**Project Synopsis-**

**Real Estate Property Management System**

# Introduction:

The Real Estate Management System is designed to facilitate property listings, inquiries, transactions, and user interactions related to real estate properties. It allows agents to list properties, buyers to make inquiries and favorite properties, and administrators to manage users and reports. The system is built using Maven, MySQL, and JPA Hibernate, ensuring scalability and efficient data management.

# Technologies Used:

- Programming Language: Java  
- Frameworks: JPA with Hibernate (ORM)  
- Database: MySQL  
- Build Tool: Maven  
- Architecture: DAO, Service Layer, and Entity Design

# Modules Overview:

## 1. Users:

Attributes: UserID (Primary Key), Username, Password, FullName, Email, Phone, Role (Admin/Agent/Buyer), CreatedAt, UpdatedAt.  
Relationships:  
- One-to-Many with Properties: A user (typically an agent) can list multiple properties.  
- One-to-Many with Favorites: Users can favorite multiple properties.  
- One-to-Many with Inquiries: Users can make multiple inquiries.  
- One-to-Many with Reports: Admins can generate reports.  
- One-to-Many with Transactions: A user can be a buyer or seller in multiple transactions.

## 2. Properties:

Attributes: PropertyID (Primary Key), Title, Description, Address, City, State, ZipCode, Price, PropertyType (Apartment, House, etc.), Status (Available/Sold), ListedBy (Foreign Key referencing UserID), CreatedAt, UpdatedAt.  
Relationships:  
- Many-to-One with Users (agents list properties).  
- One-to-Many with PropertyImages: A property can have multiple images.  
- One-to-Many with Favorites: Multiple users can favorite a property.  
- One-to-Many with Inquiries: Properties can receive multiple inquiries.  
- One-to-Many with Transactions: A property can be involved in multiple transactions.

## 3. PropertyImages:

Attributes: ImageID (Primary Key), PropertyID (Foreign Key), ImageURL.  
Relationships:  
- Many-to-One with Properties: A property can have multiple images.

## 4. Favorites:

Attributes: FavoriteID (Primary Key), UserID (Foreign Key), PropertyID (Foreign Key), CreatedAt.  
Relationships:  
- Many-to-One with Users: A user can have many favorites.  
- Many-to-One with Properties: A property can be favorited by many users.

## 5. Inquiries:

Attributes: InquiryID (Primary Key), UserID (Foreign Key), PropertyID (Foreign Key), Message, Status, CreatedAt.  
Relationships:  
- Many-to-One with Users: A user can make many inquiries.  
- Many-to-One with Properties: Properties can receive multiple inquiries.

## 6. Reports:

Attributes: ReportID (Primary Key), ReportType (Sales, Listings, etc.), Description, GeneratedBy (Foreign Key referencing UserID), CreatedAt.  
Relationships:  
- Many-to-One with Users: Reports are generated by administrators.

## 7. Transactions:

Attributes: TransactionID (Primary Key), PropertyID (Foreign Key), BuyerID (Foreign Key), SellerID (Foreign Key), TransactionType (Buy/Sell), Amount, TransactionDate.  
Relationships:  
- Many-to-One with Properties: A property can be involved in many transactions.  
- Many-to-One with Users (Buyer and Seller): Users participate in transactions as buyers or sellers.

# Application Structure:

1. Entity Layer:   
Each entity (User, Property, etc.) is represented as a JPA entity, mapped to the corresponding database tables using Hibernate. These entities are annotated with JPA annotations to define the relationships and attributes.  
  
2. DAO Layer:   
The Data Access Object (DAO) layer provides an abstraction for database operations. Each entity has its DAO for performing CRUD operations (Create, Read, Update, Delete).  
  
3. Service Layer:   
The service layer encapsulates business logic, utilizing the DAO layer to interact with the database. It provides methods for managing users, properties, inquiries, and transactions.

# Features:

1. User Management:   
- Users can register, log in, and manage their profile.  
- Admin users can manage other users, generate reports, and oversee transactions.  
  
2. Property Listing:   
- Agents can list properties, including uploading multiple images for each property.  
- Buyers can browse properties and filter by location, price, and type.  
  
3. Favorites:   
- Users can save properties to their favorites list for quick access later.  
  
4. Inquiries:   
- Buyers can send inquiries to agents for more information about a property.  
  
5. Transactions:   
- Buyers and sellers can be involved in property transactions, with details like the transaction amount and date stored in the system.  
  
6. Reports:   
- Admins can generate various reports to track system activity, sales, and listings.

# Conclusion:

This Real Estate Management System is a robust solution for managing users, properties, inquiries, and transactions in the real estate industry. The system's modular design ensures scalability and ease of maintenance while leveraging industry-standard technologies like JPA, Hibernate, and MySQL.