Assignment Title

Hotel Room Booking API using FastAPI

Objective

Create a REST API that allows users to view hotel rooms and make simple bookings. This project simulates a small-scale room reservation system, focusing on API design and handling user input properly.

Requirements

Implement the following endpoints:

- **GET /rooms**
  - Return a list of available hotel rooms.
- **GET /rooms/{room_id}**
  - Return details for a specific room.
- **POST /bookings**
  - Book a room by providing room ID, guest name, and number of nights.
  - Example request body:

    ```
    {
     "room_id": 101,
     "guest_name": "John Doe",
     "nights": 3
    }
    ```

- **GET /bookings**
  List all current bookings.
- **DELETE /bookings/{booking_id}**
  Cancel a booking.

Data Models

Room

```
from pydantic import BaseModel

class Room(BaseModel):
        id: int
```

```
    type: str # e.g. "Single", "Double", "Suite"
    price_per_night: float
    available: bool
```

Booking

```
class Booking(BaseModel):
  id: int
  room_id: int
  guest_name: str
  nights: int
  total_price: float  # Calculated based on room price
```

💡 Logic Hints

- When a room is booked, mark it as available: False.
- total_price = price_per_night * nights
- Store data in Python lists/dictionaries (in-memory), no DB needed.

⚙️ Bonus (Optional)

- Validate that a room exists and is available before booking.
- Allow filtering of rooms by type (GET /rooms?type=Suite)
- Add a booking date range (use datetime) and prevent overlapping bookings.

🗂️ Suggested Folder Structure

```
hotel_api/
├── main.py      # FastAPI app
├── models.py    # Pydantic schemas
├── data.py      # Dummy in-memory data
└── requirements.txt
```

📤 Submission Guidelines for Hotel Booking API Assignment

Once your assignment is complete, please submit the following via email:

☑ What to Submit

1. **Project Code** (Zipped Folder or GitHub Link)
    - All source code including:

      ```
      hotel_api/
      ├── main.py
      ├── models.py
      ├── data.py
      └── requirements.txt
      ```

  - Make sure your code is clean, well-commented, and runs without errors.

README.md (or include a README in the email body)

- Brief explanation of the project
- Instructions on how to run the app:

  ```
  pip install -r requirements.txt
  uvicorn main:app –reload
  ```

- Mention any assumptions or extra features you added

2. **Screenshots or Postman Collection (Optional but Encouraged)**
    - Screenshots of API running locally
    - Or a Postman collection with sample requests
3. **(Optional) GitHub Repo Link**
    - If you're comfortable using Git, feel free to push your project to GitHub and share the link