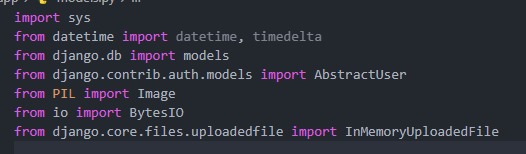
**KETIKKAN TOPIK ANDA**

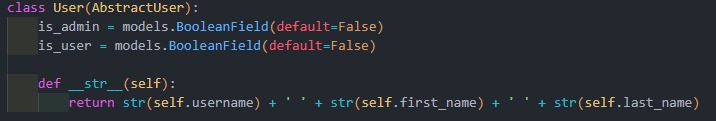
1. **REST API**
2. Framework yang digunakan : Django Rest Framework
3. Model
4. Library di models



modul sys diimpor untuk akses ke variabel dan fungsi sistem. datetime dan timedelta digunakan untuk menangani tanggal dan waktu. Library Django, models, digunakan untuk mendefinisikan model database, dan AbstractUser memungkinkan penyesuaian model pengguna.

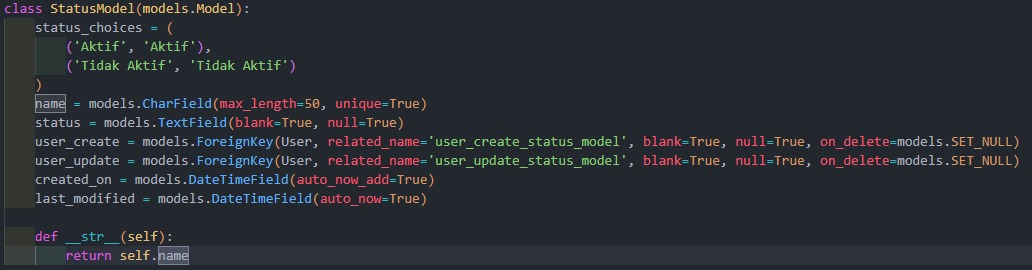
Selain itu, library Pillow (PIL) dengan modul Image digunakan untuk membuka dan memanipulasi gambar. BytesIO dari modul io mengelola stream byte dalam memori, berguna untuk operasi input/output yang tidak membutuhkan penulisan data ke disk. InMemoryUploadedFile dari Django mengelola file yang diunggah dalam memori, memungkinkan manipulasi gambar sebelum penyimpanan.

1. User



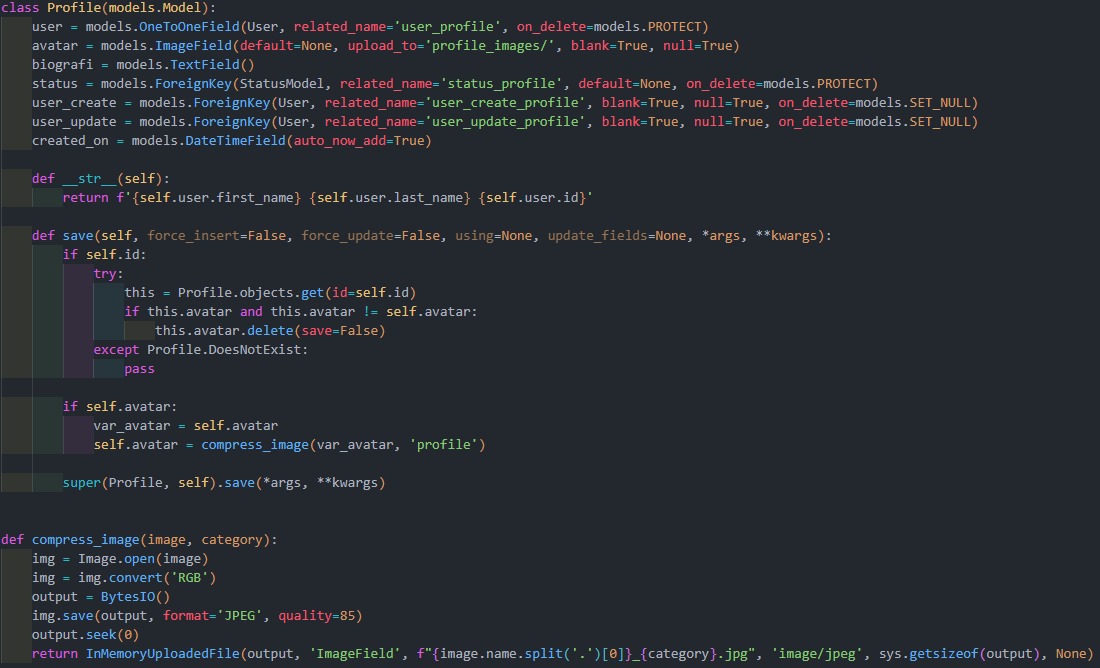
model User yang diperluas dari AbstractUser dalam aplikasi Django. Model ini menambahkan dua atribut tambahan: is\_admin dan is\_user, keduanya berupa BooleanField dengan nilai default False. Atribut ini digunakan untuk menentukan peran pengguna dalam aplikasi. Fungsi \_str\_ di-overwrite untuk mengembalikan representasi string dari objek pengguna, yang menggabungkan username, first\_name, dan last\_name. Ini memungkinkan pengenalan pengguna yang lebih mudah saat objek pengguna ditampilkan dalam antarmuka admin atau log.

1. Status Model



Model ini memiliki beberapa atribut: name yang berupa CharField dengan panjang maksimum 50 karakter dan harus unik, serta status yang berupa TextField yang opsional. Ada dua ForeignKey yang menghubungkan ke model User untuk mencatat pengguna yang membuat dan memperbarui status, dengan opsi SET\_NULL jika pengguna terkait dihapus. Selain itu, terdapat dua atribut waktu, created\_on dan last\_modified, yang otomatis diisi saat pembuatan dan pembaruan. Pilihan status status\_choices berisi dua nilai: 'Aktif' dan 'Tidak Aktif'. Metode \_str\_ mengembalikan nama status sebagai representasi string dari objek.Serta status model juga terkoneksi dengan class lain seperti kategori pulau,profile,user yang menandakan model itu aktif atau tidak

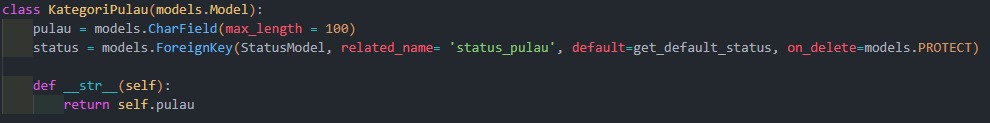
1. Profile



Model ini terhubung satu-satu dengan model User melalui OneToOneField, yang mengamankan profil pengguna agar tidak dapat dihapus dengan on\_delete=models.PROTECT. Model ini memiliki beberapa atribut tambahan seperti avatar untuk menyimpan gambar profil yang diunggah, biografi sebagai teks, dan status yang terhubung ke model StatusModel. Atribut user\_create dan user\_update mencatat pengguna yang membuat dan memperbarui profil, dengan on\_delete=models.SET\_NULL agar nilai menjadi null jika pengguna terkait dihapus. Tanggal pembuatan otomatis dicatat dengan created\_on.

Metode \_str\_ mengembalikan representasi string dari profil berupa nama depan, nama belakang, dan ID pengguna. Metode save diperbarui untuk menghapus avatar lama saat diperbarui dan memanggil fungsi compress\_image untuk mengompres gambar sebelum menyimpannya. Fungsi compress\_image mengompres gambar ke format JPEG dengan kualitas 85%, mengurangi ukuran file sebelum menyimpannya ke dalam memori.

1. Kategori Pulau



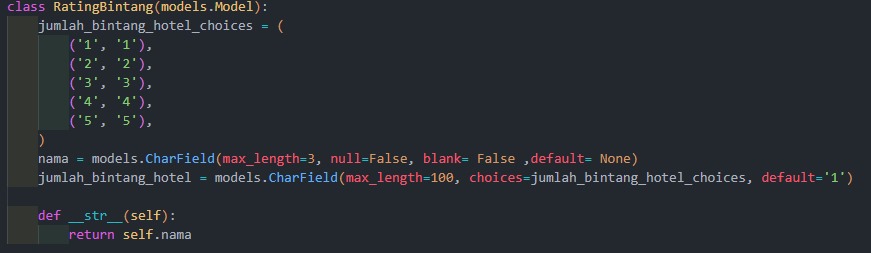
Model ini memiliki dua atribut: pulau, berupa CharField dengan panjang maksimum 100 karakter, dan status, berupa ForeignKey yang terhubung ke model StatusModel. Atribut status menggunakan fungsi get\_default\_status untuk menentukan status default, dan dilindungi dari penghapusan dengan on\_delete=models.PROTECT.

1. Table List Hotel



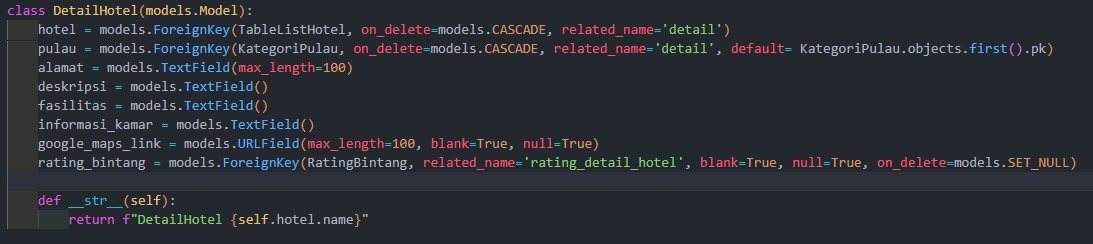
Model ini mencakup beberapa atribut: code berupa CharField dengan panjang maksimum 100 karakter, dan name berupa TextField dengan panjang maksimum 100 karakter. Atribut status menggunakan pilihan status\_choices yang bisa berupa 'Aktif' atau 'Tidak Aktif', dengan default 'Aktif'. Dua atribut user\_create dan user\_update mencatat pengguna yang membuat dan memperbarui entri, dengan on\_delete=models.SET\_NULL jika pengguna terkait dihapus. Tanggal pembuatan otomatis dicatat dengan created\_on, dan tanggal modifikasi terakhir dicatat dengan last\_modified.

1. Table RatingBintang



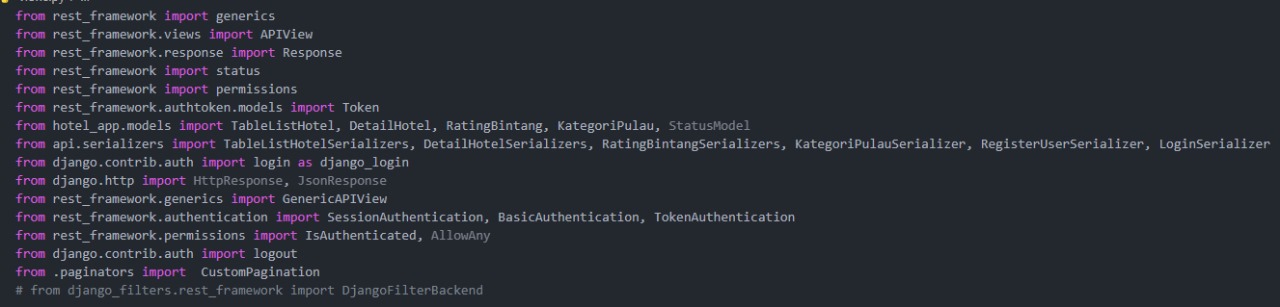
Model ini memiliki dua atribut utama: nama, berupa CharField dengan panjang maksimum 3 karakter dan tidak boleh kosong, serta jumlah\_bintang\_hotel, berupa CharField dengan pilihan jumlah\_bintang\_hotel\_choices yang bisa bernilai dari '1' hingga '5', dengan default '1'.

1. Table Detail Hotel



model DetailHotel ini digunakan untuk menyimpan detail dari suatu hotel, seperti nama hotel, pulau tempat hotel tersebut berada, alamat hotel, deskripsi hotel, fasilitas yang disediakan, informasi tentang kamar-kamar yang tersedia, serta tautan Google Maps jika ada.

1. Controller
2. Library di controller



Import Modul: Kode dimulai dengan mengimpor beberapa modul yang diperlukan dari Django REST Framework dan juga beberapa modul khusus dari aplikasi hotel\_app.

Deklarasi Class Views: Kita mendefinisikan beberapa class untuk views menggunakan Django REST Framework:

generics: Digunakan untuk membuat views generik seperti ListCreateAPIView.

APIView: Digunakan untuk membuat views berbasis fungsi khusus yang lebih fleksibel.

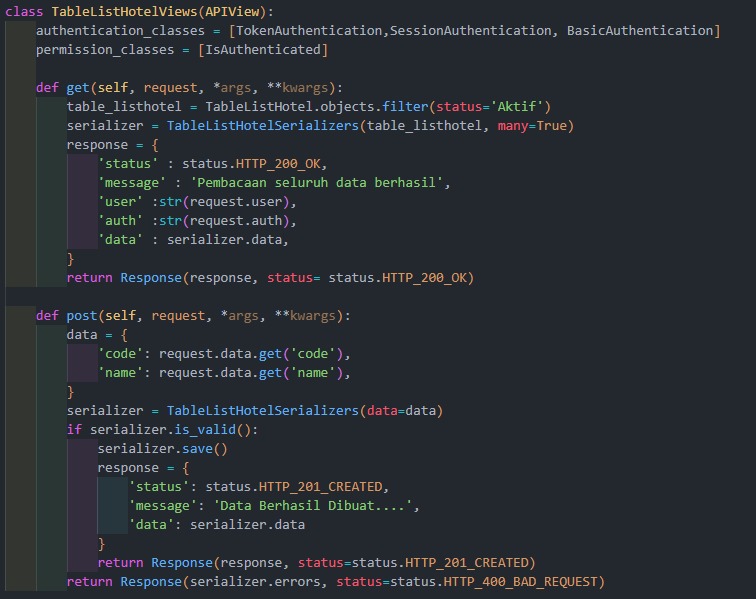
Import Model dan Serializer: Model dan serializer dari aplikasi hotel\_app diimpor untuk digunakan dalam views.

Import dan Penggunaan Autentikasi dan Perizinan: Kita mengimpor dan menggunakan berbagai jenis autentikasi seperti SessionAuthentication, BasicAuthentication, dan TokenAuthentication serta perizinan seperti IsAuthenticated dan AllowAny.

Login dan Register User: Terdapat implementasi untuk fungsi login dan register user. Fungsi django\_login digunakan untuk masuk (login) pengguna dan juga terdapat serialisasi untuk registrasi pengguna baru.

Paginasi Kustom: Digunakan untuk membuat paginasi kustom dengan mengimpor CustomPagination dari file paginators.

1. Table List Hotel Views

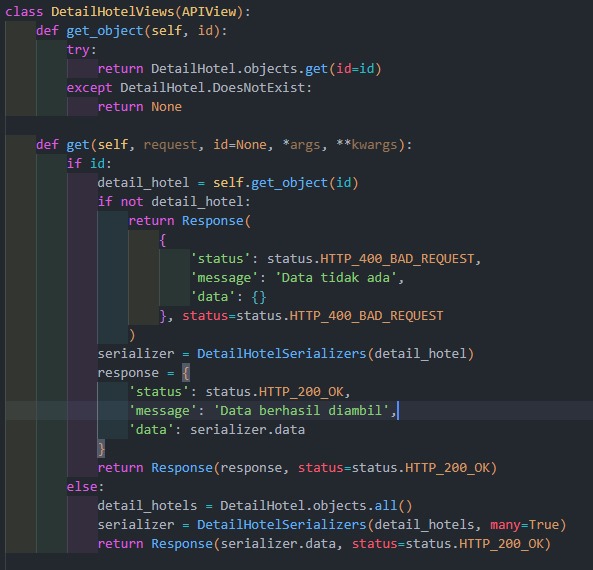


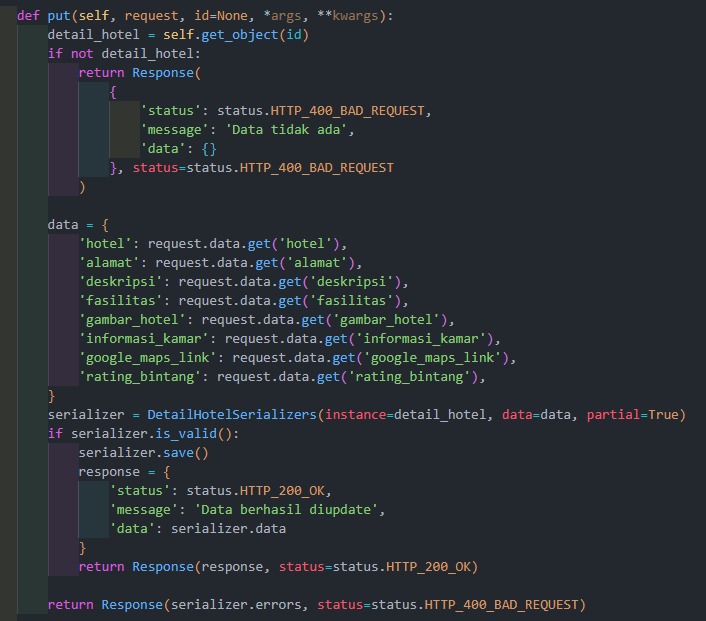
model TableListHotel. Di sini, kita menggunakan APIView dari Django REST Framework untuk mengelola permintaan HTTP. Hanya pengguna yang sudah terautentikasi (authenticated) yang bisa mengakses views ini, dan autentikasi dilakukan dengan token, session, atau basic authentication.

Ketika permintaan GET diterima, views akan mengambil semua objek TableListHotel yang memiliki status 'Aktif'. Data ini kemudian diserialisasikan dan dikirim sebagai respons bersama dengan pesan sukses dan status HTTP 200 OK.

Ketika permintaan POST diterima, views akan mencoba untuk membuat objek TableListHotel baru berdasarkan data yang diberikan dalam permintaan. Jika data valid, objek akan disimpan dan respons yang sesuai dengan pesan sukses dan status HTTP 201 CREATED akan dikirimkan. Jika data tidak valid, respons akan berisi pesan kesalahan dan status HTTP 400 BAD REQUEST.

1. Table Detail Hotel Views







model DetailHotel. Views ini mengimplementasikan operasi dasar CRUD (Create, Read, Update, Delete) untuk objek DetailHotel dalam aplikasi.

Ketika permintaan GET diterima, views akan mencari objek DetailHotel berdasarkan ID yang diberikan. Jika ID ditemukan, data akan diserialisasikan dan dikirim sebagai respons dengan pesan sukses dan status HTTP 200 OK. Jika ID tidak ditemukan, respons akan berisi pesan kesalahan dan status HTTP 400 BAD REQUEST. Jika ID tidak disediakan, views akan mengambil semua objek DetailHotel, men-serialize-nya, dan mengirimkan sebagai respons.

Ketika permintaan PUT diterima, views akan mencoba untuk memperbarui objek DetailHotel yang sesuai dengan ID yang diberikan. Jika objek ditemukan, data akan diperbarui berdasarkan data yang diterima dalam permintaan. Jika data valid, objek akan disimpan dan respons yang sesuai dengan pesan sukses dan status HTTP 200 OK akan dikirimkan. Jika tidak valid, respons akan berisi pesan kesalahan dan status HTTP 400 BAD REQUEST.

Ketika permintaan DELETE diterima, views akan mencoba untuk menghapus objek DetailHotel yang sesuai dengan ID yang diberikan. Jika objek ditemukan, objek akan dihapus dan respons yang sesuai dengan pesan sukses dan status HTTP 200 OK akan dikirimkan. Jika ID tidak ditemukan, respons akan berisi pesan kesalahan dan status HTTP 400 BAD REQUEST.

1. Rating Bintang Views

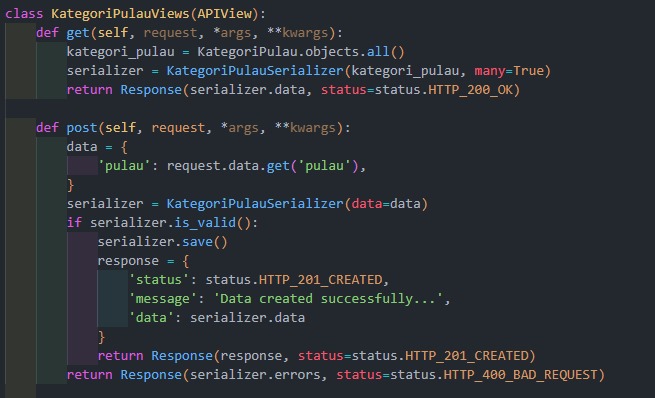


model RatingBintang. Views ini memungkinkan pengguna untuk melihat semua data rating bintang yang tersedia dan juga membuat rating baru untuk hotel.

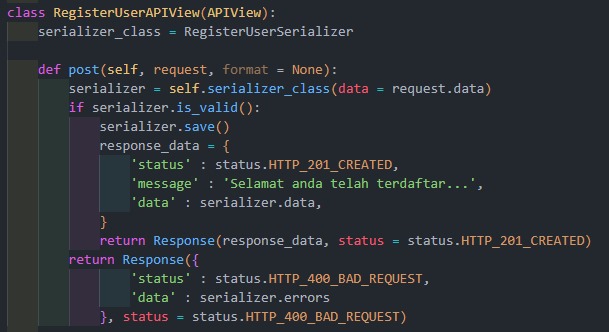
Ketika permintaan GET diterima, views akan mengambil semua objek RatingBintang, men-serialize-nya, dan mengirimkannya sebagai respons dengan status HTTP 200 OK.

Ketika permintaan POST diterima, views akan mencoba membuat objek RatingBintang baru berdasarkan data yang diberikan dalam permintaan. Data tersebut meliputi hotel yang dirating, jumlah bintang, dan ulasan. Jika data valid, objek akan disimpan dan respons yang sesuai dengan pesan sukses dan status HTTP 201 CREATED akan dikirimkan. Jika tidak valid, respons akan berisi pesan kesalahan dan status HTTP 400 BAD REQUEST.

1. Kategori Pulau Views



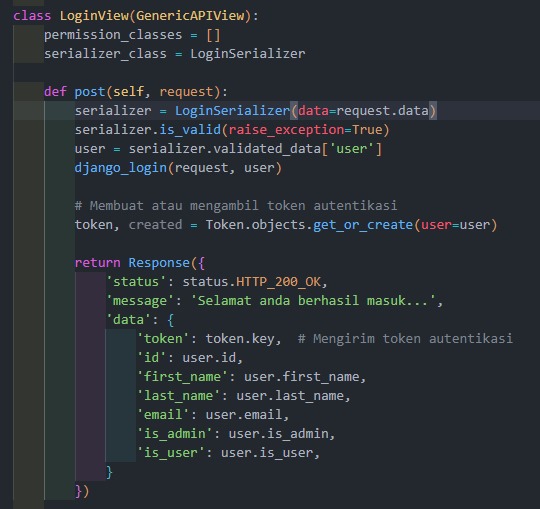
1. Register User View



Kode ini adalah bagian dari sebuah API yang digunakan untuk mendaftarkan pengguna baru. Kelas RegisterUserAPIView memiliki metode post yang menangani permintaan POST untuk pendaftaran. Ketika ada permintaan pendaftaran, data yang dikirimkan oleh pengguna akan diambil dan diperiksa apakah valid atau tidak.

Jika data tersebut valid, data pengguna akan disimpan dan API akan mengembalikan respons dengan status HTTP 201 Created, bersama dengan pesan "Selamat anda telah terdaftar..." dan data pengguna tersebut. Namun, jika data yang dikirimkan tidak valid, API akan mengembalikan respons dengan status HTTP 400 Bad Request serta pesan kesalahan yang terjadi.

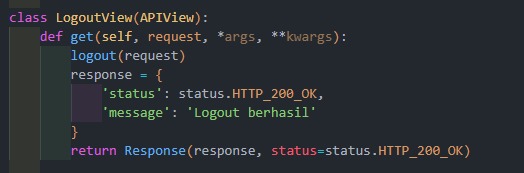
1. LoginView



Views ini menggunakan LoginSerializer untuk memvalidasi data yang diterima dari permintaan POST.

Ketika permintaan POST diterima, views akan mencoba untuk memvalidasi data yang diberikan. Jika data valid, pengguna akan diautentikasi dan token autentikasi akan dibuat atau diambil dari database. Respons yang dikirimkan akan berisi pesan sukses, status HTTP 200 OK, dan informasi pengguna yang diautentikasi, termasuk token autentikasi yang akan digunakan untuk otorisasi dalam permintaan berikutnya.

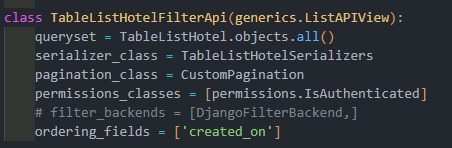
1. Logout View



ini adalah views untuk proses logout pengguna. Ketika permintaan GET logout diterima, views akan melakukan proses logout pengguna yang terkait dengan permintaan tersebut.

Setelah logout berhasil dilakukan, respons yang dikirimkan akan berisi pesan sukses dan status HTTP 200 OK

1. Table List Hotel Filter



ini adalah views untuk menyediakan endpoint API yang memungkinkan pengguna untuk melakukan filter pada model TableListHotel.

Views ini menggunakan ListAPIView dari Django REST Framework yang memungkinkan pengguna untuk melakukan operasi baca saja (GET) pada data TableListHotel.

Queryset default yang digunakan adalah semua objek TableListHotel. Serializer yang digunakan untuk menangani data adalah TableListHotelSerializers.

Selain itu, views juga menggunakan CustomPagination untuk membagi hasil data menjadi beberapa halaman.

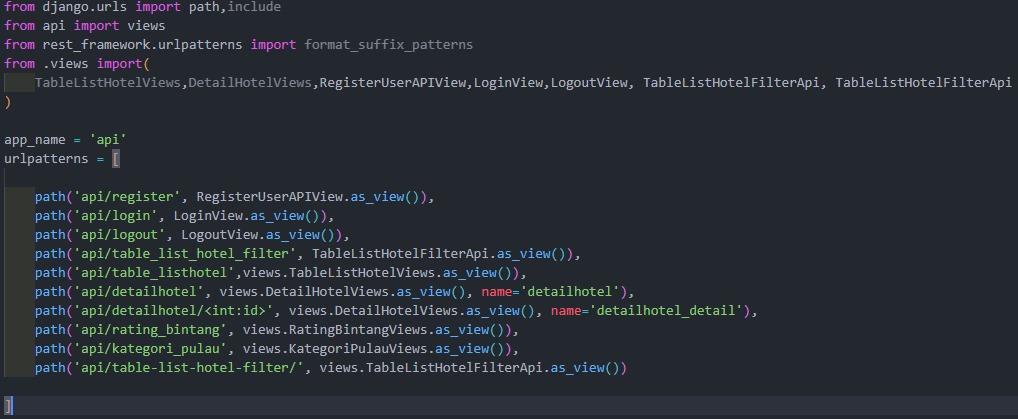
Hanya pengguna yang sudah terautentikasi yang bisa mengakses endpoint ini, yang diatur melalui permissions\_classes = [permissions.IsAuthenticated].

Pengguna juga dapat melakukan pengurutan hasil berdasarkan kolom created\_on, yang diatur menggunakan ordering\_fields = ['created\_on'].

1. Routing (url)

Screenshot dari *routing (url)* yang anda bangun dan uraikan narasi dari masing-masing *routing (url)* yang anda bangun

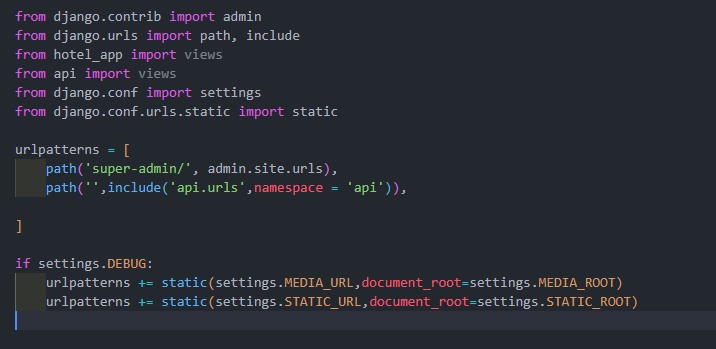
1. api/urls.py



Ini adalah file urls.py yang berisi konfigurasi URL untuk aplikasi API.

Setiap path dihubungkan dengan views yang sesuai menggunakan as\_view() untuk menghasilkan tampilan yang dapat diakses oleh Django.Pengguna dapat mendaftar (register), masuk (login), dan keluar (logout) dari akun mereka menggunakan endpoint yang disediakan.Selain itu, terdapat juga endpoint untuk mengakses data TableListHotel, DetailHotel, RatingBintang, dan KategoriPulau.Endpoint khusus juga disediakan untuk melakukan filter pada data TableListHotel.Semua URL ini terhubung dengan views yang relevan untuk mengakses atau memanipulasi data sesuai dengan permintaan pengguna.

1. Pos/urls.py



Ini adalah file urls.py yang merupakan titik awal untuk routing URL dalam aplikasi.Pertama, terdapat path untuk mengakses halaman admin Django pada endpoint super-admin/.

Kemudian, terdapat path kosong ('') yang mengarahkan ke file urls.py dalam aplikasi api, yang telah didefinisikan sebelumnya.Jika pengaturan DEBUG diaktifkan, juga ada konfigurasi untuk menyajikan file media dan statis selama pengembangan aplikasi.

Penggunaan include('api.urls', namespace='api') menunjukkan bahwa semua URL yang dimulai dengan api/ akan ditangani oleh file urls.py dalam aplikasi api dengan namespace api.Ini adalah konfigurasi dasar untuk mengarahkan permintaan pengguna ke views yang relevan dalam aplikasi, baik itu untuk mengelola data dari API atau halaman admin Django

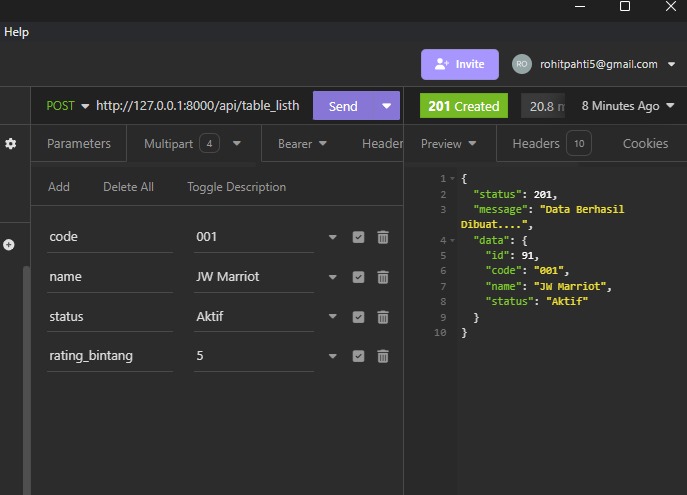
1. Middleware

Tidak ada Middleware

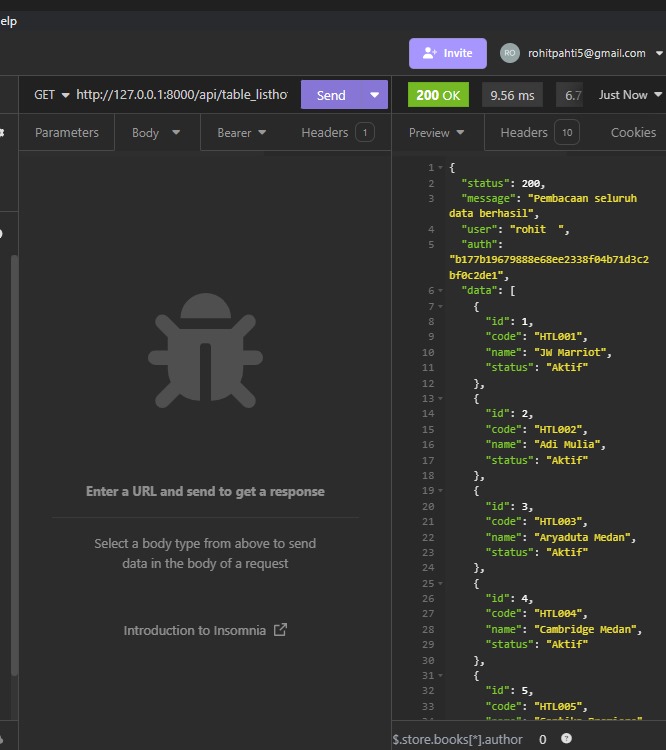
1. Input - Proses/Transaksi - Output REST API (service GET, POST, PUT dan DELETE)

Screenshot dari Input - Proses/Transaksi - Output REST API yang anda bangun dan uraikan narasi dari Input - Proses/Transaksi - Output REST API yang anda bangun. Lalu tampilkan hasil screenshot dari REST API yang dibangun menggunakan aplikasi Postman atau Insomnia *(pilih salah satu)*.

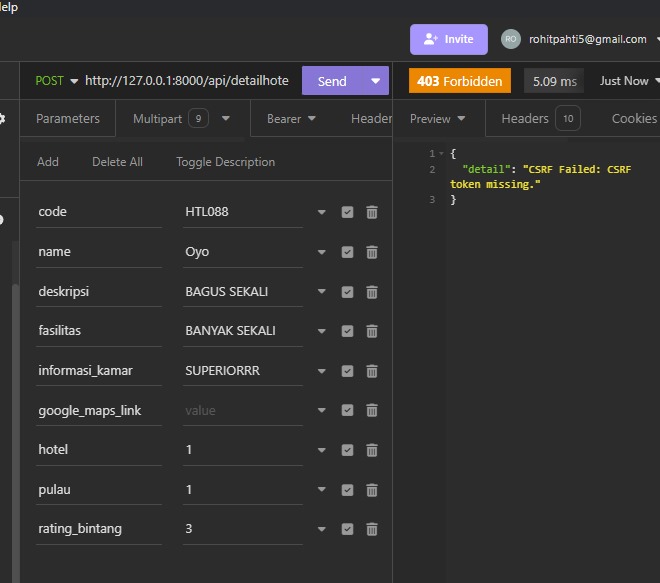
1. POST/TableListHotel



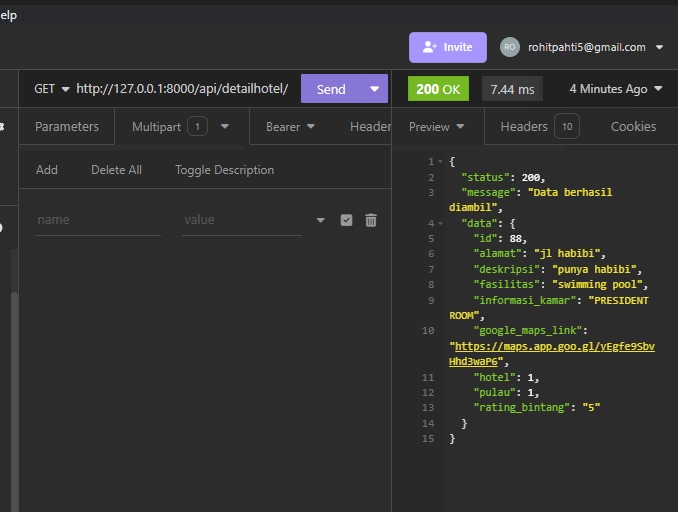
1. GET/TableListHotel



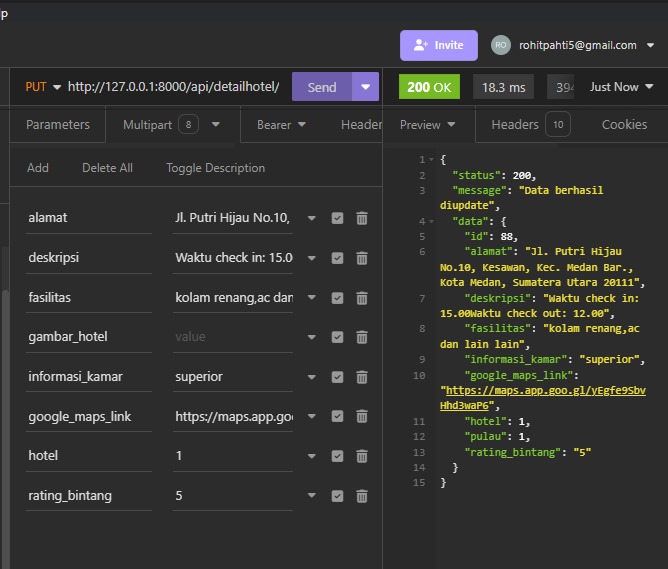
1. GET/Detail hotel



1. GET/Detail Hotel

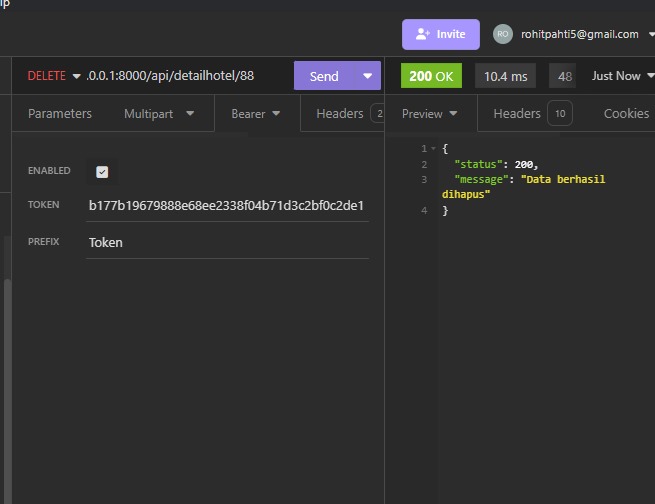


1. PUT/Detail Hotel

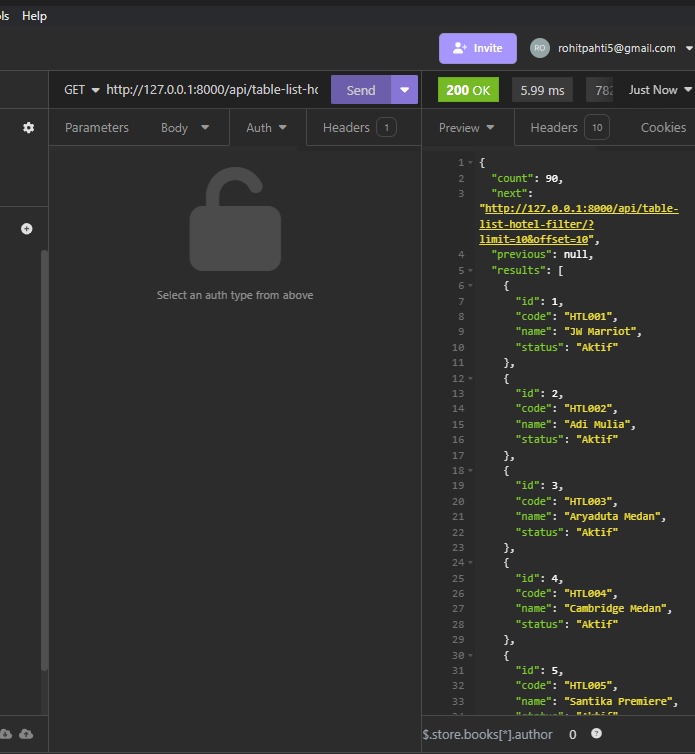


Catatan:

1. Delete/Detail Hotel



1. Pagination



* *REST API tidak harus 1 (satu) halaman.*
* *Point nomor f pada REST API disesuaikan dengan topik yang telah disetujui oleh Dosen Pengampu.*
* *Format penamaan file ini : NIM\_KETUA\_TIM\_NAMA\_KETUA\_TIM\_ NAMA\_MATA\_KULIAH\_KELAS, contoh : 2033030300007\_Andi\_Wijaya\_PM\_TI\_4\_PAGI\_A.*
* *Wajib mengikuti format penaaman file diatas untuk memudahkan proses penilaian.*