Art Gallery Management System

## 

## Team-39

## Abstract:

The Art Gallery Management System (AGMS) is an advanced software solution designed to streamline and enhance the operations of art galleries. It manages key entities such as artworks, artists, exhibitions, customers, staff, and transactions. The system offers comprehensive features for inventory management, exhibition planning, sales processing, and customer and artist relations. Additionally, it supports staff scheduling, detailed reporting, analytics, and online integration for virtual exhibitions and sales. The AGMS enables galleries to operate more efficiently, improve stakeholder interactions, and focus on showcasing and promoting art.

## Introduction:

Artist:

Attributes:

ArtistID (Primary key): Unique identifier for each artist.

Name: Name of the artist.

Birthdate: Date of birth of the artist.

Nationality: Nationality of the artist.

Style: Artistic style of the artist.

Cardinality: One artist can create many artworks, so the relationship is one-to-many with the Artwork entity.

Artwork:

Attributes:

ArtworkID (Primary key): Unique identifier for each artwork.

Title: Title of the artwork.

Description: Description of the artwork.

CreationYear: Year when the artwork was created.

Medium: Medium used in creating the artwork.

Cardinality: Each artwork is created by one artist, so the relationship is many-to-one with the Artist entity. Each artwork can be exhibited in multiple exhibitions, so the relationship with the Exhibition entity is many-to-many through the Catalog entity.

Exhibition:

Attributes:

ExhibitionID (Primary key): Unique identifier for each exhibition.

Name: Name of the exhibition.

StartDate: Start date of the exhibition.

EndDate: End date of the exhibition.

Location: Location of the exhibition.

Cardinality: Each exhibition can feature multiple artworks, and each artwork can be part of multiple exhibitions. So, the relationship with the Artwork entity is many-to-many through the Catalog entity.

Sale:

Attributes:

SaleID (Primary key): Unique identifier for each sale.

ArtworkID (Foreign key): References the artwork being sold.

Price: Price at which the artwork was sold.

Date: Date of the sale.

CollectorID (Foreign key): References the collector who purchased the artwork.

Cardinality: Each sale involves one artwork being sold to one collector, so the relationship is many-to-one with both the Artwork and Collector entities.

Collector:

Attributes:

CollectorID (Primary key): Unique identifier for each collector.

Name: Name of the collector.

Email: Email address of the collector.

Phone: Phone number of the collector.

Cardinality: One collector can make multiple purchases (sales), so the relationship with the Sale entity is one-to-many.

Catalog:

Attributes:

CatalogID (Primart key): Unique identifier for each catalog entry.

ArtworkID (Foreign key): References the artwork included in the exhibition.

ExhibitionID (Foreign key): References the exhibition in which the artwork is included.

Cardinality: Each artwork can be included in multiple exhibitions, and each exhibition can include multiple artworks. So, the relationship with both Artwork and Exhibition entities is many-to-many.

Loan:

Attributes:

LoanID (Primary key): Unique identifier for each loan.

ArtworkID (Foreign key): References the artwork being loaned.

StartDate: Start date of the loan period.

EndDate: End date of the loan period.

Cardinality: Each artwork can be loaned out multiple times, and each loan is for one artwork. So, the relationship with the Artwork entity is one-to-many.

Promotion:

Attributes:

PromotionID (Primary key): Unique identifier for each promotion.

Description: Description of the promotion.

StartDate: Start date of the promotion period.

EndDate: End date of the promotion period.

Cardinality: Each promotion is associated with either an artwork or an exhibition. So, the relationship can be one-to-one or one-to-many depending on whether the promotion is for a single artwork or an entire exhibition.

Inventory:

Attributes:

InventoryID (Primary key): Unique identifier for each inventory entry.

ArtworkID (Foreign key): References the artwork included in the inventory.

Quantity: Quantity of the artwork available in inventory.

Cardinality: Each artwork can have multiple inventory entries, indicating different quantities available. So, the relationship with the Artwork entity is one-to-many.

Conservation:

Attributes:

ConservationID (Primary key): Unique identifier for each conservation record.

ArtworkID (Foreign key): References the artwork being conserved.

Description: Description of the conservation activity.

Date: Date when the conservation activity took place.

Cardinality: Each artwork may have multiple conservation activities associated with it, so the relationship with the Artwork entity is one-to-many.

Functional requirement

1.Artwork Management

- Add Artwork: Enable users to add new artwork details, including title, artist, creation date, medium, dimensions, description, acquisition date, value, and location.

- Update Artwork: Allow users to update existing artwork information.

- Delete Artwork: Enable the removal of artwork records from the system.

- View Artwork: Provide detailed views of artwork information, including current status (on display, in storage, on loan).

2.Artist Management:

- Add Artist: Allow the addition of new artist profiles, including name, biography, date of birth, nationality, contact information, and list of artworks.

- Update Artist: Enable updates to artist information.

- Delete Artist: Allow removal of artist profiles.

- View Artist: Provide detailed views of artist profiles and associated artworks.

3. Exhibition Management:

- Create Exhibition: Enable the creation of new exhibitions with details such as title, start date, end date, description, list of artworks, and curator.

- Update Exhibition: Allow updates to existing exhibition details.

- Delete Exhibition: Enable the deletion of exhibition records.

- View Exhibition: Provide detailed views of exhibitions, including artwork lists and schedules.

4. Customer Management:

- Add Customer: Allow the addition of new customer profiles, including name, contact information, purchase history, and membership status.

- Update Customer: Enable updates to customer information.

- Delete Customer: Allow removal of customer profiles.

-View Customer: Provide detailed views of customer profiles and transaction history.

5. Transaction Management

- Record Transaction: Enable recording of sales transactions, including customer ID, artwork ID, transaction date, amount, and payment method.

- View Transaction: Provide detailed views of transaction records.

- Update Transaction: Allow updates to transaction details.

- Delete Transaction: Enable deletion of transaction records.

6. Staff Management

- Add Staff: Allow the addition of new staff profiles, including name, position, contact information, work schedule, and responsibilities.

- Update Staff: Enable updates to staff information.

- Delete Staff: Allow removal of staff profiles.

- View Staff: Provide detailed views of staff profiles and schedules.

7. Inventory Management:

- Track Inventory: Enable tracking of artwork inventory, including status updates (on display, in storage, on loan).

- Generate Inventory Reports: Provide tools to generate reports on the current status and location of artworks.

8. Exhibition Planning:

- Schedule Exhibitions: Enable scheduling and organization of exhibitions.

- Assign Artworks to Exhibitions: Allow assignment of artworks to specific exhibitions.

- Generate Exhibition Reports: Provide tools to generate reports on upcoming, ongoing, and past exhibitions.

9. Sales and Financial Management:

- Process Sales:Enable processing of artwork sales and generating receipts.

- Generate Sales Reports: Provide tools to generate detailed sales reports.

- Manage Payments: Allow management of different payment methods and records.

10.Reporting and Analytics:

- Generate Reports: Enable the generation of various reports, including sales, inventory, exhibition performance, and customer activity.

- Data Analysis: Provide tools for analyzing data to support decision-making processes.

11.Online Integration:

- Virtual Exhibitions: Enable creation and management of virtual exhibitions.

- Online Sales: Facilitate online sales and transactions.

-Customer Portal: Provide an online portal for customers to view exhibitions, make purchases, and manage their profiles.

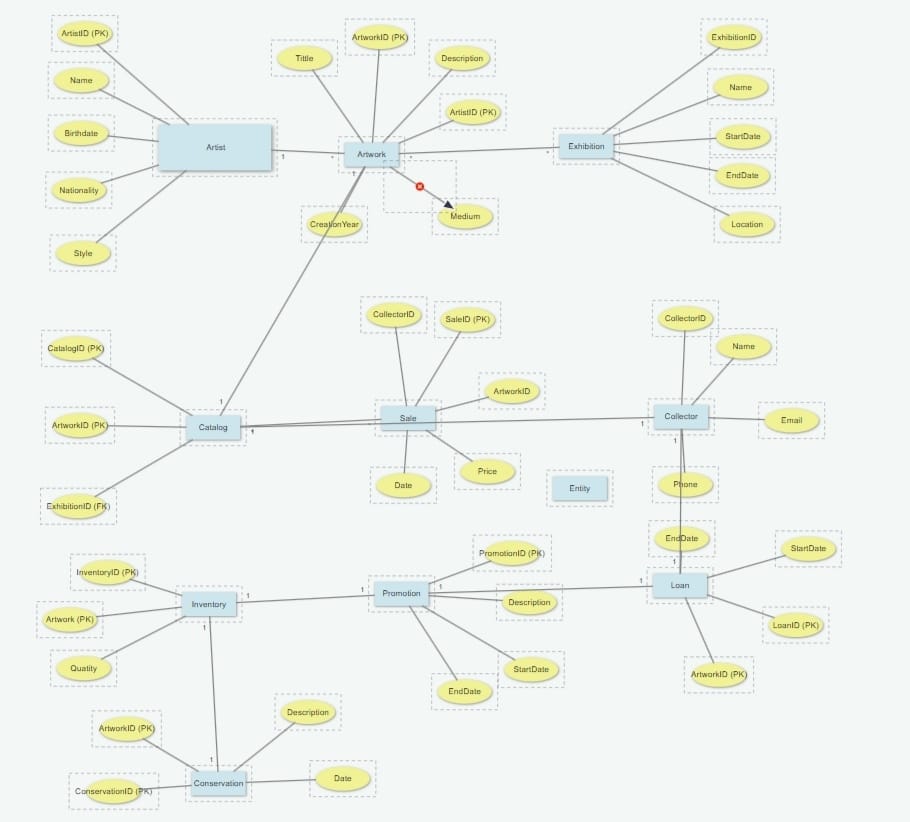
12. Security and Access Control:

- User Authentication: Implement secure login and authentication mechanisms for different user roles (admin, curator, staff).

- Role-Based Access Control : Ensure access control based on user roles, restricting access to sensitive information and functions.

These functional requirements aim to provide a comprehensive and efficient management system for art galleries, enhancing their operational capabilities and user experience.

ER Diagram:



Queries to create database

1.Artwork:

CREATE TABLE Artwork (

artwork\_id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(255) NOT NULL,

artist\_id INT,

genre VARCHAR(100),

creation\_year INT,

price DECIMAL(10, 2),

status ENUM('available', 'sold', 'on loan', 'in conservation', 'unavailable'),

CONSTRAINT fk\_artist

FOREIGN KEY (artist\_id)

REFERENCES Artist(artist\_id)

ON DELETE SET NULL

);

2.Artist:

CREATE TABLE Artist (

artist\_id INT PRIMARY KEY AUTO\_INCREMENT,

artist\_name VARCHAR(100) NOT NULL,

birth\_date DATE,

nationality VARCHAR(100)

);

3.Exhibition:

CREATE TABLE Exhibition (

exhibition\_id INT PRIMARY KEY AUTO\_INCREMENT,

title VARCHAR(255) NOT NULL,

start\_date DATE,

end\_date DATE,

curator VARCHAR(100)

);

4.Sale:

CREATE TABLE Sale (

sale\_id INT PRIMARY KEY AUTO\_INCREMENT,

artwork\_id INT,

customer\_id INT,

sale\_date DATE,

price DECIMAL(10, 2),

CONSTRAINT fk\_artwork

FOREIGN KEY (artwork\_id)

REFERENCES Artwork(artwork\_id)

ON DELETE CASCADE,

CONSTRAINT fk\_customer

FOREIGN KEY (customer\_id)

REFERENCES Collector(collector\_id)

ON DELETE SET NULL

);

5.Collector:

CREATE TABLE Collector (

collector\_id INT PRIMARY KEY AUTO\_INCREMENT,

first\_name VARCHAR(100) NOT NULL,

last\_name VARCHAR(100) NOT NULL,

email VARCHAR(255) UNIQUE,

phone\_number VARCHAR(20)

);

6.Catalog:

CREATE TABLE Catalog (

artwork\_id INT PRIMARY KEY,

description TEXT,

date\_added DATE,

CONSTRAINT fk\_catalog\_artwork

FOREIGN KEY (artwork\_id)

REFERENCES Artwork(artwork\_id)

ON DELETE CASCADE

);

7.Loan:

CREATE TABLE Loan (

loan\_id INT PRIMARY KEY AUTO\_INCREMENT,

artwork\_id INT,

lender\_id INT,

borrower\_id INT,

start\_date DATE,

end\_date DATE,

CONSTRAINT fk\_loan\_artwork

FOREIGN KEY (artwork\_id)

REFERENCES Artwork(artwork\_id)

ON DELETE CASCADE,

CONSTRAINT fk\_lender

FOREIGN KEY (lender\_id)

REFERENCES Collector(collector\_id)

ON DELETE SET NULL,

CONSTRAINT fk\_borrower

FOREIGN KEY (borrower\_id)

REFERENCES Collector(collector\_id)

ON DELETE SET NULL

);

8.Promotion:

CREATE TABLE Promotion (

promotion\_id INT PRIMARY KEY AUTO\_INCREMENT,

promotion\_name VARCHAR(255) NOT NULL,

description TEXT,

start\_date DATE,

end\_date DATE

);

9.Inventory:

CREATE TABLE Inventory (

artwork\_id INT PRIMARY KEY,

quantity INT,

CONSTRAINT fk\_inventory\_artwork

FOREIGN KEY (artwork\_id)

REFERENCES Artwork(artwork\_id)

ON DELETE CASCADE

);

10.Conservation:

CREATE TABLE Conservation (

conservation\_id INT PRIMARY KEY AUTO\_INCREMENT,

artwork\_id INT,

conservator VARCHAR(100),

conservation\_date DATE,

details TEXT,

CONSTRAINT fk\_conservation\_artwork

FOREIGN KEY (artwork\_id)

REFERENCES Artwork(artwork\_id)

ON DELETE CASCADE

);

UML Diagram

|  |
| --- |
| Artwork |
| * ID * title * description * creationDate * medium * dimensions * value |
| * displayInfo() * calcInsValue() |

|  |
| --- |
| Artist |
| * ID * name * biography * birthDate * deathDate * nationality |
| * displayProfile() * addArtwork() |

|  |
| --- |
| Exhibition |
| * ID * title * startDate * endDate * location * theme |
| * scheduleExh() * displayExhDet() |

|  |
| --- |
| Sale |
| * ID * artwork * saleDate * salePrice * buyer |
| * recordsale() * generaterec() |

|  |
| --- |
| Collector |
| * ID * name * contactInfo * collection |
| * displayCollInfo() * addArtToColl() |

|  |
| --- |
| Catalog |
| * ID * title * publicationDate * artworks |
| * addArtwork() * displayCatalog() |

|  |
| --- |
| Loan |
| * ID * artwork * borrower * loanStartDate * loanEndDate * conditions |
| * initiateLoan() * returnLoan() |

|  |
| --- |
| Promotion |
| * ID * campaignName * startDate * endDate * budget |
| * startCampaign() * endCampaign() * displayPromDet() |

|  |
| --- |
| Inventory |
| * ID * artwork * location * quantity |
| * addArtwork() * updateLocation() |

|  |
| --- |
| Conservation |
| * ID * artwork * conservationDate * description * conservator |
| * scheduleCons() * displayConsDer() |

All Class java file codes

1.Artwork

import java.util.Date;

public class Artwork {

private int artworkId;

private String title;

private int artistId;

private String genre;

private int creationYear;

private double price;

private String status;

public Artwork(int artworkId, String title, int artistId, String genre, int creationYear, double price, String status) {

this.artworkId = artworkId;

this.title = title;

this.artistId = artistId;

this.genre = genre;

this.creationYear = creationYear;

this.price = price;

this.status = status;

}

public int getArtworkId() {

return artworkId;

}

public void setArtworkId(int artworkId) {

this.artworkId = artworkId;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public int getArtistId() {

return artistId;

}

public void setArtistId(int artistId) {

this.artistId = artistId;

}

public String getGenre() {

return genre;

}

public void setGenre(String genre) {

this.genre = genre;

}

public int getCreationYear() {

return creationYear;

}

public void setCreationYear(int creationYear) {

this.creationYear = creationYear;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

public String getStatus() {

return status;

}

public void setStatus(String status) {

this.status = status;

}

}

2.Artist

public class Artist {

private int artistId;

private String name;

private String nationality;

public Artist(int artistId, String name, String nationality) {

this.artistId = artistId;

this.name = name;

this.nationality = nationality;

}

public int getArtistId() {

return artistId;

}

public void setArtistId(int artistId) {

this.artistId = artistId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getNationality() {

return nationality;

}

public void setNationality(String nationality) {

this.nationality = nationality;

}

}

3.Exhibition

import java.util.Date;

public class Exhibition {

private int exhibitionId;

private String title;

private Date startDate;

private Date endDate;

private String curator;

public Exhibition(int exhibitionId, String title, Date startDate, Date endDate, String curator) {

this.exhibitionId = exhibitionId;

this.title = title;

this.startDate = startDate;

this.endDate = endDate;

this.curator = curator;

}

public int getExhibitionId() {

return exhibitionId;

}

public void setExhibitionId(int exhibitionId) {

this.exhibitionId = exhibitionId;

}

public String getTitle() {

return title;

}

public void setTitle(String title) {

this.title = title;

}

public Date getStartDate() {

return startDate;

}

public void setStartDate(Date startDate) {

this.startDate = startDate;

}

public Date getEndDate() {

return endDate;

}

public void setEndDate(Date endDate) {

this.endDate = endDate;

}

public String getCurator() {

return curator;

}

public void setCurator(String curator) {

this.curator = curator;

}

}

4.Sale

import java.util.Date;

public class Sale {

private int saleId;

private int artworkId;

private int customerId;

private Date saleDate;

private double price;

public Sale(int saleId, int artworkId, int customerId, Date saleDate, double price) {

this.saleId = saleId;

this.artworkId = artworkId;

this.customerId = customerId;

this.saleDate = saleDate;

this.price = price;

}

public int getSaleId() {

return saleId;

}

public void setSaleId(int saleId) {

this.saleId = saleId;

}

public int getArtworkId() {

return artworkId;

}

public void setArtworkId(int artworkId) {

this.artworkId = artworkId;

}

public int getCustomerId() {

return customerId;

}

public void setCustomerId(int customerId) {

this.customerId = customerId;

}

public Date getSaleDate() {

return saleDate;

}

public void setSaleDate(Date saleDate) {

this.saleDate = saleDate;

}

public double getPrice() {

return price;

}

public void setPrice(double price) {

this.price = price;

}

}

5.Collector

public class Collector {

private int collectorId;

private String name;

private String email;

private String phoneNumber;

public Collector(int collectorId, String name, String email, String phoneNumber) {

this.collectorId = collectorId;

this.name = name;

this.email = email;

this.phoneNumber = phoneNumber;

}

public int getCollectorId() {

return collectorId;

}

public void setCollectorId(int collectorId) {

this.collectorId = collectorId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPhoneNumber() {

return phoneNumber;

}

public void setPhoneNumber(String phoneNumber) {

this.phoneNumber = phoneNumber;

}

}

6.Catalog

import java.util.Date;

public class Catalog {

private int artworkId;

private String description;

private Date dateAdded;

public Catalog(int artworkId, String description, Date dateAdded) {

this.artworkId = artworkId;

this.description = description;

this.dateAdded = dateAdded;

}

public int getArtworkId() {

return artworkId;

}

public void setArtworkId(int artworkId) {

this.artworkId = artworkId;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public Date getDateAdded() {

return dateAdded;

}

public void setDateAdded(Date dateAdded) {

this.dateAdded = dateAdded;

}

}

7.Loan

import java.util.Date;

public class Loan {

private int loanId;

private int artworkId;

private int lenderId;

private int borrowerId;

private Date startDate;

private Date endDate;

public Loan(int loanId, int artworkId, int lenderId, int borrowerId, Date startDate, Date endDate) {

this.loanId = loanId;

this.artworkId = artworkId;

this.lenderId = lenderId;

this.borrowerId = borrowerId;

this.startDate = startDate;

this.endDate = endDate;

}

public int getLoanId() {

return loanId;

}

public void setLoanId(int loanId) {

this.loanId = loanId;

}

public int getArtworkId() {

return artworkId;

}

public void setArtworkId(int artworkId) {

this.artworkId = artworkId;

}

public int getLenderId() {

return lenderId;

}

public void setLenderId(int lenderId) {

this.lenderId = lenderId;

}

public int getBorrowerId() {

return borrowerId;

}

public void setBorrowerId(int borrowerId) {

this.borrowerId = borrowerId;

}

public Date getStartDate() {

return startDate;

}

public void setStartDate(Date startDate) {

this.startDate = startDate;

}

public Date getEndDate() {

return endDate;

}

public void setEndDate(Date endDate) {

this.endDate = endDate;

}

}

8.Promotion

import java.util.Date;

public class Promotion {

private int promotionId;

private String name;

private String description;

private Date startDate;

private Date endDate;

public Promotion(int promotionId, String name, String description, Date startDate, Date endDate) {

this.promotionId = promotionId;

this.name = name;

this.description = description;

this.startDate = startDate;

this.endDate = endDate;

}

public int getPromotionId() {

return promotionId;

}

public void setPromotionId(int promotionId) {

this.promotionId = promotionId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDescription() {

return description;

}

public void setDescription(String description) {

this.description = description;

}

public Date getStartDate() {

return startDate;

}

public void setStartDate(Date startDate) {

this.startDate = startDate;

}

public Date getEndDate() {

return endDate;

}

public void setEndDate(Date endDate) {

this.endDate = endDate;

}

}

9.Inventory

public class Inventory {

private int artworkId;

private int quantity;

public Inventory(int artworkId, int quantity) {

this.artworkId = artworkId;

this.quantity = quantity;

}

public int getArtworkId() {

return artworkId;

}

public void setArtworkId(int artworkId) {

this.artworkId = artworkId;

}

public int getQuantity() {

return quantity;

}

public void setQuantity(int quantity) {

this.quantity = quantity;

}

}

10.Conservation:

import java.util.Date;

public class Conservation {

private int conservationId;

private int artworkId;

private String conservator;

private Date conservationDate;

private String details;

public Conservation(int conservationId, int artworkId, String conservator, Date conservationDate, String details) {

this.conservationId = conservationId;

this.artworkId = artworkId;

this.conservator = conservator;

this.conservationDate = conservationDate;

this.details = details;

}

public int getConservationId() {

return conservationId;

}

public void setConservationId(int conservationId) {

this.conservationId = conservationId;

}

public int getArtworkId() {

return artworkId;

}

public void setArtworkId(int artworkId) {

this.artworkId = artworkId;

}

public String getConservator() {

return conservator;

}

public void setConservator(String conservator) {

this.conservator = conservator;

}

public Date getConservationDate() {

return conservationDate;

}

public void setConservationDate(Date conservationDate) {

this.conservationDate = conservationDate;

}

public String getDetails() {

return details;

}

public void setDetails(String details) {

this.details = details;

}

}

Challenges List

- Ensuring accurate and up-to-date information for artworks, artists, and exhibitions.

- Training staff and artists to effectively use the system.

- Seamlessly integrating with existing systems and online platforms.

- Protecting sensitive data from unauthorized access and cyber threats.

- Handling an expanding inventory and user base without performance degradation.

- Meeting diverse and specific needs of different galleries.

- Regularly updating the system to fix bugs and add new features.