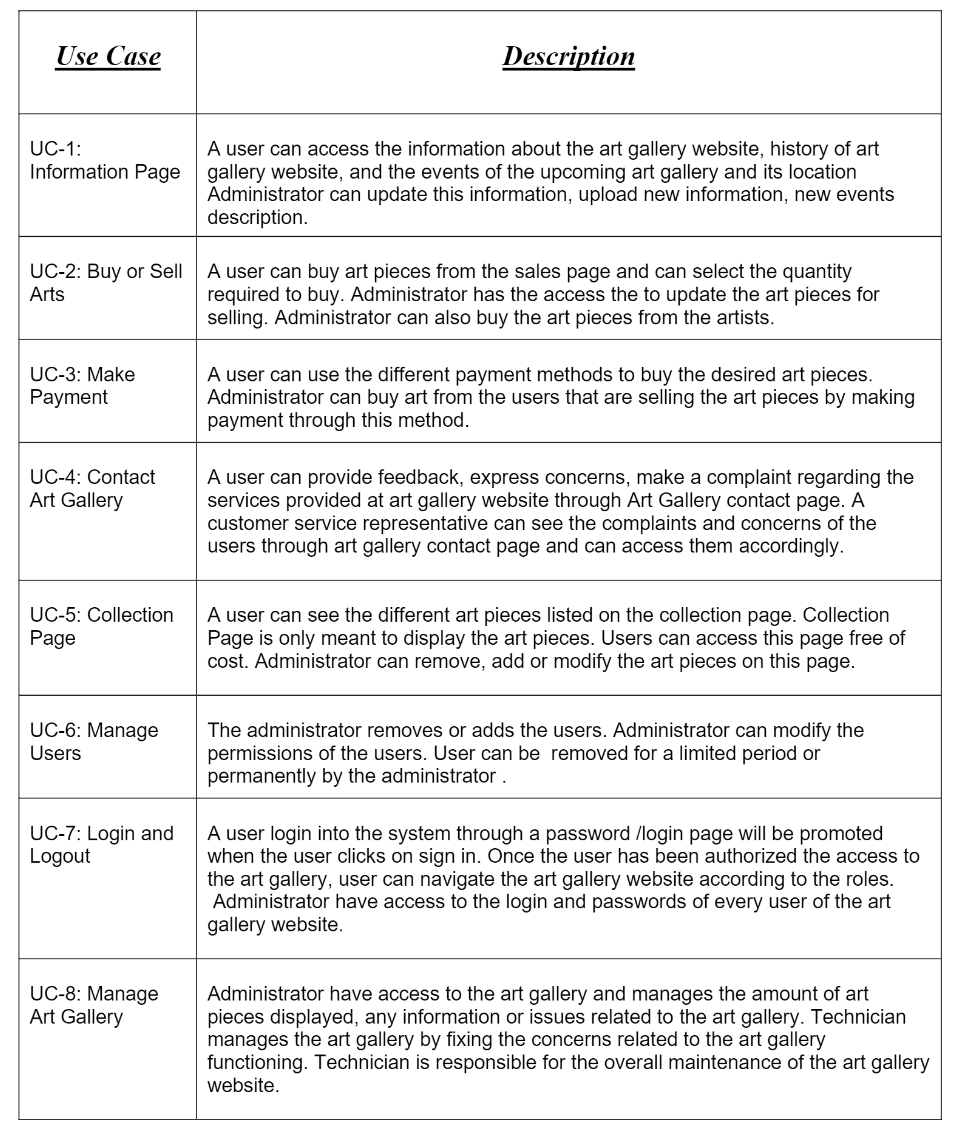
****

**SOFE 3650U Software Design and Architecture**

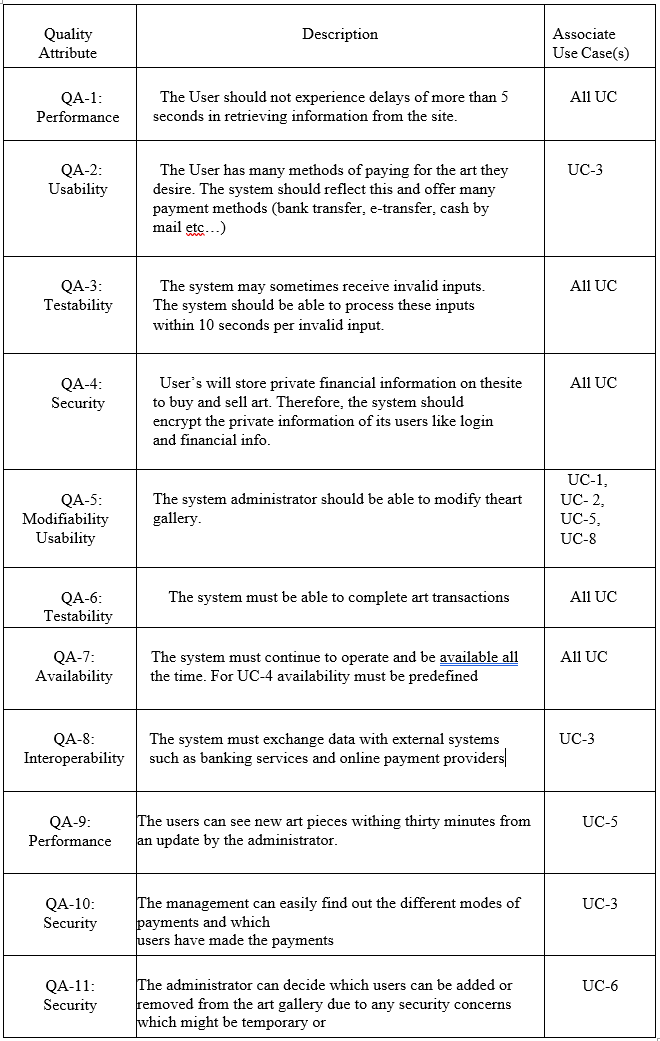
**Iteration 1 of Art Gallery Project**

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

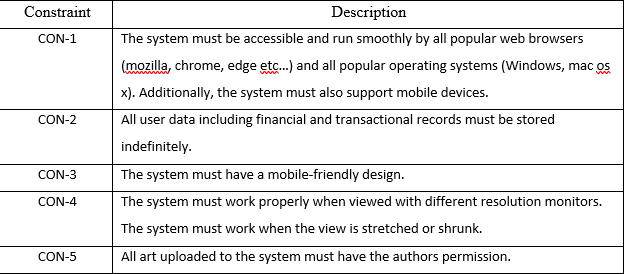
**Project Use Cases :**



**Quality Attributes :**



**Constraints:**

****

**ADD Step 1: Review Inputs**

| **Category** | **Details** |
| --- | --- |
| Design Purpose | Art Gallery is a website that allows users to create accounts to browse and buy art pieces. The system features a login/account system with a payment system for ordering art. The system administrator manages the website by adding/removing art pieces, changing prices, and editing accounts, among other privileges. |
| Primary Functional requirements | 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 |
| Quality attribute scenarios | | **Quality Attribute Scenario** | **Importance to the Customer** | **Difficulty of Implementation According to the Architect** | | --- | --- | --- | | QA-1 | low | low | | QA-2 | high | medium | | QA-3 | low | low | | QA-4 | high | medium | | QA-5 | medium | low | | QA-6 | high | low | | QA-7 | high | low | | QA-8 | high | medium | | QA-9 | low | low | | QA-10 | low | medium | | QA-11 | medium | low | |
| Constraints | All the concerns discussed in initially are included as drivers |
| Architectural concerns | All the architectural concerns discussed initially are included as drivers: |

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

The goal of this iteration is to achieve the architectural concern CNR-1 of establishing an overall system structure. Keeping this in mind, there are several drivers that we as the architects must account for;

1. QA-2: The User has many methods of paying for the art they desire. The system should reflect this and offer many payment methods (bank transfer, e-transfer, cash by mail etc…)
2. QA-4: User’s will store private financial information on the site to buy and sell art. Therefore, the system should encrypt the private information of its users like login and financial info.
3. QA-6: The system must be able to complete art transactions
4. QA-7: The system must continue to operate and be available all the time. For UC-4 availability must be predefined
5. QA-8: The system must exchange data with external systems such as banking services and online payment providers
6. CON-1: The system must be accessible and run smoothly by all popular web browsers (mozilla, chrome, edge etc…) and all popular operating systems (Windows, mac os x). Additionally, the system must also support mobile devices.
7. CON-2: All user data including financial and transactional records must be stored indefinitely.
8. CON-3: The system must have a mobile-friendly design.
9. CON-4: The system must work properly when viewed with different resolution monitors. The system must work when the view is stretched or shrunk.
10. CON-5: All art uploaded to the system must have the author’s permission.
11. CRN-1: Establishing an overall system structure.

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

This is a greenfield project for a mature domain, so the entire art gallery system is to be refined.

All components should be revised so the overall performance and functionality can be improved. refinement is performed through decomposition.

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

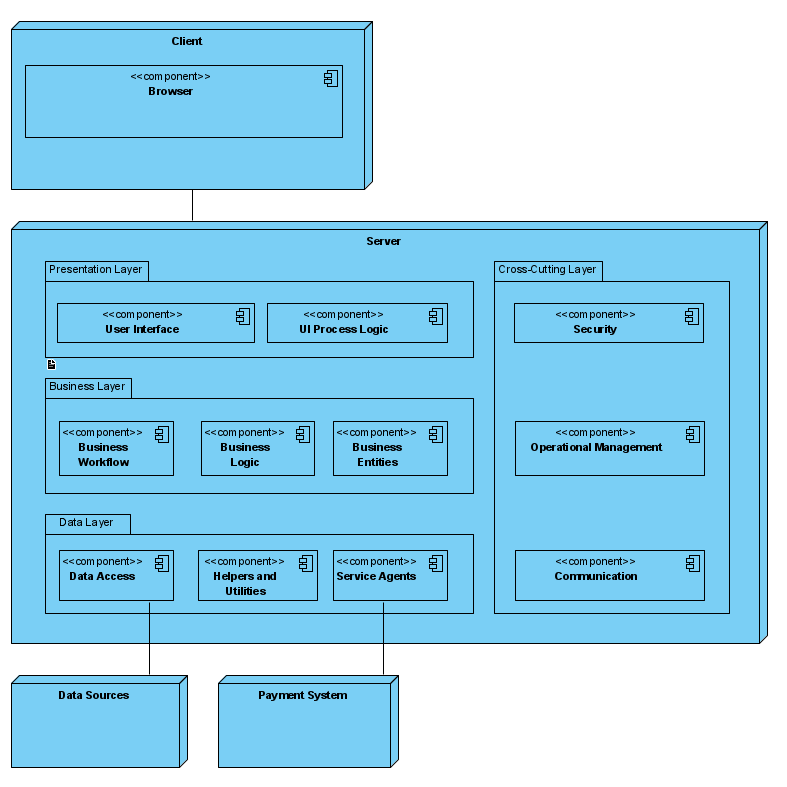
| **Design Decisions and Locations** | **Rationale** |
| --- | --- |
| Logically structure the client and server part of the system using the **Web Application** reference architecture | The Web Applications reference architecture is selected for its overall usability to construct non-rich web applications. This is because our system does not require a rich user interface, it does not need to install anything on the client machine, and it must be accessible over the internet by web browsers (CON-1). This design decision partially affects all of the other drivers, but does not directly impact any other as substantially as the first concern. |
| Physically structure the application using the **three-tier deployment pattern** | This system needs to be accessed through a web browser (CON-1) and there should be an existing database that should be used . Hence the three-tier layer deployment is appropriate . |

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

| **Design Decisions and Location** | **Rationale** |
| --- | --- |
| Remove Application Facade for the business layer of the server | The Application facade component is unnecessary to fulfill the requirements of the project overall. |
| Add Payment System as a external system that communicates with the service agents | The Payment System will be an external system to process payments for the Users. Adding the payment module ensures that many payment methods can be used (UC-3, QA-2), the system can complete art transactions (QA-6), and the system uses an external payment module (QA-8). |
|  |  |

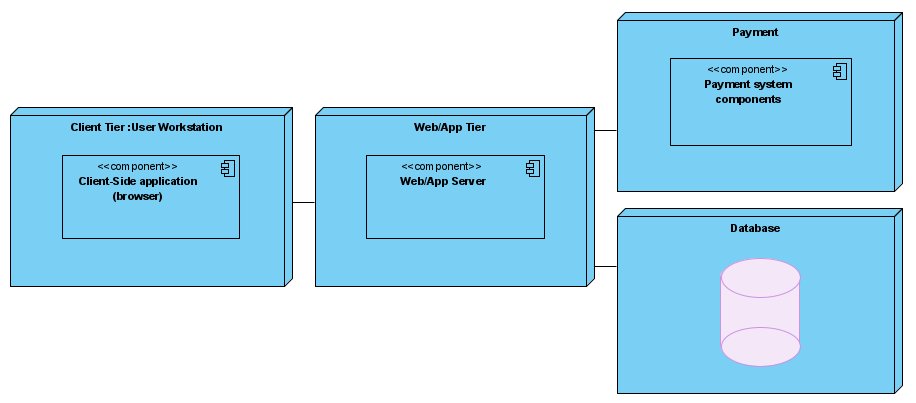
Step 6: Sketch Views and Record Design Decision

Module View of the system after initial design decisions:



| Layer | Component | Responsibility |
| --- | --- | --- |
| Client |  | The presentation layer that communicates with the client service modules |
|  | 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 |
| Server |  | This layer exposes the modules and components that users will interact with |
| Presentation |  | This layer controls user interaction with the server directly, including use case interactions |
|  | 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... |
| Business Logic |  | Contains modules that perform business logic operations on the server side |
|  | Business Workflow | These components are responsible for managing the processes of the business operations, involving the execution of use cases |
|  | Business Logic | These components retrieve and process the data with business rules. |
|  | Business Entities | These components represent the entities of the business domain |
| Data |  | This layer contains modules for data persistence and communication with external systems |
|  | Data access | These components encapsulate persistence mechanisms to provide basic operations like retrieving and storing data |
|  | Helpers and Utilities | These components contain functionality common to other modules in the data layer |
|  | Service Agents | These components are necessary for communicating and transferring data between external services and the system itself |
| Cross-Cutting |  | These modules have functionality that are designed to work across multiple layers |
|  | 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 |
|  | Communication | These components handle communication across the layers and physical tiers of the system |
| Payment Server |  | This layer is an external system that will communicate with the system to provide payment methods to the users on the website (UC-3, QA-2, QA-6, QA-8) |

Deployment Diagram of the system:



| Element | Responsibility |
| --- | --- |
| User Workstation | Users PC, hosts client side application, in this case the user would only need a browser to connect to the server |
| Web/App Server | Hosts the server-side logic and web pages of the application |
| Database Server | The server that hosts the database |
| Payment server | The external system used to make payments on the website |

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

| Not Addressed | Partially Addressed | Fully Addressed | Rationale |
| --- | --- | --- | --- |
|  | UC-2 |  | Introduction of an external payment system supports the functionality of this use-case |
|  | UC-5 |  | Selected reference architecture establishes required modules that will support this functionality |
|  | UC-8 |  | Selected reference architecture establishes required modules that will support this functionality |
|  | QA-2 |  | Introduction of an external payment system supports the functionality of this quality attribute |
|  | QA-4 |  | Selected reference architecture establishes required modules that will support this functionality |
|  | QA-6 |  | Selected reference architecture establishes required modules that will support this functionality |
|  | QA-7 |  | Selected reference architecture establishes required modules that will support this functionality |
|  |  | QA-8 | Introduction of an external payment system |
|  | CON-1 |  | Selected reference architecture establishes required modules that will support this functionality |
|  | CON-2 |  | Selected reference architecture establishes required modules that will support this functionality |
| CON-3 |  |  | No relevant design decisions were made |
|  | CON-4 |  | Selected reference architecture establishes required modules that will support this functionality |
| CON-5 |  |  | No relevant design decisions were made |
|  |  | CRN-1 | Selection of reference architecture and deployment pattern |