**Project Development Phase**

**Model Performance Test**

|  |  |
| --- | --- |
| Date | 16 July 2025 |
| Team ID | LTVIP2025TMID59690 |
| Project Name | HouseHunt – Rental Property Recommender |
| Maximum Marks | 2 Marks |

**RESTful API Design:**

**1. Introduction to RESTful API Design:**

REST (Representational State Transfer) is an architectural style for designing networked applications. RESTful APIs use HTTP methods (GET, POST, PUT, DELETE) to access and manipulate resources (data) through stateless communication.

In MERN applications, RESTful API design plays a critical role in enabling smooth communication between the frontend (React.js) and backend (Node.js/Express).

**2. Why RESTful API Design Was Used:**

RESTful API principles were implemented in this project for the following reasons:

* Ensures **standardized communication** between client and server.
* Uses **HTTP methods** for clearly defined operations.
* Provides **scalable, maintainable** endpoints.
* Separates concerns: frontend handles UI, backend handles logic.
* Enables **role-based access control** through protected routes.

**3. How RESTful APIs Were Used in the Project:**

RESTful APIs were designed for:

* 🔹 **User Authentication**
  + /api/users/register (POST)
  + /api/users/login (POST)
* 🔹 **Property Management**
  + /api/properties/create (POST)
  + /api/properties/list (GET)
  + /api/properties/:id (GET, PUT, DELETE)
* 🔹 **Bookings & Admin Controls**
  + /api/bookings/ (GET, POST)
  + /api/admin/approve/:id (PUT)

APIs returned standardized **JSON responses** with appropriate status codes (200, 201, 401, 404, 500, etc.).

**4. Sample API Route Snippet:**

// Create Property - POST /api/properties/create

router.post('/create', authMiddleware, async (req, res) => {

try {

const property = new Property(req.body);

await property.save();

res.status(201).json({ message: 'Property created successfully', property });

} catch (error) {

res.status(500).json({ message: 'Error creating property', error });

}

});

**5. Conclusion:**

RESTful API design provided a clean and organized method for managing data flow between the frontend and backend in this MERN stack project. The consistent API structure allowed developers to scale and debug efficiently while maintaining clear separation of concerns.