**Problem Statement**

To create a user-friendly, secure and high usability application for a Real Estate Management System.

**Description**

The system consists of various real estate listing, having a unique owner whose details the customers can view to contact. The system is accessible to two types of users, namely admin and customer. Admin can track user activities and see his/her interests and then create appropriate business strategies.

The customer can view all the estate listings in the system. The estate list can be sorted by advanced search filters, for example: search by city, search by square feet area and so on. Sorting can be done with price. Customer can download brochure of the estate listings and also list his/her own estate with the system.

**Scope**

* User can only view the listings of the real estate, no additional interaction can be done. If interested, he/she can download the brochure of that particular estate or contact the owner using his contact details.
* Estates once listed, will forever be in the database with the owner info. Real life transactions cannot be reflected in the system.
* User interest is being tracked by the brochures a user downloads. If a user downloads the brochure for property A, that means that user is interested in property A. This interest can be seen by the admin.
* Admin can offer one of the estates out of the estates the user is interested in to the customer at a discount. Only one estate can be offered at a discount at one point of time.

**Out of Scope**

* Buying and Selling real life transactions are out of scope.
* Owner info is displayed in estate listings, the system covers no additional functionalities like buying the property or contacting the seller.
* Dynamic offers are out of scope. An “offer” on a property means we offer a discount of Rs. 10,000

**User methods:**

**+getEstates (Customer)**

Customer can view all estates in the system. He/She can also sort them and search for specific area / city.

**+addEstate (Customer)**

Users can add estates in the system. All details have to be entered including 5 images of the property being listed.

**+viewoffer(Customer)**

Customers can view the estate which is being offered to them exclusively at a discount.

**+viewInterests(Admin)**

Admin can see the users and their interests in properties. Interest is being tracked by the downloading of property brochures.

**+updateOffer(Admin)**

Admin can update the property being offered to the user at a discount. At one time only one property can be offered to a user.

**register(Both)**

User id is generated automatically. By default user role is set to customer. The “default” property is offered to all new users at a discount.

**Assumptions and validation:**

1. Register - (customer):
   1. Userid: Automatically generated.
   2. Password: Password must contain one number,one alphabet,one special character and size should be at least 8 characters and not more than 14 characters.
   3. Phone: Phone number should be 10 digits.
   4. Name: Name should not start with special characters and should not be blank. First letter should be capital.
   5. For registration, default role is user.
   6. Email: should be a proper email
   7. By default, the default estate is added in the user object as the “offerEstate” and it is offered to all new users at a discount.
2. View Estate -
   1. User gets a list of all the estates in the system
   2. User can search and sort them.
   3. Searching can be done by 3 preset Area ranges.
   4. Searching can also be done by 3 preset City values.
   5. Sorting can be done in ascending or descending order with respect to the price of the estate list currently being displayed.
3. Add Estate:
   1. User has to enter all the details while listing the estate
   2. Estate Name should be minimum 15 characters
   3. Estate Price should be minimum Rs 10,000
   4. Estate Area should be greater than 9 sq. feet.
   5. Estate City can only be Mumbai, Delhi or Kolkata.
   6. Estate State can be anything.
   7. There is no mapping between city and state.
   8. User has to upload 5 .jpg images while adding a new estate
   9. These images will be permanently associated with the estate listing.
4. View Offer:
   1. By default, every user (new), is given the default estate of the system in an offer.
   2. This offer estate can be changed by the admin.
   3. Offer means a discount of Rs. 10,000 on that particular estate.
   4. This offer estate will be highlighted at the start of the user home page.
   5. User can contact the owner directly about it.

1. View Interests:
   1. Interest is being tracked by tracking the brochure downloads.
   2. Admin can view these interests.
   3. Once a brochure is downloaded for a particular estate, that estate will forever be listed as an interest of the customer.
   4. The admin can select one of these interesting estates and give an offer on one of them.
2. Update Offer:
   1. By default, the default estate of the system is offered to the user at a discount.
   2. Later, the admin can update this estate under offer from the list of estates in which the user is interested.
   3. At one time, only one estate can be offered to the user at a discount.
3. Schedule visit:
4. Update status:

Use case Diagram:

**Classes**: For this project, we are creating 4 Classes.

1. Estate

2. Images

3. Address

4. User

**Class Description and roles**

1. DTO Layer

1.  **Estate:** *For each estate in the system*

Attributes:

-estateId: BigInteger

-estateName: String

-estateAddress: String

-estateArea: BigInteger

-estatePrice: BigInteger

-estateOwner: User

-imageList: List<Images>

**2. Images:** *For each image present under an estate*

Attributes:

-imageId: BigInteger

-imageName: String

-url: String

-data:bytes[]

**3. Address:** Address of an estate

-addressLine: String;

-city: String

-state: String

-pincode: String

**4. User:** Depending on the type of user, a user role specific menu will be shown.

-userId: String

-userPassword: String

-userName: String

-userContact: String

-userRole: String

-userEmail: String

-birthDate:LocalDate

**5. ScheduledVisit:** All property visits are scheduled and stored in here.

**-**estateId: BigInteger

-date: LocalDate

**Class diagram**

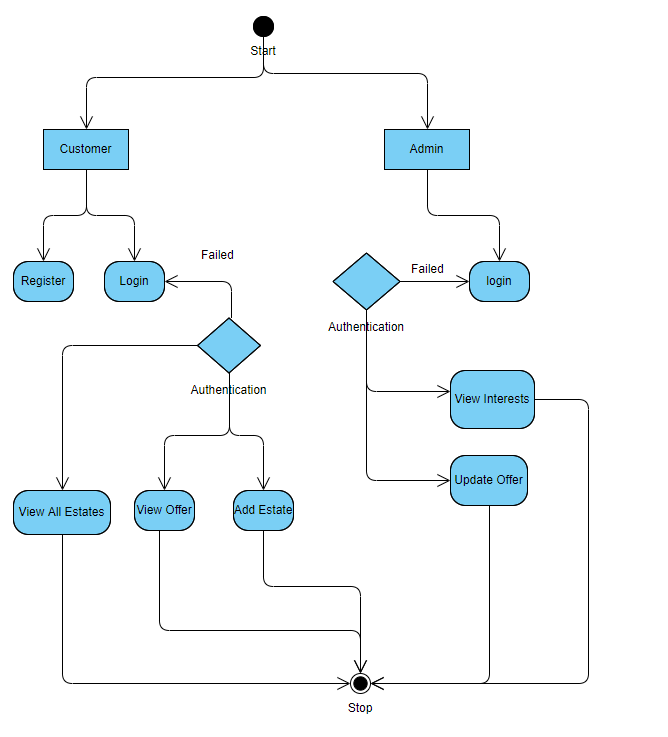


**Relationships:**

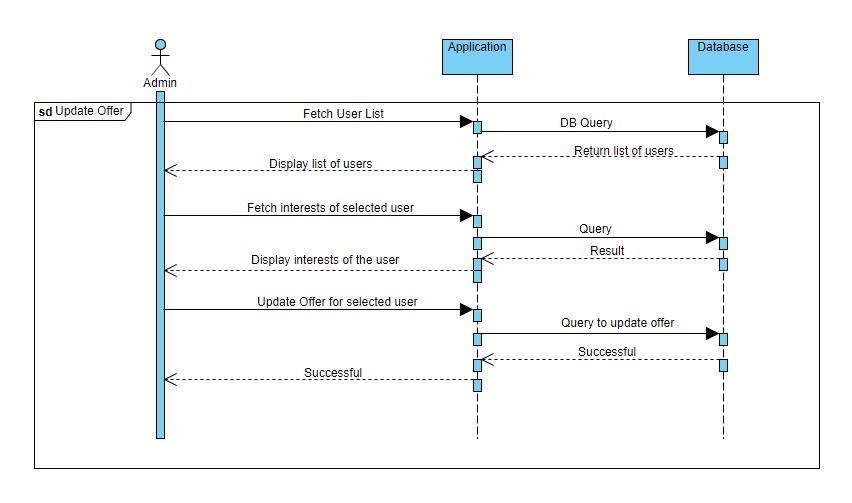
1. User and Estate have 1 to many aggregation relationship**.**
2. Estate and Address have 1 to 1 composition relationship**.**
3. Estate and Images have 1 to many composition relationship**.**
4. Estate and ScheduledVisit has 1 to many composition relationship.

**Diagrams:**

**Activity Diagram :**



**Sequence Diagram**: ADMIN



**Sequence Diagram**: Customer

