**MongoDB Data Modelling:** PG Made Eazy

**What Is Data Modelling?**

**Data modelling** is the process of organizing and structuring data in a way that makes it efficient to store, retrieve, and manage within a database system. In the context of MongoDB, which is a NoSQL document-oriented database, data modelling involves designing collections (similar to tables in relational databases) and deciding how data will be stored—either embedded within documents or referenced across collections. A well-structured data model ensures data consistency, reduces redundancy, and improves the performance of queries and updates.

For example, in the PG Made Eazy application, data modelling helps define how users (seekers or hosts), properties, bookings, reviews, and reports relate to each other. A host can list multiple properties, so a one-to-many relationship is modeled by referencing the host's ID in the properties collection. Similarly, a user can book multiple PG accommodations, leave reviews, and interact with the platform—each action mapped clearly in the database using references or embedded documents as needed. This structured approach enables the application to efficiently manage real-world operations like booking PGs, tracking user activity, and generating reports

**1. Identify Collections**

| Collection | Purpose |
| --- | --- |
| users | Stores user details including seekers and hosts |
| properties | Details of PG listings posted by hosts |
| bookings | Information about booking transactions between users and properties |
| reviews | User feedback and ratings for PGs |
| reports | Administrative analytics and activity tracking reports |

**2. Define Relationships**

| **Relationship** | **Type** | **Field Reference** | **Modelling Approach** | **Justification** |
| --- | --- | --- | --- | --- |
| Users → Properties | One-to-Many | properties.host\_id → users.\_id | Reference | Keeps listings lightweight |
| Users → Bookings | One-to-Many | bookings.user\_id → users.\_id | Reference | Tracks multiple bookings by a user |
| Properties → Bookings | One-to-Many | bookings.property\_id → properties.\_id | Reference | Efficient booking access |
| Users → Reviews | One-to-Many | reviews.user\_id → users.\_id | Reference | Tracks reviews per user |
| Properties → Reviews | One-to-Many | reviews.property\_id → properties.\_id | Reference | Associates reviews with PGs |
| Admins → Reports | One-to-Many | reports.admin\_id → users.\_id | Reference | Keeps analytics tied to admins |

**3. Embed vs Reference**

| **Use Case** | **Recommended Modeling** |
| --- | --- |
| **User profile with list of previous ratings** | **Embed ratings[]** |
| **Reviews posted by users for PG properties** | **Reference user\_id and property\_id** |
| **Properties listed by different hosts** | **Reference host\_id** |
| **Booking history of users** | **Reference user\_id and property\_id** |
| **Amenities available in a PG listing** | **Embed amenities[]** |

**4. Sample Schemas**

**a.users**

**b. properties**

**b. properties**



**c. bookings**



**d. reviews**



**e. reports**