Richard Phan, Neloy Kundu, Paul Johnson
CS 205 Software Engineering
Professor Smith
April 5, 2020

Member Responsibilities

Richard : Responsibilities include designing unit tests and system tests for the application. It is and will be important to understand all aspects of how the system functions and how it is intended to function so that sufficient testing can take place. Besides writing and designing the system Test Plan, also contributed in the implementation/coding of the backend and frontend. Specifically helped with coding the graph data structure, which is the main structure of the museum hierarchy, and how it is displayed onto the UI.

Neloy : Responsibilities included integrating the database and integrating the creation of museums along with it. This is important because it allows the entire museum to essentially be created from an sqlite document. Aside from these specialized roles, I will also contribute to writing the backend of the system, as having a firm grasp on such would logically follow with my understanding of implementing databases.

Paul: Responsibilities include the overall design of the user interface and implementing the sketches shown below. In expanding the user interface, I will be connecting major components of the database to be displayed in the program and easily navigated by the user.

Problem Definition and Design Overview

The customer would like a big virtual Haus of places with stuff about stuff for people that want to know about stuff about stuff where the owners of the stuff can control stuff about all that stuff and other people can help organize all the meta-stuff in the Haus.

The platform acting as a solution to the customer's requests contains 4 major system components. These involve the user navigation, museum management, feedback system, and user management. User navigation deals with how a visitor can explore the platform. Museum management takes care of the creation of the addition of new museum entries and its contents. The feedback system allows comments to be made and manages entire comment sections. User management deals with how the system admin, the "owner," manages the curators on the platform.

User Stories

1. As a visitor, I want to explore not only a single museum that I like, but be able to explore the museums that comprise it/ are under it because I am interested in the museum topic.
2. As a system admin, I want to be able to invite new curators to my platform because a diverse set of curators will attract more diverse visitors.
3. As a museum curator, I would like to add pictures and a description to my museum's page because I want to spread information on a certain museum entry to the visitors of my page.
4. As a museum curator, I want to be able to add new museum entries to contribute my interests to existing museum topics.
5. As a visitor, I want to be able to search for a museum by its title or key terms because then I can easily find the museum I am looking for.
6. As a visitor, I want to be able to see other museums that a museum curator I like has created (museum curator profile page) because I might like those other museums as well.
7. As a visitor, I would like the ability to comment on items in the museum because I would like to start a discussion with other visitors.
8. As a system admin, I would like the ability to choose if comments are available for certain posts or not because I want to prevent offensive statements from being said on posts covering sensitive issues.
9. As a system admin, I want to be able to remove curators who I do not want on my platform because I do not want them contributing content I do not like.

10. As a visitor, I want to visit the museum that the current museum I am viewing is part of because I have lost interest in the current museum but still have an interest in the general topic surrounding the current museum.

Use Case Descriptions

Use Case Name: Visit Museum Page	ID: 1	Importance Level: High
Primary Actor: Visitor	Use Case Type: Essential	
Stakeholders and Interests: Visitor- Wants to view the contents of the page Curator- Wants museum page to be displayed to the public		
Brief Description: This use case describes how a visitor visits a museum page.		
Trigger: User enters the museum application Type: External		
Relationships: Association: Include: View current museum's child-museum Extend: View curator's museums, Application home page, View current museum's child-museum, Generate related museums list Generalization:		
Normal Flow of Events: 1. The visitor enters the museum application 2. The visitor is able to decide next action. a. Go to the application homepage b. View the current museum's child-museum c. View the current curator's museums d. Generate a list of related museums		

Use Case Name: Create Museum Page	ID: 2	Importance Level: High
Primary Actor: Curator	Use Case Type: Essential	
Stakeholders and Interests: Curator- Wants museum page to be displayed to the public Visitor- Wants to see new and exciting museums		
Brief Description: This use case describes how a curator creates a new museum page		
Trigger: Curator logs in to the museum application and selects "Create museum page" Type: External		
Relationships: Association: Include: Add picture, Add description Extend: Add key terms, Create sub-museum Generalization:		
Normal Flow of Events: 1. The curator opens the application 2. Curator enters login information 3. System checks account priveleges 4. Curator selects museum location 5. Curator enters museum name 6. Curator enters museum description 7. Curator uploads picture 8. Curator enters key words relevant to the museum 9. Curator publishes museum		

Use Case Name: Invite new curators	ID: 3	Importance Level: Medium
Primary Actor: System Admin	Use Case Type: Essential	
Stakeholders and Interests: System Admin- Wants to add new curators to add museums to the page Curator- Wants museum page to be displayed to the public		
Brief Description: This use case describes how the system admin adds new curators to the museum application		
Trigger: System admin navigates to their tab in application page to add new curators Type: Internal		
Relationships: Association: database of current users Include: Extend: Generalization:		
Normal Flow of Events: 1. System admin navigates to their tab in application page to add new curators 2. System admin generates a unique password 3. Unique password is put into the database for a new curator to enter and access priveleges		

Use Case Name: Create New Account	ID: 4	Importance Level: Medium
Primary Actor: Any User but System Admin	Use Case Type: Somewhat Essential	
Stakeholders and Interests: Visitor- Wants to view the contents of the page, like, and comment on pages Curator- Wants to create a curator profile System Admin- Wants to be able to track users and invite users to curate pages		
Brief Description: This use case describes how a user creates a new account for the museum application.		
Trigger: User opens application and selects, "Create New Account" Type: External		
Relationships: Association: Database of users Include: Extend: Generalization:		
Normal Flow of Events: 1. The user enters the museum application 2. Without login information, the user selects "Create New Account" 3. User enters profile information, just name for now 4. User enters new username and password 5. Username and password info are entered into the user database 6. IF the user has a unique curator password from System Admin: a. User enters unique curator password b. User enters museum curator profile information ELSE a. User continues on to main museum page		

Major System Components

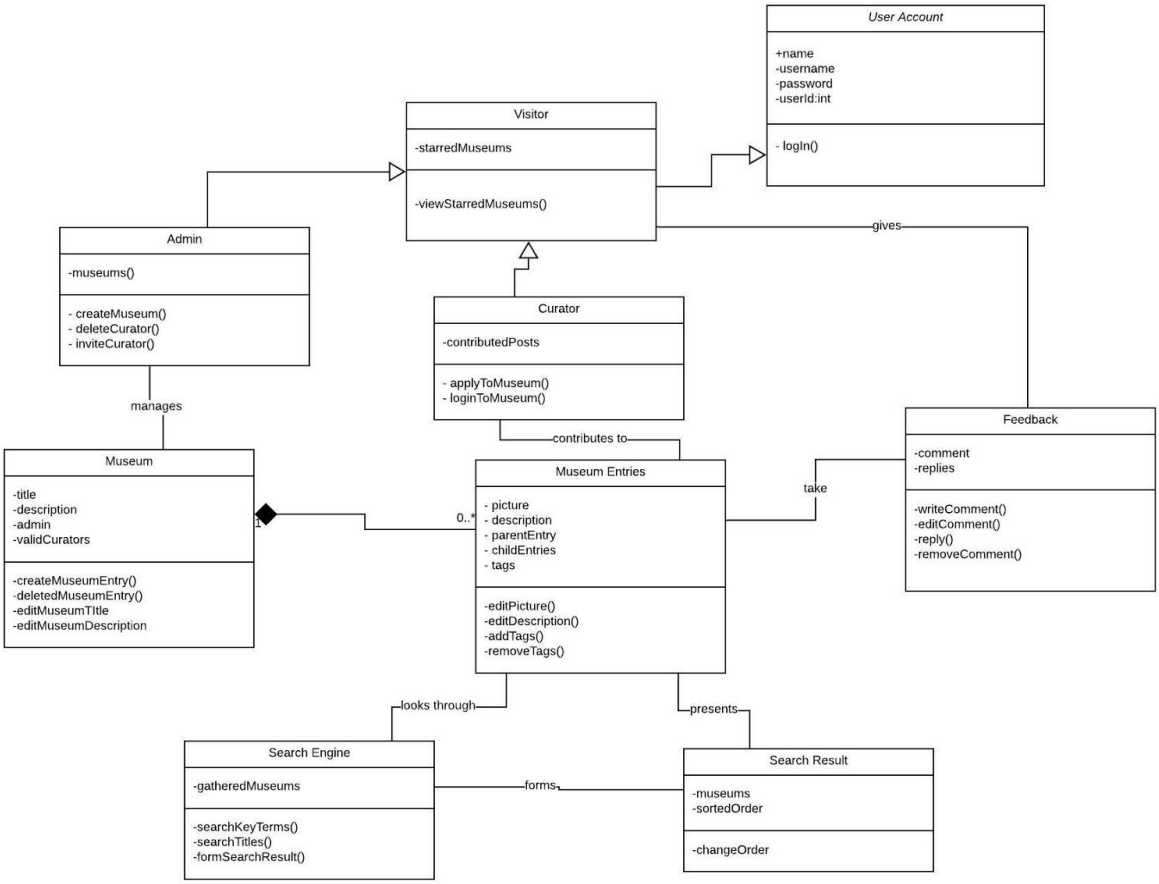
User Navigation - The user navigation system implements a method in which users can seamlessly explore museums by including features such as a search system, curator profile, and a museum hierarchy. Users are able to search for specific topics they are interested in (User story 5) while being able to continue exploring more specific (User story 1) or more broad museums/topics (User story 10) related to the current museum. In the event that the user takes particular interest in a specific curator, they have the ability to find their profile and explore all of the museums that curator has created (User story 6). Seamless user navigation is essential to the application because it will result in a positive experience for the user since finding interesting museums/topics will be efficient.

Museum Management - The museum management system incorporates the addition of museum entries (user story 4) and the addition of details to museum pages (user story 3). This system is integral to the functioning of the application because it provides insight into the creation of the content that will exist on this platform. Furthermore, the museum management system will be more hierarchy-driven, that is, museums will be added under other museums; the primary objective of this is that aspects of a museum entry are explored more in depth with other museum entries under it. For instance, a museum entry representing a car may have museum entries under it that detail different parts of the car and the history of the car.

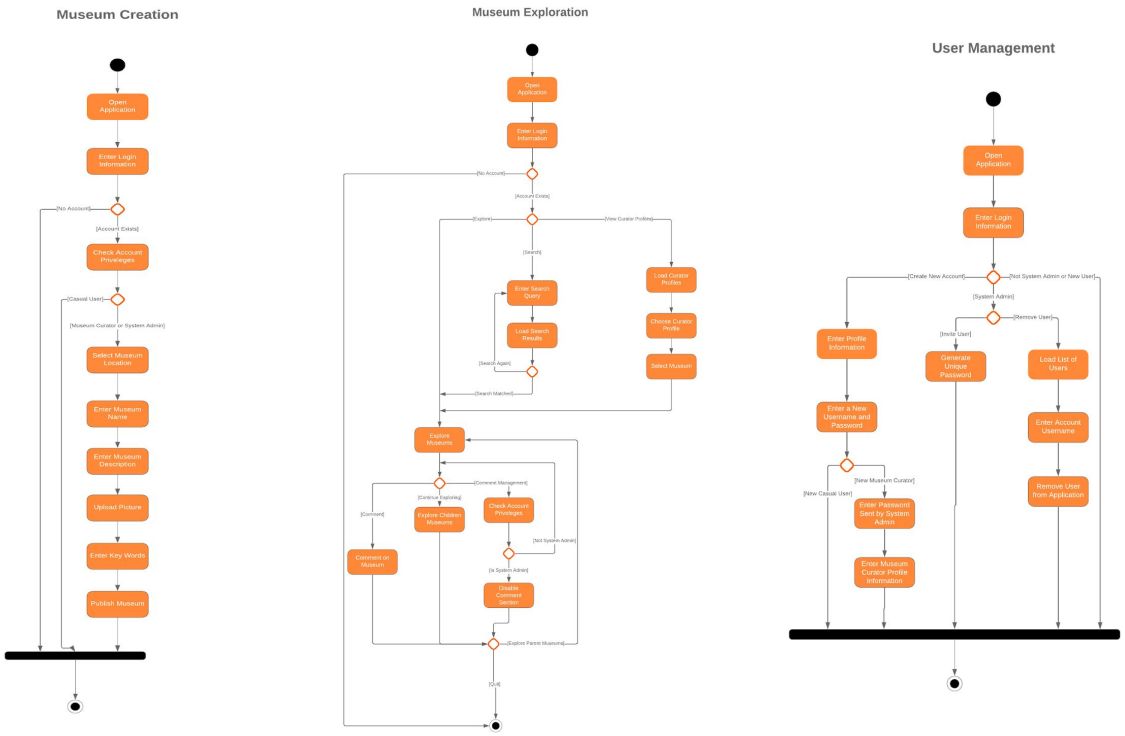
Feedback System - The feedback system allows users and museum curators to converse with each other through a comment section. Users and museum curators are able to contribute their ideas by posting comments on a specific museum (User story 7). System admins are also able to comment back on any user's comments, but they also have the ability to disable a specific museum's comment section in the case that there is a negative user or there are comments about sensitive topics (User story 8). Feedback systems help build a community with other people that like that particular museum and allows museum curators to gain feedback on their work.

User Management - The user management system allows the system admin complete control over which curators are allowed on his platform. The system admin can invite new curators to his platform (user story 2) and remove currently-active curators (user story 9) from the system. This is important because it means while curators have control over the flow of content in museum entries, system admins have a final say in what type of content/ what direction content is moving in the platform.

Logical View



Process View



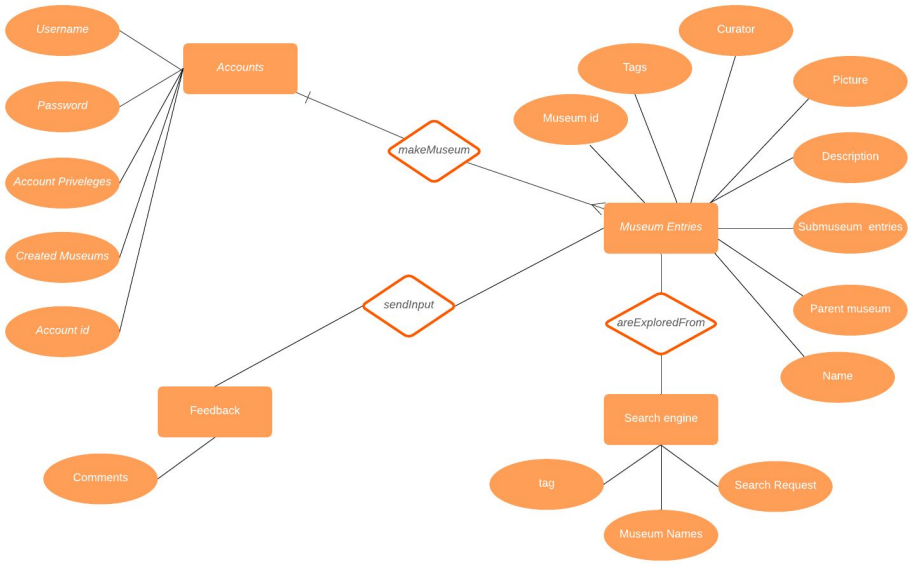
Interrelation of Logical and Process View

Museum Creation - When the user opens the application and enters their login information, the system will interact with the user account class and depending on the type of account the user has, the system will also interact with either the admin, curator, or visitor class. However, if the user does not have an account or is logged in to a visitor account, they are not authorized to create museum entries, so they are automatically forced to leave this museum creation process. In the case that the user is logged in as an admin or curator account, they will then be allowed to create museum entries. At the beginning of the museum entry creation process, the system will reference the Museum Entries class, which is the class that contains the information that is displayed onto each museum page on the UI. Interacting with the Museum Entries class means that the Museum class is also involved since the added instance of the Museum Entries class will be contained in the Museum class. Since the creation of the Museum Entries entails adding and storing data, the system will interact with the proper database table.

Museum Exploration - The user will open the application. Then, the user will try to log in. This dynamic involves the Account class and Visitor class if the user's account exists. Furthermore, the system will interact with either the Admin or Curator class. If the account exists, the user will be able to (a) explore Museums on their own by visiting the Museums on their own from a root home page that features root museum pages of museums in the gallery, which involves interaction with the Museum and Museum Entries class, (b) enter a search query and load the results, which involved interaction with the Search engine and Search results class, to bring a list of Museum Entry objects to navigate to, or (c) view a list of curators, and visit their profile pages (involving interaction with the Curator class) to view a list of museum entries the curator has contributed to, eventually clicking one to direct to a museum entry page (obviously involving interaction with the Museum Entry class). To explore a Museum once in a Museum entry page, one can explore a child entry of a museum entry or visit the parent entry (if not a root). Visitor classes will have the ability to provide feedback, interacting with the Feedback class. Furthermore, the system will check if the user is an admin or not before allowing the user one more option, namely the ability to disable the comment section. This interacts with the Museum Entry class. Aside from all this, the user has the ability to quit.

User Management - When the user opens the application and enters their login information, the system will check through the database to find the object of the User Account class with the particular login and password. If the user is making a new account, a new User Account will be created and added into the database. If the new user wants to be a curator, they will enter in the password created by the Admin class, and a new Curator object will be created with the information that the user provides. If the user logging in has Admin credentials, the system will interact with the Admin class to be able to look at the list of users, remove users, and invite curators.

Entity-Relation Diagram



Schema:
accounts(username, accountID, password, accountPrivileges, createdMuseums)
makeMuseum(accountId, accountPriveleges, museumId)
museumEntries(name,parentMuseum,submuseumEntries,description, picture, curator, tag, museumID)
areExploredFrom(searchRequest, museumId)
searchEngine(tag,museumNames, searchRequest)
feedback(comments)
sendInput(comments, museumId)

Test Plan Description

This system contains crucial class components that need to be individually tested through unit testing to ensure that the entire system can run. Further systems testing is required to verify that all class components work together without producing errors. The following is a design of how the testing will be carried throughout the development of the system.

Unit Testing:

- Testing the Graph - Consists of making sure vertices have the correct neighbors and that each vertex contains the correct information associated with a museum page. More specifically, this can be tested by using a series of get vertex methods to print out what the node contains and matching the printed output with the input file. Since this part of the system is the “backbone” of the rest of the program, a code

coverage of 85% using white box testing with at least three tests for each method is essential to verify a working system.

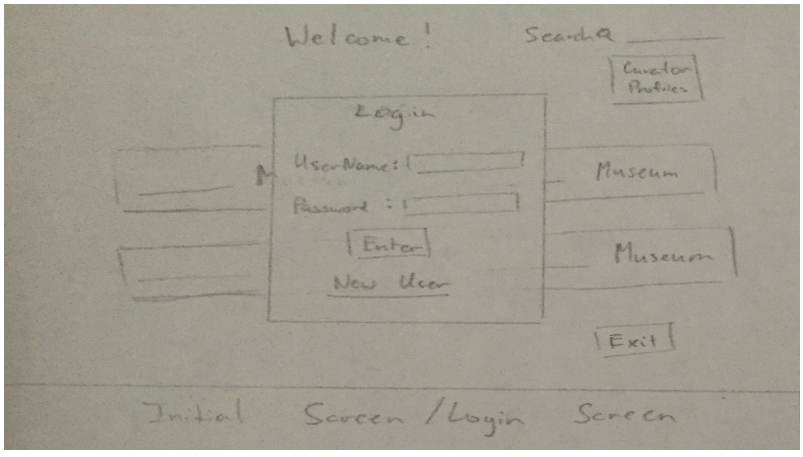
- Testing the Search Function - After ensuring the Graph is working optimally, the search function can be tested. These tests will consist of searching for vertices in different types of graphs (ie Graphs with only right children) to make sure the search function can work in any graph and find what the user is looking for. Large number of tests will help determine whether this function works optimally. Black box testing can be used for testing this function.
- Testing the Comment Function - Black box testing can also be used for testing this function because inputs and outputs can be matched by determining if the comments appear in the correct museum entry. For the deletion of comments, this can also be checked visually through the UI. This function will also have a large number of tests to make sure that it is working properly.
- Testing the Login/Account System - Since each account type has different privileges on the system, the account hierarchy can be tested using black box testing by checking the restrictions of each account. For example, visitors should not be able to create museums and curators should not be able to remove users. The addition and removal of users can be tested by both white box and black box testing. The white box method involves looking at the resulting files/database after a user is added or removed. The black box method involves trying to login after a user is added or removed and the system reacts appropriately.

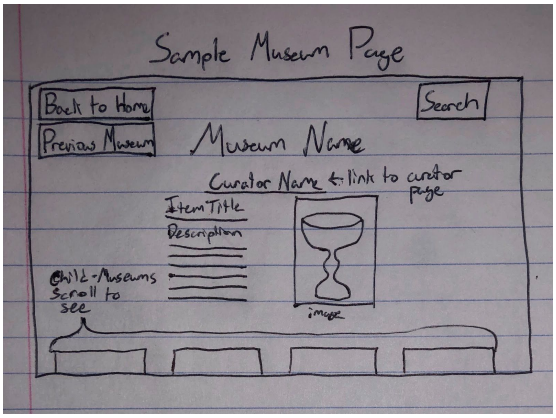
Systems Testing:

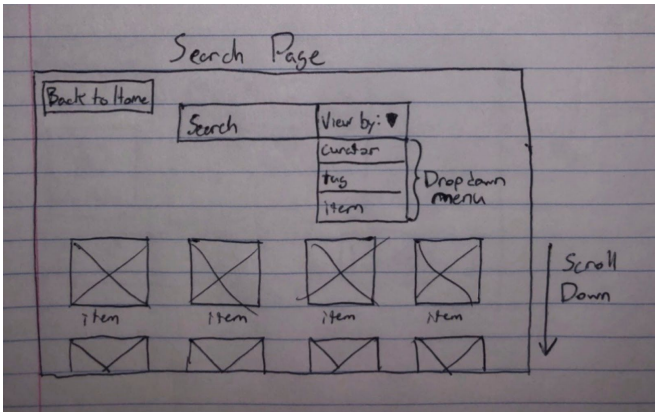
- Usability Testing - Makes sure the system is easy to use for the user. A heuristics evaluation will be performed by the team members and the customer. Since the customer specified the importance of user immersion for this system, this usability test will be important and evaluated strictly.
- Performance Testing - This test ensures that the system works efficiently when dealing with a large amount of data, which is especially important for this system because of the many parts that make up a museum entry as well as the possible size of the museum itself.
- Build Verification Testing - Tests the functionalities of the system including whether the system runs, the system UI, and testing the buttons on the UI.

The unit tests will be carried out both during the implementation of the system and after the system is completed. It is important to test the system during its implementation because it will make sure that errors in the system are identified early. Unit testing after the completion of the system is also important because each individual class must be able to work before testing the system as a whole. Systems testing will be carried out after the completion of the system because only then can proper system testing be made.

User Interface Design







The user interface design all stems from the implementation of the graph structure connecting the museums, which allows the user to go deep into the museums, each showing their child museums and giving relevant information. The museum experience starts with the login page, shown at the top of the images above, which allows a user to login or create a new account. Based on their account status (Visitor, Curator, or Admin), the user is then greeted by the main page, which allows easy access into several new and popular museums, and curators and admin will have special added tabs on the screen to access museum creation (for the curator), and user management (for the admin). Clicking into any museum on the main page will take you to the selected museum's page, shown above left, with the ability to go deeper, or go to the curator's profile page, which shows their museums and tags. To increase explorability, when a user goes to the search page, it will instantly populate with the newest museums any curators have created, which the user can navigate to, or search for a keyword, tag, or curator, thus replacing the museums with the proper results. The creation page shown directly above displays how a curator may add new museums and publish them. Overall, the user interface will be a graphical representation of the structure of museums in the database, and provide links between curators, museums, and their child museums.