Documentation for API endpoints for Hotel Management Project

Django REST Framework based Project

ISSUED BY
Sheshant

Introduction & Background

This is an API endpoint documentation of the Hotel Management Project. The Project consists of only backend and no frontend. This Project has 4 major class

1. Hotel: It consists of all the data related to the hotels in the database
2. Rooms: It consists of details of the rooms in all the hotels, their price, and their occupancy status
3. Bookings: It includes details of all the bookings done by the user and also their cancellation status
4. User Profile: Every user has to create a profile before booking. This table stores its profile data.

The database used here is the SQLite default database for the Django Project. For the beginners, we have superuser username ‘sheshant’ and password ‘recursive’. Go to the project folder and run ./manage.py runserver and got to the admin panel by login as superuser and then using browser go to the URL **http://127.0.0.1:8000/admin** to see or modify all the objects of the classes present in the database. All the dependencies are given in the file requirement.txt

API endpoints

Following are the endpoints.
Note: Most of the responses are paginated with page size 10

1. **/api/management/search_hotel/ POST**

It will give all the results hotels based on the parameters provided in the input JSON data parameters. The input JSON parameters may have any of

1. “city”: string field having the name of the city
2. “state”: string field having the name of the state
3. “pincode”: string field having the pincode

Or even all of them or none and it will filter and return the list of all records based on the input params. It will return a paginated response of Hotels their ID, names, and address

2. **/api/management/search_room/ POST**

It will give all the results hotels based on the parameters provided in the input JSON data parameters. The input JSON parameters may have any of

1. “upper_price”: integer field having the upper range of price
2. “lower_price”: integer field having the lower range of price
3. “type”: any one of them ('single_occupancy', 'double_occupancy', ‘tripple_occupancy'))
4. “hotel_id”: integer field having the hotel id

Or even all of them or none and it will filter and return the list of all It will return the list of all the vacant rooms and their hotels

3. **/api/management/get_all_bookings/ POST**

It will return the list of all the bookings based on the filtered and sorting criteria. It will also return a paginated response of all the bookings. The input params for this API are.

1. “hotel_id”: integer field having the hotel id
2. “room_id”: integer field having the room id
3. “user_id”: integer field having the user id for the booking
4. ‘date_of_booking’: the date when booking was done (DateTime field of format 'YYYY-MM-DD HH:MM:SS')
5. ‘date_of_stay’: the date of the start of the stay schedule (DateTime field of the format 'YYYY-MM-DD HH:MM:SS')
6. “status”: integer field whether canceled or not (only valid types are ‘cancelled’, ‘not_cancelled’)
7. ‘sort_by’: it is having a list of dictionaries. Each dictionary having 2 keys “field” and “order”. “field” must be one of the fields of model booking and “order” must be 0 for ascending and 1 for descending

If any of them is missing then the option will not be filtered or even sorted

4. **/api/management/book_room/ POST**

This API will be used to book a room. It requires the params

1. “hotel_id”: integer field having the hotel id
2. “room_id”: integer field having the room id
3. “user_id”: integer field having the user id for the booking
4. ‘date_of_stay’: the date of the start of the stay schedule (DateTime field of the format 'YYYY-MM-DD HH:MM:SS')
5. ‘last_date_of_stay’: the date of the end of the stay schedule (DateTime field of the format 'YYYY-MM-DD HH:MM:SS')

If any of the corresponding params are missing or improper then it will return an error. It will return the booking details if successfully booked

5. **/api/management/cancel_booking/ POST**

It will require only the ‘booking_id’ and cancel the booking. If improper booking_id provided then it will return corresponding error message else success message.

6. /api/management/checkin/ POST

It will require only the ‘booking_id’ and it will check-in the user to the room. A success message will be returned if successfully checked-in else corresponding error message

7. /api/management/checkout/ POST

It will require only the ‘booking_id’ and it will check-out the user to the room. A success message will be returned if successfully checked-out else corresponding error message