**USE CASE DIAGRAM**

Use case diagrams provide a high-level view of the system, showing the interactions between users (actors) and the system itself. They help in understanding the system’s functionality and the requirements from the user’s perspective.

**Components**:

* **Actors**: Represent the users or other systems that interact with your Hostel Hub. Common actors might include:
  + **Admin**: Manages the hostel, including room assignments, payments, and maintenance.
  + **Student**: Books rooms, makes payments, and requests services.
  + **Custodian**: Handles maintenance requests and other operational tasks.
* **Use Cases**: Represent the actions or services provided by the system. Examples for a Hostel Hub might include:
  + **Manage Rooms**: Admin can add, update, or delete room information.
  + **Book Room**: Students can search for and book available rooms.
  + **Chat wit custodians**: Students can chat with custodians for their bookings.
  + **Request Maintenance**: Students can request maintenance services.
* **Relationships**: Show how actors and use cases interact. Common relationships include:
  + **Association**: A line connecting an actor to a use case, indicating interaction.
  + **Include**: A dotted arrow from one use case to another, indicating that the second use case is always included in the first.
  + **Extend**: A dotted arrow from one use case to another, indicating that the second use case is sometimes included in the first.

**SEQUENCE DIAGRAMS**

Sequence diagrams detail how objects interact in a particular scenario of a use case. They show the sequence of messages exchanged between objects to carry out a function.

**Components**:

* **Lifelines**: Represent the different objects or actors involved in the interaction. For your Hostel Hub, lifelines might include:
  + **Student**
  + **Booking System**
  + **Chatting page**
  + **Admin**
* **Messages**: Arrows between lifelines that represent the communication between objects. Types of messages include:
  + **Synchronous**: A solid arrow indicating a call that waits for a response.
  + **Asynchronous**: A solid arrow with an open arrowhead indicating a call that does not wait for a response.
  + **Return**: A dashed arrow indicating the return of control to the caller.
* **Activation Bars**: Thin rectangles on lifelines that show when an object is active or performing an action.

**Example Scenario**: Booking a Room

1. **Student** sends a **search request** to the **Booking System**.
2. **Booking System** returns the **available rooms**.
3. **Student** selects a room and sends a **booking request**.
4. **Booking System** enable a student to **chat** with the Custodian.
5. **The Custodia** processes the room and returns a **confirmation**.
6. **Booking System** updates the booking status and sends a **confirmation** to the **student**.
7. **Booking System** notifies the **Admin** of the new booking.