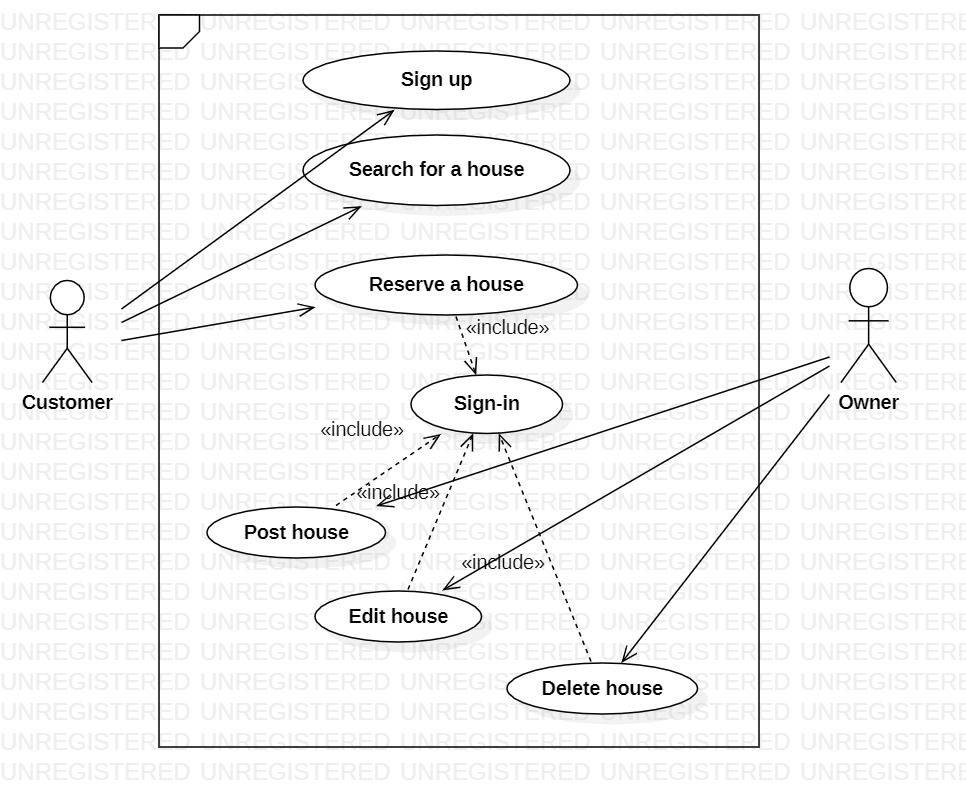
eHouseRent: Use Case Description

Our github link is <https://github.com/MIU-SE/eHouseRent>



|  |  |  |
| --- | --- | --- |
| **Use Case Number: 1** | | |
| **Name Sing-Up** | | |
| **Brief description** This use case allows to create a profile | | |
| **Actors C**ustomer/Owner | | |
|  | | |
| **Preconditions** | | |
| No precondition | | |
| **Flows of Events:** | | |
| **1. Basic Flows** | | |
| **1.0 Sign up** | | |
| **Step** | **User Actions** | **System Actions** |
| 1 | The user calls the sign-up command | The system displays a form with the fields for first name, last name, email, password. |
| 2 | The user fills out the form and requests the system to save the details | The system verify that there is no other profile in the database with the same email address, saves the user and returns the success message on success or a fail message in case of failure. In case another profile exists with the same email address, the system returns a message indicating a duplicate entry exits |
| **Postconditions** | | |
| The user is persisted in the system | | |
| **Business Rule** | | |
| No duplicate user profiles. Two profile are identical if they have the same email address | | |

****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case Number: 2** | | | |  |
| **Name** | |  | House CRUD |  |
|  |  | |  |  |
| **Brief description** | | | This use case allows the owner to create profiles for the houses | |
|  |  | |  |  |
| **Actors** | | | Owner |  |
|  |  | |  |  |
|  |  |  |  |  |
| **Preconditions** | | |  |  |
| The Owner must sign in to the system | | | |  |
|  |  | |  |  |
| **Flows of Events:** | | |  |  |
| **1. Basic Flows** | | |  |  |
|  |  | | |  |
| **1.1.0 Post a House** | | | |  |
|  |  |  |  |  |
| **Step** |  |  | **User Actions** | **System Actions** |
|  |  |  | |  |
| 1 |  | The owner select the Add new house command | | The system displays the house profile form with |
|  |  |  |  | the fields for house Category, address, price, owner Name |
|  |  |  |  |  |
|  |  |  | |  |
| 2 |  | The owner fills out the form and requests the system | | The system verifies that there’s no other profile in |
|  |  | to save the details | | the database with the same house number |
|  |  |  |  | and returns the success message on success or |
|  |  |  |  | a fail message in case of failure. In case |
|  |  |  |  | another house exists with the same data, |
|  |  |  |  | the system returns the message indicating a |
|  |  |  |  | duplicate entry exists. |
|  |  | |  |  |
| **Post Conditions** | | |  |  |
| The new house data is persisted in the system | | | |  |



**Business Rules**

No duplicate house with the same data.

**1.1.1 Read/view list of owners house**

|  |  |  |
| --- | --- | --- |
| **Step** | **Owner Actions** | **System Actions** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| 1 | The owner selects to view a list of houses | The system returns a list of all houses that he/she owns. |
|  |  |  |
|  |  |  |
| **1.1.2 Update house information** | |  |
|  |  |  |
| **Step** | **Owner Actions** | **System Action** |
|  |  |  |
| 1 | The owner selects to view a list of houses | The system returns a list of all houses. |
|  |  |  |
| 2 | The owner selects the profile of the house that he wants to update | The system displays an editable house profile |
|  |  | form pre-populated with the house profile details |
|  |  |  |
| 3 | The owner updates the fields he want to update | The system updates the record and returns the |
|  | and requests system to save the new details | Success message or a fail message on exception. |
|  |  |  |
| **Post conditions** | |  |
| The house profile will be updated | |  |
|  |  |  |
| **Business Rule** | |  |
| The house Number, address fields should be un-editable. | |  |
|  |  |  |
|  |  |  |
| **1.1.3 Delete/remove house** | |  |
|  |  |  |
| **Step** | **Owner Actions** | **System Actions** |
|  |  |  |
| 1 | The owner selects to view a list of houses | The system returns a list of all house profile. |
|  |  |  |
| 2 | The owner selects house to delete/remove from a list. | The system displays a confirmation dialogue |
|  |  | window |
|  |  |  |
| 3 | The owner selects OK on the confirmation dialog | The system confirms the house is no more registered in the list |
|  | window to confirm deleting the house | system returns message a success message on |
|  |  | success or a failure message. The system should |
|  |  | return a message indicating the house could not |
|  |  | be deleted because it is already reserved by someone. |
|  |  |  |
|  |  |  |
|  |  |  |
| **Post conditions** | |  |
| The house will be deleted | |  |
|  |  |  |
| **Business Rule** | |  |
| Only a house that is not reserved can be removed. | | |
|  |  |  |
|  | |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Use Case Number: 3** | | |
| **Name** House search | | |
| **Brief description** This use case allows the customer to search for available house based on specific location | | |
| **Actors customer** | | |
|  | | |
| **Preconditions** | | |
| No **Preconditions, any one can search for the available house** | | |
| **Flows of Events:** | | |
| **1. Basic Flows** | | |
| **1.1.0 Search for available house** | | |
| **Step** | **User Actions** | **System Actions** |
| 1 | The customer selects a location | The system displays the list of houses at that location |
| 2 | The customer selects a house they want from the list | The system displays the selected house’s detail information (type, price, availability date of the house…. ). |

|  |  |  |
| --- | --- | --- |
| **Use Case Number: 4** | | |
| **Name** booking house | | |
| **Brief description** This use case provides a functionality for booking house | | |
| **Actors** customer | | |
|  | | |
| **Preconditions** | | |
| The customer must be logged in to the system | | |
| **Flows of Events:** | | |
| **1. Basic Flows** | | |
| **1.0 book a house** | | |
| **Step** | **User Actions** | **System Actions** |
| 1 | The customer select a house from the list of available house and then click reserve | The system displays a form with check in and check out date |
| 2 | The customer fills out the form and requests the system to save the details | The system saves the booking and returns a success message or a fail message in case of an exception. |
| **Post conditions** | | |
| The booking is persisted in the system | | |
| **Business Rule** | | |
| The same house cannot be booked more than once at the same time. | | |

**Supplementary Specification**

The purpose of this document is to define the non functional requirements of the eHouseRent system. Those specifications with the use case model together captures a complete set of requirements on the system.In our system we will use specifications such us

**Performance** : Our system should record a new booking and give response to the user with in one minute.

**Reliability**  
The main system must be running 95% of the time.

Glossary

1. Customer: a user who is a member of the system who has the permission to book a house from an owner
2. Owner: a user who owns a house to rent
3. Reserve a house: the steps taken for a customer to rent a house from an owner
4. Post house: the step taken for an owner to post or add a house
5. Search house: the steps to search for a list of house