Art Exhibition Management System (AEMS) Coding Project Report

Written by Group 2: Ayush Patel, Bogdan Catana, Anirudh Yallapragada, Shane Abraham, CS 440
at the
University of Illinois Chicago
April 2025

Table of Contents

List of Figures

List of Tables

I Project Description

1 Project Overview

The Art Exhibition Management System (AEMS) is a system that manages art exhibitions. There are two main components: the manager and the user. The manager creates the art exhibitions and manages them in their profile. They add information about the artworks, can edit or delete the exhibitions they have added, and decide when they will take place. The user can browse through the different art exhibitions that are available, go through the different artworks in the exhibition by viewing a 3D gallery, and then can "purchase" a ticket to attend the exhibition. There is a recommendation system if the user wants to view a certain type of artwork in an exhibition. The user can refund their ticket if they choose not to go. They can also view the exhibitions they signed up for through a calendar page.

Essentially, AEMS is an exhibition application that allows various art exhibitions to be present, which is inputted from the managers of the respective art exhibition, while that information is presented to the users, who are usually the visitors. The users will have the chance to view the various art exhibitions that are available, in which the artwork information is available to view, as well as some other features for the user to get an idea of what the exhibition may look like. If interested, they can reserve a spot at the exhibition, and if they change their mind, they can cancel their reservation. AEMS will be continuous as there will be art exhibitions added throughout.

2 Project Domain

The domain for AEMS would be artwork and exhibitions. For artwork, there are artists, different types, different mediums, and other details about artwork. For exhibitions, there are managers, ticket sales, different events, and other details about the artwork. The subparts within the domain are being implemented in AEMS, which are important to implement the manager and user (visitor) parts.

3 Relationship to Other Documents

The relationship between this document and the overall design project report of the original group is the different features and guidelines that AEMS has. This document covers the main idea of AEMS and the implementation that we have. This document has the progress that was made during each deliverable phase, which is similar to the plan that the design project report had. Some of the features from the design project description document were implemented in our version of AEMS, which was noted in this document. While there were some differences in terms of the implementation of AEMS, the majority of it was similar as stated in the design project requirements document. For instance, having an authentication system for the user was included in

the requirements in the design project requirements document, while that was one of the components tested and inspected in this document.

4 Naming Conventions and Definitions

4a Definitions of Key Terms

AEMS: Acronym for the “Art and Exhibition Management System” application.

Artist: Creator of the artwork.

Artwork: A piece of art that is created by different mediums.

Exhibition: An event that entails a display of various artworks related to the given theme.

Ticket: Reservation for a visitor to an exhibition.

3D Walkthrough: A virtual interface that shows the different artworks in an exhibition style.

Manager: Creates an exhibition in the AEMS, with the ability to edit and delete them.

Medium: The type of the artwork.

User/Visitor: Attends the exhibition by reserving a ticket, views the various exhibitions available on AEMS.

Calendar: Shows the dates of the exhibition(s) that the visitor reserved tickets for.

4b UML and Other Notation Used in This Document

This document follows the guidelines for UML as described by Fowler in [4]. Any deviations from this standard will be made clear where used.

4c Data Dictionary for Any Included Models

Artwork record: Date, name, artist, and medium of artwork.

Exhibition record: Various artworks that the manager inputs.

Account Record: User information: username, password, manager/visitor.

II Project Deliverables

The deliverables for AEMS have been split into two main components. One component focused on the manager's side of the application, whereas the second component focused on the user's (visitor) side of the application. After these two components were completed, the main focus became polishing up the application to make it more aesthetically pleasing and creating easier functions for both the managers and visitors to use.

5 First Release

The first release was February 28th, 2025. The focus of the first release was to create the functionality of the manager's side of AEMS. As the art exhibitions depend on the input of the managers, it was important to start with the manager's side of the application. At the point of the first release, we had an authentication system working by creating an account and logging in by username and password, making sure that the same username and the same email isn't used more than once. We had two options on the main page, which were to either browse the exhibitions or manage exhibitions. We worked on the manage exhibitions part, which had the ability to create exhibitions. To create an exhibition, the manager would enter the exhibition name, location, start/end date, and start/end time. After creating an exhibition, the manager can add artwork that would be on the exhibition. The manager would upload an image of the art piece, the artist's name, title, medium, year it was made, and a description. The manager had the option to either edit or delete the exhibition that they created on the application. The same features were also available for the artwork that the manager would add to the exhibition. The first release was a sole focus on the manager's side of the application.

6 Second Release

The second release was April 4th, 2025. The focus of the second release was to create the functionality of the user side of AEMS. From the first release, the input of art exhibitions and information related to it was set as the manager's side was complete. At this point in the release, the same authentication system was used for the user as well. This time around, we worked on the browse exhibitions section for the users to view and select art exhibitions that they wanted to visit. On this page, when there are art exhibitions available, they would be present in the middle of the screen with the option to visit them. The user can search for the art exhibitions that they have in mind to visit and can view the exhibitions that are available at the bottom of the screen, which can be sorted by exhibition title and start/end date. The user can either reserve a ticket for that exhibition, get a 3D view of the exhibition with the artworks that are in the exhibition, get an augmented reality view of the artwork, or read more about

the artworks. The user can view the tickets for exhibitions that they have reserved, and delete them if they no longer wish to attend. There is also a calendar that the user can view that shows what art exhibitions they have reserved for the date and time of the exhibition. The second release was a sole focus on the user's side of the application.

7 Comparison with Original Project Design Document

The prototype produced compares with the full project described by the previous group by having the main functionality of the manager and user side. There is an authentication for the manager and user to log in to AEMS. We have a section that allows the manager to manage exhibitions, while the user can view the various art exhibitions. We have a section to manage the art and the exhibition, and we have a section to manage the reservations of the visitors wanting to visit an art exhibition through a ticket system. With the ticket system, we don't have a specific purchase payment method that is used to reserve a ticket, instead, we just let the user indicate that they are attending an art exhibition. We don't have a section that manages the order and manages the order updates. We do have augmented reality and a 3D walkthrough of the exhibition with the artwork, which was briefly mentioned in the original project report. We also added a calendar for the user to have access to know which art exhibitions they reserved and what exhibitions are upcoming, which wasn't mentioned in the original project report.

III Testing

8 Items to be Tested

UI Functionality (item ID 1)

Login and signup functionality backend (item ID 2)

Exhibition creation backend (item ID 3)

Artwork upload backend (item ID 4)

Fetching an exhibition backend (item ID 5)

Editing uploaded art backend (item ID 6)

3D Tour Functionality and Performance (item ID 7)

AR Functionality and Performance (item ID 8)

Ease of Use (item ID 9)

Look and Style (item ID 10)

9 Test Specifications

ID# 1- UI Functionality Test

Description: Checks whether the buttons and UI features are functioning like they're supposed to.

Items covered by this test: UI Functionality (item ID 1), Ease of Use (item ID 9), Look and Style (item ID 10)

Requirements addressed by this test: ID# HR-UR-01 - Intuitive Navigation

Environmental needs: The server and the client both need to be run after installing the needed packages using Node Package Manager.

Intercase Dependencies: NA

Test Procedures: A person will go through the website and perform normal tasks that a user would do while monitoring the state of the backend to see that the buttons and forms send messages as intended. There are console outputs to see when the server performs actions, so the person just needs to click buttons and see that they have the expected behavior.

Input Specification: Clicking on the buttons and inputting data into the forms. Exact data does not matter here since it's just testing that the UI is properly interacting with the backend and that it is usable.

Output Specifications: The server should print to the console when an action that interacts with it is performed on the UI.

Pass/Fail Criteria: This test passes if the UI has the expected output on the backend and the person using it feels that the UI offers a good experience with a design that is pleasing to the eyes. This is a test that is meant to be done constantly by the entire development team to catch any bad UX problems.

ID# 2- Login and signup functionality Test

Description: Checks whether the server backend is able to create an account in the database and then verify a login to that account.

Items covered by this test: Login and signup functionality backend (item ID 2)

Requirements addressed by this test: ID# - F-01 -User Registration and Login, ID# - DR-4 -Customer Data

Environmental needs: Node package manager to install all the required Node packages to run the server and the tests.

Intercase Dependencies: NA

Test Procedures: run the command “npm start” to get Node to run the script for testing that is included with the project

Input Specification: A mock database created in temporary memory that has the same tables as the real one. An account with the name “testuser” and password “secure123” for the signup, and then the same for the login successful test, and the password was changed to “wrong123” for the failed login test.

Output Specifications: The test expects an output of a response that contains the account ID created in the database for the signup portion. For the login successful test, it expects the test to send a message of “success,” and for the failure, it expects “failed login”.

Pass/Fail Criteria: the script runs successfully and doesn’t output any differences

ID# 3- Exhibition Creation Test

Description: Checks that the server is able to create an entry in the database for a new exhibition.

Items covered by this test: Exhibition creation backend (item ID 3)

Requirements addressed by this test: ID# - F-04 - Exhibition Management, ID# - DR-3 -Exhibition Data

Environmental needs: Node package manager to install all the required Node packages to run the server and the tests.

Intercase Dependencies: ID# 5- Fetching An Exhibition Test

Test Procedures: run the command “npm start” to get Node to run the script for testing that is included with the project

Input Specification: A database of exhibitions loaded into temporary memory. The server will then take as input a test exhibition with some data for each of the fields that is expected from the client (exhibitionName, exhibitionLocation, exhibitionStartDate, exhibitionEndDate, organizerID, exhibitionStartTime, exhibitionEndTime).

Output Specifications: The test expects the return from the server to be a message with success.

Pass/Fail Criteria: This test is dependent on the output of the exhibition fetching test as well. It will pass only when the output for both this test and the fetch test returns and passes the script.

ID# 4- Artwork Upload Test

Description: This tests that the server is able to add a new entry to the artwork database with the proper fields added and add the image to the folder on the backend.

Items covered by this test: Artwork upload backend (item ID 4)

Requirements addressed by this test: ID# - DR-2 - Artwork Data, ID# - F-03 - Artwork Management

Environmental needs: Node package manager to install all the required Node packages to run the server and the tests..

Intercase Dependencies: ID# 7- Editing An Artwork Test, ID# 6- Fetching An Artwork Test

Test Procedures: run the command “npm start” to get Node to run the script for testing that is included with the project

Input Specification: An art exhibition created in temporary memory, as well as a mock artwork with all the data specified (artist, title, medium, year, aspect ratio, and a test image that is included in the project).

Output Specifications: The server should output success, and the test image should be moved into the appropriate folder on the backend.

Pass/Fail Criteria: This test is dependent on the successful completion of the artwork fetch test that follows it. It passes if the fetch test passes and if the correct output is generated.

ID# 5- Fetching An Exhibition Test

Description: Checks that the server is able to fetch the exhibition that was added to memory in test #3.

Items covered by this test: Fetching an exhibition backend (item ID 5)

Requirements addressed by this test: ID# - F-04 - Exhibition Management, ID# - DR-3 -Exhibition Data

Environmental needs: Node package manager to install all the required Node packages to run the server and the tests.

Intercase Dependencies: ID# 3- Exhibition Creation Test

Test Procedures: run the command “npm start” to get Node to run the script for testing that is included with the project

Input Specification: The database that was created in memory for test #3, along with the ID of the organizer who created that exhibition. (1 in this case since there is only one).

Output Specifications: The output expects that the data outputted here is the same as the data entered in test case #3.

Pass/Fail Criteria: This test passes if the output has the same data as the input in test #3.

ID# 6- Fetching an Artwork Test

Description: This tests that the server is able to fetch an artwork that was added in test case #4

Items covered by this test: Artwork upload backend (item ID 4)

Requirements addressed by this test: ID# - DR-2 - Artwork Data, ID# - F-03 - Artwork Management

Environmental needs: Node package manager to install all the required Node packages to run the server and the tests.

Intercase Dependencies: ID# 7- Editing An Artwork Test, ID# 4- Artwork Upload Test

Test Procedures: run the command “npm start” to get Node to run the script for testing that is included with the project

Input Specification: The exhibition was created in #4 with an exhibit ID of 1 (since there is only one).

Output Specifications: The server should output the data from the input in test #4.

Pass/Fail Criteria: This test passes if the output is the same data that was entered in test #4.

ID# 7- Editing an Artwork Test

Description: This tests that the server is able to edit an artwork on the backend with new information.

Items covered by this test: Artwork upload backend (item ID 4)

Requirements addressed by this test: ID# - DR-2 - Artwork Data, ID# - F-03 - Artwork Management

Environmental needs: Node package manager to install all the required Node packages to run the server and the tests..

Intercase Dependencies: ID# 6- Fetching An Artwork Test, ID# 4- Artwork Upload Test

Test Procedures: run the command “npm start” to get Node to run the script for testing that is included with the project

Input Specification: The exhibition created in #4 with new data for each field that is specified in test number #4. It will also need to re-run test #6 to check that the data has changed properly, so an exhibit ID of 1 (to check that it did not add a new one but edited the old one).

Output Specifications: The server should output success on the editing portion, and then the new fetch should return the edited data, not the old data.

Pass/Fail Criteria: This test passes if the output data is the same as the input data and not the same as the old data entered in #4.

ID# 8- AR and 3D Test

Description: Checks that the UI is able to handle the 3D and AR functionality on new exhibitions on any device.

Items covered by this test: 3D Tour Functionality and Performance (item ID 7), AR Functionality and Performance (item ID 8)

Requirements addressed by this test: NA

Environmental needs: A device with a camera is needed, and with WebGL enabled in their browser (this is enabled by default on most browsers).

Intercase Dependencies: NA

Test Procedures: Use the website to do a 3D walkthrough of multiple exhibitions with different numbers of artworks. At the same time, also test that the AR is able to work on any camera and with all the artwork in said exhibitions.

Input Specification: 3 Exhibitions, one with 0 art in it, another with 2 art in it, and then another with 9 art in it.

Output Specifications: The website should display an empty exhibition in 3D for the 0 art exhibition, an exhibition with one corridor module and an image on each side for the 2 art exhibition and an exhibition with 5 corridor modules with art on each side except for the last one that has 1 art on a side and the other is empty for the 9 art exhibition. For the AR, for each piece of art, the software should be able to project the artwork to whatever square-shaped object the person has, and is able to follow that object if the object moves.

Pass/Fail Criteria: The test passes if it is able to produce the expected output on both mobile and desktop on Safari, Chrome, and Firefox.

10 Test Results

ID# 1- UI Functionality Test

Date(s) of Execution: Every Saturday from February 8, 2025 to April 26, 2025

Staff conducting tests: The entire development team: Bogdan. Anirudh, Ayush, Shane

Expected Results: UI is functioning properly and is able to deliver a good UX to the user while looking good.

Actual Results: The UI has generally not had any problems, except for on the last test run on April 26, 2025, we decided that the color scheme of the website needed to be reworked since the website should have a consistent theme. This led to the refactoring of the CSS code.

Test Status: Pass

ID# 2 through 7- Backend tests

Date(s) of Execution: May 2, 2025

Staff conducting tests: Bogdan

Expected Results: All tests pass as reported by the script.

Actual Results: All tests passed.

Test Status: Pass

ID# 8- 3D and AR test

Date(s) of Execution: April 4, 2025, March 21, 2025

Staff conducting tests: Bogdan, Anirudh, Ayush, Shane

Expected Results: The AR is able to work with any camera and any image, while the 3D is able to display images in any configuration of exhibition

Actual Results: Same as expected

Test Status: Pass

11 Regression Testing

The UI test (ID #1) is to be run every week after everyone has added their functionality for the week's sprint, then every developer will go and test the UI to see that it works. This was done at the end of every week, and it has caught bugs with the buttons on the ticket list not working properly, and the exhibition buttons leading to the wrong exhibition. It is also what allowed us to incrementally improve the look and feel of the website.

IV Inspection

12 Items to be Inspected

User Authentication (Login/Signup)

Exhibition Management Module (Creating/Deleting/Editing Exhibitions)

- Artwork Management Module (Creating/Deleting/Editing Artworks from Exhibitions)

3D Tour and AR Functionality

User Reservation System and Attendance Planning (Booking tickets, seeing upcoming events)

13 Inspection Procedures

- Checklist:
  - Functionality (Usability and consistency)
  - Correctness (Logic errors and edge cases)
  - Performance (Speed and database query efficiency)
  - Maintainability (Code readability)

14 Inspection Results

1) User Authentication (Inspected by Shane)

- This went quite smoothly and worked as expected. Admittedly, not much effort was put into the authentication aside from making sure the user picked a secure username and password that was unique. We didn't put any conditions for including special characters. Most of the development effort was placed on the substantive part of the project.

2) Exhibition Management Module (Inspected by Anirudh)
    - Editing the Artworks had an issue, which was that when the edit button was clicked, the artworks would not appear unless the user refreshed the page
    - This was caused by the SQL statement on the backend switching the artwork ID and the image name
    - This was unfortunately not found in inspection, as it worked fine, but did sadly show up for the first time during our demo. This was fixed shortly after the discovery.

2a) Artwork Management Module (Inspected by Anirudh)

- Most of the edited information seemed to register fine in the backend, but there were attribute issues in making them appear on the front end, such as using an incorrect ID for the exhibition and images.

3) 3D Tour and AR Functionality (Inspected by Bogdan)
    - Discovered that the AR would be much easier to implement if the usual React methods were to be ignored and instead regular HTML would be inserted into the React scene.
4) User Reservation System and Attendance Planning (Inspected by Ayush)
    - Discovered that refunding a ticket for a particular exhibition would cause all tickets to disappear on the client end. The user would have to refresh the page to see their tickets again without the refunded ticket.
    - This was caused by calling the wrong attribute. ticket.ticketid and exhibition.exhibitionid were called when ticket.id and exhibition.id were supposed to be called.

V Recommendations and Conclusions

- Most of our inspection and testing processes passed and went smoothly. Our key findings include:
    - Authentication: Functionally works as expected, but could be improved with more security measures and potentially more ways to expand accessibility, such as logging in with the user's Google account.
    - Editing the exhibition: Tragically, not discovered during inspection and testing but was instead discovered during the demo, and was quickly resolved with an improved SQL query validation.

- 3D Tour & AR: The 3D Tour functionality works as expected. For the AR, we were able to resolve lagging issues and improve surface detection.
- Reservation System: Users' ability to efficiently refund tickets was quickly fixed with the right attribute.

VI Project Issues

15 Open Issues

An issue that came up during the development of AEMS was with the AR portion. While we were able to get the AR part to work, we weren't able to get the proper surface detection in AR on the web to work. That is because we needed to have certain Android phones and versions of Android for the proper surface detection in AR to work on the web. Therefore, we had to modify the AR section, however, there wasn't much of a problem that came with the modification, as we had the AR portion of AEMS to work with.

Another issue that came up during the development of AEMS was that for the description box, we wanted to use a Rich Text Editor API. However, we weren't able to do so because React Version 19, the framework we used, was incompatible with the API. What we ended up doing was we had our own three buttons for italicizing, bolding, and underlining. Though we could've added more text box features, such as header sizes, bullet points, and other features apart from the three features we had. Having a Rich Text Editor API would've been beneficial towards areas of AEMS that needed certain styles of text.

Another issue that is more of an aesthetic situation is having a robust login system. We had a basic login system that asked for an email, username, and password. We made sure that the same username wasn't used again. We didn't have any conditions when choosing a password about how safe the password is, specific character requirements such as a digit or a special character, and conditions that ensure a secure username and password are used. The authentication system was something to be focused on later as our main focus was getting the manager and visitor's side to function properly.

16 Waiting Room

We had some ideas for AEMS that couldn't be implemented now, but definitely can be implemented in the near future. Those ideas include reviews, a payment system, a stats page from the manager's side, a Google Maps API integration, and the formatting on mobile devices.

Having reviews for the art exhibitions that are in the system would be helpful for the visitors to know how each exhibition is doing. A review page is common for many applications, so it would be easy to implement in the future. Having a review page would be helpful for the manager and the visitor.

Having a payment system for the art exhibitions would be used to have the visitors reserve a spot at the art exhibition they choose to go to. As of now, we have a ticket system that indicates whether the visitor wants to attend the art exhibition they chose. However, in the future, there will be a payment system that will replace the ticket system and ask the visitor for payment through a digital transaction (credit card, debit card). The implementation of this system, in the near future, can be feasible.

Having a stats page from the manager's side would be helpful for the manager to know the general success of the art exhibition that they're handling. The stats page will show the number of people who bought a ticket to see the art exhibition, who are attending, how many people viewed the exhibition on the system, and many other statistics that can help the manager get an idea of the success they're having. This was something that we thought of doing during the development of AEMS, but didn't get the chance to do so. We can implement this feature in the future.

Having a Google Maps API integration for the location of the art exhibitions would be a useful feature for the visitors, as they would get an idea of where the art exhibition would be from their current location. Instead of having the visitor go to the Google Maps app and manually search the address of the art exhibition, the visitor can directly look at the directions in AEMS. The Google Maps API integration would be helpful for the visitor to decide which art exhibitions to go to. With the use of the API, implementing this feature in the future will be feasible.

Having proper UI formatting in the mobile application of AEMS would be useful for the manager and the visitor, as they can use AEMS without worrying about formatting issues. Having proper formatting in the mobile application of AEMS would encourage managers and visitors to use AEMS on their phones instead of using the application on their laptops/computers. Depending on the tools for formatting the mobile application of AEMS, having proper formatting in the near future can be feasible.

17 Ideas for Solutions

Technical Improvements:

- Rich Text Editor Integration:

    - Given the issue that numerous text editor APIs would not work with React version 19, we could try migrating to React version 18 to be compatible with these APIs
    - Another option we could try is putting in some more time in our own lightweight editor. Currently the user can italicize, bolden, and underline their text, but we can include more features such as heading sizes or bullet point formatting.
- AR Surface Detection:
    - AR surface detection is only possible on the web through an API developed by Google (WebXR). However, there are many issues with this API. The API could only be implemented on Chrome and can function only on the Android version of the app since the API is not widely adopted, and the API is slow and prone to errors.
    - The API could be implemented as a feature available only when an Android device is accessing the website and leave the rest of the users of other devices to use the marker AR.
- Authentication:
    - More password policies could be implemented.
    - Allowing the user to make an account with their Google account or other established social media services would be an excellent feature to implement in the near future.

18 Project Retrospective

At the conclusion of AEMS, we found a way to communicate together as a group, in order to make sure to deliver the finest version of an art exhibition management system. As we had to develop AEMS based on a report made by a previous group, we made sure to highlight the main components of the app which ended up being on the manager's and the visitor's side. Based on those two components of AEMS, we decided to add features from the report within each component to make the application. We made sure to discuss the individual parts that everyone is doing, discussed the progress we made during our weekly meetings and over Discord. We made sure to help each other if we were stuck on a certain part. We made sure to communicate with each other about any changes we made to the code before someone else makes edits. Something that can be improved upon in the future is to have a strict schedule on the completion of certain parts of the application rather than having loose deadlines that cause us to work on certain parts at the very end. Regardless, we made sure to confirm with everyone about a change being made in the application and with any other parts that needed to be added or changed. We had a clear plan and excellent communication that led to AEMS being delivered in the finest quality possible given the components and the time frame.

VII Glossary

AEMS: Art Exhibition Management System.

Manager: Creates the art exhibitions on the system that they will manage.

Visitor: Views the art exhibitions on the system that they will attend.

AR: Augmented reality feature that allows the visitor to see an artwork from the exhibition through a closer lens.

3-D Tour: An online visual of the artworks of the selected art exhibition attached to a wall in an exhibition-type setting.

Ticket: A way for the visitor to reserve a spot in the selected art exhibition.

Artwork: Creative work that is set in the art exhibition.

Medium: The type of artwork that the artist created.

Exhibition: An event where different artworks are displayed based on a set theme.

VIII References / Bibliography

[1] Robertson and Robertson, Mastering the Requirements Process.

[2] A. Silberschatz, P. B. Galvin and G. Gagne, Operating System Concepts, Ninth ed., Wiley, 2013.

[3] J. Bell, "Underwater Archaeological Survey Report Template: A Sample Document for Generating Consistent Professional Reports," Underwater Archaeological Society of Chicago, Chicago, 2012.

[4] M. Fowler, UML Distilled, Third Edition, Boston: Pearson Education, 2004.

[5] J. Patel, Z.Awaidah, P.Rantisi, A.Alfadhel, AEMS, Design Project Report, 2023.

IX Index