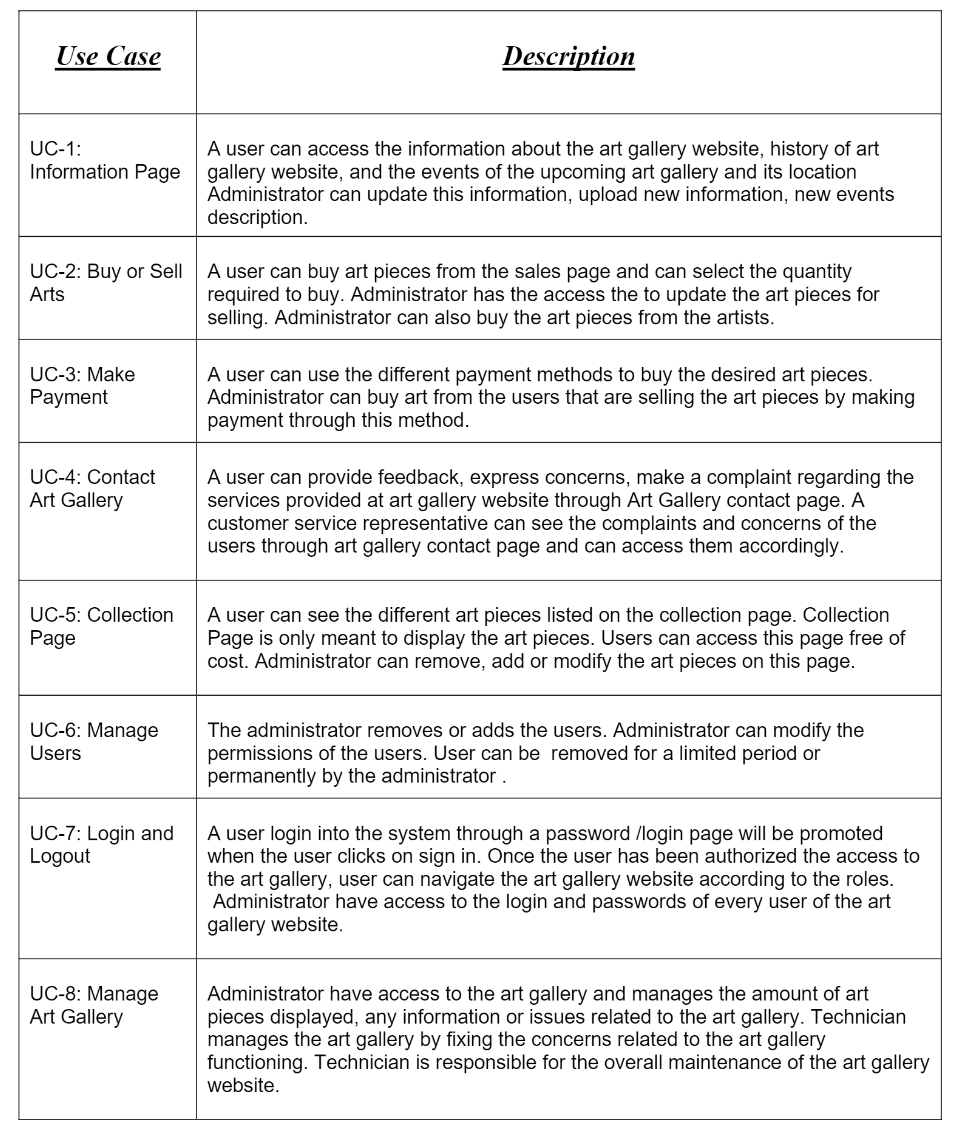
****

**SOFE 3650U Software Design and Architecture**

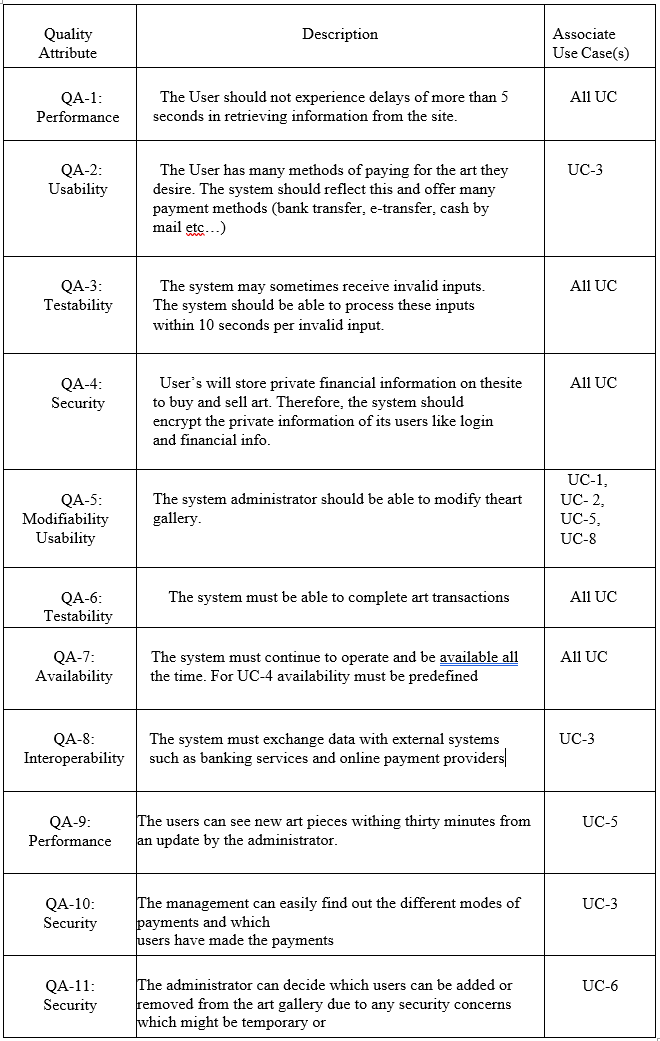
**Iteration 2 of Art Gallery Project**

|  |  |
| --- | --- |
| **Student** | **Name** |
| Manreet Kaur | 100766207 |
| Haiqa Tikka | 100739498 |
| Matthew Gardiner | 100768198 |
| Ammar Salmawy | 100756573 |

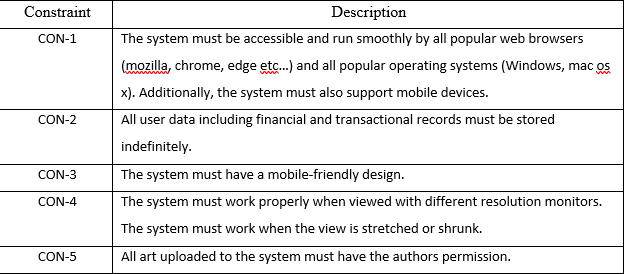
**Project Use Cases:**



**Quality Attributes :**



**Constraints:**

****

**Iteration 2: Identifying Structures to Support Primary Functionality**

The goal of this iteration is to reason the units of implementation. This will affect the team’s formation , interfaces and means by which the development task may be implemented .

**Step 2: Establish Iteration Goal by Selecting Drivers**

The goal of this iteration is to address the general architectural concern of identifying structures to support primary functionality. Keeping this in mind, the following drivers that we as the architects must account for:

* UC-2: Buy or Sell arts: This directly supports the core of the project through the purchasing of art pieces and the ability to add to the database by the administrator
* UC-5: Collection Page: This directly supports the core of the project through the ability to modify the art piece collections page
* UC-8: Manage Art Gallery: This directly supports the core of the project through the ability to modify the art pieces database. It also supports modifiability of the back end with a technician

**Step 3: Choose One or More Elements of the System to Refine**

The elements that are to be refined are those that are directly associated with the different layers that were previously defined by the reference architectures. Specifically, the main functional requirements must be refined in reference to the architecture designed in iteration 1.

**Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers**

**Selected design concepts**

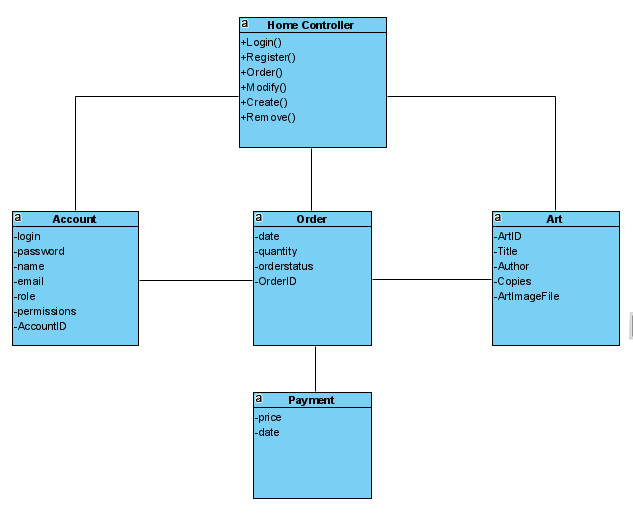
|  |  |
| --- | --- |
| **Design Decisions and Location** | **Rationale and Assumptions** |
| Create a Domain Model for the application | A Domain Model is useful for the system to identify major entities and their relationships within the domain. The Domain Model always exists within a system, however, the earlier it is designed the easier it becomes to understand. So, an initial Domain Model must be created early in the design process. |
| Identify Domain Objects that map to the functional requirements | After the initial Domain Model, each Domain Object must then be identified and encapsulated in its own building block. |
| Decompose Domain Objects into generalized and specialized Components | Once the Domain Objects have been identified and encapsulated, they must be specialized into modules and components that are specific to the layer they are located in. |

**Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces**

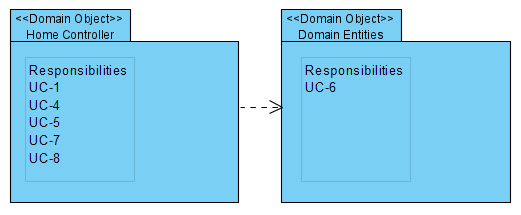
|  |  |
| --- | --- |
| **Design Decisions and Location** | **Rationale** |
| Create only an initial domain model | An initial domain model is created to accelerate this design phase. In this domain model, the entities of the primary use cases are identified and modeled . |
| Map the system use cases to domain objects | System’s use cases are analyzed to identify domain objects associated with primary use cases. |
| Decompose the domain objects across the layers to identify layer-specific modules with an explicit interface | This method of working confirms that the module that maintains all of the features are recognized. The use case will be handled by the architect. This helps the remaining team members to identify the module, which allows the work to be equally distributed. |

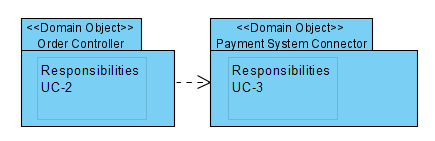
**Step 6: Sketch Views and Record Design Decisions**

Initial Domain model for the system encapsulating some use cases:

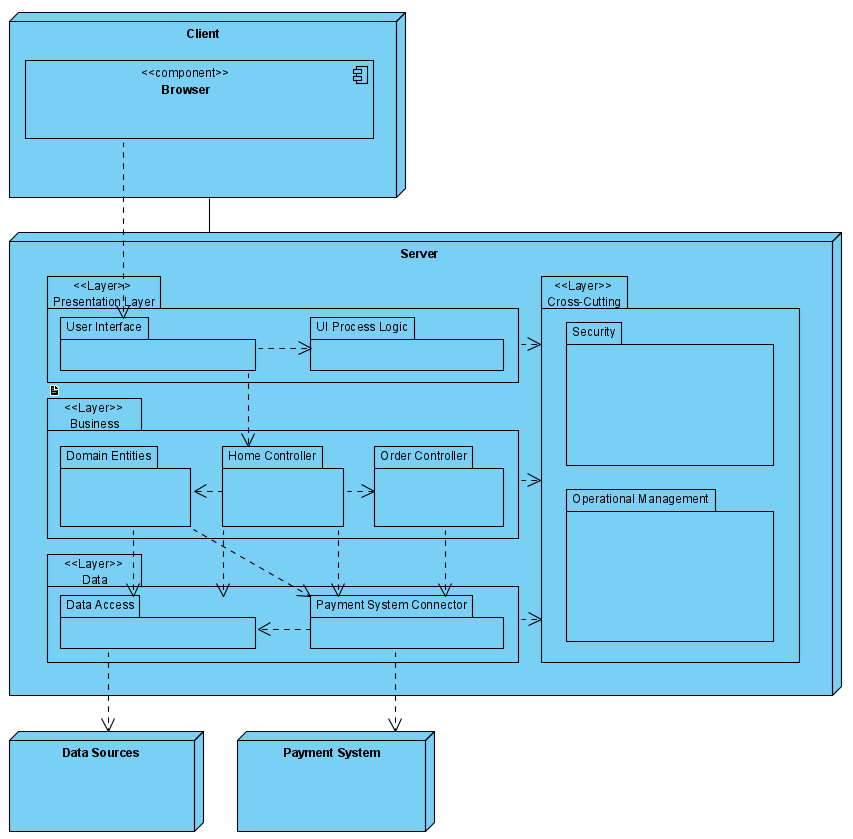
****

Domain Objects for some use cases.





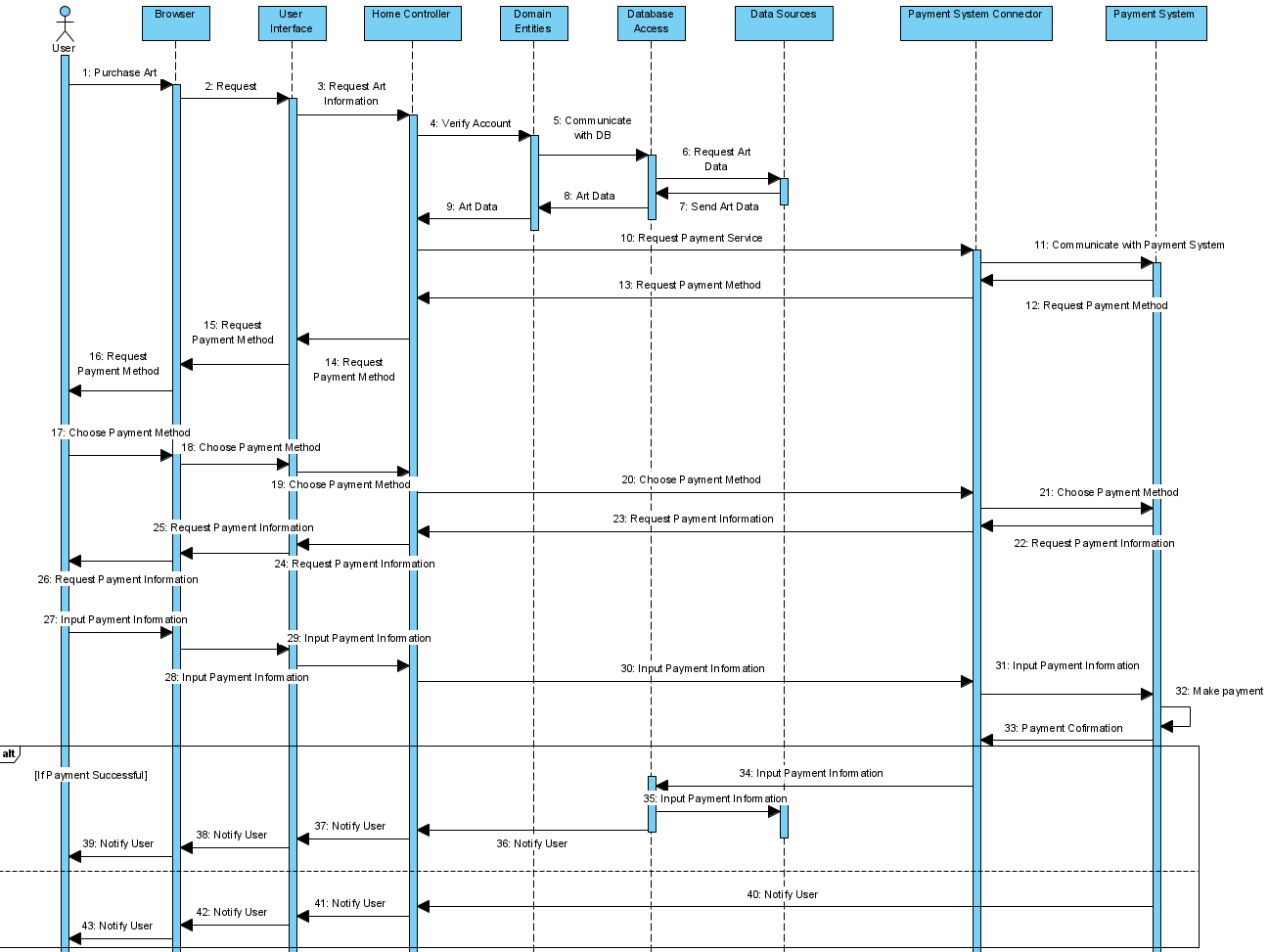
Module view with modules that support primary use cases:

****

|  |  |
| --- | --- |
| **Element** | **Responsibility** |
| **Browser** | Application module that is used by the client to interact with the server applications to provide or display information. Runs on the client machine |
| **User Interface** | These components are responsible for receiving/sending information to the users through inputs like buttons, text fields, etc... |
| **UI Process Logic** | These components are used to direct the flow of the applications use cases. This can include data validation, providing data from business layer to presentation layer, etc... |
| **Domain Entities** | Contains the entities from the domain model. These include account processing, registration, etc... |
| **Home Controller** | Contains business logic pertaining to most use cases of the system. This includes logging in, registering, modifying the art gallery, etc… (UC-1, UC-4, UC-5, UC-7, UC-8) |
| **Order Controller** | This controller processes business logic pertaining to the ordering of artwork (UC-2) |
| **Data Access** | This module encapsulates persistence mechanisms to provide basic operations like retrieving and storing data |
| **Payment System Connector** | This connector is responsible for communication between the order controller and the external payment system, in other words a service agent |
| **Security** | These components include functionality to handle security aspects such as authorization and authentication |
| **Operational Management** | These components handle cross-cutting concerns such as exception management, logging, and instrumentation and validation |

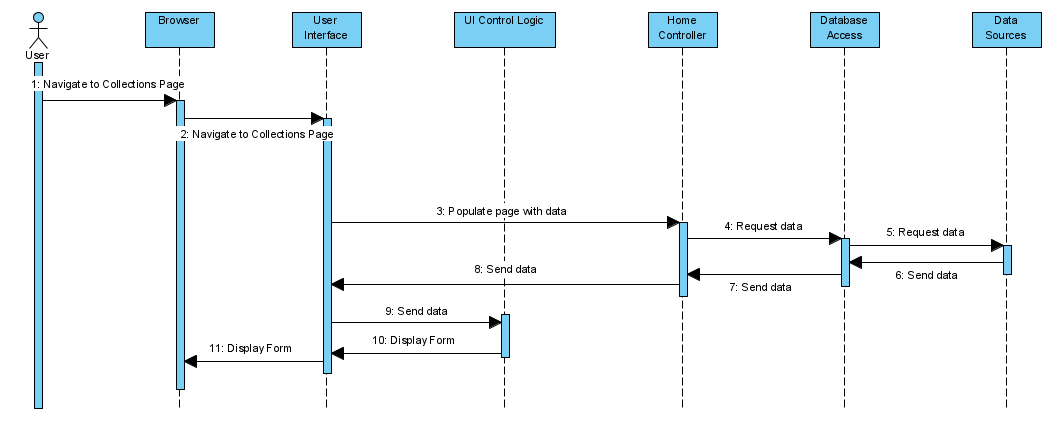
**Sequence Diagrams for primary use cases:**

**UC-2: *Buy or Sell Art***

****

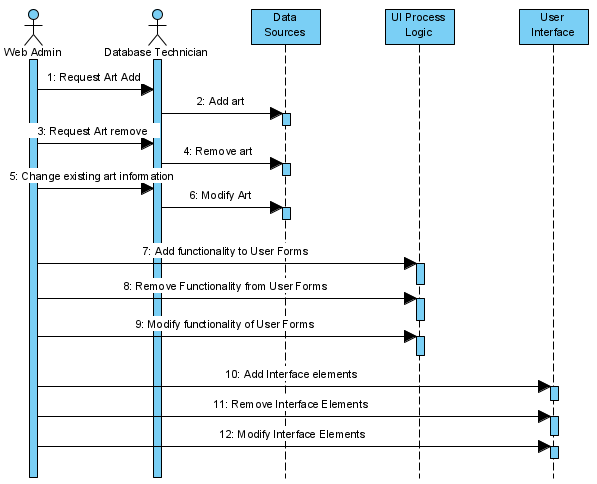
|  |  |
| --- | --- |
| **Method** | **Description** |
| **Element: Browser** | |
| Request | Browser is requesting the information pertaining to the art requested by the user |
| Choose Payment Method | Payment System asks user what payment method they would like to use, then user selects from given list and browser is passing along parameters |
| Input Payment Information | User inputs information for given payment method and browser is passing along parameters |
| **Element: User Interface** | |
| Request Art information | User interface is requesting the information pertaining to the art requested by the user to display to send to the browser |
| Choose Payment Method | User Interface is passing parameters of payment system chosen to payment system connector |
| Input Payment Information | User Interface is passing parameters of payment information to payment system connector |
| **Element: Home Controller** | |
| Verify Account (Request Art Information) | This method verifies that the user account is valid with the account management system in Domain Entities and requesting art information through that |
| **Element: Domain Entities** | |
| Communicate with DB | This method requests art information that the user requested to purchase |
| **Element: Database Access** | |
| Request Art Data | This method requests art information that the user requested to purchase |
| Input Payment Information | Access Database to input payment information (invoice etc) |
| **Element: Payment System Connector** | |
| Communicate with Payment System | Connect to the external Payment system Application |
| Choose Payment Method | Transfer user inputted information to the Payment System Application |
| Input Payment Method | Transfer user inputted information to the Payment System Application |
| Input Payment Information | Transfer information from payment (invoice etc) to the database |
| **Element: Payment System** | |
| Make Payment | Payment System Application uses user inputted information to make a payment |

**UC-5: *Collection Page***

******

|  |  |
| --- | --- |
| **Method** | **Description** |
| **Element: Browser** | |
| Navigate to Collections Page | This method calls the User Interface to provide a form for the User Interface |
| **Element: User Interface** | |
| Populate page with data | Request information from database to display to the user viewing the collections page |
| Send data | Send database information to UI Control Logic to receive updated element information |
| **Element: Home Controller** | |
| Request Data | Request information from database |
| **Element: Database Access** | |
| Request Data | Request access to read database information |

**UC-8: Manage Art Gallery**

******

|  |  |
| --- | --- |
| **Method** | **Definition** |
| **User: Web Admin** | |
| Request Art Add | Web Admin makes decision about which art to add and then notifies Database Technician |
| Request Art Remove | Web Admin makes decision about which art to remove and then notifies Database Technician |
| Change existing art information | Web Admin makes decision about which art to modify and then notifies Database Technician |
| Add functionality to User Forms | Web Admin makes decision about which functionalities should be added to the system and directly accesses the UI Process Logic to add them |
| Remove functionality to User Forms | Web Admin makes decision about which functionalities should be removed to the system and directly accesses the UI Process Logic to remove them |
| Modify functionality to User Forms | Web Admin makes decision about which functionalities should be modified to the system and directly accesses the UI Process Logic to modify them |
| Add Interface Elements | Web Admin makes decision about which interface elements should be added to the system and directly accesses the User Interface to add them |
| Remove Interface Elements | Web Admin makes decision about which interface elements should be removed to the system and directly accesses the User Interface to remove them |
| Modify Interface Elements | Web Admin makes decision about which interface elements should be modified to the system and directly accesses the User Interface to modify them |
| **User: Database Technician** | |
| Add art | Database Technician receives information from Web Admin to update database by adding art information |
| Remove Art | Database Technician receives information from Web Admin to update database by removing art information |
| Modify Art | Database Technician receives information from Web Admin to update database by modifying existing art information |

**Step 7: Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose**

The design decisions in this iteration provided an initial understanding of how functionality is supported in the system with regards to the primary functional use cases. Modules associated with the functionality of the system were identified and defined.

|  |  |  |  |
| --- | --- | --- | --- |
| Not Addressed | Partially Addressed | Fully Addressed | Rationale |
|  | UC-2 |  | The payment system to pay for art has been implemented, but a system that pays the users selling art on the website must still be implemented to fulfill UC-2. |
|  |  | UC-5 | The collection page is fully implemented as a modifiable page that can be displayed to the user if they wish to browse art on the art gallery |
|  |  | UC-8 | The art gallery is fully modifiable through direct access to the database and the user interface forms are also modifiable directly |