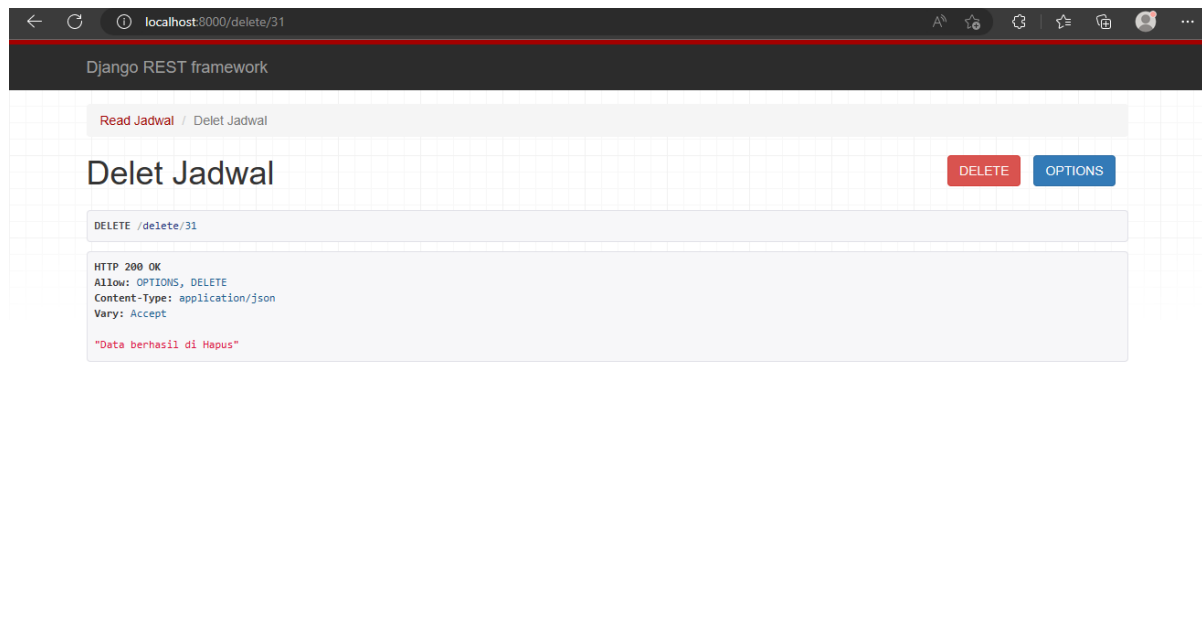
POST /create/

```
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

4. Menghapus data



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"}
```

5. Mengedit data yang sudah ada

←↻🔍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 RAMADHANssss 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