**UML Assignment for The Use case, Sequence and Class diagrams.**

1. A **Use Case Diagram** visually represents the interactions between users (actors) and the system (use cases). It captures the functional requirements of the system and shows what the system is expected to do.

**Components of the use case include**:

**Actors**: Represent users or other systems that interact with the application.

**Use Cases**: Actions or services the system performs (functionalities).

**System Boundary**: Defines the scope of the system.

**Components of the project (BrokerUG a mobile app for house rentals) include**:

Actors

House Owner

Tenant

Administrator

Payment Processor (external system)

Some of the Use Cases for the project above include:

Listing a property.

Searching for rental houses.

Viewing property details.

Booking a rental.

Making payments.

Managing rental agreement.

2. A **Sequence Diagram** models the interaction between objects in a system in a time sequence. It shows how processes operate with each other and in what order.

**Components**:

**Objects/Actors**: Represent system components or users interacting with the system.

**Lifelines**: Vertical dashed lines representing the lifetime of the object during interaction.

**Messages**: Horizontal arrows showing communication between objects.

**Activation Bars**: Indicate when an object is active in a process.

**Conditions/Loops**: Represent conditional flows or loops within interactions.

**As used in the Project (BrokerUG)**:

**House Rental Process** in a Sequence Diagram:

Objects include:

Tenant, House Listing, Booking System, Payment Gateway

Process:

Tenant searches for available houses.

Tenant selects a property.

Tenant initiates the booking process.

Booking system confirms the reservation.

Payment gateway processes the transaction.

3. A **Class Diagram** is a static structure diagram in UML that shows the system's classes, their attributes, methods, and relationships. It represents the blueprint of the system

.

**Components**:

**Classes**: Represent the main entities in the system. Each class has:

**Attributes**: Properties or data fields (e.g., name, ID, price).

**Methods**: Functions or behaviours (e.g., login, book Property).