## Consulta Inmueble

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Task Card** | **Vista de consulta inmueble** | |  | Programador | |  | |
|  |  |  |  |  | |  |  |
| Fecha |  |  |  | Task Points | |  |  |
|  |  |  |  |  | |  |  |
| Número de Historia | | 4 |  | Número de TaskCard | | | 20 |
|  |  |  |  |  | |  |  |
| Descripción:  Se deberá presentar una vista donde permita consultar un inmueble ingresando provincia (string:30), localidad (string:30), barrio (string:30), tipo (TipoInmueble), cantidad de dormitorios (Entero > 0) y precio (Decimal) (seleccionando un rango, mínimo y máximo).  Si ocurre un error en la validación de uno o más campos, la interfaz deberá mostrar el error y explicar brevemente que ha sucedido.  Si todo es correcto se debe pasar a la lógica un objeto filtro con los campos completados por el usuario. Se debe mostrar cualquier error recibido. Si todo es correcto se deben presentar los inmuebles obtenidos como resultado de la consulta en forma de lista mostrando ubicación (provincia, localidad, barrio), dimensión (frente, fondo, superficie) y precio.  Luego de la consulta se permitirá seleccionar algún resultado y luego elegir alguna de las siguientes funciones:  Baja Inmueble  Modificación Inmueble.  Catálogo de inmueble (Genera un catálogo con los inmuebles listados)  Generar Reserva  Venta  Si se selecciona alguna de esas funciones, se debe invocar a la acción correspondiente.  Las UI deben ser similares a los spikes realizados en la fase de exploración de la planificación. | | | | | | | |
|  |  |  |  |  | |  |  |
| Notas | | | | | | | |
|
|
| Fecha | Realizado | | A realizar | | Comentarios | | |
|  |  | |  | |  | | |
|  |  | |  | |  | | |
|  |  | |  | |  | | |
|  |  | |  | |  | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Task Card** | **Lógica y persistidor de consulta inmueble** | |  | Programador | |  | |
|  |  |  |  |  | |  |  |
| Fecha |  |  |  | Task Points | |  |  |
|  |  |  |  |  | |  |  |
| Número de Historia | | 4 |  | Número de TaskCard | | | 21 |
|  |  |  |  |  | |  |  |
| Descripción:  Se recibe de la vista un objeto filtro con los datos necesarios para realizar una consulta a la base de datos (Provincia, localidad, barrio, TipoInmueble, cantidad de dormitorios y rango de precios).  Se realiza la consulta en la base de datos con el filtro recibido. Si ocurre algún error durante la consulta se debe devolver una excepción ConsultaException que extienda de PersistenceException.  Si no ha ocurrido ningún error se debe devolver un arreglo con los inmuebles que arrojó como resultado la consulta.  La realización de la consulta en la base de datos se debe implementar mediante hibernate y se deben completar las entidades con las anotaciones necesarias. | | | | | | | |
|  |  |  |  |  | |  |  |
| Notas | | | | | | | |
|
|
| Fecha | Realizado | | A realizar | | Comentarios | | |
|  |  | |  | |  | | |
|  |  | |  | |  | | |
|  |  | |  | |  | | |
|  |  | |  | |  | | |

## Catálogo Inmueble

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Task Card** | **Vista Alta Catálogo Inmueble** | |  | Programador |  | |
|  |  |  |  |  |  |  |
| Fecha |  |  |  | Task Points |  |  |
|  |  |  |  |  |  |  |
| Número de Historia | | 5 |  | Número de TaskCard | | 22 |
|  |  |  |  |  |  |  |
| Descripción:  La carga de un catálogo es invocada desde la administración de clientes; al seleccionar uno, se habilita la opción de generar un catálogo para el mismo.  Se deberá presentar una vista donde permita crear un nuevo catálogo seleccionando los inmuebles que se incluirán en el mismo. Para realizar esta operación, se presenta un listado (inicialmente vacío) con los inmuebles que serán parte del catálogo, junto con las opciones de “agregar nuevo inmueble”, “eliminar del catálogo” y “cambiar foto”.  Cuando se presiona “Agregar nuevo inmueble”, se presenta la vista correspondiente a consultar inmuebles, sólo con la opción de “agregar”. Se permitirá agregar más de un inmueble por consulta.  En la vista de administración del catálogo, se visualizarán los inmuebles a incluir en el catálogo junto con la foto que va a ser incluida en el mismo (por defecto la foto es seleccionada aleatoriamente entre las asociadas al inmueble). La foto puede ser cambiada desde esta lista, seleccionando un inmueble y habilitando la opción de “cambiar foto”. Cuando se selecciona esta opción, se presenta una vista que muestre las fotos asociadas al inmueble, permitiendo seleccionar solo una foto.  Si se selecciona un inmueble de la lista de inmuebles que formarán parte del catálogo, se habilita la opción para eliminarlo del catálogo; esta operación sólo lo removerá de la lista. (estaría bueno un deshacer).  Cuando la lista de inmuebles no está vacía, se habilita la opción de “Generar catálogo”. Esta operación debe invocar a la generación del catálogo. Si el catálogo es generado exitosamente, se visualiza automáticamente con el visor de archivos pdf por defecto.  Si ocurre un error en la generación del catálogo se debe visualizar con un cartel describiendo lo que sucedió.  Las UI deben ser similares a los spikes realizados en la fase de exploración de la planificación. | | | | | | |
| Notas | | | | | | |
|
|
| Fecha | Realizado | | A realizar | | Comentarios | |
|  |  | |  | |  | |
|  |  | |  | |  | |
|  |  | |  | |  | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Task Card** | **Persistidor y entidad catálogo** | |  | Programador |  | |
|  |  |  |  |  |  |  |
| Fecha |  |  |  | Task Points |  |  |
|  |  |  |  |  |  |  |
| Número de Historia | | 5 |  | Número de TaskCard | | 24 |
|  |  |  |  |  |  |  |
| Descripción:  Implementar la clase Catálogo. La clase debe tener los siguientes atributos: id(integer), fechaEmision(Date) y archivoCatalogo(Archivo).  Para cada atributo deben hacerse los métodos set y get.  Para persistir un catálogo se debe hacer una interfaz con métodos que permitan:  - Guardar un catálogo (si resulta en error lanza excepción SaveUpdateException)  - Eliminar un catálogo (si resulta en error lanza excepción SaveUpdateException)  - Obtener un catálogo (si resulta en error lanza excepción ConsultaException)  - Obtener todos los catálogos (por cliente) (si resulta en error lanza excepción ConsultaException)  Todas las excepciones mencionadas extienden a PersistenceException.  Se debe hacer una clase que implemente dicha interfaz mediante hibernate y completar la entidad catálogo con las anotaciones necesarias. | | | | | | |
| Notas | | | | | | |
|
|
| Fecha | Realizado | | A realizar | | Comentarios | |
|  |  | |  | |  | |
|  |  | |  | |  | |
|  |  | |  | |  | |