

Models

RoleModel

Field Name	Datatype	Primary Key	Comments
id	Integer	Yes	Autoincrement
rolename	String	No	Values: "CUSTOMER", "MANAGER"

UserModel

Field Name	Datatype	Primary Key	Foreign Key	Comments
id	Integer	Yes	No	Autoincrement
username	String	No	No	-
email	String	No	No	Unique
password	String	No	No	-
role	Integer	No	Yes	FK to RoleModel

BookingModel

Field Name	Datatype	Primary Key	Foreign Key	Comments
id	Integer	Yes	No	Autoincrement
customerId	Integer	No	Yes	FK to UserModel
roomType	String	No	No	Values: "Deluxe", "Suite", "Standard"
nights	Integer	No	No	-
basePrice	Double	No	No	Price per night based on roomType
discount	Double	No	No	Default: 0.0
totalPrice	Double	No	No	Calculated: (nights * basePrice) - discount
bookingDate	String	No	No	Default: Current date (dd/MM/yyyy)
status	String	No	No	Default: "pending"

JWT Authentication

- JWT token required in Authorization header:

,

Authorization: Bearer <JWT_TOKEN>

,

- Roles (CUSTOMER / MANAGER) are identified from the token.

- All endpoints except /login require authentication.

Endpoints

1. POST /login

- Purpose: Authenticate user and return JWT.

- Request Body:

```
`json
{
  "email": "cust1@gmail.com",
  "password": "cust123$"
}
```

- Success Response (200 OK):

```
`json
{
  "email": "cust1@gmail.com",
  "access": "<access_token>",
  "refresh": "<refresh_token>",
  "status": 200
}
```

2. POST /booking/create (CUSTOMER only)

- Creates a new booking.

- Request Body:

```
`json
{
  "roomType": "Deluxe",
  "nights": 3,
  "discount": 500
}
```

- Manipulation:

- basePrice auto-set based on roomType:

- Deluxe → 4000/night

- Suite → 7000/night

- Standard → 2500/night

- totalPrice = (nights * basePrice) - discount

- Success Response (201 CREATED):

```
`json
{
  "id": 9001,
  "customerId": 1,
  "roomType": "Deluxe",
  "nights": 3,
  "basePrice": 4000,
  "discount": 500,
```

```
"totalPrice": 11500,  
"bookingDate": "29/11/2025",  
"status": "pending"  
}  
,
```

3. GET /booking/list?roomType=Suite&minNights=2&maxPrice=20000 (Both roles)

- Returns bookings filtered by:
 - roomType
 - minNights (bookings with nights \geq given value)
 - maxPrice (bookings with totalPrice \leq given value)
- Success: 200 OK with list.
- Error: 400 Bad Request \rightarrow "no bookings found"

4. PATCH /booking/update/{id} (MANAGER only)

- Updates booking status.
- Request Body:

```
`json  
{ "status": "confirmed" }
```
- Success Response (200 OK): Updated booking object.
- Error: 400 Bad Request \rightarrow "booking not found"

5. DELETE /booking/delete/{id} (Conditional access)

- Accessible by:
 - Manager
 - Customer who created the booking
- Success: 204 No Content \rightarrow "deleted successfully"
- Error:
 - 400 Bad Request \rightarrow "booking not found"
 - 403 Forbidden \rightarrow "you don't have permission"



Response Codes

- 200 OK – Success
- 201 CREATED – Resource created
- 204 No Content – Deleted successfully
- 400 Bad Request – Invalid input / Not found
- 403 Forbidden – Permission denied

- ✅ This new problem statement adds extra manipulations:
- Auto-set basePrice based on roomType.
 - Calculate totalPrice using $(\text{nights} * \text{basePrice}) - \text{discount}$.
 - Query params with three filters: roomType, minNights, maxPrice.
