

Free Artists

Name:

Gherman Serban

Group: 30236

Table of Contents

[Deliverable 1 3](#_Toc64843130)

[Project Specification 3](#_Toc64843131)

[Functional Requirements 3](#_Toc64843132)

[Use Case Model 3](#_Toc64843133)

[Use Cases Identification 3](#_Toc64843134)

[UML Use Case Diagrams 3](#_Toc64843135)

[Supplementary Specification 3](#_Toc64843136)

[Non-functional Requirements 3](#_Toc64843137)

[Design Constraints 3](#_Toc64843138)

[Glossary 3](#_Toc64843139)

[Deliverable 2 3](#_Toc64843140)

[Domain Model 3](#_Toc64843141)

[Architectural Design 4](#_Toc64843142)

[Conceptual Architecture 4](#_Toc64843143)

[Package Design 4](#_Toc64843144)

[Component and Deployment Diagram 4](#_Toc64843145)

[Deliverable 3 4](#_Toc64843146)

[Design Model 4](#_Toc64843147)

[Dynamic Behavior 4](#_Toc64843148)

[Class Diagram 4](#_Toc64843149)

[Data Model 4](#_Toc64843150)

[System Testing 4](#_Toc64843151)

[Future Improvements 4](#_Toc64843152)

[Conclusion 4](#_Toc64843153)

[Bibliography 4](#_Toc64843154)

# Deliverable 1

## Project Specification

Free Artists is a platform that connects artists with clients who are interested in commissioning art or purchasing art pieces. The application allows artists to showcase their work through posts, which can include a description, the type of art, and the price. Clients can browse through posts, search for specific art types, and make transactions for either purchasing existing art or commissioning new pieces.

## Functional Requirements

1. User registration and authentication: Artists and clients can create accounts and log in to the platform.
2. User profile management: Users can manage their profiles, including updating personal information and changing passwords.
3. Artist post creation: Artists can create posts showcasing their art and providing details, such as description, art type, and price.
4. Browsing and searching for art: Clients can browse and search for art based on various criteria, such as art type or artist.
5. Purchasing art: Clients can purchase existing art pieces showcased in posts.
6. Commissioning art: Clients can commission new art pieces from artists.
7. Transaction management: The platform handles transactions between artists and clients, ensuring secure payments and communication.

## Use Case Model 1

### Use Cases Identification

Use Case 1: Register Account

Level: User

Primary Actor: User (Artist or Client)

Main success scenario:

1. User navigates to the registration page.
2. User enters required information, such as username, password, email, display name, and role (Artist or Client).
3. User submits the registration form.
4. The system validates the provided information and checks for any existing accounts with the same username or email.
5. If validation is successful and no conflicts are found, the system creates a new account for the user and stores it in the database.
6. The user receives a confirmation message and is redirected to the login page.

Extensions:

4a. The provided information is invalid or incomplete.

The system displays an error message and prompts the user to correct the information.

4b. An existing account with the same username or email is found.

The system displays an error message and prompts the user to choose a different username or email.

Use Case 2: User Authentication

Level: User

Primary Actor: User (Artist or Client)

Main success scenario:

1. User navigates to the login page.
2. User enters their username and password.
3. User submits the login form.
4. The system validates the provided credentials against the database.
5. If the validation is successful, the user is granted access to their account and is redirected to their homepage or dashboard.

Extensions:

4a. The provided credentials are incorrect or incomplete.

The system displays an error message and prompts the user to correct the information or reset their password.

Use Case 3: Create a Post

Level: User

Primary Actor: Artist/Client

Main success scenario:

1. Artist/Client navigates to the "Create a Post" page.
2. Artist/Client fills in the required information, such as title, description, art type, and any additional details.
3. Artist can optionally attach a file or image of the artwork.
4. Artist submits the post creation form.
5. The system validates the provided information and creates a new post in the database, associating it with the artist's account.
6. The post is now visible on the artist's profile and in relevant site listings.

Extensions:

5a. The provided information is invalid or incomplete.

The system displays an error message and prompts the artist to correct the information.

Use Case 4: Update a Post

Level: User

Primary Actor: Artist

Main success scenario:

1. Artist navigates to the post they want to update, either through their profile or a direct link.
2. Artist clicks on the "Edit" or "Update Post" button.
3. The system displays the post editing form with the current post details pre-filled.
4. Artist updates the necessary information, such as title, description, art type, price, or status.
5. Artist submits the updated post form.
6. The system validates the provided information and updates the existing post in the database.
7. The updated post is now visible on the artist's profile and in relevant site listings with the new information.

Extensions:

6a. The provided information is invalid or incomplete.

The system displays an error message and prompts the artist to correct the information.

Use Case 5: Delete a Post

Level: User

Primary Actor: Client or Artist

Main success scenario:

1. User (Client or Artist) navigates to the post they want to delete, either through their profile or a direct link.
2. User clicks on the "Delete" or "Remove Post" button.
3. The system prompts the user to confirm the deletion.
4. User confirms the deletion.
5. The system removes the post from the database.
6. The deleted post is no longer visible on the user's profile or in relevant site listings.

Extensions:

4a. User cancels the deletion.

The system returns the user to the post view without deleting the post.

Use Case 6: Browse Posts

Level: User

Primary Actor: Client or Artist

Main success scenario:

1. User (Client or Artist) accesses the main page or a specific category of posts (e.g., sales or commissions).
2. The system displays a list of available posts with their titles, descriptions, and other relevant details.
3. User can filter or sort the list by specific criteria, such as art type, price, or date.
4. User selects a post to view more information.
5. The system displays the detailed view of the selected post, including artist or client information, price, and additional details.

Use Case 7: Search for Posts

Level: User-goal

Primary Actor: Client, Artist

Main success scenario:

1. User navigates to the search page or uses the search bar.
2. User enters a search query (keywords, tags, or other criteria).
3. The system filters and displays the relevant posts based on the search criteria.
4. User can view the search results and click on a post for more details.

Extensions:

1a. No posts match the search criteria.

The system displays a message informing the user that no results were found.

Use Case 8: Rate and Review Artists or Clients

Level: User-goal

Primary Actor: Client, Artist

Main success scenario:

1. After completing a sale or commission, the user navigates to the profile of the other party involved (artist or client).
2. User clicks on the "Rate and Review" button.
3. User provides a rating (e.g., 1-5 stars) and writes a review describing their experience.
4. User submits the rating and review.
5. The system saves the rating and review, and displays it on the profile of the reviewed user.

Extensions:

5a. The user submits an incomplete or inappropriate review.

The system shows an error message and prompts the user to correct the issue before submitting again.

Use Case 9: Follow Artists or Clients

Level: User-goal

Primary Actor: Client, Artist

Main success scenario:

1. User visits the profile of an artist or client they are interested in.
2. User clicks the "Follow" button.
3. The system saves the following relationship and sends notifications to the user when the followed artist or client creates or updates posts.

Extensions:

2a. User has already followed the artist or client.

The "Follow" button is replaced with an "Unfollow" button, allowing the user to stop following the artist or client.

Use Case 10: Manage Commission Requests

Level: User-goal

Primary Actor: Artist

Main success scenario:

1. Artist receives a commission request from a client.
2. Artist reviews the commission request details.
3. Artist decides whether to accept or decline the request.
4. Artist clicks on the "Accept" or "Decline" button.
5. The system updates the commission request status and notifies the client.

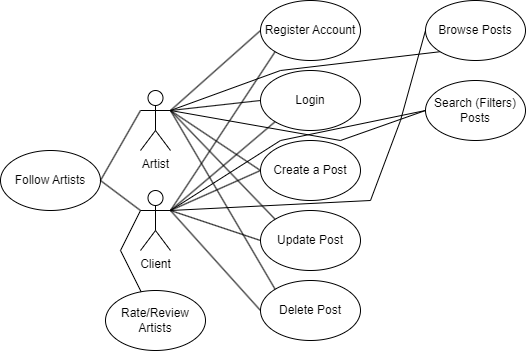
Extensions:

3a. Artist requests more information from the client before making a decision.

Artist sends a message to the client asking for additional details or clarification.

**(Use Cases 1-6 take priority, other Use Cases may be added in the future)**

### UML Use Case Diagrams

****

## Supplementary Specification

### Non-functional Requirements

1. Performance: The system should be able to handle a large number of users and posts without significant performance degradation.
2. Security: User data, including personal information and payment details, must be securely stored and transmitted.
3. Usability: The platform should be easy to navigate and use for both artists and clients, with a user-friendly interface.
4. Scalability: The system should be able to handle an increasing number of users and posts as the platform grows in popularity.

### Design Constraints

* The platform will be built using Java and the Spring Boot framework.
* A relational database, such as MySQL or PostgreSQL, will be used for data storage.
* The platform's frontend will use a modern web framework, such as React or Angular.
* The platform must comply with relevant data protection and privacy regulations, such as GDPR.

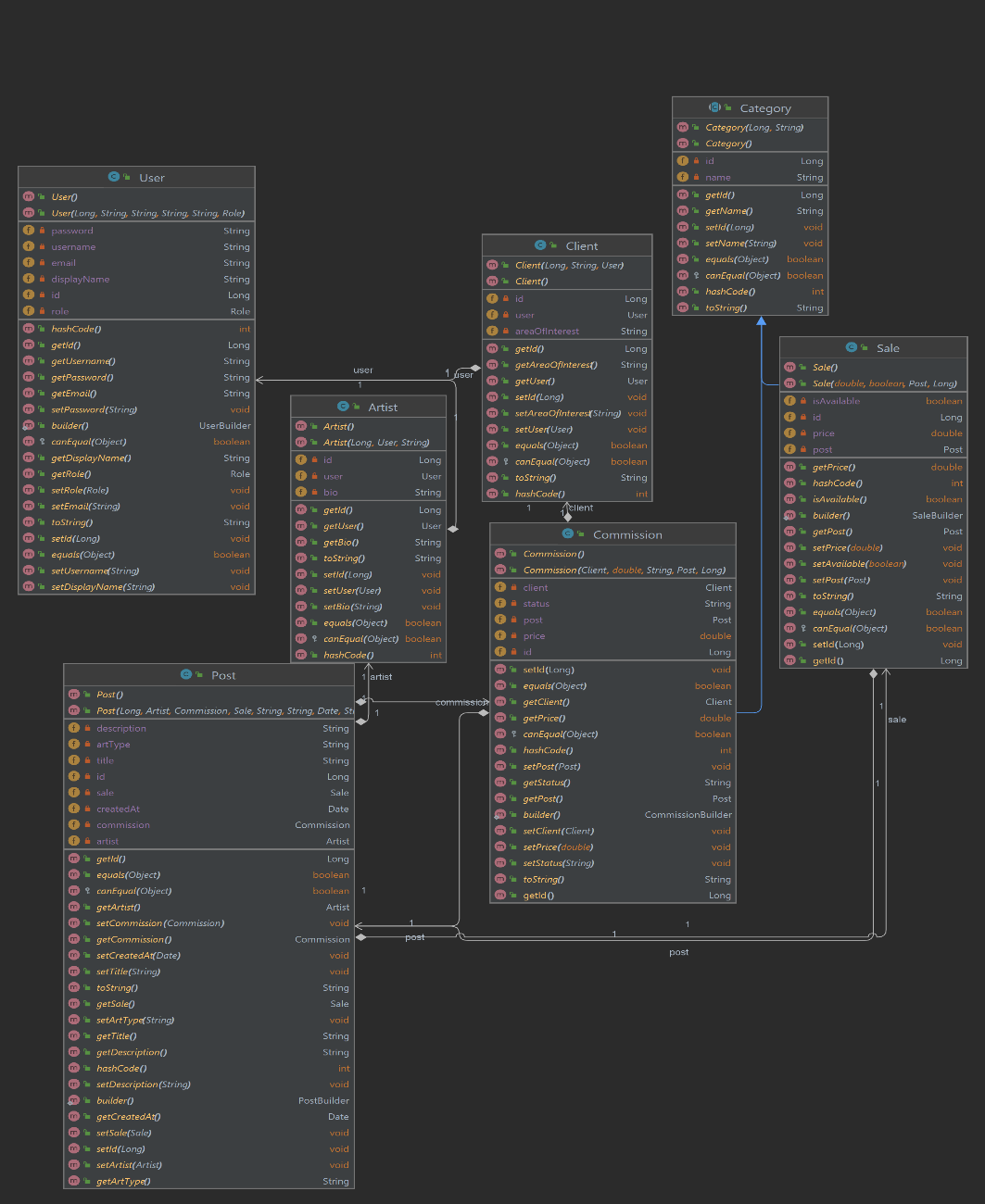
## Glossary

1. Artist: A user who creates and shares art on the platform.
2. Client: A user who browses, purchases, or commissions art on the platform.
3. Post: A showcase of an artist's work, including details such as description, art type, and price.
4. Commission: A custom art piece requested by a client from an artist.
5. Sale: An existing art piece showcased in a post that can be purchased by a client.
6. Art type: The category or style of art showcased in a post, such as painting, sculpture, or digital art.

# Deliverable 2

## Domain Model

The domain model is made up of the following primary entities : User ( either Client or Artist ) and Post (either Commission or Sale ). These entities represent the main components of the system and their relationships. The conceptual class diagram illustrates the relationship between these entities , including their attributes and associations.



## Architectural Design

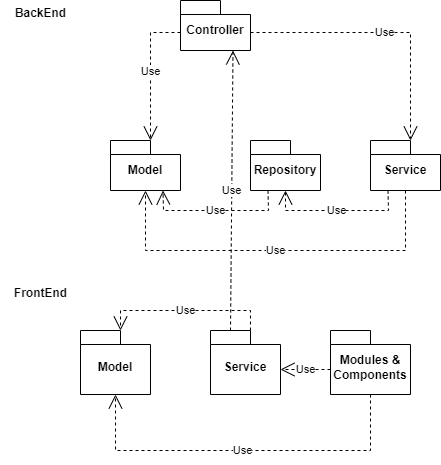
### Conceptual Architecture

The Free Artists system follows a Layered Architecture pattern, dividing the application into different layers with specific tasks. Layered Architecture allows for a clear separation of concerns , promotes maintainability , testability , and ease of access and understanding. The primary layers in the system are as follows:

1. Presentation Layer: Represents the user interface and presentation of data to the user. In this case, this consists of the Angular components which handle the frontend behaviour. They display information and allow user interaction.
2. Application Layer: Handles user input and manages the flow of data between the other layers. It consists of an API (RESTful) with controllers for receiving requests from the frontend and interacting with the services to perform operations on different types of data.
3. Business Logic Layer: Represents the application’s data and business logic. Made up of the domain model entities described above coupled with the corresponding services that handle data manipulation and storage.
4. Data Access Layer: Provides and interface to the data storage mechanism. In this case , repository interfaces tasked with database access and data storage. We use an SQL database for this layer.

### Package Design

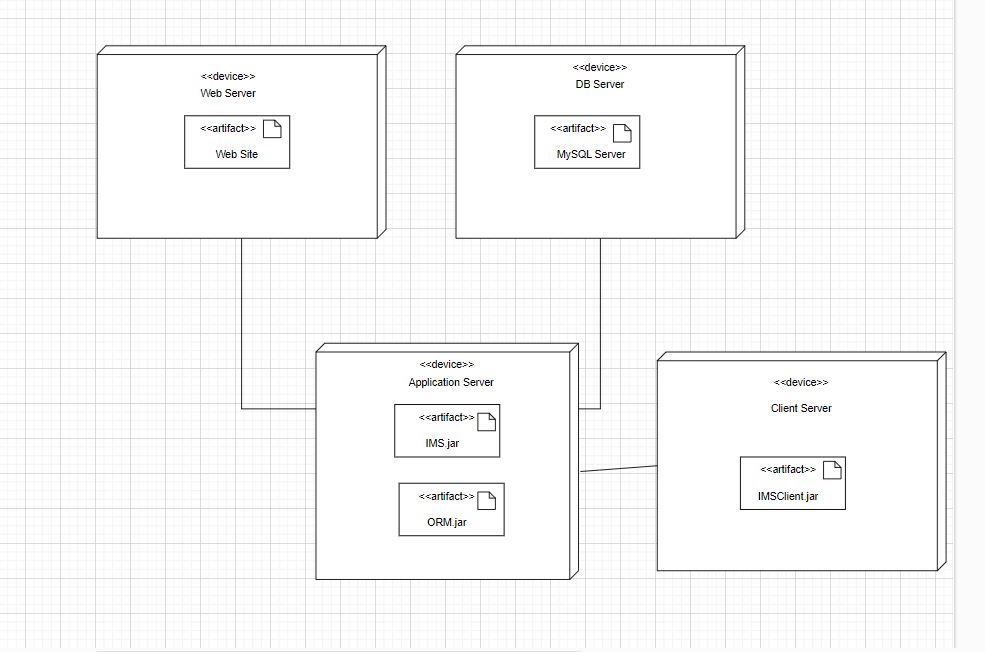
The package design for the Free Artists system organizes the code into cohesive and modular packages following the Layered Architecture. The package diagram shows the main packages and their dependencies, highlighting the separation of concerns and the flow of data between the layers.



### Component and Deployment Diagram

The component diagram illustrates the primary components of the Free Artists application and their relationships, focusing on the interactions between the backend, and the database. The deployment diagram shows the system's deployment infrastructure, including the web server, application server, and the database server, detailing how the components are distributed and interact in a production environment.

Deployment Diagram:



Component Diagram:

A picture containing text

Description automatically generated

# Deliverable 3

## Design Model

### Dynamic Behavior

A screenshot of a computer screen

Description automatically generated with low confidence

SCENARIO 1. LOGIN

1. User inputs data
2. Data is sent to API through a put method
3. The controller sends the login-request data to the User Repository
4. The data Is checked against existing users
5. If successful a TRUE value is returned to the controller so a Websocket connection can be made, after the connection is made we return the HTTP ok Status.
6. If the Login credentials aren’t adequate a message pops-up in the console.

Scenario 2. LIKING POSTS

1. The user can browse posts on the posts page.
2. When he sees a post the he likes, he can press a button with an adequate icon for a thumbs-up.
3. The post-id is remembered and sent to the back-end together with the user-id who like it.
4. In the backend we first check if the post and the user exists, if they do
5. We check whether the post is liked already by the user. If it is we remove it from the liked posts, if not we add it to the liked posts
6. The updated data is sent back to the frontend.

### Class Diagram

The UML class diagram depicts the classes and their relationships in the system. The GoF pattern used in this design is the Singleton pattern, which ensures that only one instance of the class is created and shared by all clients.

A screenshot of a computer screen

Description automatically generated with medium confidence

## Data Model

1. User:

* UserID (Primary Key)
* Username
* Email
* Password (hashed and salted)
* Logged (indicating if the user is currently logged in)

1. Post:

* PostID (Primary Key)
* UserID (Foreign Key referencing User.UserID)
* Title
* Description
* PostTimestamp (to track when the post was made)

In a relational database, these tables can be linked together through a foreign key, which in this case would be the UserID in the Post table. This means that each post is associated with a user, and each user can have multiple posts. This establishes a one-to-many relationship from User to Post.

# System Testing

For system testing methods we used unit testing. One of these tests for example is made by adding a fictional post with a type of post that isn’t found on the app so we return a exception with Post not Found.

# Future Improvements

1. Follow/Unfollow Mechanism: Not implemented , allowing users to follow others will add an extra layer of interaction. It could also provide a feed of followed user's posts, similar to many social media platforms.
2. Enhanced Search: We mentioned that users can search for different posts. We could enhance this feature by introducing filters and sort options. For example, users could search for posts based on tags, date, popularity, or user.
3. Post Analytics: Providing users with analytics on their posts can be quite useful. For instance, the number of views, likes, shares, comments, etc. can all provide valuable feedback.
4. User Rankings or Badges: Implementing a ranking or badge system based on user activity or contributions. This can make the platform more engaging and foster a sense of community and competition.
5. Customizable User Profiles: Allowing users to customize their profiles further, such as changing layout, colors, or even adding a custom bio or links to other social media.
6. Content Sharing: If users are creating great content, we can make it easy for them to share this content on other platforms. Integrating with other social media platforms for easy content sharing could be beneficial.
7. Private Messaging: Allow users to send private messages to each other. This can encourage more user interaction and keep users engaged with the platform.
8. Moderation Tools: Depending on the size of our user base, adding more robust moderation tools could be necessary. This could include report systems for users to report inappropriate content, automatic filters for explicit content, or even a system for users to moderate content (like Reddit's upvote/downvote system).

# Conclusion

In this deliverable, we have created the interaction diagrams, UML class diagram, data model, and testing methods for the system. We have also identified some features that could be added to the system in the future to improve its functionality.

# Bibliography

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Professional.

Fowler, M. (2003). UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd ed.). Addison-Wesley Professional.