Case Study
Virtual Art Galllery
SaiPrabath Chowdary S

Schema design:

Entities:

• Designing the schema for a Virtual Art Gallery involves creating a structured representation of the database that will store information about artworks, artists, users, galleries, and various relationships between them. Below is a schema design for a Virtual Art Gallery database.

```
CREATE DATABASE VirtualArtGallery
USE VirtualArtGallery
```

• Entities and Attributes:

• Artwork
ArtworkID (Primary Key)
Title
Description
CreationDate
Medium
ImageURL (or any reference to the digital representation)

```
CREATE TABLE Artwork (
ArtworkID INT PRIMARY KEY AUTO_INCREMENT,
Title VARCHAR(255),
Description TEXT,
CreationDate DATE,
Medium VARCHAR(100),
ImageURL VARCHAR(255),
ArtistID INT
);
```

• Artist
ArtistID (Primary Key)
Name
Biography
BirthDate
Nationality
Website
Contact Information

```sql
CREATE TABLE Artist (
ArtistID INT PRIMARY KEY AUTO_INCREMENT,
Name VARCHAR(255),
Biography TEXT,
BirthDate DATE,
Nationality VARCHAR(100),
Website VARCHAR(255),
ContactInformation VARCHAR(255)
);
```

• **User**
UserID (Primary Key)
Username
Password
Email
First Name
Last Name
Date of Birth
Profile Picture
FavoriteArtworks (a list of references to ArtworkIDs)

```sql
CREATE TABLE User (
UserID INT PRIMARY KEY AUTO_INCREMENT,
Username VARCHAR(50),
Password VARCHAR(255),
Email VARCHAR(255),
FirstName VARCHAR(100),
LastName VARCHAR(100),
DateOfBirth DATE,
ProfilePicture VARCHAR(255)
);
```

• **Gallery**
GalleryID (Primary Key)
Name
Description
Location
Curator (Reference to ArtistID)
OpeningHours

```
CREATE TABLE Gallery (
GalleryID INT PRIMARY KEY AUTO_INCREMENT,
Name VARCHAR(255),
Description TEXT,
Location VARCHAR(255),
Curator INT,
OpeningHours VARCHAR(255),
FOREIGN KEY (Curator) REFERENCES Artist(ArtistID)
);
```

• **Relationships:**

• **Artwork - Artist (Many-to-One)**
An artwork is created by one artist.
Artwork.ArtistID (Foreign Key) references Artist.ArtistID.

```
ALTER TABLE Artwork ADD FOREIGN KEY (ArtistID) REFERENCES
Artist(ArtistID);
```

• **User - Favorite Artwork (Many-to-Many)**
A user can have many favorite artworks, and an artwork can be a favorite of multiple users.
User_Favorite_Artwork (junction table):
UserID (Foreign Key) references User.UserID.
ArtworkID (Foreign Key) references Artwork.ArtworkID.

```
CREATE TABLE User_Favorite_Artwork (
UserID INT,
ArtworkID INT,
FOREIGN KEY (UserID) REFERENCES User(UserID),
FOREIGN KEY (ArtworkID) REFERENCES Artwork(ArtworkID),
PRIMARY KEY (UserID, ArtworkID)
);
```

• **Artist - Gallery (One-to-Many)**
An artist can be associated with multiple galleries, but a gallery can have only one curator (artist).
Gallery.ArtistID (Foreign Key) references Artist.ArtistID.

• **Artwork - Gallery (Many-to-Many)**
An artwork can be displayed in multiple galleries, and a gallery can have multiple artworks.
Artwork_Gallery (junction table):
ArtworkID (Foreign Key) references Artwork.ArtworkID.
GalleryID (Foreign Key) references Gallery.GalleryID.

```sql
CREATE TABLE Artwork_Gallery (
ArtworkID INT,
GalleryID INT,
FOREIGN KEY (ArtworkID) REFERENCES Artwork(ArtworkID),
FOREIGN KEY (GalleryID) REFERENCES Gallery(GalleryID),
PRIMARY KEY (ArtworkID, GalleryID)
);
```

Coding :

Create the model/entity classes corresponding to the schema **within package** entity **with variables declared private, constructors(default and parametrized) and getters,setters) Service Provider Interface/Abstract class**

1. Artist class

```python
class Artist:
    def __init__(self, artistID, name, biography, birthDate, nationality, website,
contactInformation):
        self.__artistID = artistID
        self.__name = name
        self.__biography = biography
        self.__birthDate = birthDate
        self.__nationality = nationality
        self.__website = website
        self.__contactInformation = contactInformation

    # Getters
    def getArtistID(self):
        return self.__artistID

    def getName(self):
        return self.__name

    def getBiography(self):
        return self.__biography

    def getBirthDate(self):
        return self.__birthDate

    def getNationality(self):
        return self.__nationality

    def getWebsite(self):
        return self.__website
    def getContactInformation(self):
        return self.__contactInformation

    # Setters
    def setName(self, name):
        self.__name = name

    def setBiography(self, biography):
        self.__biography = biography

    def setBirthDate(self, birthDate):
        self.__birthDate = birthDate

    def setNationality(self, nationality):
        self.__nationality = nationality

    def setWebsite(self, website):
        self.__website = website

    def setContactInformation(self, contactInformation):
        self.__contactInformation = contactInformation

    def __str__(self):
        return f"Artist ID: {self.__artistID}\nArtist Name: {self.__name}, Contact: 
{self.__contactInformation}\nBirth date: {self.__birthDate}, Nationality: 
{self.__nationality}, Website: {self.__website}\n"
```

2. Artwork class

```
class Artwork:
    def __init__(self, artworkID, title, description, creationDate, medium, imageURL, 
artistID):
        self.__artworkID = artworkID
        self.__title = title
        self.__description = description
        self.__creationDate = creationDate
        self.__medium = medium
        self.__imageURL = imageURL
        self.__artistID = artistID

    # Getters
    def getArtworkID(self):
        return self.__artworkID
```

```python
    def getTitle(self):
        return self.__title

    def getDescription(self):
        return self.__description

    def getCreationDate(self):
        return self.__creationDate

    def getMedium(self):
        return self.__medium

    def getImageURL(self):
        return self.__imageURL

    def getArtistID(self):
        return self.__ArtistID

    # Setters
    def setTitle(self, title):
        self.__title = title

    def setDescription(self, description):
        self.__description = description

    def setCreationDate(self, creationDate):
        self.__creationDate = creationDate

    def setMedium(self, medium):
        self.__medium = medium

    def setImageURL(self, imageURL):
        self.__imageURL = imageURL

    def setArtistID(self, ArtistID):
        self.__ArtistID = ArtistID

    def __str__(self):
        return f"Artwork ID: {self.__artworkID}\nTitle: {self.__title}, Description: 
{self.__description}\nDate: {self.__creationDate}, Medium: {self.__medium}\nURL: 
{self.__imageURL}, Artist ID: {self.__artistID}\n"
```

3. Gallery class

```python
class Gallery:
    def __init__(self, galleryID, name, description, location, curator, openingHours):
        self.__galleryID = galleryID
        self.__name = name
        self.__description = description
        self.__location = location
        self.__curator = curator
        self.__openingHours = openingHours

    # Getters
    def getGalleryID(self):
        return self.__galleryID

    def getName(self):
        return self.__name

    def getDescription(self):
        return self.__description

    def getLocation(self):
        return self.__location

    def getCurator(self):
        return self.__curator

    def getOpeningHours(self):
        return self.__openingHours

    # Setters
    def setName(self, name):
        self.__name = name

    def setDescription(self, description):
        self.__description = description

    def setLocation(self, location):
        self.__location = location

    def setCurator(self, curator):
        self.__curator = curator

    def setOpeningHours(self, openingHours):
        self.__openingHours = openingHours
    def __str__(self):
        return f"Gallery ID: {self.__galleryID}\nName: {self.__name}, Description: {self.__description}\nLocation: {self.__location}, Curator: {self.__curator}, Opening Hours: {self.__openingHours}\n"
```

4. User class

```python
class User:
    def __init__(self, userID, username, password, email, firstName, lastName, birthDate, profilePicture):
        self.__userID = userID
        self.__username = username
        self.__password = password
        self.__email = email
        self.__firstName = firstName
        self.__lastName = lastName
        self.__birthDate = birthDate
        self.__profilePicture = profilePicture

    # Getters
    def getUserID(self):
        return self.__userID

    def getUsername(self):
        return self.__username

    def getPassword(self):
        return self.__password

    def getEmail(self):
        return self.__email

    def getFirstName(self):
        return self.__firstName

    def getLastName(self):
        return self.__lastName

    def getBirthDate(self):
        return self.__birthDate

    def getProfilePicture(self):
        return self.__profilePicture
    # Setters
    def setUsername(self, username):
        self.__username = username

    def setPassword(self, password):
        self.__password = password

    def setEmail(self, email):
        self.__email = email

    def setFirstName(self, firstName):
        self.__firstName = firstName

    def setLastName(self, lastName):
        self.__lastName = lastName

    def setBirthDate(self, birthDate):
        self.__birthDate = birthDate

    def setProfilePicture(self, profilePicture):
        self.__profilePicture = profilePicture

    def __str__(self):
        return f"User ID: {self.__userID}\nUserName: {self.__username}, email id: 
{self.__email}\nFirst Name: {self.__firstName}, Last Name: {self.__lastName}\nBirth 
Date: {self.__birthDate}, Profile Pic: {self.__profilePicture}\n"
```

Keep the interfaces and implementation classes in package dao

Create **IVirtualArtGallery** Interface/abstract class with the following methods

// Artwork Management

addArtwork();
parameters- Artwork object
return type Boolean

updateArtwork();
parameters- Artwork object
return type Boolean

removeArtwork()
parameters-artworkID
return type Boolean

getArtworkById();
parameters-artworkID
return type Artwork
searchArtworks()

searchArtworks();
parameters- keyword
return type list of Artwork Object

// User Favorites

addArtworkToFavorite();
parameters- userId, artworkId
return type boolean

removeArtworkFromFavorite()
parameters- userId, artworkId
return type boolean

getUserFavoriteArtworks()
parameters- userId
return type boolean
}

```python
from abc import ABC, abstractmethod
class IVirtualArtGallery(ABC):

    @abstractmethod
    def createUser(self,user):
        pass

    @abstractmethod
    def getAllArtworks(self):
        pass

    @abstractmethod
    def addArtwork(self, artwork):
        pass

    @abstractmethod
    def updateArtwork(self, artwork):
        pass

    @abstractmethod
    def removeArtwork(self, artworkID):
        pass
    @abstractmethod
    def getArtworkById(self, artworkID):
        pass

    @abstractmethod
    def searchArtworks(self, keyword):
        pass

    @abstractmethod
    def addArtworkToFavorite(self, userID, artworkID):
        pass

    @abstractmethod
    def removeArtworkFromFavorite(self, userID, artworkID):
        pass

    @abstractmethod
    def getUserFavoriteArtworks(self, userID):
        pass

    @abstractmethod
    def displayGalleries(self):
        pass

    @abstractmethod
    def addArtist(self,artist):
        pass

    @abstractmethod
    def addGallery(self, gallery):
        pass

    @abstractmethod
    def updateGallery(self, gallery):
        pass

    @abstractmethod
    def removeGallery(self, gallery_id):
        pass

    @abstractmethod
    def searchGalleries(self, keyword):
        pass
```

7: Connect your application to the SQL database:

- Write code to establish a connection to your SQL database.
- Create a utility class **DBConnection** in a package **util** with a static variable **connection** of Type
- **Connection** and a static method **getConnection()** which returns connection. Connection properties supplied in the connection string should be read from a property file.
- Create a utility class **PropertyUtil** which contains a static method named **getPropertyString()** which reads a property fie containing connection details like hostname, dbname, username, password, port number and returns a connection string.

```
import mysql.connectorfrom mysql.connector import Error
class DBConnection:
    connection = None

    @staticmethod
    def getConnection():
        try:
            if DBConnection.connection is None or not
DBConnection.connection.is_connected():
                connection_string = PropertyUtil.getPropertyString("DBdata.txt")
                DBConnection.connection = mysql.connector.connect(**connection_string)
            return DBConnection.connection
        except Error as e:
            print("Error:", e)
            return None

class PropertyUtil:
    @staticmethod
    def getPropertyString(property_file):
        properties = {}
        with open(property_file,'r') as file:
            for line in file:
                key, value = line.strip().split('=')
                properties[key] = value
        return properties
```

```
DBdata.txt

host=localhost
user=root
password=chowdary@22
port=3306
database=VirtualArtGallery
```

8: Service implementation

1. Create a Service class **IVirtualArtGalleryImpl** in **dao** with a static variable named connection of type **Connection** which can be assigned in the constructor by invoking the **getConnection()** method in **DBConnection** class

2. Provide implementation for all the methods in the interface.

```
from VirtualArtGallery.dao.IVirtualArtGallery import IVirtualArtGallery
from VirtualArtGallery.util.dbutil import DBConnection
from VirtualArtGallery.exception.myexceptions import ArtWorkNotFoundException
from VirtualArtGallery.entity.artwork import Artwork
from VirtualArtGallery.entity.gallery import Gallery
import mysql.connector

class IVirtualArtGalleryImpl(IVirtualArtGallery):

    def __init__(self):
        self.connection = DBConnection.getConnection()

    def createUser(self,user):
        try:
            cursor = self.connection.cursor()
            query = "INSERT INTO User (username, password, email, FirstName, LastName,
DateOfBirth, ProfilePicture) VALUES (%s, %s, %s, %s, %s, %s, %s)"
            values = (user.getUsername(), user.getPassword(), user.getEmail(),
user.getFirstName(), user.getLastName(), user.getBirthDate(), user.getProfilePicture())
            cursor.execute(query, values)
            self.connection.commit()

            query = "SELECT max(userID) FROM User"
            cursor.execute(query)
            uid = cursor.fetchone()
            self.connection.commit()
            cursor.close()
            return [True, uid]

        except mysql.connector.Error as err:
            print("Error adding user:", err)
            return False

    def getAllArtworks(self):
        try:
            cursor = self.connection.cursor()
            query = "SELECT * FROM Artwork"
            cursor.execute(query)
            artwork_data = cursor.fetchall()
            cursor.close()
            artworks = [Artwork(*data) for data in artwork_data]
            return artworks
        except mysql.connector.Error as err:
            print("Error:", err)
            return None

    def addArtwork(self, artwork):
        try:
            cursor = self.connection.cursor()
            query = "INSERT INTO Artwork (title, description, creationDate, medium, 
imageURL, ArtistID) VALUES (%s, %s, %s, %s, %s, %s)"
            values = (artwork.getTitle(), artwork.getDescription(), 
artwork.getCreationDate(), artwork.getMedium(), artwork.getImageURL(), 
artwork.getArtistID())
            cursor.execute(query, values)
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False

    def updateArtwork(self, artwork):
        try:
            cursor = self.connection.cursor()
            query = "UPDATE Artwork SET title=%s, description=%s, creationDate=%s, 
medium=%s, imageURL=%s, ArtistID=%s WHERE artworkID=%s"
            values = (artwork.getTitle(), artwork.getDescription(), 
artwork.getCreationDate(), artwork.getMedium(), artwork.getImageURL(), 
artwork.getArtistID(), artwork.getArtworkID())
            cursor.execute(query, values)
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False

    def removeArtwork(self, artworkID):
        try:
            cursor = self.connection.cursor()
            query = "DELETE FROM Artwork WHERE artworkID=%s"
            cursor.execute(query, (artworkID,))
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False

    def getArtworkById(self, artworkID):
        try:
            cursor = self.connection.cursor()
            query = "SELECT * FROM Artwork WHERE artworkID=%s"
            cursor.execute(query, (artworkID,))
            result = cursor.fetchone()
            if result:
                artwork = Artwork(*result)
                return artwork

        except mysql.connector.Error as err:
            print("Error:", err)
            return None

    def searchArtworks(self, keyword):
        try:
            cursor = self.connection.cursor()
            query = "SELECT * FROM Artwork WHERE title LIKE %s OR description LIKE %s"
            cursor.execute(query, (f"%{keyword}%", f"%{keyword}%"))
            artwork_data = cursor.fetchall()
            cursor.close()
            artworks = [Artwork(*data) for data in artwork_data]
            return artworks
        except mysql.connector.Error as err:
            print("Error:", err)
            return []
    def addArtworkToFavorite(self, userId, artworkId):
        try:
            cursor = self.connection.cursor()
            query = "INSERT INTO User_Favorite_Artwork (userID, artworkID) VALUES 
(%s, %s)"
            cursor.execute(query, (userId, artworkId))
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False

    def removeArtworkFromFavorite(self, userId, artworkId):
        try:
            cursor = self.connection.cursor()
            query = "DELETE FROM User_Favorite_Artwork WHERE userID=%s AND 
artworkID=%s"
            cursor.execute(query, (userId, artworkId))
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False

    def getUserFavoriteArtworks(self, userId):
        try:
            cursor = self.connection.cursor()
            query = "SELECT artworkID, title, description, creationDate, medium, 
imageURL, artistID FROM User_Favorite_Artwork uf join artwork aw on 
uf.artworkID=aw.artworkID WHERE userID=%s"
            cursor.execute(query, (userId,))
            artwork_data = cursor.fetchall()
            cursor.close()
            favoriteArtworks = [Artwork(*data) for data in artwork_data]
            return favoriteArtworks
        except mysql.connector.Error as err:
            print("Error:", err)
            return []
```

```
 def displayGalleries(self):
        try:
            cursor = self.connection.cursor()
            query = "SELECT * FROM gallery"
            cursor.execute(query)
            gallery_data = cursor.fetchall()
            cursor.close()
            galleries = [Gallery(*data) for data in gallery_data]
            return galleries
        except mysql.connector.Error as err:
            print("Error:", err)
            return []

    def addArtist(self,artist):
        try:
            cursor = self.connection.cursor()
            query = "INSERT INTO Artist (name, biography, birthDate, nationality, 
website, ContactInformation) VALUES (%s, %s, %s, %s, %s, %s)"
            values = (artist.getName(), artist.getBiography(), artist.getBirthDate(), 
artist.getNationality(),
                      artist.getWebsite(), artist.getContactInformation())
            cursor.execute(query, values)
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False

    def addGallery(self, gallery):
        try:
            cursor = self.connection.cursor()
            query = "INSERT INTO Gallery (name, description, location, openingHours, 
curator) VALUES (%s, %s, %s, %s, %s)"
            values = (gallery.getName(), gallery.getDescription(), 
gallery.getLocation(), gallery.getOpeningHours(),
                      gallery.getCurator())
            cursor.execute(query, values)
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False
    def updateGallery(self, gallery):
        try:
            cursor = self.connection.cursor()
            query = "UPDATE Gallery SET name = %s, description = %s, location = %s, 
openingHours = %s, curator = %s WHERE galleryID = %s"
            values = (gallery.getName(), gallery.getDescription(), 
gallery.getLocation(), gallery.getOpeningHours(),
                      gallery.getCurator(), gallery.getGalleryID())
            cursor.execute(query, values)
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False

    def removeGallery(self, gallery_id):
        try:
            cursor = self.connection.cursor()
            query = "DELETE FROM Gallery WHERE galleryID = %s"
            cursor.execute(query, (gallery_id,))
            self.connection.commit()
            cursor.close()
            return True
        except mysql.connector.Error as err:
            print("Error:", err)
            return False

    def searchGalleries(self, keyword):
        try:
            cursor = self.connection.cursor()
            query = "SELECT * FROM Gallery WHERE name LIKE %s OR description LIKE %s"
            cursor.execute(query, ('%' + keyword + '%', '%' + keyword + '%'))
            gallery_data = cursor.fetchall()
            cursor.close()
            galleries=[Gallery(*data) for data in gallery_data]
            return galleries
        except mysql.connector.Error as err:
            print("Error:", err)
            return []
```

9: Exception Handling

Create the exceptions in package **myexceptions** Define the following custom exceptions and throw them in methods whenever needed. Handle all the exceptions in main method,

1. **ArtWorkNotFoundException** :throw this exception when user enters an invalid id which doesn't exist in db

3. **UserNotFoundException** :throw this exception when user enters an invalid id which doesn't exist in db

```
class ArtWorkNotFoundException(Exception):
    def __init__(self, message="Artwork not found."):
        self.message = message
        super().__init__(self.message)
class UserNotFoundException(Exception):
    def __init__(self, message="User not found."):
        self.message = message
        super().__init__(self.message)
class ArtistNotFoundException(Exception):
    def __init__(self, message="Artist not found."):
        self.message = message
        super().__init__(self.message)
class GalleryNotFoundException(Exception):
    def __init__(self, message="Gallery not found."):
        self.message = message
        super().__init__(self.message)
```

9. Main Method

Create class named MainModule with main method in main package.
Trigger all the methods in service implementation class.

```
from VirtualArtGallery.dao.IVirtualArtGalleryimpl import IVirtualArtGalleryImpl
from VirtualArtGallery.exception.myexceptions import ArtWorkNotFoundException,
UserNotFoundException

from VirtualArtGallery.exception.myexceptions import ArtistNotFoundException,
GalleryNotFoundException

from VirtualArtGallery.entity.artwork import Artwork
from VirtualArtGallery.entity.user import User
from VirtualArtGallery.entity.gallery import Gallery
from VirtualArtGallery.entity.artist import Artist
```

```python
class MainModule:
    def __init__(self):
        self.virtual_gallery = IVirtualArtGalleryImpl()

    # artwork management
    def create_user(self):
        try:
            username = input("Enter username: ")
            email = input("Enter email: ")
            password = input("Enter password: ")
            firstName = input("Enter first Name: ")
            lastName = input("Enter last Name: ")
            birthDate = input("Enter : birthDate")
            profilePicture = input("select profilePicture(enter url): ")
            user = User(None, username, password, email, firstName, lastName,
birthDate, profilePicture)

            result = self.virtual_gallery.createUser(user)
            if result[0]:
                print("User created successfully!")
                return result[1]

        except Exception as e:
            print(f"Error creating user: {e}")
            return None

    def display_artworks(self):
        try:
            artworks = self.virtual_gallery.getAllArtworks()
            if artworks:
                print("Artworks:")
                for artwork in artworks:
                    print(artwork)
                return True
            else:
                print("No artworks found.")
                return False
        except Exception as e:
            print(f"Error displaying Artworks: {e}")
    def add_artwork(self):
        try:
            title = input("Enter Title: ")
            description = input("Enter Description: ")
            creation_date = input("Enter Creation Date: ")
            medium = input("Enter Medium: ")
            image_url = input("Enter Image URL: ")
            artist_id = int(input("Enter Artist ID: "))

            artwork = Artwork(None, title, description, creation_date, medium,
image_url, artist_id)
            if self.virtual_gallery.addArtwork(artwork):
                print("Artwork Added Successfully")
        except Exception as e:
            print(f"Error adding artwork: {e}")

    def update_artwork(self):
        arts = self.display_artworks()
        if not arts:
            return

        try:
            artwork_id = int(input("Enter Artwork ID to update: "))
            title = input("Enter new Title: ")
            description = input("Enter new Description: ")
            creation_date = input("Enter new Creation Date: ")
            medium = input("Enter new Medium: ")
            image_url = input("Enter new Image URL: ")
            artist_id = int(input("Enter Artist ID: "))

            artwork = Artwork(artwork_id, title, description, creation_date, medium,
image_url, artist_id)

            if self.virtual_gallery.updateArtwork(artwork):
                print("Artwork updated successfully!")

        except ArtWorkNotFoundException as e:
            print(f"Artwork not found: {e}")
        except Exception as e:
            print(f"Error updating artwork: {e}")
```

```
def remove_artwork(self):
    arts = self.display_artworks()
    if not arts:
        return

    try:
        artwork_id = int(input("\nEnter Artwork ID to remove: "))
        if self.virtual_gallery.removeArtwork(artwork_id):
            print("Artwork removed successfully!")

    except ArtWorkNotFoundException as e:
        print(f"Artwork not found: {e}")
    except Exception as e:
        print(f"Error removing artwork: {e}")

def get_artwork_by_id(self):
    try:
        artwork_id = input("Enter Artwork ID to retrieve: ")
        artwork = self.virtual_gallery.getArtworkById(artwork_id)
        if artwork:
            print("Artwork details:")
            print(artwork)
        else:
            print(f"Artwork {artwork_id} not found")

    except ArtWorkNotFoundException as e:
        print(f"Artwork not found: {e}")
    except Exception as e:
        print(f"Error retrieving artwork: {e}")

def search_artworks(self):
    keyword = input("Enter keyword to search artworks: ")
    try:
        artworks = self.virtual_gallery.searchArtworks(keyword)
        if artworks:
            print("Search results:")
            for artwork in artworks:
                print(artwork)
        else:
            print("No Artworks Found")
    except Exception as e:
        print(f"Error searching artworks: {e}")
```

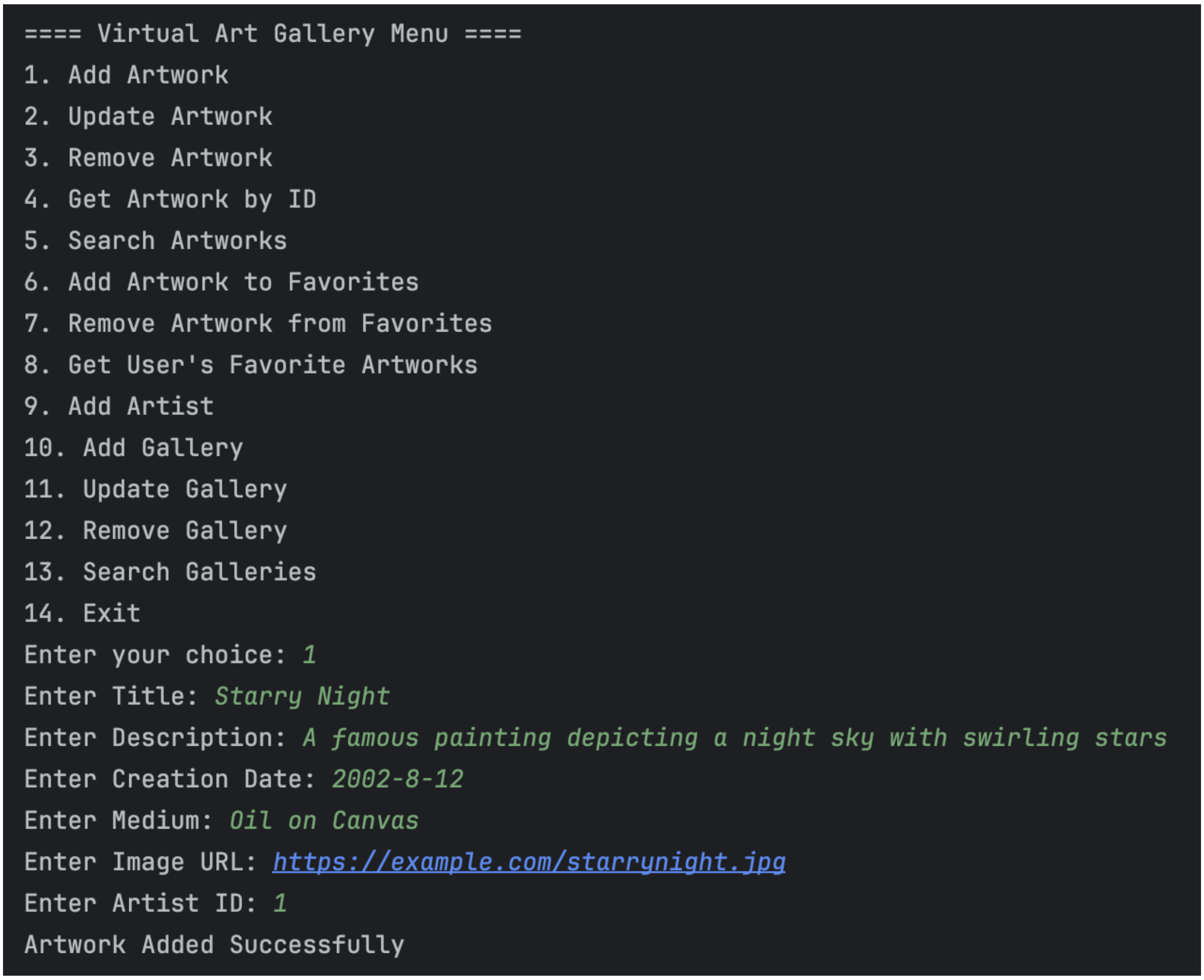
```python
def add_artwork_to_favorite(self):
    try:
        existing_user = input("Are you an existing user? (y/n): ").lower()
        if existing_user == 'y':
            user_id = input("Enter your user ID: ")
        else:
            user_id = self.create_user()
            print(f"Your user id = {user_id}")

        if not user_id:
            print("Failed to add artwork to favorites. User ID not provided.")
            return

        arts = self.display_artworks()

        if not arts:
            print("no artworks found to add")
            return

        artwork_id = input("Enter Artwork ID to add to favorites: ")
        if self.virtual_gallery.addArtworkToFavorite(user_id, artwork_id):
            print("Artwork added to favorites successfully!")

    except (ArtWorkNotFoundException, UserNotFoundException) as e:
        print(f"Error adding artwork to favorites: {e}")
    except Exception as e:
        print(f"Error adding artwork to favorites: {e}")

def remove_artwork_from_favorite(self):
    try:
        user_id = self.get_user_favorite_artworks()
        artwork_id = input("Enter Artwork ID to remove from favorites: ")
        if self.virtual_gallery.removeArtworkFromFavorite(user_id, artwork_id):
            print("Artwork removed from favorites successfully!")
    except (ArtWorkNotFoundException, UserNotFoundException) as e:
        print(f"Error removing artwork from favorites: {e}")
    except Exception as e:
        print(f"Error removing artwork from favorites: {e}")
    def get_user_favorite_artworks(self):
        try:
            user_id = input("Enter User ID to retrieve favorite artworks: ")
            favorite_artworks = self.virtual_gallery.getUserFavoriteArtworks(user_id)
            if favorite_artworks:
                print("User's favorite artworks:")
                for artwork in favorite_artworks:
                    print(artwork)
            else:
                print("No favourite artworks found")

            return user_id
        except UserNotFoundException as e:
            print(f"User not found: {e}")
        except Exception as e:
            print(f"Error retrieving user's favorite artworks: {e}")

# Gallery management
    def display_galleries(self):
        try:
            galleries = self.virtual_gallery.displayGalleries()
            if galleries:
                print("Available Galleries:")
                for gallery in galleries:
                    print(gallery)
                    return True
            else:
                print("No galleries found.")
                return False
        except Exception as e:
            print(f"Error displaying galleries: {e}")

    def create_artist(self):
        try:
            name = input("Enter artist name: ")
            biography = input("Enter artist biography: ")
            birth_date = input("Enter artist birth date: ")
            nationality = input("Enter artist nationality: ")
            website = input("Enter artist website: ")
            contact_info = input("Enter artist contact information: ")
            artist = Artist(None, name, biography, birth_date, nationality, website,
contact_info)
            if self.virtual_gallery.addArtist(artist):
                print("Artist added successfully!")
        except Exception as e:
            print(f"Error adding artist: {e}")
    def create_gallery(self):
        try:
            name = input("Enter gallery name: ")
            description = input("Enter gallery description: ")
            location = input("Enter gallery location: ")
            opening_hours = input("Enter opening hours: ")
            curator = input("Enter curator name : ")
            gallery = Gallery(None, name, description, location, curator,
opening_hours)
            if self.virtual_gallery.addGallery(gallery):
                print("Gallery created successfully!")
        except Exception as e:
            print(f"Error creating gallery: {e}")

    def update_gallery(self):
        gall = self.display_galleries()
        if not gall:
            return

        try:
            gallery_id = int(input("Enter the ID of the gallery you want to update: "))
            name = input("Enter new name : ")
            description = input("Enter new description: ")
            location = input("Enter new location : ")
            opening_hours = input("Enter new opening hours : ")
            curator = input("Enter new curator name : ")
            gallery = Gallery(gallery_id, name, description, location, curator,
opening_hours)
            if self.virtual_gallery.updateGallery(gallery):
                print("Gallery updated successfully!")
            else:
                print("Failed to update gallery.")
        except GalleryNotFoundException as e:
            print(f"Gallery not found: {e}")
        except Exception as e:
            print(f"Error updating gallery: {e}")

    def remove_gallery(self):
        gall = self.display_galleries()
        if not gall:
            return

        try:
            gallery_id = int(input("Enter the ID of the gallery you want to remove: "))
            self.virtual_gallery.removeGallery(gallery_id)
            print("Gallery removed successfully!")
        except GalleryNotFoundException as e:
            print(f"Gallery not found: {e}")
        except Exception as e:
            print(f"Error removing gallery: {e}")

    def search_galleries(self):
        keyword = input("Enter keyword to search galleries: ")
        try:
            galleries = self.virtual_gallery.searchGalleries(keyword)
            if galleries:
                print("Search Results:")
                for gallery in galleries:
                    print(gallery)
            else:
                print("No galleries found matching the keyword.")
        except Exception as e:
            print(f"Error searching galleries: {e}")

    def main(self):
        while True:
            print("\n==== Virtual Art Gallery Menu ====")
            print("1. Add Artwork")
            print("2. Update Artwork")
            print("3. Remove Artwork")
            print("4. Get Artwork by ID")
            print("5. Search Artworks")
            print("6. Add Artwork to Favorites")
            print("7. Remove Artwork from Favorites")
            print("8. Get User's Favorite Artworks")
            print("9. Add Artist")
            print("10. Add Gallery")
            print("11. Update Gallery")
            print("12. Remove Gallery")
            print("13. Search Galleries")
            print("14. Exit")
            choice = input("Enter your choice: ")

            if choice == "1":
                self.add_artwork()
            elif choice == "2":
                self.update_artwork()
            elif choice == "3":
                self.remove_artwork()
            elif choice == "4":
                self.get_artwork_by_id()
            elif choice == "5":
                self.search_artworks()
            elif choice == "6":
                self.add_artwork_to_favorite()
            elif choice == "7":
                self.remove_artwork_from_favorite()
            elif choice == "8":
                self.get_user_favorite_artworks()
            elif choice == "9":
                self.create_artist()
            elif choice == "10":
                self.create_gallery()
            elif choice == "11":
                self.update_gallery()
            elif choice == "12":
                self.remove_gallery()
            elif choice == "13":
                self.search_galleries()
            elif choice == "14":
                print("Exiting...")
                break
            else:
                print("Invalid choice. Please try again.")
```

10. Unit Testing

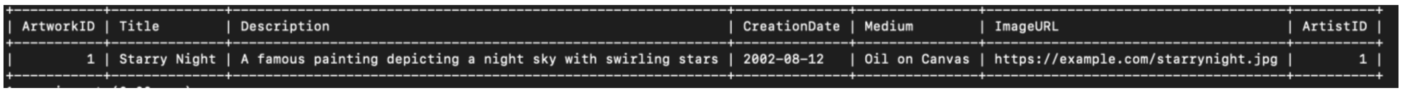
Creating Unit test cases for a Virtual Art Gallery system is essential to ensure that the system functions correctly. Below are sample test case questions that can serve as a starting point for your JUnit test suite:

1. Artwork Management:

a. Test the ability to upload a new artwork to the gallery.

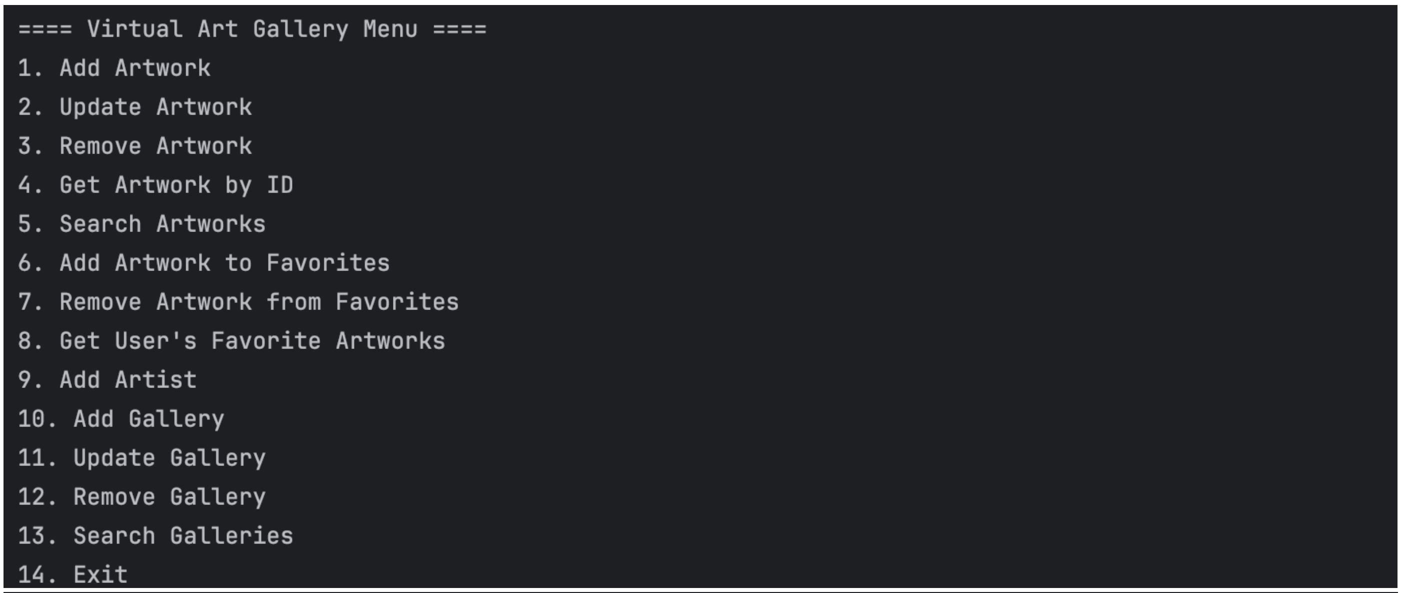


```
==== Virtual Art Gallery Menu ====
1. Add Artwork
2. Update Artwork
3. Remove Artwork
4. Get Artwork by ID
5. Search Artworks
6. Add Artwork to Favorites
7. Remove Artwork from Favorites
8. Get User's Favorite Artworks
9. Add Artist
10. Add Gallery
11. Update Gallery
12. Remove Gallery
13. Search Galleries
14. Exit
Enter your choice: 1
Enter Title: Starry Night
Enter Description: A famous painting depicting a night sky with swirling stars
Enter Creation Date: 2002-8-12
Enter Medium: Oil on Canvas
Enter Image URL: https://example.com/starrynight.jpg
Enter Artist ID: 1
Artwork Added Successfully
```

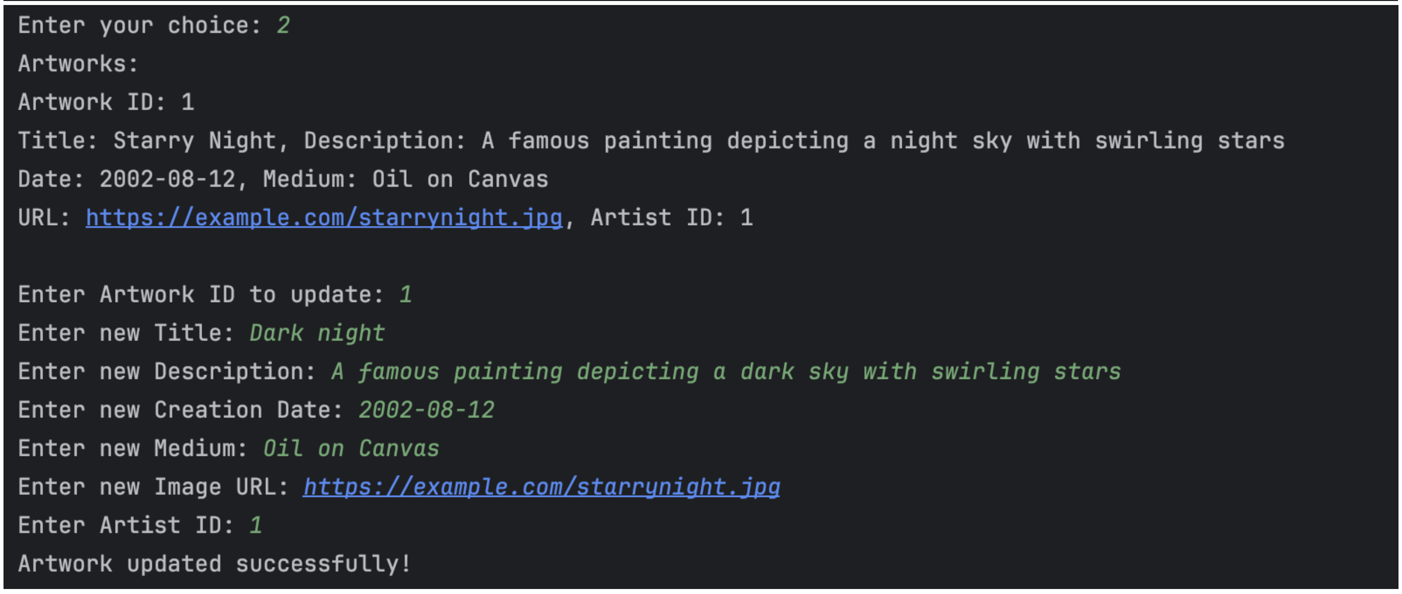


```
+-----------+--------------+--------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
| ArtworkID | Title        | Description                                                  | CreationDate | Medium        | ImageURL                            | ArtistID |
+-----------+--------------+--------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
|         1 | Starry Night | A famous painting depicting a night sky with swirling stars  | 2002-08-12   | Oil on Canvas | https://example.com/starrynight.jpg |        1 |
+-----------+--------------+--------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
```

b. Verify that updating artwork details works correctly.

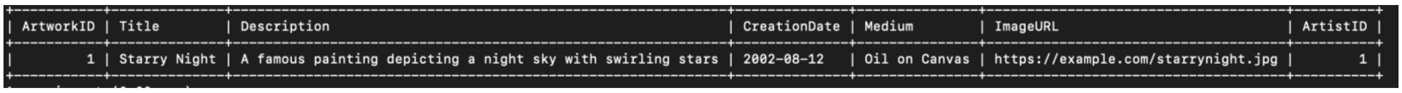


```
==== Virtual Art Gallery Menu ====
1. Add Artwork
2. Update Artwork
3. Remove Artwork
4. Get Artwork by ID
5. Search Artworks
6. Add Artwork to Favorites
7. Remove Artwork from Favorites
8. Get User's Favorite Artworks
9. Add Artist
10. Add Gallery
11. Update Gallery
12. Remove Gallery
13. Search Galleries
14. Exit
```



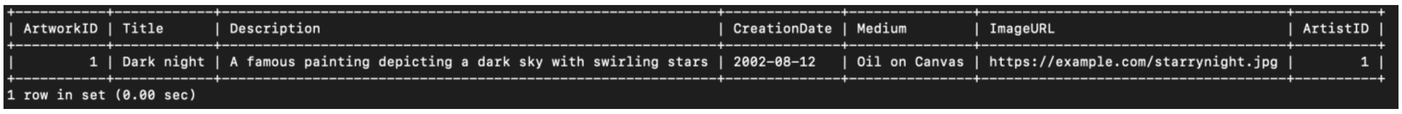
```
Enter your choice: 2
Artworks:
Artwork ID: 1
Title: Starry Night, Description: A famous painting depicting a night sky with swirling stars
Date: 2002-08-12, Medium: Oil on Canvas
URL: https://example.com/starrynight.jpg, Artist ID: 1

Enter Artwork ID to update: 1
Enter new Title: Dark night
Enter new Description: A famous painting depicting a dark sky with swirling stars
Enter new Creation Date: 2002-08-12
Enter new Medium: Oil on Canvas
Enter new Image URL: https://example.com/starrynight.jpg
Enter Artist ID: 1
Artwork updated successfully!
```

old :



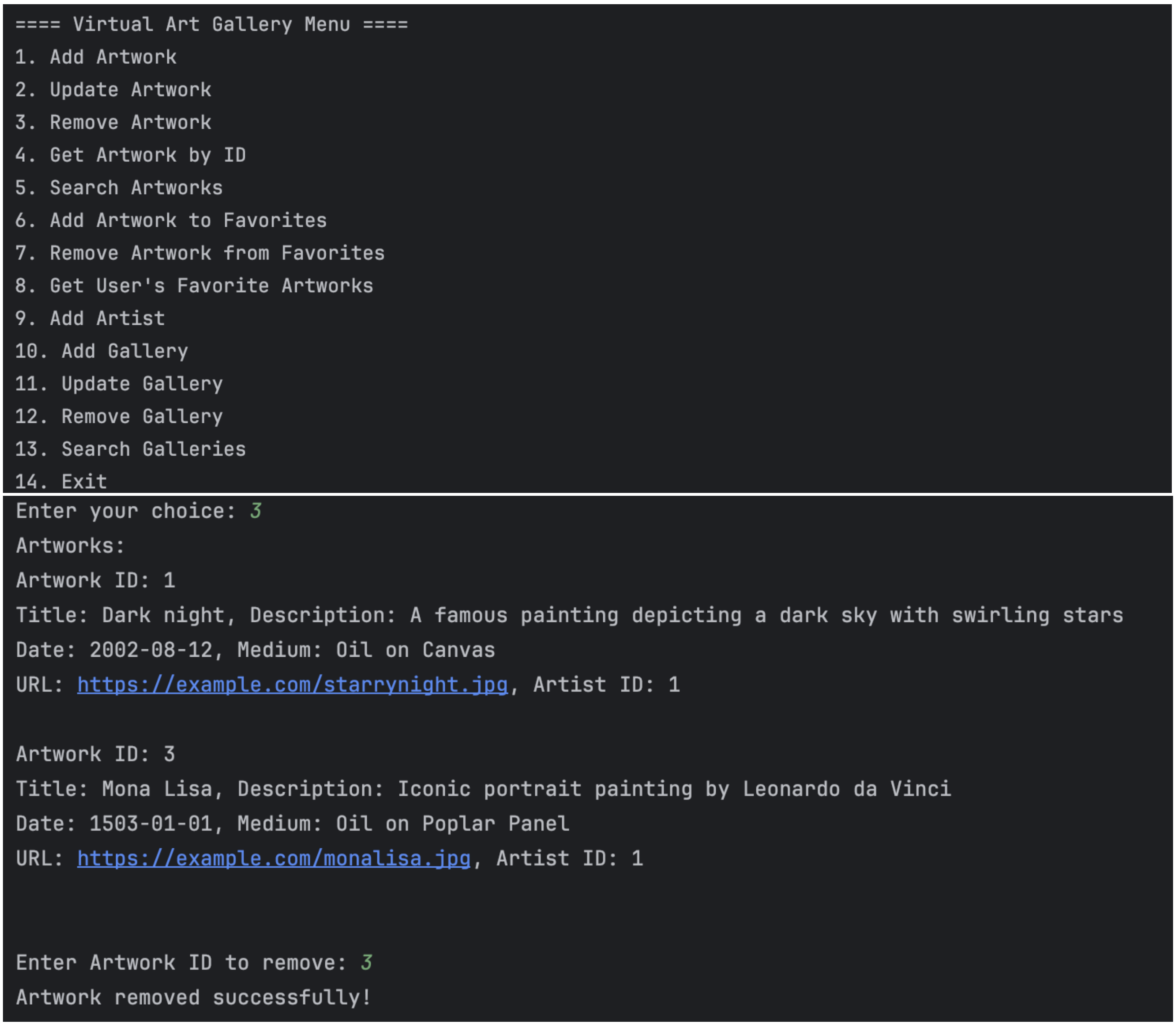
```
+-----------+--------------+-------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
| ArtworkID | Title        | Description                                                 | CreationDate | Medium        | ImageURL                            | ArtistID |
+-----------+--------------+-------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
|         1 | Starry Night | A famous painting depicting a night sky with swirling stars | 2002-08-12   | Oil on Canvas | https://example.com/starrynight.jpg |        1 |
+-----------+--------------+-------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
```

updated:



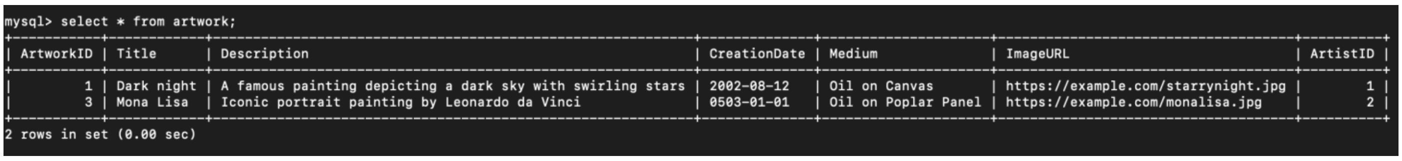
```
+-----------+------------+------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
| ArtworkID | Title      | Description                                                | CreationDate | Medium        | ImageURL                            | ArtistID |
+-----------+------------+------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
|         1 | Dark night | A famous painting depicting a dark sky with swirling stars | 2002-08-12   | Oil on Canvas | https://example.com/starrynight.jpg |        1 |
+-----------+------------+------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
1 row in set (0.00 sec)
```

c. Test removing an artwork from the gallery.



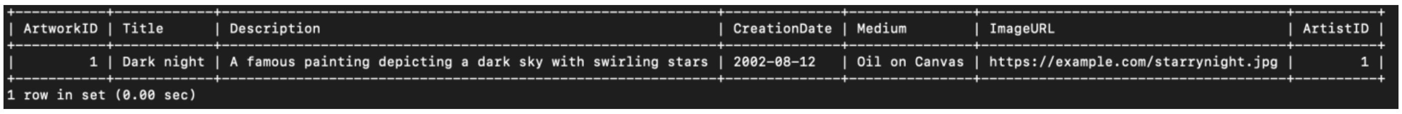
```
==== Virtual Art Gallery Menu ====
1. Add Artwork
2. Update Artwork
3. Remove Artwork
4. Get Artwork by ID
5. Search Artworks
6. Add Artwork to Favorites
7. Remove Artwork from Favorites
8. Get User's Favorite Artworks
9. Add Artist
10. Add Gallery
11. Update Gallery
12. Remove Gallery
13. Search Galleries
14. Exit
Enter your choice: 3
Artworks:
Artwork ID: 1
Title: Dark night, Description: A famous painting depicting a dark sky with swirling stars
Date: 2002-08-12, Medium: Oil on Canvas
URL: https://example.com/starrynight.jpg, Artist ID: 1

Artwork ID: 3
Title: Mona Lisa, Description: Iconic portrait painting by Leonardo da Vinci
Date: 1503-01-01, Medium: Oil on Poplar Panel
URL: https://example.com/monalisa.jpg, Artist ID: 1


Enter Artwork ID to remove: 3
Artwork removed successfully!
```

old:



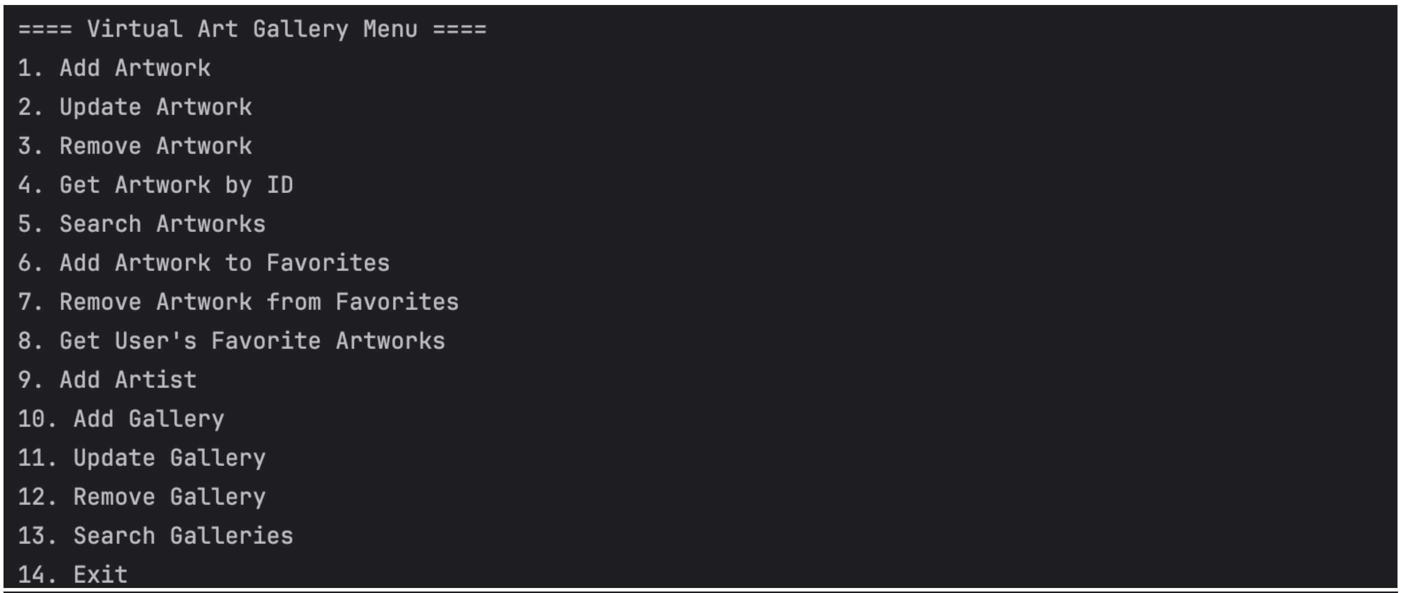
```
mysql> select * from artwork;
+-----------+------------+------------------------------------------------------------+--------------+----------------------+-------------------------------------+----------+
| ArtworkID | Title      | Description                                                | CreationDate | Medium               | ImageURL                            | ArtistID |
+-----------+------------+------------------------------------------------------------+--------------+----------------------+-------------------------------------+----------+
|         1 | Dark night | A famous painting depicting a dark sky with swirling stars | 2002-08-12   | Oil on Canvas        | https://example.com/starrynight.jpg |        1 |
|         3 | Mona Lisa  | Iconic portrait painting by Leonardo da Vinci              | 0503-01-01   | Oil on Poplar Panel  | https://example.com/monalisa.jpg    |        2 |
+-----------+------------+------------------------------------------------------------+--------------+----------------------+-------------------------------------+----------+
2 rows in set (0.00 sec)
```

Removed:

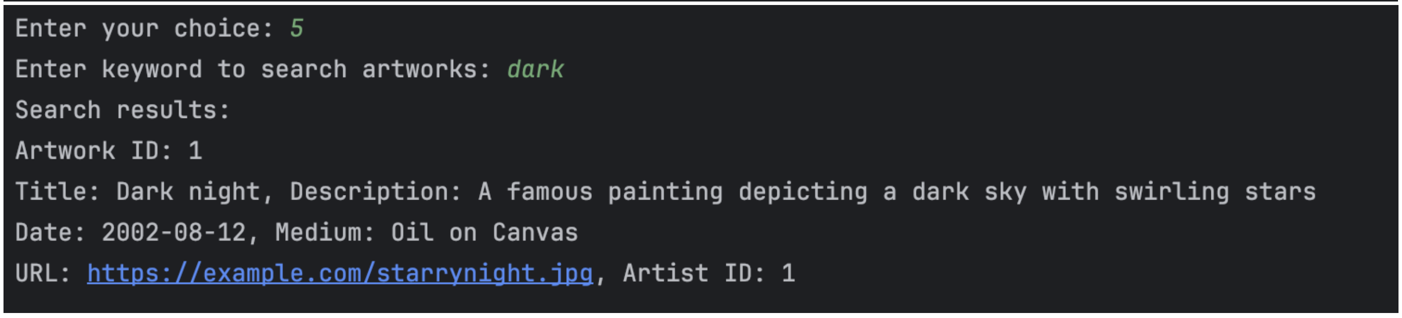


```
+-----------+------------+------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
| ArtworkID | Title      | Description                                                | CreationDate | Medium        | ImageURL                            | ArtistID |
+-----------+------------+------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
|         1 | Dark night | A famous painting depicting a dark sky with swirling stars | 2002-08-12   | Oil on Canvas | https://example.com/starrynight.jpg |        1 |
+-----------+------------+------------------------------------------------------------+--------------+---------------+-------------------------------------+----------+
1 row in set (0.00 sec)
```

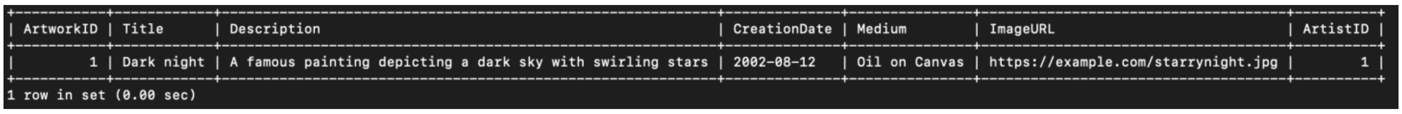
d. Check if searching for artworks returns the expected results.



```
==== Virtual Art Gallery Menu ====
1. Add Artwork
2. Update Artwork
3. Remove Artwork
4. Get Artwork by ID
5. Search Artworks
6. Add Artwork to Favorites
7. Remove Artwork from Favorites
8. Get User's Favorite Artworks
9. Add Artist
10. Add Gallery
11. Update Gallery
12. Remove Gallery
13. Search Galleries
14. Exit
```



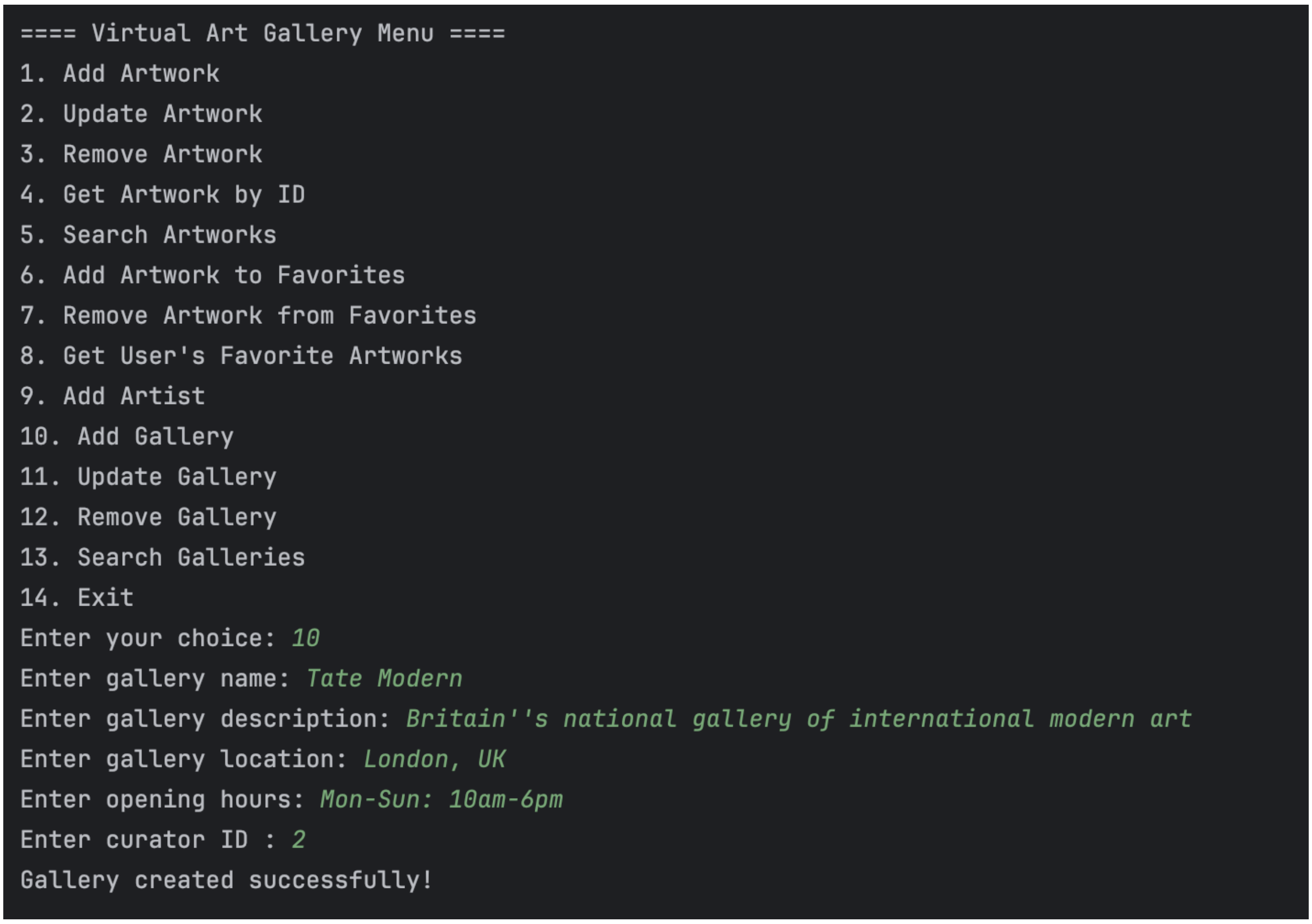
```
Enter your choice: 5
Enter keyword to search artworks: dark
Search results:
Artwork ID: 1
Title: Dark night, Description: A famous painting depicting a dark sky with swirling stars
Date: 2002-08-12, Medium: Oil on Canvas
URL: https://example.com/starrynight.jpg, Artist ID: 1
```



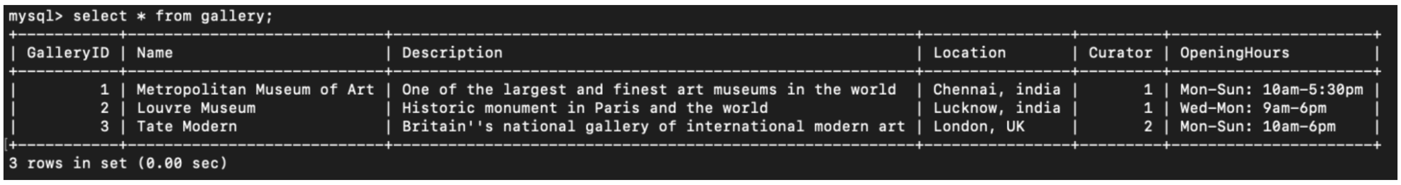
```
+-----------+------------+----------------------------------------------------------+--------------+--------------+-------------------------------------+----------+
| ArtworkID | Title      | Description                                              | CreationDate | Medium       | ImageURL                            | ArtistID |
+-----------+------------+----------------------------------------------------------+--------------+--------------+-------------------------------------+----------+
|         1 | Dark night | A famous painting depicting a dark sky with swirling stars | 2002-08-12   | Oil on Canvas | https://example.com/starrynight.jpg |        1 |
+-----------+------------+----------------------------------------------------------+--------------+--------------+-------------------------------------+----------+
1 row in set (0.00 sec)
```

2. Gallery Management:

a. Test creating a new gallery.

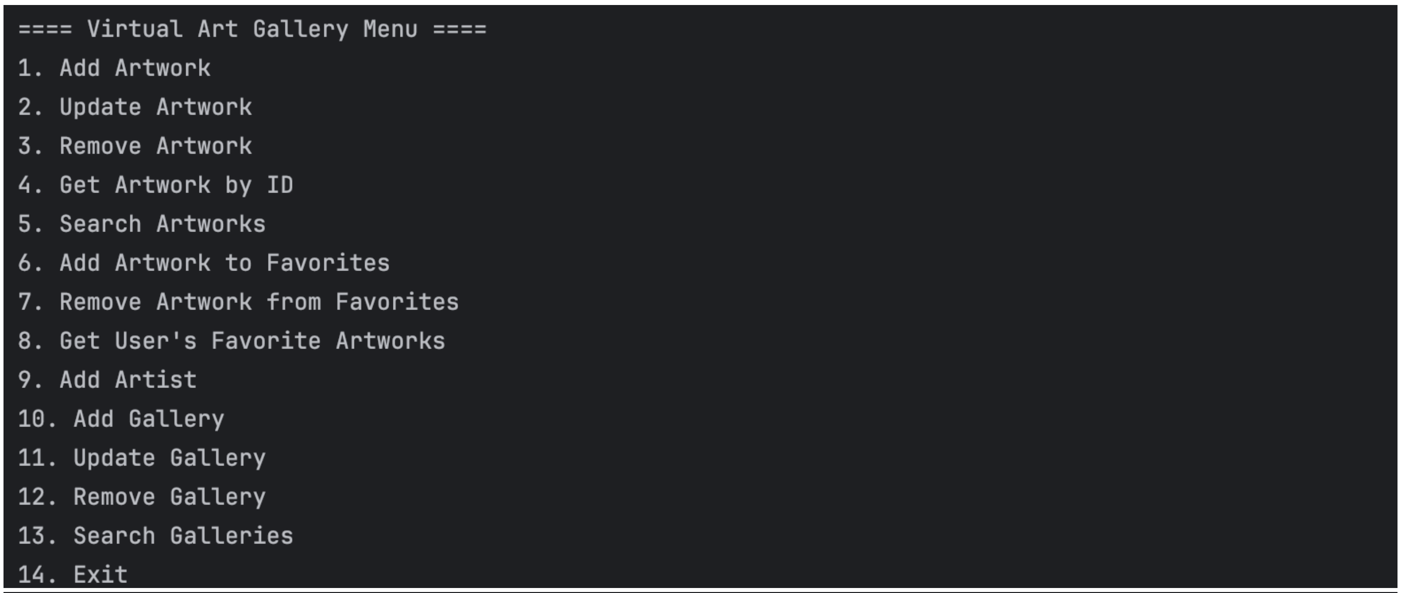


```
==== Virtual Art Gallery Menu ====
1. Add Artwork
2. Update Artwork
3. Remove Artwork
4. Get Artwork by ID
5. Search Artworks
6. Add Artwork to Favorites
7. Remove Artwork from Favorites
8. Get User's Favorite Artworks
9. Add Artist
10. Add Gallery
11. Update Gallery
12. Remove Gallery
13. Search Galleries
14. Exit
Enter your choice: 10
Enter gallery name: Tate Modern
Enter gallery description: Britain''s national gallery of international modern art
Enter gallery location: London, UK
Enter opening hours: Mon-Sun: 10am-6pm
Enter curator ID : 2
Gallery created successfully!
```

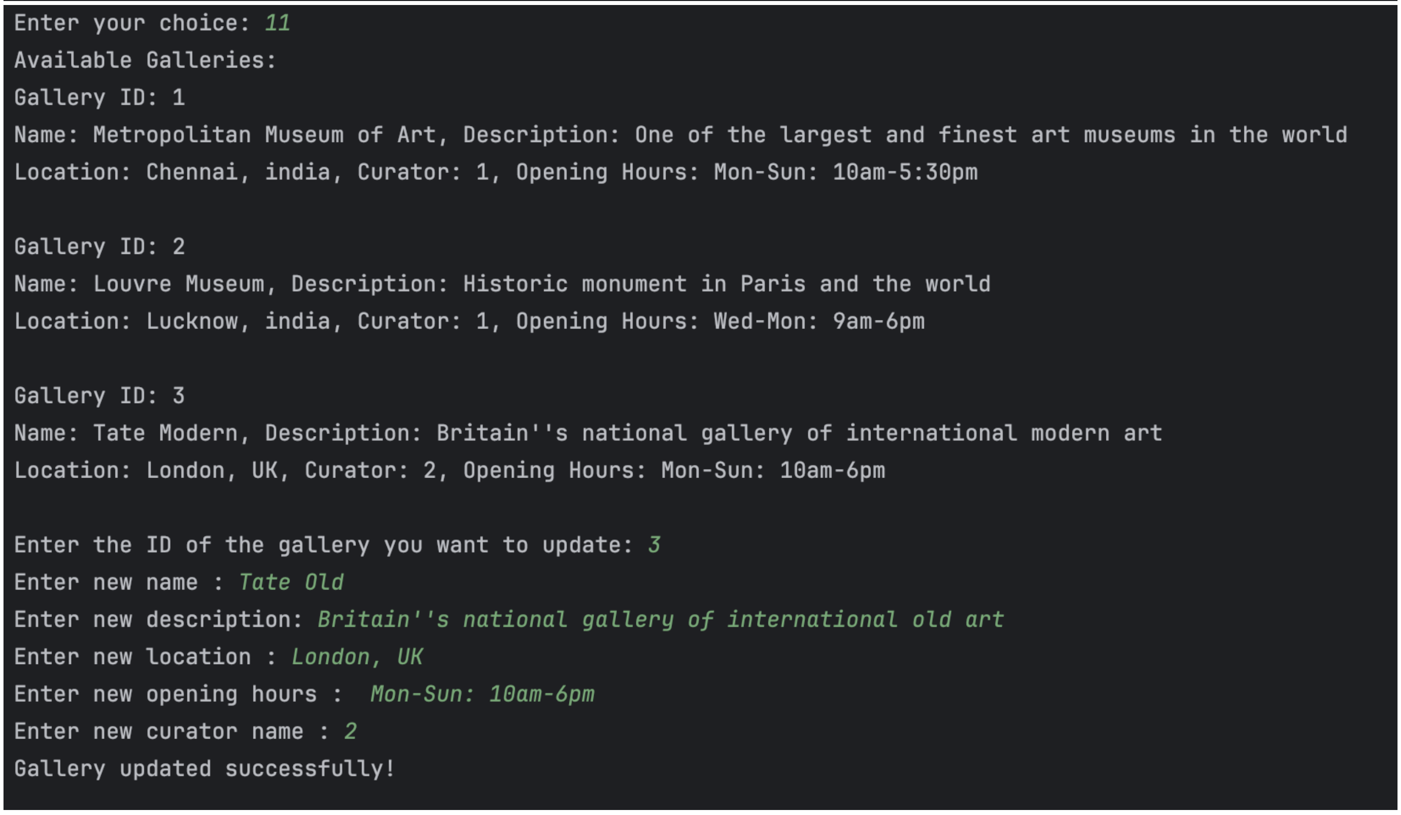


```
mysql> select * from gallery;
+-----------+---------------------------+---------------------------------------------------------+----------------+---------+----------------------+
| GalleryID | Name                      | Description                                             | Location       | Curator | OpeningHours         |
+-----------+---------------------------+---------------------------------------------------------+----------------+---------+----------------------+
|         1 | Metropolitan Museum of Art | One of the largest and finest art museums in the world  | Chennai, india |       1 | Mon-Sun: 10am-5:30pm |
|         2 | Louvre Museum             | Historic monument in Paris and the world                | Lucknow, india |       1 | Wed-Mon: 9am-6pm     |
|         3 | Tate Modern               | Britain''s national gallery of international modern art | London, UK     |       2 | Mon-Sun: 10am-6pm    |
+-----------+---------------------------+---------------------------------------------------------+----------------+---------+----------------------+
3 rows in set (0.00 sec)
```

b. Verify that updating gallery information works correctly.

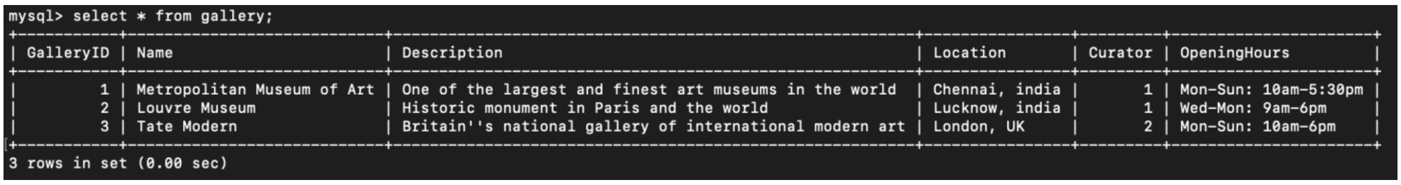


```
==== Virtual Art Gallery Menu ====
1. Add Artwork
2. Update Artwork
3. Remove Artwork
4. Get Artwork by ID
5. Search Artworks
6. Add Artwork to Favorites
7. Remove Artwork from Favorites
8. Get User's Favorite Artworks
9. Add Artist
10. Add Gallery
11. Update Gallery
12. Remove Gallery
13. Search Galleries
14. Exit
```



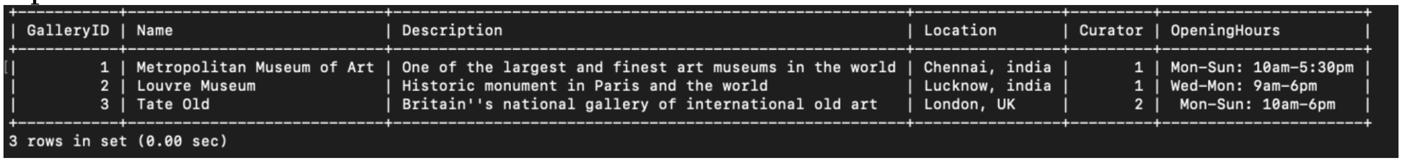
```
Enter your choice: 11
Available Galleries:
Gallery ID: 1
Name: Metropolitan Museum of Art, Description: One of the largest and finest art museums in the world
Location: Chennai, india, Curator: 1, Opening Hours: Mon-Sun: 10am-5:30pm

Gallery ID: 2
Name: Louvre Museum, Description: Historic monument in Paris and the world
Location: Lucknow, india, Curator: 1, Opening Hours: Wed-Mon: 9am-6pm

Gallery ID: 3
Name: Tate Modern, Description: Britain''s national gallery of international modern art
Location: London, UK, Curator: 2, Opening Hours: Mon-Sun: 10am-6pm

Enter the ID of the gallery you want to update: 3
Enter new name : Tate Old
Enter new description: Britain''s national gallery of international old art
Enter new location : London, UK
Enter new opening hours :  Mon-Sun: 10am-6pm
Enter new curator name : 2
Gallery updated successfully!
```

Old:



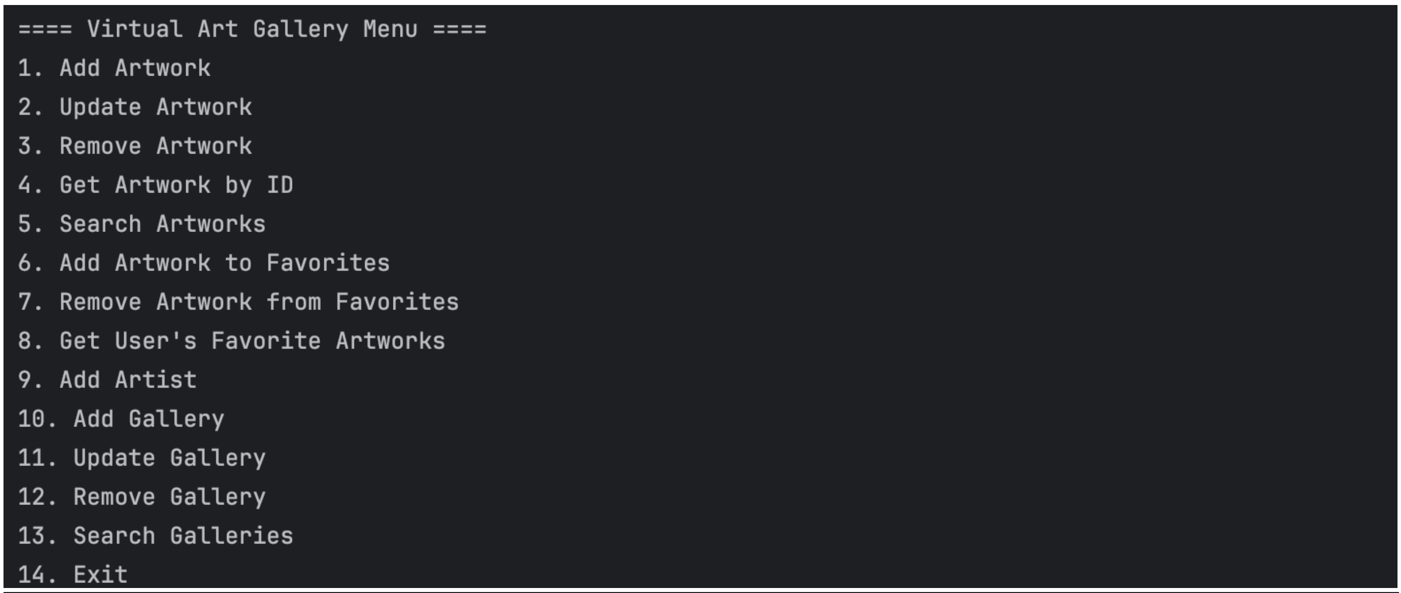
```
mysql> select * from gallery;
+-----------+----------------------------+-------------------------------------------------------+-----------------+---------+--------------------+
| GalleryID | Name                       | Description                                           | Location        | Curator | OpeningHours       |
+-----------+----------------------------+-------------------------------------------------------+-----------------+---------+--------------------+
|         1 | Metropolitan Museum of Art | One of the largest and finest art museums in the world | Chennai, india |       1 | Mon-Sun: 10am-5:30pm |
|         2 | Louvre Museum              | Historic monument in Paris and the world              | Lucknow, india  |       1 | Wed-Mon: 9am-6pm   |
|         3 | Tate Modern                | Britain''s national gallery of international modern art | London, UK    |       2 | Mon-Sun: 10am-6pm  |
+-----------+----------------------------+-------------------------------------------------------+-----------------+---------+--------------------+
3 rows in set (0.00 sec)
```

Updated:

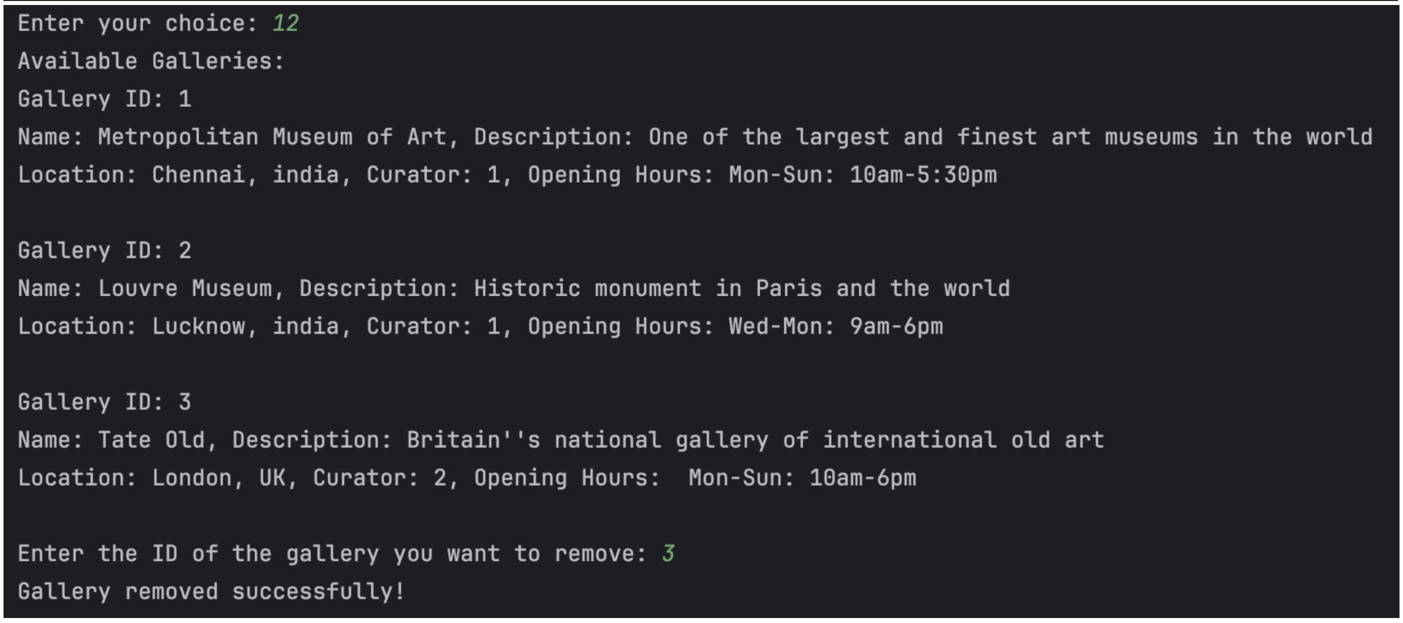


```
+-----------+----------------------------+--------------------------------------------------------+-----------------+---------+----------------------+
| GalleryID | Name                       | Description                                            | Location        | Curator | OpeningHours         |
+-----------+----------------------------+--------------------------------------------------------+-----------------+---------+----------------------+
|         1 | Metropolitan Museum of Art | One of the largest and finest art museums in the world | Chennai, india  |       1 | Mon-Sun: 10am-5:30pm |
|         2 | Louvre Museum              | Historic monument in Paris and the world               | Lucknow, india  |       1 | Wed-Mon: 9am-6pm     |
|         3 | Tate Old                   | Britain''s national gallery of international old art   | London, UK      |       2 |  Mon-Sun: 10am-6pm   |
+-----------+----------------------------+--------------------------------------------------------+-----------------+---------+----------------------+
3 rows in set (0.00 sec)
```

c. Test removing a gallery from the system.

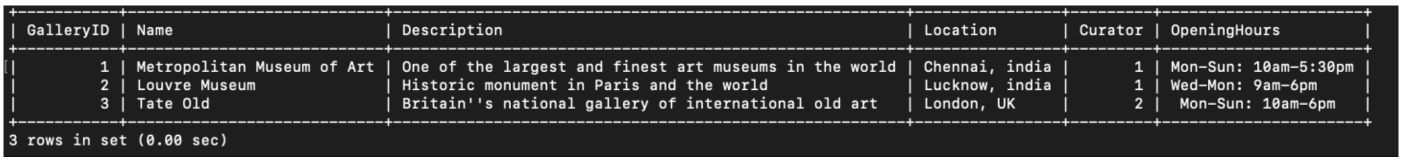


```
==== Virtual Art Gallery Menu ====
1. Add Artwork
2. Update Artwork
3. Remove Artwork
4. Get Artwork by ID
5. Search Artworks
6. Add Artwork to Favorites
7. Remove Artwork from Favorites
8. Get User's Favorite Artworks
9. Add Artist
10. Add Gallery
11. Update Gallery
12. Remove Gallery
13. Search Galleries
14. Exit
```



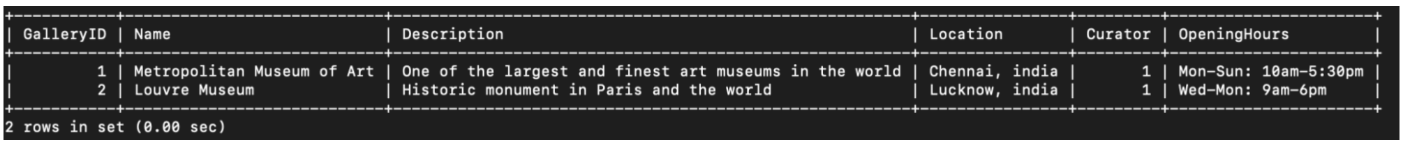
```
Enter your choice: 12
Available Galleries:
Gallery ID: 1
Name: Metropolitan Museum of Art, Description: One of the largest and finest art museums in the world
Location: Chennai, india, Curator: 1, Opening Hours: Mon-Sun: 10am-5:30pm

Gallery ID: 2
Name: Louvre Museum, Description: Historic monument in Paris and the world
Location: Lucknow, india, Curator: 1, Opening Hours: Wed-Mon: 9am-6pm

Gallery ID: 3
Name: Tate Old, Description: Britain''s national gallery of international old art
Location: London, UK, Curator: 2, Opening Hours:  Mon-Sun: 10am-6pm

Enter the ID of the gallery you want to remove: 3
Gallery removed successfully!
```

Old:



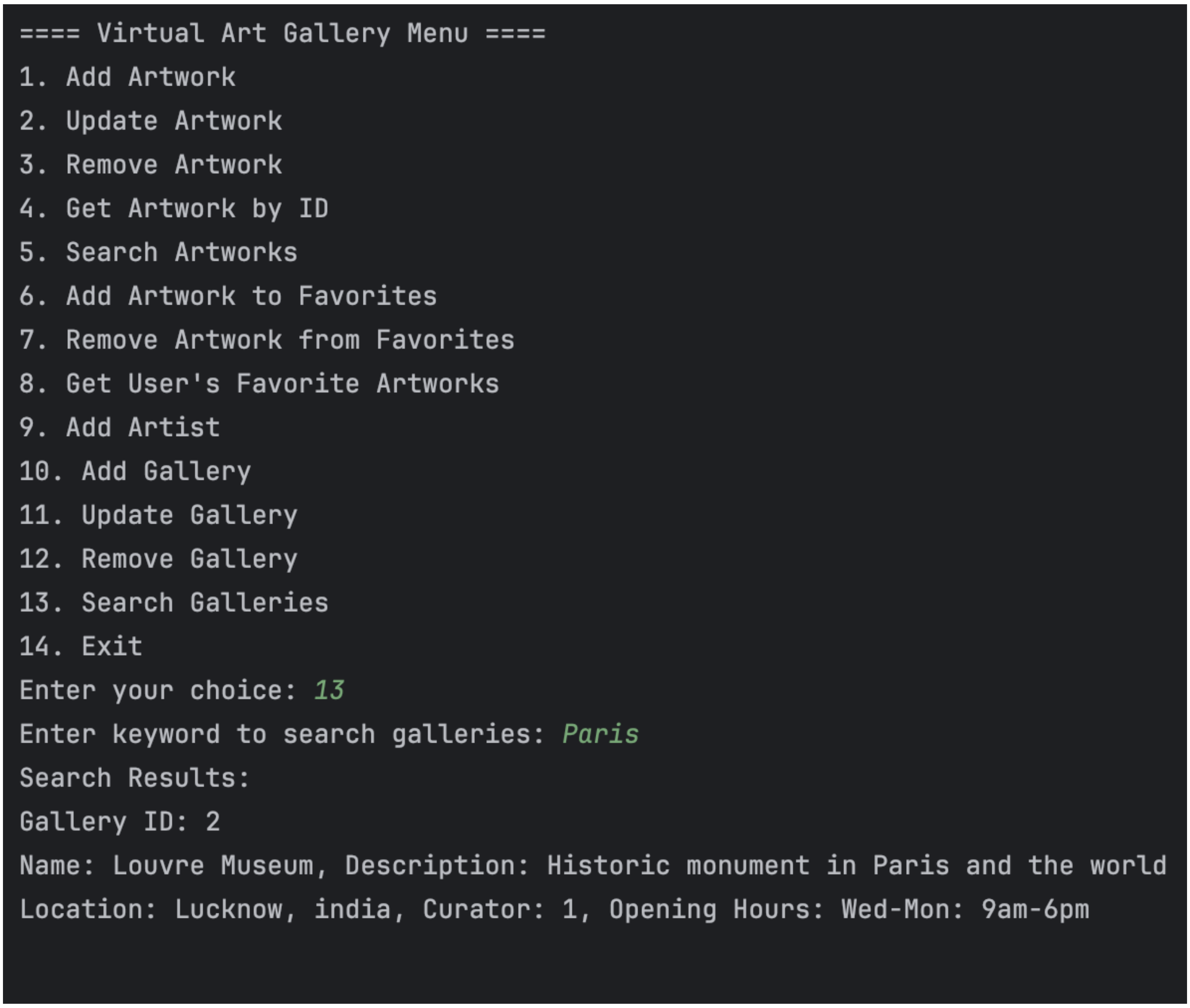
```
+-----------+----------------------------+------------------------------------------------------+----------------+---------+---------------------+
| GalleryID | Name                       | Description                                          | Location       | Curator | OpeningHours        |
+-----------+----------------------------+------------------------------------------------------+----------------+---------+---------------------+
|         1 | Metropolitan Museum of Art | One of the largest and finest art museums in the world | Chennai, india |       1 | Mon-Sun: 10am-5:30pm |
|         2 | Louvre Museum              | Historic monument in Paris and the world             | Lucknow, india |       1 | Wed-Mon: 9am-6pm    |
|         3 | Tate Old                   | Britain''s national gallery of international old art | London, UK     |       2 |  Mon-Sun: 10am-6pm  |
+-----------+----------------------------+------------------------------------------------------+----------------+---------+---------------------+
3 rows in set (0.00 sec)
```

Removed:



```
+-----------+----------------------------+--------------------------------------------------------+----------------+---------+----------------------+
| GalleryID | Name                       | Description                                            | Location       | Curator | OpeningHours         |
+-----------+----------------------------+--------------------------------------------------------+----------------+---------+----------------------+
|         1 | Metropolitan Museum of Art | One of the largest and finest art museums in the world | Chennai, india |       1 | Mon-Sun: 10am-5:30pm |
|         2 | Louvre Museum              | Historic monument in Paris and the world               | Lucknow, india |       1 | Wed-Mon: 9am-6pm     |
+-----------+----------------------------+--------------------------------------------------------+----------------+---------+----------------------+
2 rows in set (0.00 sec)
```

d. Check if searching for galleries returns the expected results.



```
==== Virtual Art Gallery Menu ====
1. Add Artwork
2. Update Artwork
3. Remove Artwork
4. Get Artwork by ID
5. Search Artworks
6. Add Artwork to Favorites
7. Remove Artwork from Favorites
8. Get User's Favorite Artworks
9. Add Artist
10. Add Gallery
11. Update Gallery
12. Remove Gallery
13. Search Galleries
14. Exit
Enter your choice: 13
Enter keyword to search galleries: Paris
Search Results:
Gallery ID: 2
Name: Louvre Museum, Description: Historic monument in Paris and the world
Location: Lucknow, india, Curator: 1, Opening Hours: Wed-Mon: 9am-6pm
```