TABLE OF CONTENTS

SYNOPSIS

The Art Gallery Management System is a Python and MySQL-based application designed to simplify the management of artists, artworks, exhibitions, and sales. It provides a comprehensive solution for art galleries by automating tedious manual tasks and ensuring data integrity. The system features a menu-driven interface that allows users to perform various operations, such as adding, searching, updating and deleting records.

PROJECT AIM AND OBJECTIVES

AIM

Develop a user-friendly Art Gallery Management System for efficient data handling and retrieval.

OBJECTIVES

- **Data Management**: Store detailed information about artists, artworks ,exhibitions ,sales
- **Efficient Retrieval**: Enable easy search and display of artists, artworks ,exhibitions ,sales
- **Data Integrity**: Record and document sales transactions accurately
- **Update/Delete Operations**: Allow updating and deleting Art gallery data
- **Exhibition Management**: Manage and schedule exhibitions
- **User-Friendly Interface**: Provide an intuitive interface for easy interaction

ABOUT THE PROJECT

As the world becomes increasingly reliant on computer systems, this project introduces the Art Gallery Management System, a solution designed to streamline the management of artists, artworks, exhibitions, and sales. The system reduces the need for manual record-keeping by offering robust features for adding, viewing, updating and deleting records across these domains. Built using Python and MySQL, it leverages a user-friendly, menu-driven interface to ensure easy navigation and interaction.

Key features include artist and artwork management, comprehensive reporting, exhibition tracking, and sales recording, all backed by secure data storage and integrity through relational database design, and tabular data presentation improves accessibility. The system simplifies data organization, ensures consistency, and aids decision-making by providing real-time access to well-structured information.

This project modernizes art gallery operations, making management tasks simpler, efficient, and scalable for galleries and small exhibitions alike, contributing to improved productivity and operational excellence.

Key Modules and Features:

1. Artist Management:

- Add new artists with their biography and contact details.
- Search for artists by ID or name.
- Delete artists and their associated artworks.

2. Artwork Management:

- Add details about artworks, including title, medium, size, and price.
- Link artworks to artists using artist IDs.

3. Exhibition Management:

- Manage exhibitions with details like title, location, start date, and end date.

4. Sales Management:

- Record sales, including artwork ID, client details, sale date, and price.

5. Comprehensive Reporting:

- View all records (artists, artworks, exhibitions, and sales) in a tabular format.

6. Interactive User Interface:

- Menu-driven interface for navigation.
- Easy-to-use prompts for data entry and operations.

7. Update data

- Menu driven interface to update artist, artwork, exhibition, or sale.
- Displays the current data on the database enabling the user to update only desired data

DEVLOPMENT ENVIRONMENT

HARDWARE

PROCESSOR : Intel(R) Core(TM) i5-10400 CPU @ 2.90GHz

RAM : 8.00 GB (7.83 GB usable)

MONITOR : DELL D1918H

KEYBOARD : DELL KB216

MOUSE : DELL MS116

SOFTWARE

1) **Operating System (Windows 11 Pro)**: Windows 11 Pro is chosen for its enhanced security features, improved performance on multi-core processors, faster boot times, and modern user interface. It also offers advanced file system support, optimized for both productivity and gaming, with features like BitLocker encryption, TPM 2.0, and seamless integration with Microsoft Teams.

2) **Database (MySQL):** MySQL is selected as database; main reason is quite obvious it is in our syllabus. MySQL is free and open-source relational database management system.

3) **Programming Language (Python):** The reason of selecting Python is same as with MySQL. Python is also free and open-source programming language

ABOUT PYTHON

Python is an object-oriented programming (OOP) language. It was created by Guido Van Rossum and released in 1991. It is used for developing desktop GUI applications, websites, and web applications.

Advantages and Features of Python:

- Simple: Python is a simple and minimalistic language. The pseudo-code nature of Python is one of its greatest strengths.
- Easy to Learn: Python has an extraordinarily simple syntax, making it easier to learn.
- Free and Open Source: One can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs.
- High-Level Language: While writing programs in Python, one does not need to worry about low-level details such as managing the memory used by the program.
- Portable: Python can work on many platforms, such as Linux and Windows.
- Interpreted: Python does not require compilation to binary. The program is run directly from the source code.
- Extensible: A Python code can be written in C or C++ language that can be compiled in C/C++ language.
- Embeddable: Python can be embedded within C/C++ to give scripting capabilities for the program's users.
- Extensive Libraries: The Python Standard Library is a collection of script modules accessible to a Python program, simplifying the programming process and removing the need to rewrite commonly used commands. They can be used by calling/importing them at the beginning of a script**.**

TABLE DESIGN

Table : Artists

```
mysql> desc Artist;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| artist_id    | int          | NO   | PRI | NULL    | auto_increment |
| name         | varchar(100) | NO   |     | NULL    |                |
| bio          | text         | YES  |     | NULL    |                |
| contact_info | varchar(150) | YES  |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)
```

Table : Artworks

```
mysql> desc artworks;
+------------+---------------+------+-----+---------+----------------+
| Field      | Type          | Null | Key | Default | Extra          |
+------------+---------------+------+-----+---------+----------------+
| artwork_id | int           | NO   | PRI | NULL    | auto_increment |
| artist_id  | int           | NO   | MUL | NULL    |                |
| title      | varchar(255)  | NO   |     | NULL    |                |
| medium     | varchar(255)  | YES  |     | NULL    |                |
| size       | varchar(255)  | YES  |     | NULL    |                |
| price      | decimal(10,2) | YES  |     | NULL    |                |
+------------+---------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

Table : Exhibition

```
mysql> desc exhibition;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| exhibition_id | int          | NO   | PRI | NULL    | auto_increment |
| artist_id     | int          | NO   | MUL | NULL    |                |
| title         | varchar(255) | NO   |     | NULL    |                |
| location      | varchar(255) | YES  |     | NULL    |                |
| start_date    | date         | YES  |     | NULL    |                |
| end_date      | date         | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

Table : Sales

```
mysql> desc sales;
+-------------+---------------+------+-----+---------+----------------+
| Field       | Type          | Null | Key | Default | Extra          |
+-------------+---------------+------+-----+---------+----------------+
| sale_id     | int           | NO   | PRI | NULL    | auto_increment |
| artist_id   | int           | NO   | MUL | NULL    |                |
| artwork_id  | int           | NO   | MUL | NULL    |                |
| client_name | varchar(255)  | YES  |     | NULL    |                |
| sale_date   | date          | YES  |     | NULL    |                |
| sale_price  | decimal(10,2) | YES  |     | NULL    |                |
+-------------+---------------+------+-----+---------+----------------+
6 rows in set (0.00 sec)
```

CODE

MySQL CODE

```
CREATE DATABASE IF NOT EXISTS art_gallery;

USE art_gallery;

CREATE TABLE artist (artist_id INT AUTO_INCREMENT PRIMARY KEY,name
VARCHAR(255) NOT NULL,bio TEXT,contact_info VARCHAR(255));

CREATE TABLE artworks (artwork_id INT AUTO_INCREMENT PRIMARY
KEY,artist_id INT NOT NULL,title VARCHAR(255) NOT NULL,medium
VARCHAR(255),size VARCHAR(255),price DECIMAL(10, 2),FOREIGN KEY (artist_id)
REFERENCES artist(artist_id) ON DELETE CASCADE);

CREATE TABLE exhibition (exhibition_id INT AUTO_INCREMENT PRIMARY
KEY,artist_id INT NOT NULL,title VARCHAR(255) NOT NULL,location
VARCHAR(255),start_date DATE,end_date DATE,FOREIGN KEY (artist_id)
REFERENCES artist(artist_id) ON DELETE CASCADE);

CREATE TABLE sales (sale_id INT AUTO_INCREMENT PRIMARY KEY,artist_id INT
NOT NULL,artwork_id INT NOT NULL,client_name VARCHAR(255),sale_date
DATE,sale_price DECIMAL(10, 2),FOREIGN KEY (artist_id) REFERENCES
artist(artist_id) ON DELETE CASCADE,FOREIGN KEY (artwork_id) REFERENCES
artworks(artwork_id) ON DELETE CASCADE);
```

PYTHON CODE

```
import mysql.connector

from tabulate import tabulate

def connect_db():

    return mysql.connector.connect(host="localhost",user="root",password="admin@123",

database="art_gallery")
```

```
#Adding data of artists and related data
def add_artist():
    name = input("Enter Artist's Name: ")
    bio = input("Enter Artist's Bio: ")
    contact_info = input("Enter Artist's Contact Info: ")
    conn = connect_db()
    cursor = conn.cursor()
    cursor.execute("INSERT INTO artist (name, bio, contact_info) VALUES (%s, %s, %s)",
                   (name, bio, contact_info))
    conn.commit()
    # Fetch the last inserted artist_id
    artist_id = cursor.lastrowid
    print(f"Artist added successfully with Artist ID: {artist_id}")
    conn.close()
def add_artwork():
    artist_id = input("Enter Artist ID: ")
    title = input("Enter Artwork Title: ")
    medium = input("Enter Artwork Medium: ")
    size = input("Enter Artwork Size: ")
    price = float(input("Enter Artwork Price: "))
    conn = connect_db()
    cursor = conn.cursor()
```

```
    cursor.execute("INSERT INTO artworks (artist_id, title, medium, size, price) VALUES (%s, %s, %s, %s, %s)",(artist_id, title, medium, size, price))
    conn.commit()
    print("Artwork added successfully.")
    conn.close()
def add_exhibition():
    artist_id = input("Enter Artist ID: ")
    title = input("Enter Exhibition Title: ")
    location = input("Enter Exhibition Location: ")
    start_date = input("Enter Start Date (YYYY-MM-DD): ")
    end_date = input("Enter End Date (YYYY-MM-DD): ")
    conn = connect_db()
    cursor = conn.cursor()
    cursor.execute("INSERT INTO exhibition (artist_id, title, location, start_date, end_date) VALUES (%s, %s, %s, %s, %s)",(artist_id, title, location, start_date, end_date))
    conn.commit()
    print("Exhibition added successfully.")
    conn.close()
def add_sale():
    artist_id = input("Enter Artist ID: ")
    artwork_id = input("Enter Artwork ID: ")
    client_name = input("Enter Client Name: ")
    sale_date = input("Enter Sale Date (YYYY-MM-DD): ")
```

```
    sale_price = float(input("Enter Sale Price: "))
    conn = connect_db()
    cursor = conn.cursor()
    cursor.execute("INSERT INTO sales (artist_id, artwork_id, client_name, sale_date, sale_price) VALUES (%s, %s, %s, %s, %s)",(artist_id, artwork_id, client_name, sale_date, sale_price))
    conn.commit()
    print("Sale recorded successfully.")
    conn.close()
#Update data
def update_artist():
    artist_id = input("Enter Artist ID to update: ")
    conn = connect_db()
    cursor = conn.cursor(dictionary=True)
    cursor.execute("SELECT * FROM artist WHERE artist_id = %s", (artist_id,))
    artist = cursor.fetchone()
    if artist:
        print("Current Artist Details:", artist)
        name = input(f"Enter new name (leave empty to keep '{artist['name']}'): ") or artist['name']
        bio = input(f"Enter new bio (leave empty to keep '{artist['bio']}'): ") or artist['bio']
        contact_info = input(f"Enter new contact info (leave empty to keep '{artist['contact_info']}'): ") or artist['contact_info']
```

```
        cursor.execute("UPDATE artist SET name = %s, bio = %s, contact_info = %s WHERE artist_id = %s",(name, bio, contact_info, artist_id))
        conn.commit()
        print("Artist details updated successfully.")
    else:
        print("Artist not found.")
    conn.close()
def update_artwork():
    artwork_id = input("Enter Artwork ID to update: ")
    conn = connect_db()
    cursor = conn.cursor(dictionary=True)
    cursor.execute("SELECT * FROM artworks WHERE artwork_id = %s", (artwork_id,))
    artwork = cursor.fetchone()
    if artwork:
        print("Current Artwork Details:", artwork)
        title = input(f"Enter new title (leave empty to keep '{artwork['title']}'): ") or artwork['title']
        medium = input(f"Enter new medium (leave empty to keep '{artwork['medium']}'): ") or artwork['medium']
        size = input(f"Enter new size (leave empty to keep '{artwork['size']}'): ") or artwork['size']
        price = input(f"Enter new price (leave empty to keep '{artwork['price']}'): ") or artwork['price']
        cursor.execute("UPDATE artworks SET title = %s, medium = %s, size = %s, price = %s WHERE artwork_id = %s",(title, medium, size, price, artwork_id))
```

```
        conn.commit()
        print("Artwork details updated successfully.")
    else:
        print("Artwork not found.")
    conn.close()
def update_exhibition():
    exhibition_id = input("Enter Exhibition ID to update: ")
    conn = connect_db()
    cursor = conn.cursor(dictionary=True)
    cursor.execute("SELECT * FROM exhibition WHERE exhibition_id = %s",
(exhibition_id,))
    exhibition = cursor.fetchone()
    if exhibition:
        print("Current Exhibition Details:", exhibition)
        title = input(f"Enter new title (leave empty to keep '{exhibition['title']}'): ") or
exhibition['title']
        location = input(f"Enter new location (leave empty to keep '{exhibition['location']}'): ")
or exhibition['location']
        start_date = input(f"Enter new start date (leave empty to keep
'{exhibition['start_date']}'): ") or exhibition['start_date']
        end_date = input(f"Enter new end date (leave empty to keep '{exhibition['end_date']}'):
") or exhibition['end_date']
        cursor.execute("UPDATE exhibition SET title = %s, location = %s, start_date = %s,
end_date = %s WHERE exhibition_id = %s",(title, location, start_date, end_date,
exhibition_id))
```

```
        conn.commit()
        print("Exhibition details updated successfully.")
    else:
        print("Exhibition not found.")
    conn.close()
def update_sale():
    sale_id = input("Enter Sale ID to update: ")
    conn = connect_db()
    cursor = conn.cursor(dictionary=True)
    cursor.execute("SELECT * FROM sales WHERE sale_id = %s", (sale_id,))
    sale = cursor.fetchone()
    if sale:
        print("Current Sale Details:", sale)
        client_name = input(f"Enter new client name (leave empty to keep '{sale['client_name']}'): ") or sale['client_name']
        sale_date = input(f"Enter new sale date (leave empty to keep '{sale['sale_date']}'): ") or sale['sale_date']
        sale_price = input(f"Enter new sale price (leave empty to keep '{sale['sale_price']}'): ") or sale['sale_price']
        cursor.execute("UPDATE sales SET client_name = %s, sale_date = %s, sale_price = %s WHERE sale_id = %s",(client_name, sale_date, sale_price, sale_id))
        conn.commit()
        print("Sale details updated successfully.")
    else:
```

```
        print("Sale not found.")
    conn.close()
# Update Data Menu
def update_data_menu():
    while True:
        print("\nUpdate Data Menu")
        print("1. Update Artist")
        print("2. Update Artwork")
        print("3. Update Exhibition")
        print("4. Update Sale")
        print("5. Go Back")
        choice = input("Enter your choice: ")
        if choice == '1':
            update_artist()
        elif choice == '2':
            update_artwork()
        elif choice == '3':
            update_exhibition()
        elif choice == '4':
            update_sale()
        elif choice == '5':
            break
        else:
```

```
            print("Invalid choice. Please try again.")
    # Delete Artist and Related Data
    def delete_artist():
        artist_id = input("Enter Artist ID to delete: ")
        conn = connect_db()
        cursor = conn.cursor()
        cursor.execute("DELETE FROM artworks WHERE artist_id = %s", (artist_id,))
        cursor.execute("DELETE FROM exhibition WHERE artist_id = %s", (artist_id,))
        cursor.execute("DELETE FROM sales WHERE artist_id = %s", (artist_id,))
        cursor.execute("DELETE FROM artist WHERE artist_id = %s", (artist_id,))
        conn.commit()
        # Reset AUTO_INCREMENT to maintain chronological order
        cursor.execute("ALTER TABLE artist AUTO_INCREMENT = 1")
        cursor.execute("ALTER TABLE artworks AUTO_INCREMENT = 1")
        cursor.execute("ALTER TABLE exhibition AUTO_INCREMENT = 1")
        cursor.execute("ALTER TABLE sales AUTO_INCREMENT = 1")
        conn.commit()
        print("Artist and all related data deleted successfully, and IDs reset.")
        conn.close()
    # Search and Display Artist and Related Data
    def search_artist():
        artist_id = input("Enter Artist ID or press Enter to search by name: ")
        conn = connect_db()
```

```
    cursor = conn.cursor(dictionary=True)
    if artist_id:
        cursor.execute("SELECT * FROM artist WHERE artist_id = %s", (artist_id,))
    else:
        name = input("Enter Artist Name: ")
        cursor.execute("SELECT * FROM artist WHERE name = %s", (name,))
    artist = cursor.fetchall()
    if artist:
        print("\nArtist Details:")
        print(tabulate(artist, headers="keys", tablefmt="grid"))
        # Display Related Data: Artworks, Exhibitions, and Sales
        for table, label in [("artworks", "Artworks"), ("exhibition", "Exhibitions"), ("sales",
"Sales")]:
            cursor.execute(f"SELECT * FROM {table} WHERE artist_id = %s",
(artist[0]['artist_id'],))
            data = cursor.fetchall()
            if data:
                print(f"\n{label}:")
                print(tabulate(data, headers="keys", tablefmt="grid"))
            else:
                print(f"\n{label}: No records found.")
    else:
        print("Artist not found.")
```

```
    conn.close()
def display_tables():
    conn = connect_db()
    cursor = conn.cursor(dictionary=True)
    for table in ["artist", "artworks", "exhibition", "sales"]:
        cursor.execute(f"SELECT * FROM {table}")
        data = cursor.fetchall()
        print(f"\nTable: {table.capitalize()}")
        print(tabulate(data, headers="keys", tablefmt="grid"))
    conn.close()
def main_menu():
    while True:
        print("\nArt Gallery Management System")
        print("1. Add Artist")
        print("2. Add Artwork")
        print("3. Add Exhibition")
        print("4. Add Sale")
        print("5. Search and Display Artist Details")
        print("6. Display All Tables")
        print("7. Update Data")
        print("8. Delete Artist and Related Data")
        print("9. Exit")
        choice = input("Enter your choice: ")
```

```
        if choice == '1':
            add_artist()
        elif choice == '2':
            add_artwork()
        elif choice == '3':
            add_exhibition()
        elif choice == '4':
            add_sale()
        elif choice == '5':
            search_artist()
        elif choice == '6':
            display_tables()
        elif choice == '7':
            update_data_menu()
        elif choice == '8':
            delete_artist()
        elif choice == '9':
            print("Exiting program.")
            break
        else:
            print("Invalid choice. Please try again.")
if __name__ == "__main__":
    main_menu()
```

OUTPUT

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 1

Enter Artist's Name: Raju

Enter Artist's Bio: modern artist

Enter Artist's Contact Info: raju@gmail.com

Artist added successfully with Artist ID: 3

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 2

Enter Artist ID: 3

Enter Artwork Title: emotion

Enter Artwork Medium: oil on canvas

Enter Artwork Size: 77 cm × 53 cm

Enter Artwork Price: 500000

Artwork added successfully.

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 3

Enter Artist ID: 3

Enter Exhibition Title: Artexpo

Enter Exhibition Location: kochin

Enter Start Date (YYYY-MM-DD): 2024-05-01

Enter End Date (YYYY-MM-DD): 2024-06-01

Exhibition added successfully.

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 4

Enter Artist ID: 3

Enter Artwork ID: 3

Enter Client Name: soman

Enter Sale Date (YYYY-MM-DD): 2024-05-15

Enter Sale Price: 500000

Sale recorded successfully.

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 5

Enter Artist ID or press Enter to search by name: 3

Artist Details:

artist_id	name	bio	contact_info
3	Mithun	modern artist	raju@gmail.com

Artworks:

artwork_id	artist_id	title	medium	size	price
3	3	Rage	oil on canvas	77 cm x 53 cm	500000

Exhibitions:

exhibition_id	artist_id	title	location	start_date	end_date
3	3	Artexpo	kochin	2024-05-01	2024-06-01

Sales:

sale_id	artist_id	artwork_id	client_name	sale_date	sale_price
3	3	3	soman	2024-05-15	500000

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 7

Update Data Menu

1. Update Artist

2. Update Artwork

3. Update Exhibition

4. Update Sale

5. Go Back

Enter your choice: 1

Enter Artist ID to update: 3

Current Artist Details: {'artist_id': 1, 'name': 'Raju', 'bio': 'modern artist', 'contact_info': 'raju@gmail.com'}

Enter new name (leave empty to keep 'Raju'): Mithun

Enter new bio (leave empty to keep 'modern artist'):

Enter new contact info (leave empty to keep 'raju@gmail.com'):

Artist details updated successfully

Update Data Menu

1. Update Artist

2. Update Artwork

3. Update Exhibition

4. Update Sale

5. Go Back

Enter your choice: 2

Enter Artwork ID to update: 3

Current Artwork Details: {'artwork_id': 1, 'artist_id': 1, 'title': 'emotion', 'medium': 'oil on canvas', 'size': '77 cm × 53 cm', 'price': Decimal('500000.00')}

Enter new title (leave empty to keep 'emotion'): Rage

Enter new medium (leave empty to keep 'oil on canvas'):

Enter new size (leave empty to keep '77 cm × 53 cm'):

Enter new price (leave empty to keep '500000.00'):

Artwork details updated successfully

Update Data Menu

1. Update Artist

2. Update Artwork

3. Update Exhibition

4. Update Sale

5. Go Back

Enter your choice: 5

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 6

Table: Artist

artist_id	name	bio	contact_info
1	Amit	Abstract painter.	amit@example.com
2	Priya	Sculptor.	priya@example.com
3	Mithun	modern artist	raju@gmail.com

Table: Artworks

artwork_id	artist_id	title	medium	size	price
1	1	Dreams	Oil on Canvas	24x36 inches	15000
2	2	Majesty	Marble	30x40 inches	25000
3	3	Rage	oil on canvas	77 cm x 53 cm	500000

Table: Exhibition

exhibition_id	artist_id	title	location	start_date	end_date
1	1	Art Expo	Mumbai	2025-02-01	2025-02-15
2	2	Showcase	Delhi	2025-03-01	2025-03-10
3	3	Artexpo	kochin	2024-05-01	2024-06-01

Table: Sales

sale_id	artist_id	artwork_id	client_name	sale_date	sale_price
1	1	1	Ravi	2025-01-15	15000
2	2	2	Anjali	2025-01-20	25000
3	3	3	soman	2024-05-15	500000

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 8

Enter Artist ID to delete: 3

Artist and all related data deleted successfully, and IDs reset.

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 6

Table: Artist

artist_id	name	bio	contact_info
1	Amit	Abstract painter.	amit@example.com
2	Priya	Sculptor.	priya@example.com

Table: Artworks

artwork_id	artist_id	title	medium	size	price
1	1	Dreams	Oil on Canvas	24x36 inches	15000
2	2	Majesty	Marble	30x40 inches	25000

Table: Exhibition

exhibition_id	artist_id	title	location	start_date	end_date
1	1	Art Expo	Mumbai	2025-02-01	2025-02-15
2	2	Showcase	Delhi	2025-03-01	2025-03-10

Table: Sales

sale_id	artist_id	artwork_id	client_name	sale_date	sale_price
1	1	1	Ravi	2025-01-15	15000
2	2	2	Anjali	2025-01-20	25000

Art Gallery Management System

1. Add Artist

2. Add Artwork

3. Add Exhibition

4. Add Sale

5. Search and Display Artist Details

6. Display All Tables

7. Update Data

8. Delete Artist and Related Data

9. Exit

Enter your choice: 9

Exiting program

BIBLIOGRAPHY

COMPUTER SCIENCE WITH PYTHON XI - Sumita Arora

COMPUTER SCIENCE WITH PYTHON XII - Sumita Arora

COMPUTER SCIENCE WITH PYTHON XII - Preeti Arora

https://www.w3schools.com/