Virtual Art Gallery - Execution Report (EXE Document)

Virtual Art Gallery System

Purpose of the Application

To create an interactive virtual platform for artists to upload, manage, and share their artworks, while allowing users to view and mark their favorite pieces. This system simulates a real-world gallery experience with complete database and CRUD functionalities.

Features and Functional Modules

- Manage Artists: Add, View, Update, Delete
- Manage Artworks: Add, View, Update, Delete
- Manage Galleries: Add, View, Update, Delete
- Manage Favorites: Add, View, Remove
- View all Users
- Search artworks and galleries by keyword

Project Directory Structure

virtual_art_gallery_python

entity

```
|—— artist.py
|—— artwork.py
|—— user.py
|—— Gallery.py
```

```
dao
 ├── virtual_gallery_interface.py
 ├── virtual_gallery_impl.py

util
 ├── DBConnUtil.py
 ├── PropertyUtil.py

exception
 ├── ArtworkNotFoundException.py
 ├── UserNotFoundException.py

tests
 ├── test_virtual_gallery.py

main.py
```

Software Requirements

- Python 3.10 or above
- MySQL Server / Workbench
- Visual Studio Code
- Required Python library: pip install mysql-connector-python

Object-Oriented Design

- **Classes** used for entity modeling

- **Abstraction** through DAO interface
- **Encapsulation** of fields in entity classes
- **Inheritance** for custom exceptions
- **Polymorphism** via DAO method overrides

Database Setup

- Database: virtualartgalleryy
- Tables: Artist, Artwork, User, Gallery, FavoriteArtworks
- Relationships via Foreign Keys
- Connection via DBConnUtil.py
- DB config in PropertyUtil.py

DAO Pattern

- IVirtualArtGallery: Interface for all DB methods
- VirtualGalleryImpl: Full implementation of data operations
- Clean modularization and separation of concerns

Exception Handling

- try-except-finally used in DB methods
- Custom Exceptions:
    - ArtworkNotFoundException
    - UserNotFoundException

Unit Testing

- Framework: unittest (PyUnit)
- File: tests/test_virtual_gallery.py
- Tested: Add, Update, Delete Artworks, Add/Remove Favorite, Search Gallery

- Run Comment : python -m unittest tests.test_virtual_gallery

SQL Highlights

- **DDL**: Tables created with PK, FK
- **DML**: Used INSERT, UPDATE, DELETE, SELECT
- **JOINS**: Used in favorites retrieval
- **GROUP BY / HAVING**: Artwork filtering
- **SUBQUERY**: Used in advanced filters

Execution Steps

- Run virtual art gallery database.sql in MySQL
- Configure DB in PropertyUtil.py
- Run: python main.py

  Menu:

  1. Add Artist
  2. Add Artwork
  3. View Artists
  4. View Artworks
  5. View Artworks by Artist
  6. View Users
  7. View Favorites
  8. Search Artworks
  9. Add Favorite
  10. Remove Favorite
  11. Update Artwork

12. Remove Artwork

13. Exit

Tools & Technologies

- **Language**: Python 3
- **DBMS**: MySQL
- **IDE**: VS Code
- **Library**: mysql-connector-python, unittest

Project Summary

- Core CRUD Operations implemented
- Database connectivity established
- Object-Oriented Design applied
- Code modularized using DAO pattern
- Error and exception handling in place
- Unit tests successfully executed
- SQL queries structured and optimized