## **Step 1 : Defining the Project Purpose and Scope**

### **Case study: Virtual Art Gallery**

**1. Introduction**

The Virtual Art Gallery is a Python-based application that simulates an online platform where users can explore and manage artwork collections. This project is developed as part of a case study to apply core programming skills in Python and reinforce concepts such as object-oriented design, control structures, exception handling, and database integration.

The primary intention is to build a menu-driven, interactive system that allows users to create personal galleries, browse digital artworks, and manage data effectively using SQL databases. It also emphasizes structured project architecture using packages (entity, dao, util, exception, main), unit testing, and reusable code principles. This project bridges the gap between academic learning and real-world software development by providing hands-on experience with core technologies in a practical application.

### **2. Project Objectives (Purpose)**

* To simulate a digital environment where users can interact with a collection of virtual artworks.
* To allow users to create personal galleries and select artworks.
* To implement core Python concepts such as loops, functions, lists, classes, and exception handling.
* To establish database connectivity using SQL and perform CRUD operations.
* To practice software design principles through modular code organization and separation of concerns.
* To develop, test, and maintain scalable code through unit testing and proper error handling.

### **3. Scope of the Project**

#### **✅ In Scope:**

* Users are able to register and log in.
* Users are able to add, view, update, or delete artworks.
* Users can create their own personal galleries and add artworks to them.
* All the data will be stored in a database (like SQL).
* We will create custom error messages and handle them properly.
* We’ll use a **.ini** file to manage database settings.
* We’ll write tests using Python’s **unittest** to check if everything works as expected.

**❌ Out of Scope:**

* No fancy graphics or web interface – this will be a simple text-based app.
* No uploading images of the artwork.
* No online payments or real art sales.
* No live collaboration or chatting between users.
* No use of AI or recommendation features.

### **4. Target Users**

* **Art Enthusiasts**: Users who wish to explore digital art collections and organize them into personal galleries.
* **Students and Developers**: Learners aiming to understand practical implementation of Python, SQL, and modular architecture.
* **Project Reviewers/Trainers**: Evaluators assessing coding standards, database integration, and functionality implementation.

**5. Assumptions**

* The app will run on your local computer using Python.
* The user will interact using the keyboard and a menu shown on the screen.
* We'll use a database like My SQL to store all the data.
* One person will use the app at a time (not multiple users at once).

### **7. Technology Stack**

| **COMPONENT** | **TECHNOLOGY** |
| --- | --- |
| Programming | Python 3.11+ |
| Database | SQLite / MySQL |
| DB Access | sqlite3 / mysql.connector |
| Configuration | configparser (.ini files) |
| Testing | unittest |
| Version Control | Git + GitHub |
| Development IDE | VS Code / PyCharm |

### **8. Project Structure Overview**

### **virtual\_art\_gallery**

### ├── entity/

### │ ├── artwork.py

### │ ├── user.py

### │ └── gallery.py

### ├── dao/

### │ ├── gallery\_dao.py

### │ └── gallery\_dao\_impl.py

### ├── exception/

### │ ├── base\_exception.py

### │ └── custom\_exceptions.py

### ├── util/

### │ ├── db\_config.ini

### │ ├── db\_property\_util.py

### │ └── db\_conn\_util.py

### ├── main/

### │ └── main\_module.py

### ├── test/

### │ └── test\_gallery.py

### └── README.md

### 

### Each directory will encapsulate specific functionality:

### **entity** – Defines the data models.

### **dao** – Contains interfaces and implementations for DB interactions.

### **exception** – Custom exceptions and error-handling mechanisms.

### **util** – Utility functions for configuration and DB connection.

### **main** – Entry point to run the application.

### **test** – Unit test cases for critical modules.

### **9. Conclusion**

### The Virtual Art Gallery project blends software engineering principles with real-world programming. Through this case study, learners develop a deeper understanding of modular design, SQL operations, error handling, and test-driven development in Python. The application offers a simplified yet powerful simulation of digital art management, providing a strong foundation for more advanced projects in software development.

### 

### 

|  |  |
| --- | --- |