

# **ART GALLERY MANAGEMENT SYSTEM**

*Project Report Submitted By*

**MIDHI V MOHAN**

**Reg. No.:AJC17MCA-I039**

*In Partial fulfillment for the Award of the Degree Of*

**INTEGRATED MASTER OF COMPUTER APPLICATIONS  
(INMCA)**

**APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**



**AMAL JYOTHI COLLEGE OF ENGINEERING  
KANJIRAPPALLY**

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE,  
Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

**2017-2022**

**DEPARTMENT OF COMPUTER APPLICATIONS**  
**AMAL JYOTHI COLLEGE OF ENGINEERING**  
**KANJIRAPPALLY**



**CERTIFICATE**

This is to certify that the Project report, “**ART GALLERY MANAGEMENT SYSTEM**” is the bonafide work of **MIDHI V MOHAN (Reg.No: AJC17MCA-I039)** in partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2017-22.

**Ms. Sona Maria Sebastian**  
**Internal Guide**

**Mrs. Meera Rose Mathew**  
**Coordinator**

**Rev.Fr.Dr. RubinThottupurathu Jose**  
**Head of the Department**

**External Examiner**

## **DECLARATION**

I hereby declare that the project report “**ART GALLERY MANAGEMENT SYSTEM**” is a bonafided work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Degree of Integrated Master of Computer Applications (INMCA) from APJ Abdul Kalam Technological University, during the academic year 2017-2022.

**Date:**

**KANJIRAPPALLY**

**MIDHI V MOHAN**

**Reg. No: AJC17MCA-I039**

## ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Mrs. Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also like to express sincere gratitude to my guide, **Ms. Sona Maria Sebastian** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

MIDHI V MOHAN

## **ABSTRACT**

Art gallery management system is an application that allows buyers to purchase paintings, scriptures, models. This system includes upcoming events advertisement of an artist that rented the gallery to help also their client, and this feature of the system may also use by the art gallery business to announce their own events.

The system will store the information of each painting or sculpture including the artist who created. Each art that is stored in the system, the management can upload multiple views of a certain artwork. This system includes also simple sale management which is the art gallery business may publish also the artworks that is for sale and the customer may request for an order of the artwork and when the management will contact the customer to confirm the transaction and for the scheduling of the delivery. Art lovers have to go to the art exhibition to collect their favorite arts or painting. But now-a-days they are not getting enough time to go to the galleries and collect the arts and paintings.

# CONTENT

Sl. No	Topic	Page No
1	INTRODUCTION	1
1.1	PROJECT OVERVIEW	2
1.2	PROJECT SPECIFICATION	2
2	SYSTEM STUDY	3
2.1	INTRODUCTION	4
2.2	EXISTING SYSTEM	5
2.3	DRAWBACKS OF EXISTING SYSTEM	5
2.4	PROPOSED SYSTEM	5
2.5	ADVANTAGES OF PROPOSED SYSTEM	6
3	REQUIREMENT ANALYSIS	7
3.1	FEASIBILITY STUDY	8
3.1.1	ECONOMICAL FEASIBILITY	8
3.1.2	TECHNICAL FEASIBILITY	9
3.1.3	BEHAVIORAL FEASIBILITY	9
3.2	SYSTEM SPECIFICATION	10
3.2.1	HARDWARE SPECIFICATION	10
3.2.2	SOFTWARE SPECIFICATION	10
3.3	SOFTWARE DESCRIPTION	10
3.3.1	PHP	10
3.3.2	MYSQL	11
4	SYSTEM DESIGN	13
4.1	INTRODUCTION	14
4.2	UML DIAGRAM	14
4.2.1	USE CASE DIAGRAM	15
4.2.2	SEQUENCE DIAGRAM	18
4.2.3	OBJECT DIAGRAM	22
4.2.4	CLASS DIAGRAM	25
4.2.5	COLLABORATION DIAGRAM	29
4.2.6	ACTIVITY DIAGRAM	33
4.2.7	STATE CHART DIAGRAM	38
4.2.8	DEPLOYMENT DIAGRAM	40

<b>4.2.9</b>	<b>COMPONENT DIAGRAM</b>	<b>43</b>
<b>4.5</b>	<b>USER INTERFACE DESIGN USING FIGMA</b>	<b>46</b>
<b>4.5.1</b>	<b>INPUT DESIGN</b>	<b>46</b>
<b>4.5.2</b>	<b>OUTPUT DESIGN</b>	<b>49</b>
<b>4.6</b>	<b>DATA BASE DESIGN</b>	<b>52</b>
<b>5</b>	<b>SYSTEM TESTING</b>	<b>59</b>
<b>5.1</b>	<b>INTRODUCTION</b>	<b>60</b>
<b>5.2</b>	<b>TEST PLAN</b>	<b>61</b>
<b>5.2.1</b>	<b>UNIT TESTING</b>	<b>61</b>
<b>5.2.2</b>	<b>INTEGRATION TESTING</b>	<b>62</b>
<b>5.2.3</b>	<b>VALIDATION TESTING</b>	<b>62</b>
<b>5.2.4</b>	<b>USER ACCEPTANCE TESTING</b>	<b>63</b>
<b>5.2.5</b>	<b>SELENIUM TESTING</b>	<b>63</b>
<b>5.2.6</b>	<b>TEST CASE</b>	<b>64</b>
<b>6</b>	<b>IMPLEMENTATION</b>	<b>69</b>
<b>6.1</b>	<b>INTRODUCTION</b>	<b>70</b>
<b>6.2</b>	<b>IMPLEMENTATION PROCEDURE</b>	<b>70</b>
<b>6.2.1</b>	<b>USER TRAINING</b>	<b>71</b>
<b>6.2.2</b>	<b>TRAINING ON APPLICATION SOFTWARE</b>	<b>71</b>
<b>6.2.3</b>	<b>SYSTEM MAINTENANCE</b>	<b>71</b>
<b>7</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	<b>72</b>
<b>7.1</b>	<b>CONCLUSION</b>	<b>73</b>
<b>7.2</b>	<b>FUTURE SCOPE</b>	<b>73</b>
<b>8</b>	<b>BIBLIOGRAPHY</b>	<b>74</b>
<b>9</b>	<b>APPENDIX</b>	<b>76</b>
<b>9.1</b>	<b>SAMPLE CODE</b>	<b>77</b>
<b>9.2</b>	<b>SCREEN SHOTS</b>	<b>84</b>

## **List of Abbreviation**

IDE	-	Integrated Development Environment
HTML	-	Hyper Text Markup Language.
CSS	-	Cascading Style Sheet
SQL	-	Structured Query Language
UML	-	Unified Modeling Language



## **CHAPTER 1**

### **INTRODUCTION**

## **1.1 PROJECT OVERVIEW**

*The art gallery* is where the arts or art crafts of the artists can be displayed for the visitors' view. People additionally comment on arts that artist's show so that there will be some scope of improvement of the arts that is displayed. The art gallery is the application that enables the clients to transfer their paintings and make it available to the visitors' view.

New thing that included in this project is that the paintings can be sell through online at big discounts. Customers who visit the website they get to know about the paintings. And also, the customer can buy paintings through online.

## **1.2 PROJECT SPECIFICATION**

Art Gallery Management System is a website and it is very helpful for the art lovers and others who want to buy an art work. This website helps the end users to search their arts and paintings and they can place order for the selected pieces.

The system includes 2 modules. They are:

### **1. Admin Module**

Admin must have a login into this system. He has the overall control of the system. Admin can add or update art details, manage user data etc. Admin can View all the registered users and also manage all his data.

### **2. Customer Module**

Customer can register and they can buy art online and also view also information about his/her art.

## **CHAPTER 2**

### **SYSTEM STUDY**

## 2.1 INTRODUCTION

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minute's detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

## **2.2 EXISTING SYSTEM**

Suppose we went to the museum or art gallery there we found something which we really want to purchase so for that we need to contact to the manager and talk to him to get aware about the procedure of purchasing, but at the last moment, we come to know that someone has already booked that painting or art there we feel helpless because that painting was on sale for three days and you were not able to purchase it just because of lack of time and the distance you need to cover from your home to art gallery.

Each customer can create their own profile. The proposed system rectify the drawbacks of the present system. It is necessary to modify the existing system in order to include additional information and make the system efficient, flexible and secure. Using the new system customers can view all information about his art.

## **2.3 DRAWBACKS OF EXISTING SYSTEM**

- No proper online management of system
- Human effort is needed.
- It is difficult to maintain important information in books.

## **2.4 PROPOSED SYSTEM**

The proposed system is defined to meets all the disadvantages of the existing system. It is necessary to have a system that is more user friendly on such consideration the system is proposed. In our proposed system there is admin who can view all the customers. It allows customers to buy art through online and do their transactions by using online payment method. Users of this proposed system are admin and customer.

The software application which avoids more manual hours that need to spend in record keeping and generating reports. This application keeps the data in a centralized way which is available to all the users simultaneously. It is very easy to manage historical data in database. No specific training is required for the distributors to use this application. They can easily use the tool that decreases manual hours spending for normal things and hence increases the performance. It is very easy to record the information of online sales and purchases in the databases.

## **2.5 ADVANTAGES OF PROPOSED SYSTEM**

The system is very simple in design and to implement. The system requires very low system resources, and the system will work in almost all configurations. It has got following features:

### **➤ Better security: -**

For data to remain secure measures must be taken to prevent unauthorized access. Security means that data are protected from various forms of destruction. The system security problem can be divided into four related issues: security, integrity, privacy and confidentiality. Username and password requirement to sign in ensures security. It will also provide data security as we are using the secured databases for maintaining the documents.

### **➤ Ensure data accuracy: -**

The proposed system eliminates the manual errors while entering the details of the users during the registration.

### **➤ Better service: -**

The product will avoid the burden of hard copy storage. We can also conserve the time and human resources for doing the same task. The data can be maintained for longer period with no loss of data.

## **CHAPTER 3**

# **REQUIREMENT ANALYSIS**

### **3.1 FEASIBILITY STUDY**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development.

The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features: -

#### **3.1.1 Economical Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

The proposed system is developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it give an indication of the system is economically possible for development.

Cost of project and they divided according to the system used, its development cost and cost for hosting the project. According to all the calculations the project was developed in a low cost. As it is completely developed using open-source software.



### 3.1.2 Technical Feasibility

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical issues raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project requires High Resolution Scanning device and utilizes Cryptographic techniques. Through the technology may become obsolete after some period of time, due to the fact that newer version of same software supports older versions, the system may still be used. So, there are minimal constraints involved with this project. The system has been developed using PHP in front end and MySQL in server in back end, the project is technically feasible for development. The System used was also of good performance of Processor Intel i3-6006U core; RAM 8.00GB and, Hard disk 1TB

### 3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

## 3.2 SYSTEM SPECIFICATION

### 3.2.1 Hardware Specification

Processor - Intel core i3-6006U

RAM - 8.00 GB

Hard disk - 1 TB

### 3.2.2 Software Specification

Front End - HTML, CSS

Backend - MYSQL

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, AJAX, J Query, PHP, CSS, Bootstrap

## 3.3 SOFTWARE DESCRIPTION

### 3.3.1 PHP

PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language. PHP is now installed on more than 244 million websites and 2.1 million web servers. Originally created by Rasmus Ledorf in 1995, the reference implementation of PHP is now produced by the PHP group. While PHP originally stood for personal home page, it now stands for PHP: Hypertext Preprocessor, a recursive acronym. PHP code is interpreted by a web server with a PHP processor module which generates the resulting web page. PHP commands can be embedded directly into a HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP. PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

### 3.3.2 MySQL

MySQL, the most popular Open-Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Web site provides the latest information about MySQL software.

- **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You setup rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL92” refers to the standard released in 1992, “SQL: 1999” refers to the standard released in 1999, and “SQL: 2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

- **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available.

- **MySQL Server works in client/server or embedded systems.**

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

## **CHAPTER 4**

### **SYSTEM DESIGN**

## 4.1 INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user-oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

## 4.2 UML DIAGRAM

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

UML stands for **Unified Modeling Language**. UML is different from the other common programming languages such as C++, Java, COBOL, etc. UML is a pictorial language used to make software blueprints. UML can be described as a general-purpose visual modeling language to visualize, specify, construct, and document software system. Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design. After some standardization, UML has

become an OMG standard. All the elements, relationships are used to make a complete UML diagram and the diagram represents a system. The visual effect of the UML diagram is the most important part of the entire process. All the other elements are used to make it complete. UML includes the following nine diagrams.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

### **4.2.1 USE CASE DIAGRAM**

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. In this context, the term "system" refers to something being developed or operated, such as art sales. Use case diagrams are employed in UML (Unified Modeling Language), a standard notation for the modeling of real-world objects and systems.

System objectives can include planning overall requirements, validating a hardware design, testing and debugging a software product under development, creating an online help reference, or performing a consumer-service-oriented task. For example, use cases in a product sales environment would include item ordering, catalog updating, payment processing, and customer relations. A use case diagram contains four components.

- The boundary, which defines the system of interest in relation to the world around it.
- The actors, usually individuals involved with the system defined according to their roles.
- The use cases, which are the specific roles are played by the actors within and around the system.

- The relationships between and among the actors and the use cases.

Use case diagrams are drawn to capture the functional requirements of a system. After identifying the above items, we have to use the following guidelines to draw an efficient use case diagram

- The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
- Use notes whenever required to clarify some important points.





## 4.2.2 SEQUENCE DIAGRAM

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

### Sequence Diagram Notations –

- i. **Actors** – An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram. We use actors to depict various roles including human users and other external subjects. We represent an actor in a UML diagram using a stick person notation. We can have multiple actors in a sequence diagram.
- ii. **Lifelines** – A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically, each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram
- iii. **Messages** – Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.

Messages can be broadly classified into the following categories:

- Synchronous messages
- Asynchronous Messages
- Create message
- Delete Message
- Self-Message
- Reply Message

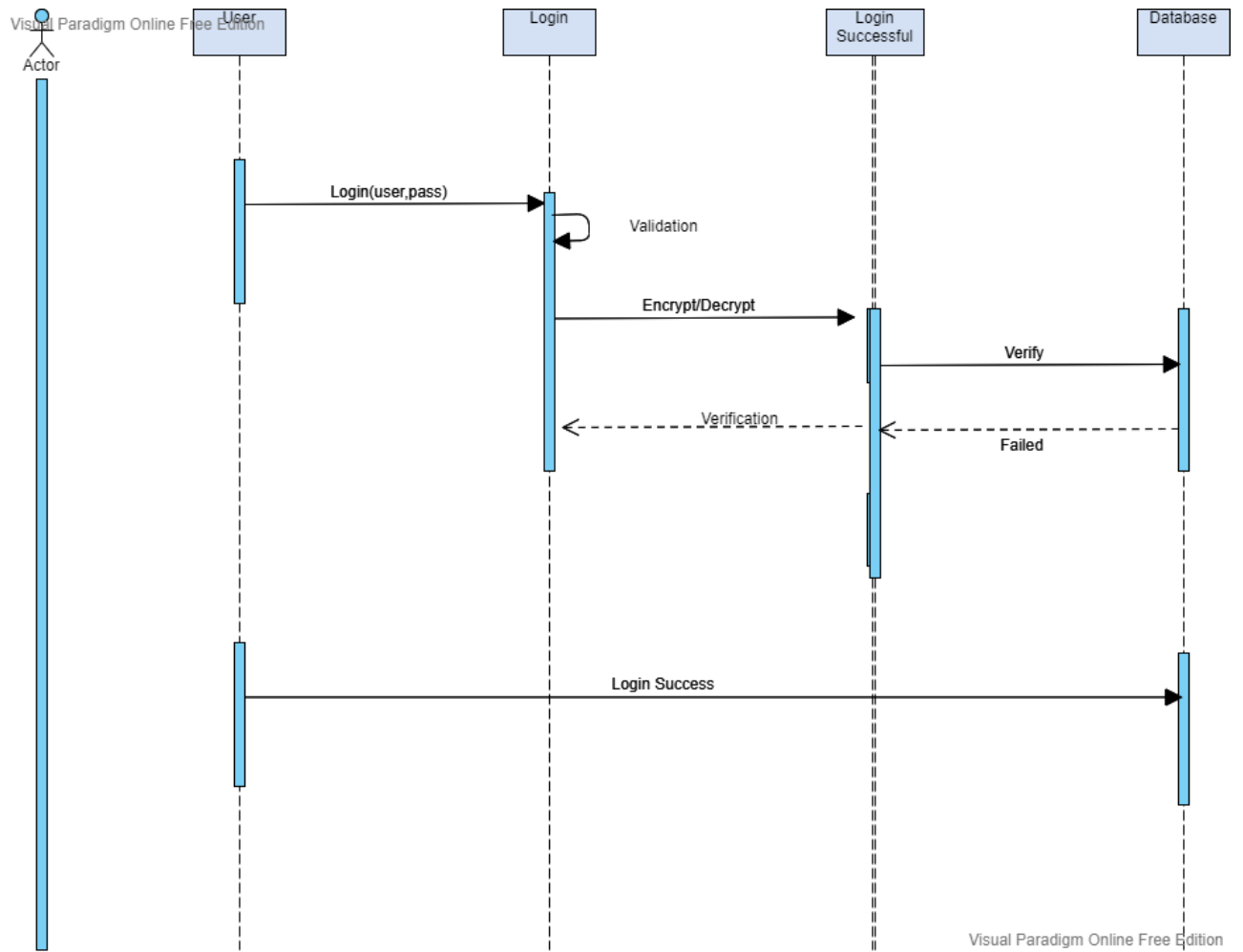
- Found Message
- Lost Message

**iv. Guards** – To model conditions we use guards in UML. They are used when we need to restrict the flow of messages on the pretext of a condition being met. Guards play an important role in letting software developers know the constraints attached to a system or a particular process.

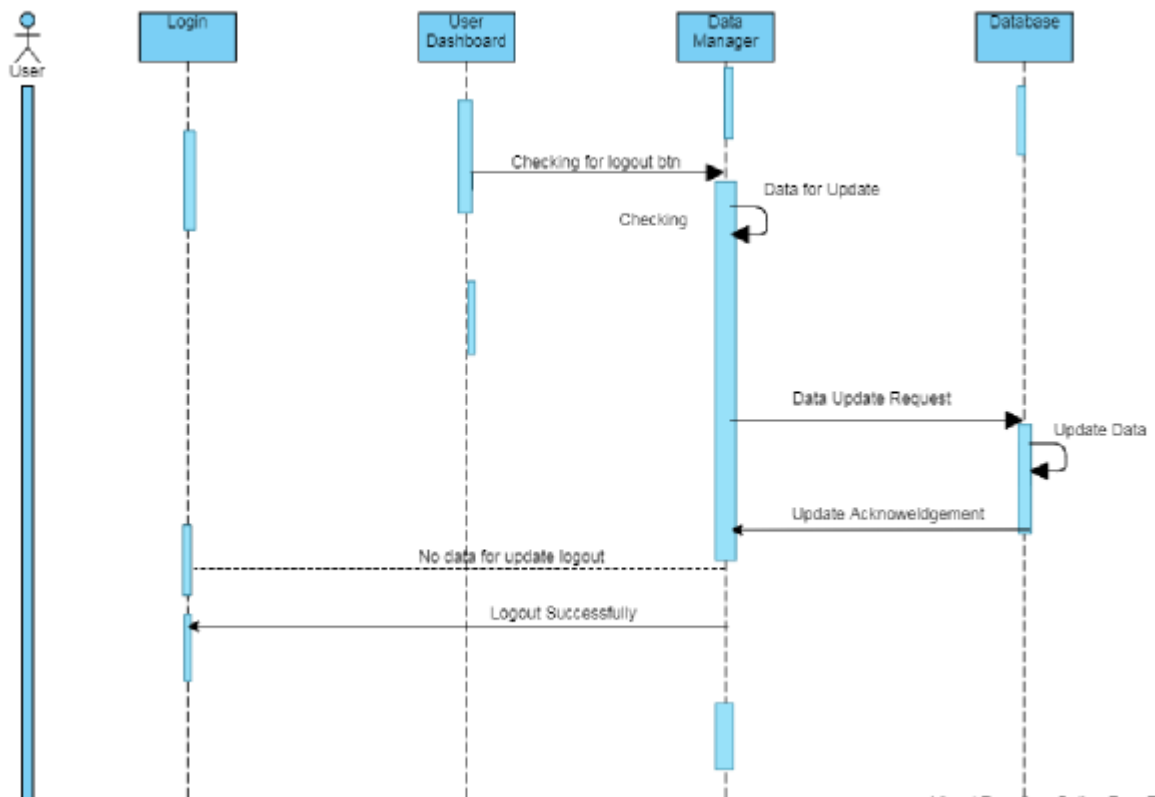
**Uses of sequence diagrams –**

- Used to model and visualize the logic behind a sophisticated function, operation or procedure.
- They are also used to show details of UML use case diagrams.
- Used to understand the detailed functionality of current or future systems.
- Visualize how messages and tasks move between objects or components in a system.

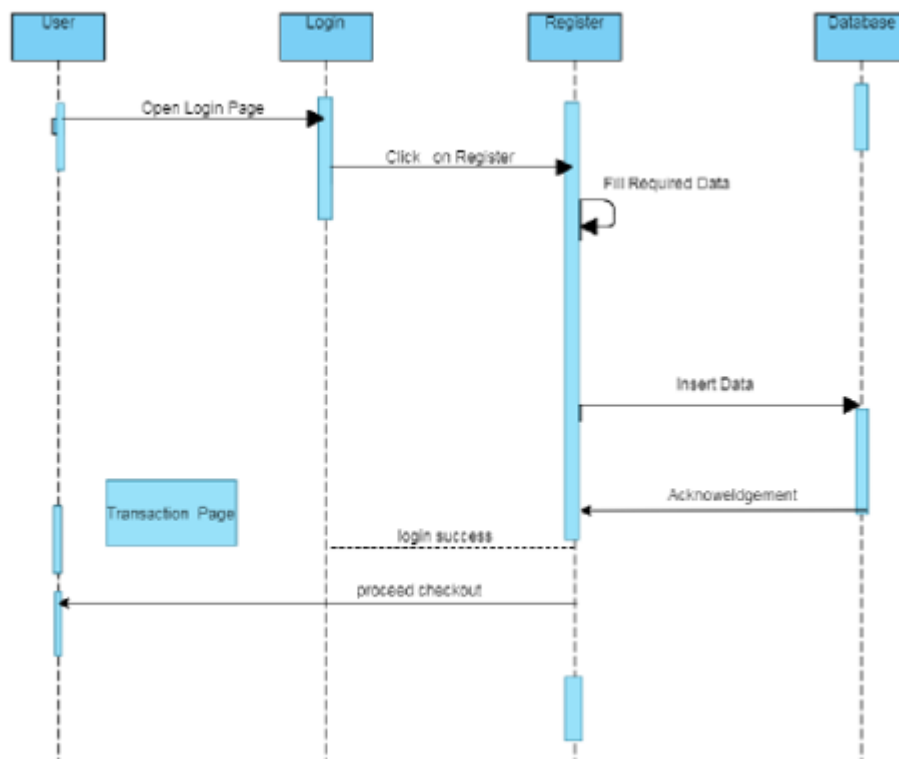
**SEQUENCE DIAGRAM (login)**



Logout



## Registration



### 4.2.3 OBJECT DIAGRAM

Object diagrams represent an instance of a class diagram. The basic concepts are similar for class diagrams and object diagrams. Object diagrams also represent the static view of a system but this static view is a snapshot of the system at a particular moment. Object diagrams are used to render a set of objects and their relationships as an instance.

#### **Purpose of Object Diagrams**

The purpose of a diagram should be understood clearly to implement it practically. The purposes of object diagrams are similar to class diagrams.

The difference is that a class diagram represents an abstract model consisting of classes and their relationships. However, an object diagram represents an instance at a particular moment, which is concrete in nature.

It means the object diagram is closer to the actual system behavior. The purpose is to capture the static view of a system at a particular moment.

The purpose of the object diagram can be summarized as –

- Forward and reverse engineering.
- Object relationships of a system
- Static view of an interaction.
- Understand object behaviour and their relationship from practical perspective

#### **How to Draw an Object Diagram?**

an object diagram is an instance of a class diagram. It implies that an object diagram consists of instances of things used in a class diagram.

So both diagrams are made of same basic elements but in different form. In class diagram elements are in abstract form to represent the blue print and in object diagram the elements are in concrete form to represent the real-world object.

To capture a particular system, numbers of class diagrams are limited. However, if we consider object diagrams then we can have unlimited number of instances, which are unique in nature.

Only those instances are considered, which have an impact on the system. From the above discussion, it is clear that a single object diagram cannot capture all the necessary instances or rather cannot specify all the objects of a system. Hence, the solution is –

- First, analyze the system and decide which instances have important data and association.
- Second, consider only those instances, which will cover the functionality.
- Third, make some optimization as the number of instances are unlimited.

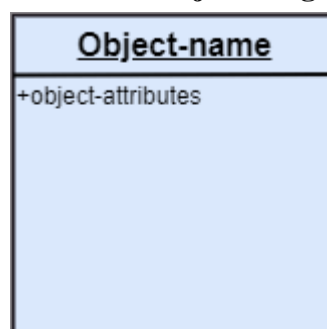
Before drawing an object diagram, the following things should be remembered and understood clearly –

- Object diagrams consist of objects.
- The link in object diagram is used to connect objects.
- Objects and links are the two elements used to construct an object diagram.

After this, the following things are to be decided before starting the construction of the diagram –

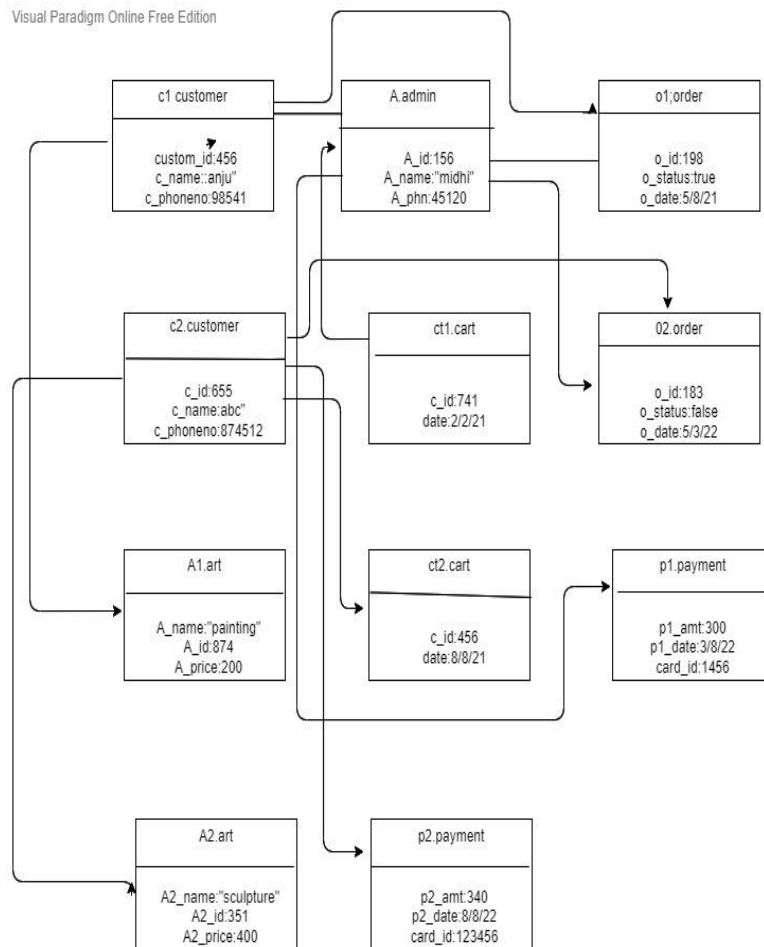
- The object diagram should have a meaningful name to indicate its purpose.
- The most important elements are to be identified.
- The association among objects should be clarified.
- Values of different elements need to be captured to include in the object diagram.
- Add proper notes at points where more clarity is required.

#### Notation of an Object Diagram



object diagrams are used for –

- Making the prototype of a system.
- Reverse engineering.
- Modeling complex data structures.
- Understanding the system from practical perspective.



Visual Paradigm Online Free Edition



## 4.2.4 CLASS DIAGRAM

The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and also it may inherit from other classes. A class diagram is used to visualize, describe, document various different aspects of the system, and also construct executable software code.

It shows the attributes, classes, functions, and relationships to give an overview of the software system. It constitutes class names, attributes, and functions in a separate compartment that helps in software development. Since it is a collection of classes, interfaces, associations, collaborations, and constraints, it is termed as a structural diagram.

### Purpose of Class Diagrams

The main purpose of class diagrams is to build a static view of an application. It is the only diagram that is widely used for construction, and it can be mapped with object-oriented languages. It is one of the most popular UML diagrams. Following are the purpose of class diagrams given below:

1. It analyses and designs a static view of an application.
2. It describes the major responsibilities of a system.
3. It is a base for component and deployment diagrams.
4. It incorporates forward and reverse engineering.

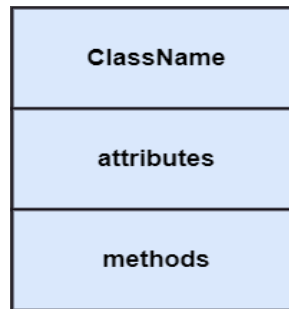
### Benefits of Class Diagrams

1. It can represent the object model for complex systems.
  2. It reduces the maintenance time by providing an overview of how an application is structured before coding.
  3. It provides a general schematic of an application for better understanding.
  4. It represents a detailed chart by highlighting the desired code, which is to be programmed.
  5. It is helpful for the stakeholders and the developers.
-

## Vital components of a Class Diagram

The class diagram is made up of three sections:

- **Upper Section:** The upper section encompasses the name of the class. A class is a representation of similar objects that shares the same relationships, attributes, operations, and semantics. Some of the following rules that should be taken into account while representing a class are given below:
  1. Capitalize the initial letter of the class name.
  2. Place the class name in the center of the upper section.
  3. A class name must be written in bold format.
  4. The name of the abstract class should be written in italics format.
- **Middle Section:** The middle section constitutes the attributes, which describe the quality of the class. The attributes have the following characteristics:
  - The attributes are written along with its visibility factors, which are public (+), private (-), protected (#), and package (~).
  5. The accessibility of an attribute class is illustrated by the visibility factors.
  6. A meaningful name should be assigned to the attribute, which will explain its usage inside the class.
- **Lower Section:** The lower section contain methods or operations. The methods are represented in the form of a list, where each method is written in a single line. It demonstrates how a class interacts with data.



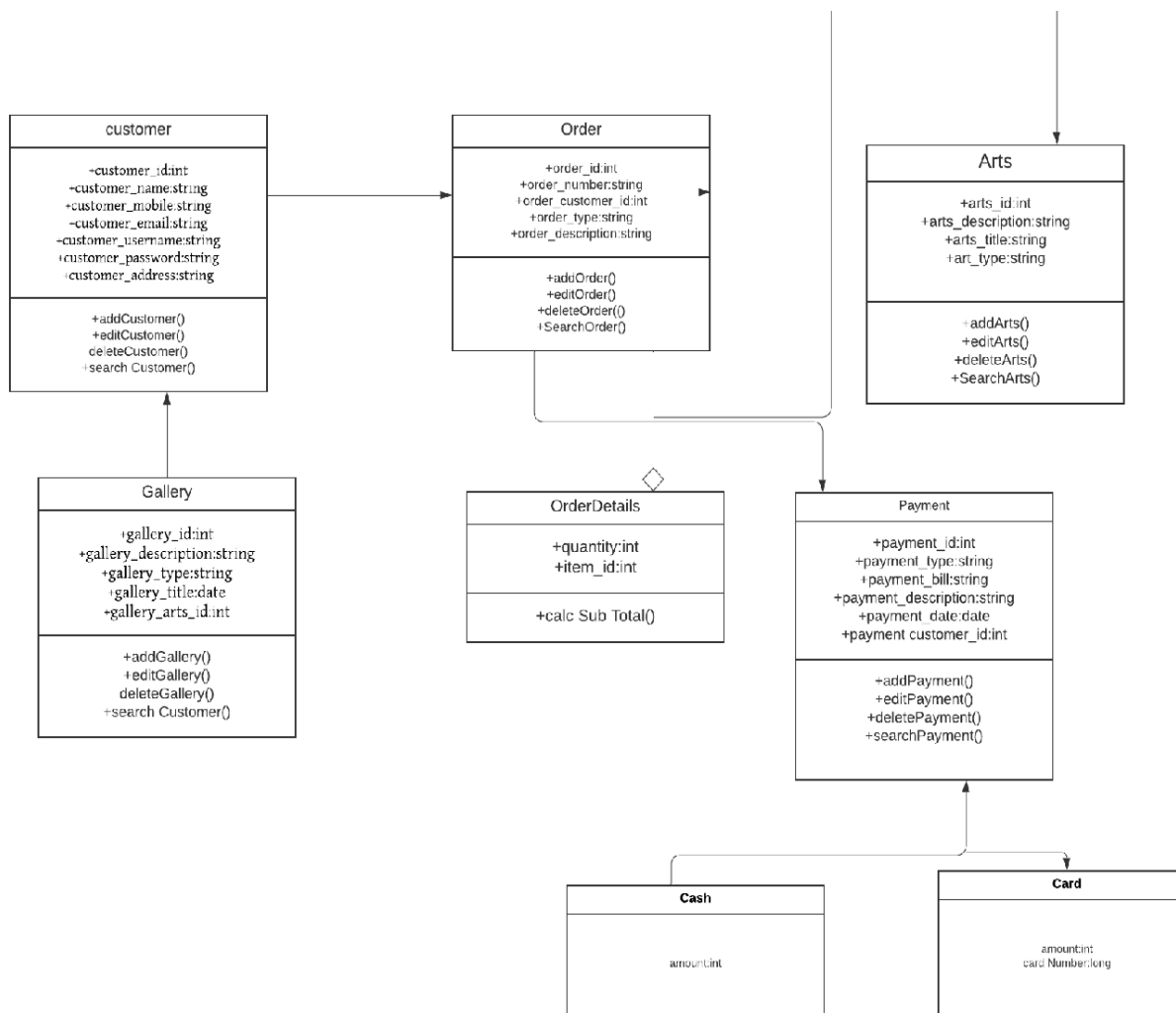
### **Abstract Classes**

In abstract class, no objects can be a direct entity of the abstract class. The abstract class can neither be declared nor be instantiated. It is used to find the functionalities across the classes. The notation of the abstract class is similar to that of class; the only difference is that the name of the class is written in italics. Since it does not involve any implementation for a given function, it is best to use the abstract class with multiple objects.

### **How to draw a Class Diagram?**

Some key points that are needed to keep in mind while drawing a class diagram are given below:

1. To describe a complete aspect of the system, it is suggested to give a meaningful name to the class diagram.
2. The objects and their relationships should be acknowledged in advance.
3. The attributes and methods (responsibilities) of each class must be known.
4. A minimum number of desired properties should be specified as more number of the unwanted property will lead to a complex diagram.
5. Notes can be used as and when required by the developer to describe the aspects of a diagram.
6. The diagrams should be redrawn and reworked as many times to make it correct before producing its final version.



## 4.2.5 COLLABORATION DIAGRAM

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

### Notations of a Collaboration Diagram

Following are the components of a component diagram that are enlisted below:

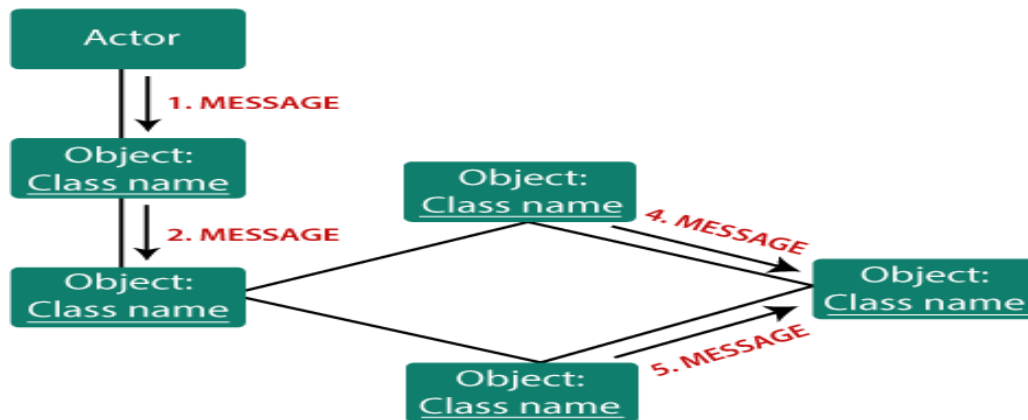
1. The representation of an object is done by an object symbol with its name and class underlined, separated by a colon.

In the collaboration diagram, objects are utilized in the following ways:

- The object is represented by specifying their name and class.
  - It is not mandatory for every class to appear.
  - A class may constitute more than one object.
  - In the collaboration diagram, firstly, the object is created, and then its class is specified.
  - To differentiate one object from another object, it is necessary to name them.
2. **Actors:** In the collaboration diagram, the actor plays the main role as it invokes the interaction. Each actor has its respective role and name. In this, one actor initiates the use case.
  3. **Links:** The link is an instance of association, which associates the objects and actors. It portrays a relationship between the objects through which the messages are sent. It is represented by a solid line. The link helps an object to connect with or navigate to another object, such that the message flows are attached to links.

4. **Messages:** It is a communication between objects which carries information and includes a sequence number, so that the activity may take place. It is represented by a labeled arrow, which is placed near a link. The messages are sent from the sender to the receiver, and the direction must be navigable in that particular direction. The receiver must understand the message.

#### Components of a collaboration diagram



#### When to use a Collaboration Diagram?

The collaborations are used when it is essential to depict the relationship between the object. Both the sequence and collaboration diagrams represent the same information, but the way of portraying it quite different. The collaboration diagrams are best suited for analyzing use cases.

Following are some of the use cases enlisted below for which the collaboration diagram is implemented:

1. To model collaboration among the objects or roles that carry the functionalities of use cases and operations.
2. To model the mechanism inside the architectural design of the system.
3. To capture the interactions that represent the flow of messages between the objects and the roles inside the collaboration.
4. To model different scenarios within the use case or operation, involving a collaboration of several objects and interactions.
5. To support the identification of objects participating in the use case.
6. In the collaboration diagram, each message constitutes a sequence number, such that the top-level message is marked as one and so on. The messages sent during the same call are denoted with the same decimal prefix, but with different suffixes of 1, 2, etc. as per their occurrence.

---

### **Steps for creating a Collaboration Diagram**

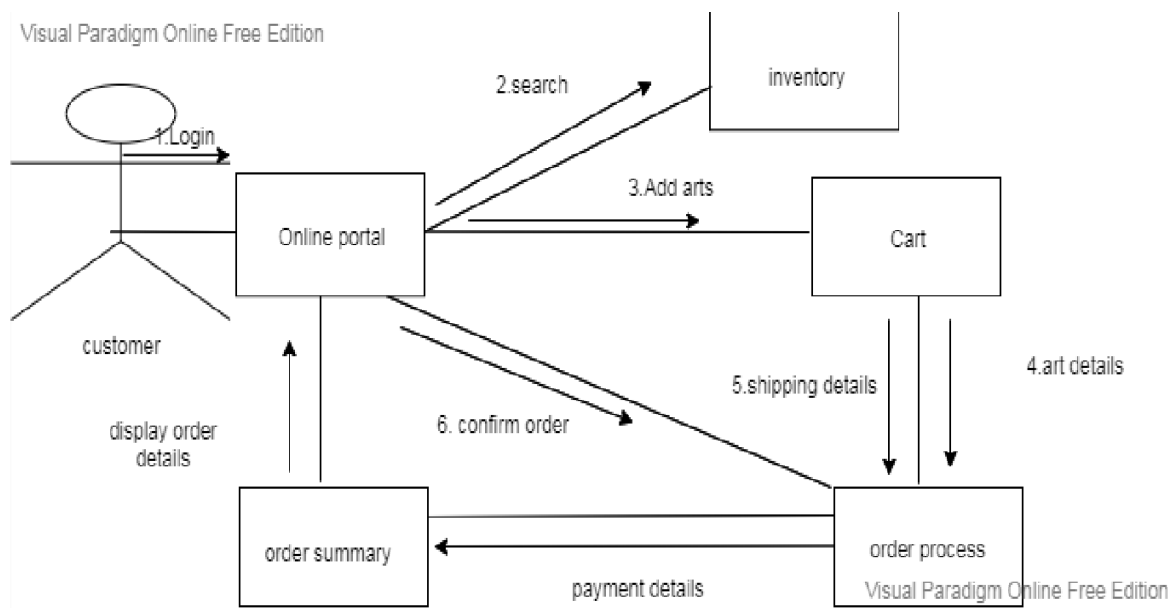
1. Determine the behavior for which the realization and implementation are specified.
2. Discover the structural elements that are class roles, objects, and subsystems for performing the functionality of collaboration.
  - Choose the context of an interaction: system, subsystem, use case, and operation.
3. Think through alternative situations that may be involved.
  - Implementation of a collaboration diagram at an instance level, if needed.
  - A specification level diagram may be made in the instance level sequence diagram for summarizing alternative situations.

### **Benefits of a Collaboration Diagram**

1. The collaboration diagram is also known as Communication Diagram.
2. It mainly puts emphasis on the structural aspect of an interaction diagram, i.e., how lifelines are connected.
3. The syntax of a collaboration diagram is similar to the sequence diagram; just the difference is that the lifeline does not consist of tails.
4. The messages transmitted over sequencing is represented by numbering each individual message.
5. The collaboration diagram is semantically weak in comparison to the sequence diagram.
6. The special case of a collaboration diagram is the object diagram.
7. It focuses on the elements and not the message flow, like sequence diagrams.
8. Since the collaboration diagrams are not that expensive, the sequence diagram can be directly converted to the collaboration diagram.
9. There may be a chance of losing some amount of information while implementing a collaboration diagram with respect to the sequence diagram.

### Drawback of a Collaboration Diagram

1. Multiple objects residing in the system can make a complex collaboration diagram, as it becomes quite hard to explore the objects.
2. It is a time-consuming diagram.
3. After the program terminates, the object is destroyed.
4. As the object state changes momentarily, it becomes difficult to keep an eye on every single that has occurred inside the object of a system.





## 4.2.6 ACTIVITY DIAGRAM

The activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities.

The activity diagram helps in envisioning the workflow from one activity to another. It put emphasis on the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc.

It is also termed as an object-oriented flowchart. It encompasses activities composed of a set of actions or operations that are applied to model the behavioural diagram.

### Components of an Activity Diagram

Following are the component of an activity diagram:

#### Activities

The categorization of behaviour into one or more actions is termed as an activity. In other words, it can be said that an activity is a network of nodes that are connected by edges. The edges depict the flow of execution. It may contain action nodes, control nodes, or object nodes.

The control flow of activity is represented by control nodes and object nodes that illustrates the objects used within an activity. The activities are initiated at the initial node and are terminated at the final node.



### Activity partition /swim lane

The swim lane is used to cluster all the related activities in one column or one row. It can be either vertical or horizontal. It used to add modularity to the activity diagram. It is not necessary to incorporate swim lane in the activity diagram. But it is used to add more transparency to the activity diagram.

### Forks

Forks and join nodes generate the concurrent flow inside the activity. A fork node consists of one inward edge and several outward edges. It is the same as that of various decision parameters. Whenever a data is received at an inward edge, it gets copied and split crossways various outward edges. It split a single inward flow into multiple parallel flows.

### Join Nodes

Join nodes are the opposite of fork nodes. A Logical AND operation is performed on all of the inward edges as it synchronizes the flow of input across one single output (outward) edge.

### Pins

It is a small rectangle, which is attached to the action rectangle. It clears out all the messy and complicated thing to manage the execution flow of activities. It is an object node that precisely represents one input to or output from the action.

### Notation of an Activity diagram

Activity diagram constitutes following notations:

- **Initial State:** It depicts the initial stage or beginning of the set of actions
- **Final State:** It is the stage where all the control flows and object flows end.
- **Decision Box:** It makes sure that the control flow or object flow will follow only one path.
- **Action Box:** It represents the set of actions that are to be performed.

### **How to draw an Activity Diagram?**

An activity diagram is a flowchart of activities, as it represents the workflow among various activities. They are identical to the flowcharts, but they themselves are not exactly the flowchart. In other words, it can be said that an activity diagram is an enhancement of the flowchart, which encompasses several unique skills.

Since it incorporates swim lanes, branching, parallel flows, join nodes, control nodes, and forks, it supports exception handling. A system must be explored as a whole before drawing an activity diagram to provide a clearer view of the user. All of the activities are explored after they are properly analysed for finding out the constraints applied to the activities. Each and every activity, condition, and association must be recognized.

After gathering all the essential information, an abstract or a prototype is built, which is then transformed into the actual diagram.

Rules that are to be followed for drawing an activity diagram:

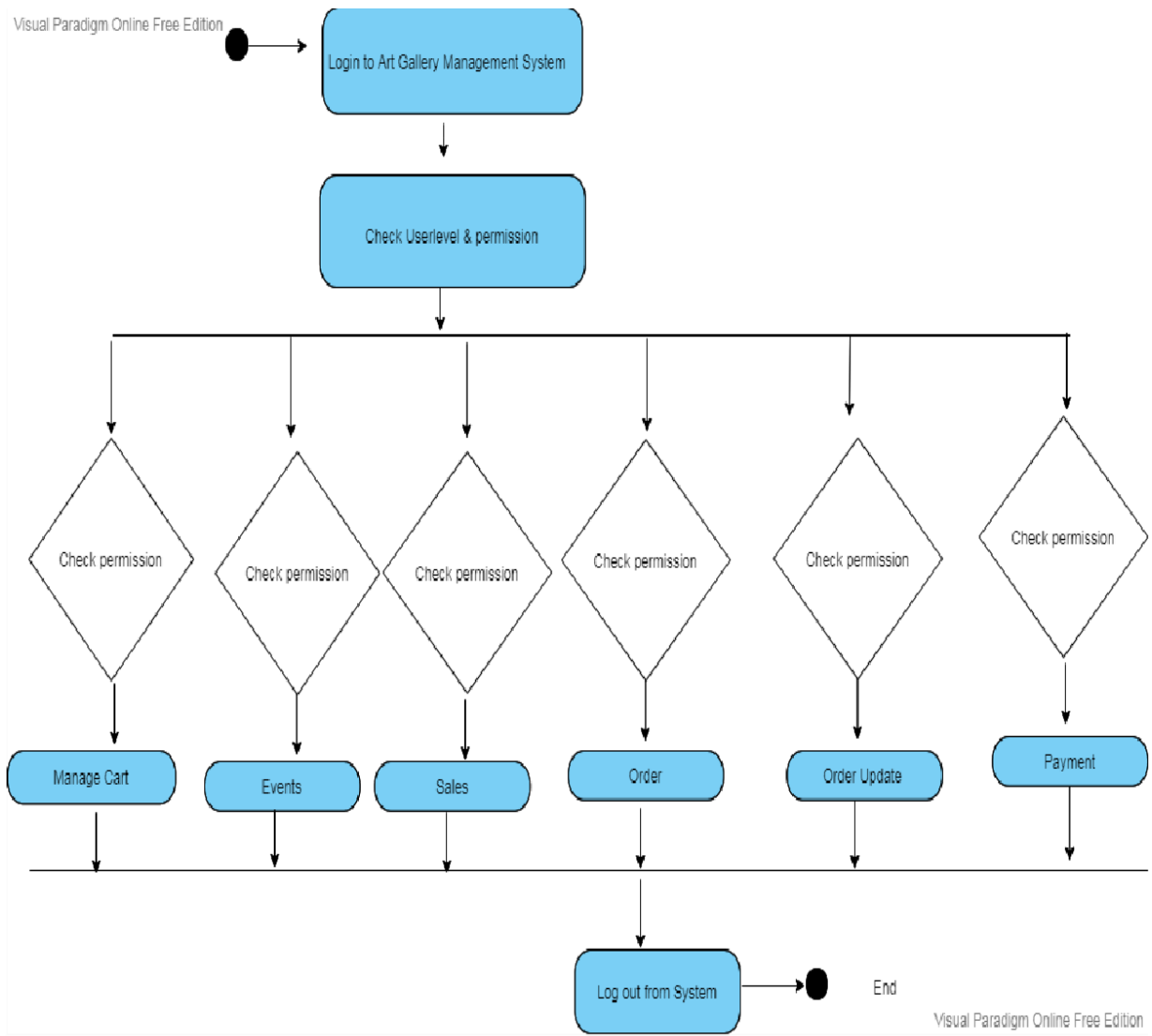
1. A meaningful name should be given to each and every activity.
2. Identify all of the constraints.
3. Acknowledge the activity associations.

### **When to use an Activity Diagram?**

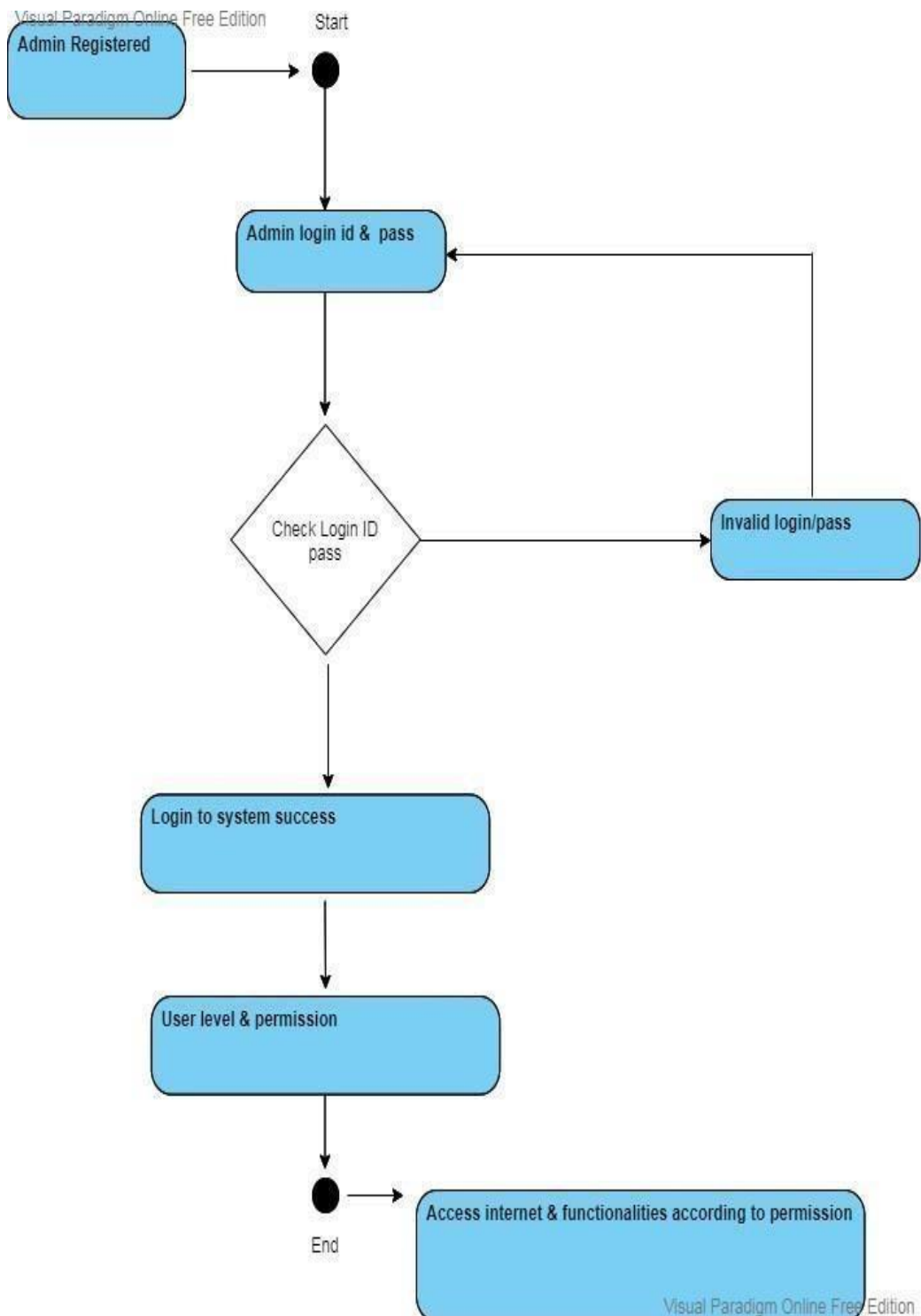
An activity diagram is utilized for the followings:

- To graphically model the workflow in an easier and understandable way.
- To model the execution flow among several activities.
- To model comprehensive information of a function or an algorithm employed within the system.

- To model the business process and its workflow.
- To envision the dynamic aspect of a system.
- To generate the top-level flowcharts for representing the workflow of an application.
- To represent a high-level view of a distributed or an object-oriented system



## Login



## STATE CHART DIAGRAM

A State chart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events.

### Purpose of State chart Diagrams

State chart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. State chart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

State chart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of State chart diagram is to model lifetime of an object from creation to termination.

State chart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system.

Following are the main purposes of using State chart diagrams –

- To model the dynamic aspect of a system.
- To model the life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model the states of an object.

### How to Draw a State chart Diagram?

State chart diagram is used to describe the states of different objects in its life cycle. Emphasis is placed on the state changes upon some internal or external events. These states of objects are important to analyze and implement them accurately.

State chart diagrams are very important for describing the states. States can be identified as the condition of objects when a particular event occurs.

Before drawing a state chart diagram, we should clarify the following points –

- Identify the important objects to be analyzed.
- Identify the states.

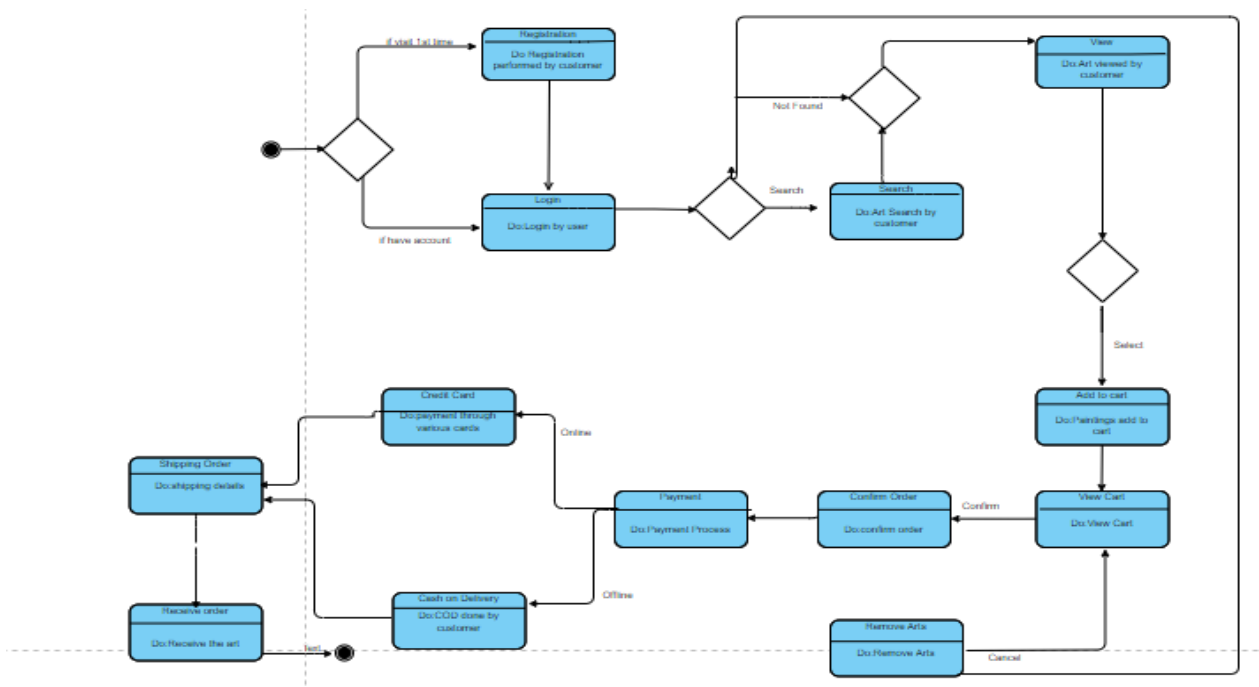
### Where to Use State chart Diagrams?

State chart diagram defines the states of a component and these state changes are dynamic in nature. Its specific purpose is to define the state changes triggered by events. Events are internal or external factors influencing the system.

State chart diagrams are used to model the states and also the events operating on the system. When implementing a system, it is very important to clarify different states of an object during its life time and State chart diagrams are used for this purpose. When these states and events are identified, they are used to model it and these models are used during the implementation of the system.

The main usage can be described as –

- To model the object states of a system.
- To model the reactive system. Reactive system consists of reactive objects.
- To identify the events responsible for state changes.
- Forward and reverse engineering.



## 4.2.7 DEPLOYMENT DIAGRAM

The deployment diagram visualizes the physical hardware on which the software will be deployed. It portrays the static deployment view of a system. It involves the nodes and their relationships.

It ascertains how software is deployed on the hardware. It maps the software architecture created in design to the physical system architecture, where the software will be executed as a node. Since it involves many nodes, the relationship is shown by utilizing communication paths.

### Purpose of Deployment Diagram

The main purpose of the deployment diagram is to represent how software is installed on the hardware component. It depicts in what manner a software interacts with hardware to perform its execution.

Both the deployment diagram and the component diagram are closely interrelated to each other as they focus on software and hardware components. The component diagram represents the components of a system, whereas the deployment diagram describes how they are actually deployed on the hardware

Following are the purposes of deployment diagram enlisted below:

1. To envision the hardware topology of the system.
2. To represent the hardware components on which the software components are installed.
3. To describe the processing of nodes at the runtime.

### Symbol and notation of Deployment diagram

The deployment diagram consists of the following notations:

1. A component
2. An artifact
3. An interface
4. A node



### **How to draw a Deployment Diagram?**

The deployment diagram portrays the deployment view of the system. It helps in visualizing the topological view of a system. It incorporates nodes, which are physical hardware. The nodes are used to execute the artifacts. The instances of artifacts can be deployed on the instances of nodes.

Since it plays a critical role during the administrative process, it involves the following parameters:

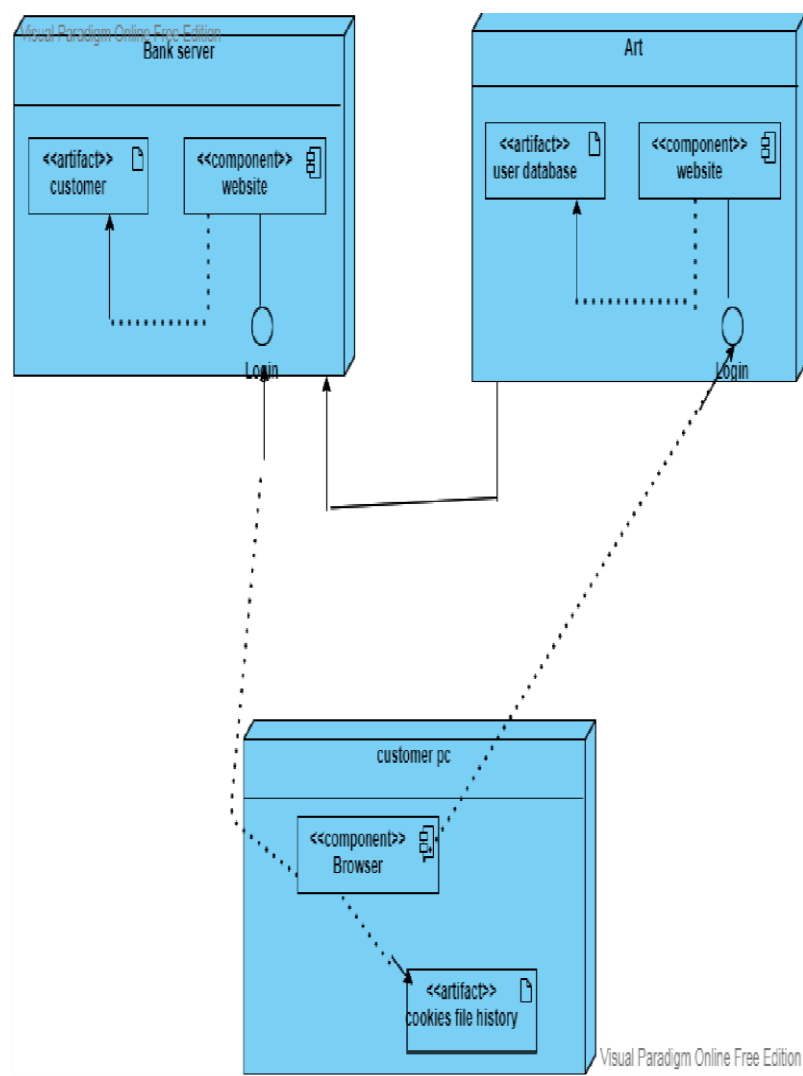
1. High performance
2. Scalability
3. Maintainability
4. Portability
5. Easily understandable

### **When to use a Deployment Diagram?**

The deployment diagram is mostly employed by network engineers, system administrators, etc. with the purpose of representing the deployment of software on the hardware system. It envisions the interaction of the software with the hardware to accomplish the execution. The selected hardware must be of good quality so that the software can work more efficiently at a faster rate by producing accurate results in no time.

Deployment diagrams can be used for the followings:

1. To model the network and hardware topology of a system.
2. To model the distributed networks and systems.
3. Implement forwarding and reverse engineering processes.
4. To model the hardware details for a client/server system.
5. For modelling the embedded system.



## 4.2.8 COMPONENT DIAGRAM

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node.

It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, and it necessitates an interface to execute a function. It is like a black box whose behaviour is explained by the provided and required interfaces.

### Notation of a Component Diagram

- a) A component
- b) A node

### Purpose of a Component Diagram

it holds distinct purposes. It describes all the individual components that are used to make the functionalities, but not the functionalities of the system. It visualizes the physical components inside the system. The components can be a library, packages, files, etc.

The component diagram also describes the static view of a system, which includes the organization of components at a particular instant. The collection of component diagrams represents a whole system.

**The main purpose of the component diagram are enlisted below:**

1. It envisions each component of a system.
2. It constructs the executable by incorporating forward and reverse engineering.
3. It depicts the relationships and organization of components.

### **Why use Component Diagram?**

The component diagrams have remarkable importance. It is used to depict the functionality and behavior of all the components present in the system, unlike other diagrams that are used to represent the architecture of the system, working of a system, or simply the system itself.

In UML, the component diagram portrays the behavior and organization of components at any instant of time. The system cannot be visualized by any individual component, but it can be by the collection of components.

Following are some reasons for the requirement of the component diagram:

1. It portrays the components of a system at the runtime.
2. It is helpful in testing a system.
3. It envisions the links between several connections.

### **When to use a Component Diagram?**

It represents various physical components of a system at runtime. It is helpful in visualizing the structure and the organization of a system. It describes how individual components can together form a single system. Following are some reasons, which tells when to use component diagram:

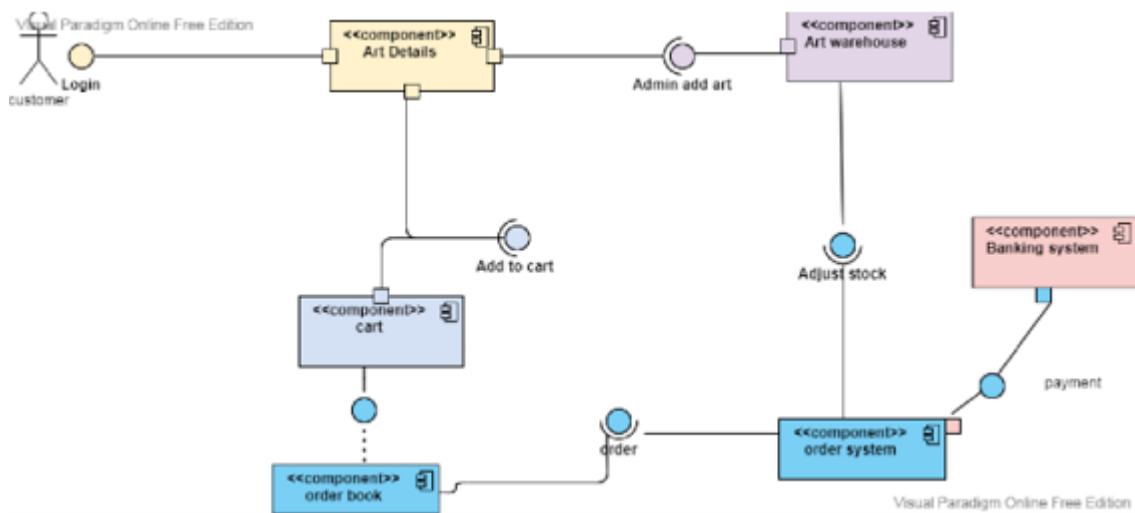
1. To divide a single system into multiple components according to the functionality.
2. To represent the component organization of the system.

### **Where to use Component Diagrams?**

The component diagram is a special purpose diagram, which is used to visualize the static implementation view of a system. It represents the physical components of a system, or we can say it portrays the organization of the components inside a system. The components, such as libraries, files, executables, etc. are first needed to be organized before the implementation.

The component diagram can be used for the followings:

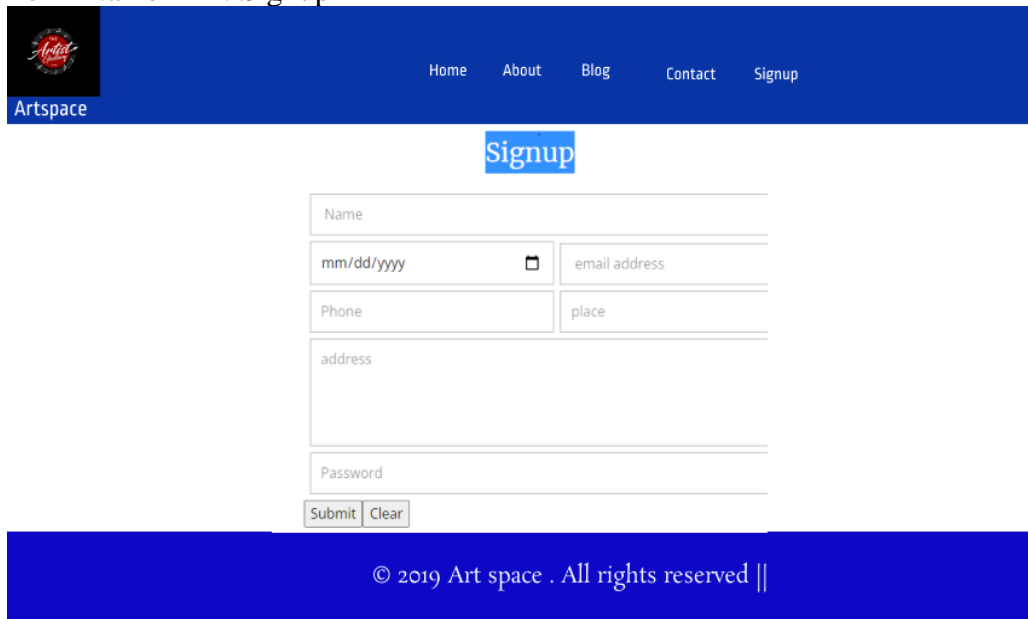
1. To model the components of the system.
2. To model the schemas of a database.
3. To model the applications of an application.
4. To model the system's source code.



## 4.5 USER INTERFACE DESIGN USING FIGMA

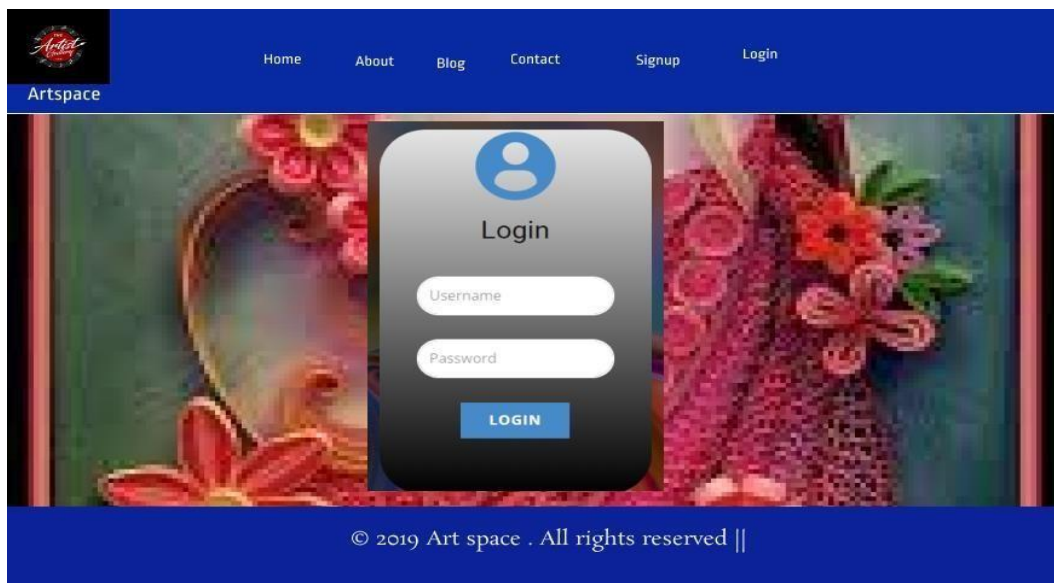
### 4.5.1-INPUT DESIGN

Form Name : Signup

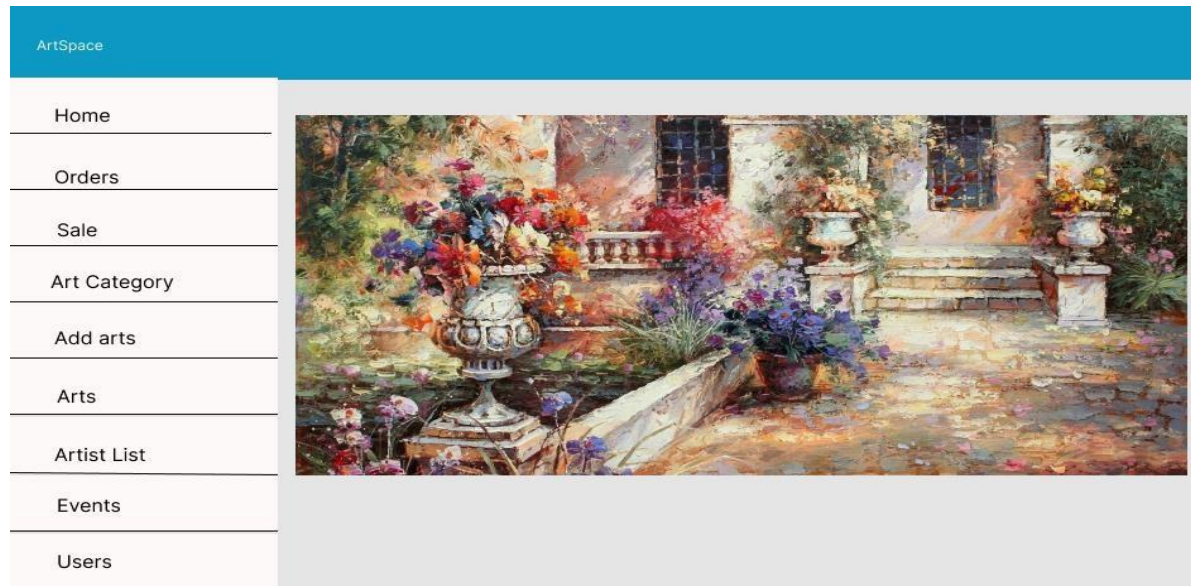





The Signup form is displayed on a blue background. At the top left is the 'Artspace' logo. To the right is a navigation menu with links: Home, About, Blog, Contact, and Signup. Below the navigation menu is a 'Signup' button. The form itself consists of several input fields: 'Name', 'mm/dd/yyyy' (with a calendar icon), 'email address', 'Phone', 'place', 'address', and 'Password'. At the bottom of the form are 'Submit' and 'Clear' buttons. Below the form is a blue footer bar with the text '© 2019 Art space . All rights reserved ||'.

Form Name: Login



The Login form is displayed on a blue background. At the top left is the 'Artspace' logo. To the right is a navigation menu with links: Home, About, Blog, Contact, Signup, and Login. Below the navigation menu is a 'Login' button. The form itself consists of a 'Login' title, a 'Username' input field, a 'Password' input field, and a 'LOGIN' button. The form is overlaid on a background image of a painting. Below the form is a blue footer bar with the text '© 2019 Art space . All rights reserved ||'.

**Form Name : Admin Dashboard****Form Name : Sales**

ArtSpace							
Home	SNo:-	Date	Image	Cavas Information	Price	Status	Action
Orders	1	May 10,2022 11:51 AM		Title: <b>Dancing Theme</b> Artist: <b>Ram Sharma</b> Description: <i>Attracting the Beauty Of Everyone.</i>	200.00	For Sale	<a href="#">Edit</a> <a href="#">Delete</a>
Sale	2	May 04,2022 12:26 PM		Title: <b>Lady In Colours</b> Artist: <b>Williams</b> Description: <i>Beautiful Lady</i>	250.00	For Sale	<a href="#">Edit</a> <a href="#">Delete</a>
Art Category	3	Apr 04,2022 09:09 AM		Title: <b>Mars</b> Artist: <b>Ravi Verma</b> Description: <i>sample</i>	164.00	For Sale	<a href="#">Edit</a> <a href="#">Delete</a>
Add arts							
Arts							
Artist List							
Events							
Users							

**Form Name : Arts**

ArtSpace							
Home	SNo:- ▲	Art Title	Description	Artist	Status	Action	
Orders	1	Mixed Colours	Attractive	Ravi Verma	Unpublished	View	Edit
Sale						Delete	
Art Category	2	Lady In Colours	Beautiful Lady	Williams	Published	View	Edit
						Delete	
Add arts	3	Seashore	Mixed Colours	Anjana	Published	View	Edit
Arts						Delete	
Artist List	4	Mixed Designs	Different Styles	Samuel Varghuese	Published	View	Edit
						Delete	
Events	5	Portrait	Amazing Paintings	Williams	Published	View	Edit
Users						Delete	

**Form Name: Events**

ArtSpace						
Home	Show 10 entries			Search: <input type="text"/>		
Orders	SNo:- ▲	Schedule	Event	Artist	Action	
Sale	1	Apr 06, 2022 06:21 AM	Kalasuthram	Samuel Varghuese	View	Edit
Art Category					Delete	
Add arts	2	Apr 15, 2022 06:36 AM	VI Monisha Exhibition & Events	Ravi Verma	View	Edit
Arts					Delete	
Artist List	3	Apr 18, 2022 09:02 AM	Dubai Expo	Anjana	View	Edit
Events					Delete	
Users	4	Apr 29, 2022 09:03 AM	International Art Events 2022	Ravi Verma	View	Edit
					Delete	



## 4.5.2 OUTPUT DESIGN

### Login



### Registration



### Signup

Username	
dd-mm-yyyy	Email
Phone	place
address	
password	
Submit	Clear

## Admin Dashboard

ArtSpace

Admin

Home

Orders

Sale

Art Category

Add Arts


Arts

Artist List

Events

Users

Welcome to Art Gallery



## Sale

ArtSpace

Admin

Home

Orders

Sale

Art Category

Add Arts

Arts

Artist List

Events



Users

List of Arts for Sale

+ New

Show 10 entries

Search:

SNo:-	Date	Image	Canvas Information	Price	Status	Action
1	May 10, 2022 11:51 AM		Title: <b>Dancing Theme</b> Artist: <b>Ram Sharma</b> Description: <i>Attracting the Beauty Of Everyone.</i>	200.00	For Sale	<a href="#">Edit</a> <a href="#">Delete</a>
2	May 04, 2022 12:26 PM		Title: <b>Lady In Colours</b> Artist: <b>Williams</b> Description: <i>Beautiful Lady</i>	250.00	For Sale	<a href="#">Edit</a> <a href="#">Delete</a>

## Arts

ArtSpace

Admin

Home

Orders

Sale

Art Category

Add Arts

Arts

Artist List

Events

Users

List of Arts

+ New Entry

Show 10 entries

Search:

SNo:-	Art Title	Description	Artist	Status	Action
1	Mixed Colours	Attractive	Ravi Verma	Unpublished	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
2	Lady In Colours	Beautiful Lady	Williams	Published	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
3	Seashore	Mixed Colours	Anjana	Published	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>
4	Mixed Designs	Different Styles	Samuel Varghuese	Published	<a href="#">View</a> <a href="#">Edit</a> <a href="#">Delete</a>

## Events

ArtSpace

Admin ▾

Home

Orders

Sale

Art Category

Add Arts

Arts

Artist List

**Events**

Users

List of Events

+ New Entry

Show 10 entries

Search:

SNo:-	Schedule	Event	Artist	Action
1	Apr 06, 2022 06:21 AM	Kalasuthram	Samuel Varghuese	<div>View Edit</div> <div>Delete</div>
2	Apr 15, 2022 06:36 AM	VI Monisha Exhibition & Events	Ravi Verma	<div>View Edit</div> <div>Delete</div>
3	Apr 18, 2022 09:02 AM	Dubai Expo	Anjana	<div>View Edit</div> <div>Delete</div>
4	Apr 29, 2022 09:03 AM	International Art Events 2022	Ravi Verma	<div>View Edit</div> <div>Delete</div>

## 4.6. DATABASE DESIGN

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two-level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

### 4.6.1 Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a table represents a set of related values.

#### Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of  $n$  elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain  $D$  is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values.

Every value in a relation is atomic, that is not decomposable.

## Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

### 4.6.2 Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- ✓ Normalize the data.
- ✓ Choose proper names for the tables and columns.
- ✓ Choose the proper name for the data.

### **First Normal Form**

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words, 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be done by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

### **Second Normal Form**

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

### **Third Normal Form**

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on other non-key attribute.

**TABLE DESIGN**

Tbl\_registration

Primary Key: -regis\_id

Field Name	Type	Constrain	Description
regis_id	Int	PRIMARY KEY	Describes id of table
Name	Varchar	-----	Represents name of the table
Dob	Date		
Email	Varchar		Shows email in table
Phone	Varchar		Represents phone No
Place	Varchar		Represents place name
Address	Varchar		Shows the address
Password	Varchar		
Status	Varchar		

**Tbl\_login\_admin**

Field name	Type	Constrain	Description
Lid	Int	Primary Key	
Username	Varchar		
Password	Varchar		

**tbl\_art**

Field name	Type	constrain	Description
art_id	Int	Primary key	Shows id of table
Name	Varchar		Shows the name of the table
Image	Varchar		Inserting image to table
Unit price	Bigint		Shows unit price
Status	Varchar		

**Tbl\_artcategory**

Field	Type	Constrain	Description
Art_id	Int	Foreign Key	
Art_category	Varchar		



**Tbl\_events**

Field	Type	Constrain	Description
event_id	Int	Primary key	Shows id of the table
Title	Varchar		Shows content of something
Artist_id	Int	Foreign Key	Shows artists id
Content	Varchar		Shows content
Event date_time	Datetime		Shows date and time
Date_created	Datetime		Shows the datetime
Status	Varchar		

**Tbl\_artist**

Field name	Type	Constrain	Description
artist_id	Int	Primary Key	
regis_id	Int	Foreign Key	
Style	Varchar		

**Tbl\_payment**

Field name	Type	Constrain	Description
order_id	Int	Foreign key	Shows the id
Name	Varchar	Foreign Key	Shows customer name
Address	Varchar	Foreign Key	Shows address
Amount	int		Shows price
Phone_no	Varchar	Foreign Key	Show contact data
Payment_id	int		Shows id
Payment Status	Varchar		Shows current status
Status	Varchar		

**Tbl art\_order**

Field name	Type	Constrain	Description
order_id	Int	Primary key	Shows primary key of the table
—			
art id	Int	Foreign Key	Shows name
Quantity	Varchar		Shows quantity
Status	Varchar		Check the current situation

## **CHAPTER 5**

### **SYSTEM TESTING**

## 5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appear to be working according to the specification, that performance requirement appear to have been met.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

## 5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

- ❖ Unit testing
- ❖ Integration Testing
- ❖ Data validation Testing
- ❖ Output Testing

### 5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code were removed and ensured that all modules are working, and gives the expected result.

### **5.2.2 Integration Testing**

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop. After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover differences in program structures were removed and a unique program structure was evolved.

### **5.2.3 Validation Testing or System Testing**

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

### 5.2.4 Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

### 5.2.5 Selenium Testing

Selenium is one of the most widely used open source Web UI (User Interface) automation testing suite. It was originally developed by Jason Huggins in 2004 as an internal tool at Thought Works. Selenium supports automation across different browsers, platforms and programming languages. Selenium can be easily deployed on platforms such as Windows, Linux, Solaris and Macintosh. Moreover, it supports OS for mobile applications like iOS, windows mobile and android Selenium supports a variety of programming languages through the use of drivers specific to each language. Languages supported by Selenium include C#, Java, Perl, PHP, Python and Ruby with Java and C#. Browsers supported by Selenium include Internet Explorer, Mozilla Firefox, Google Chrome and Safari. Selenium can be used to automate functional tests and can be integrated with automation test tools such as Maven, Jenkins & Docker to achieve continuous testing. It can also be integrated with tools such as TestNG, & JUnit for managing test cases and generating reports.

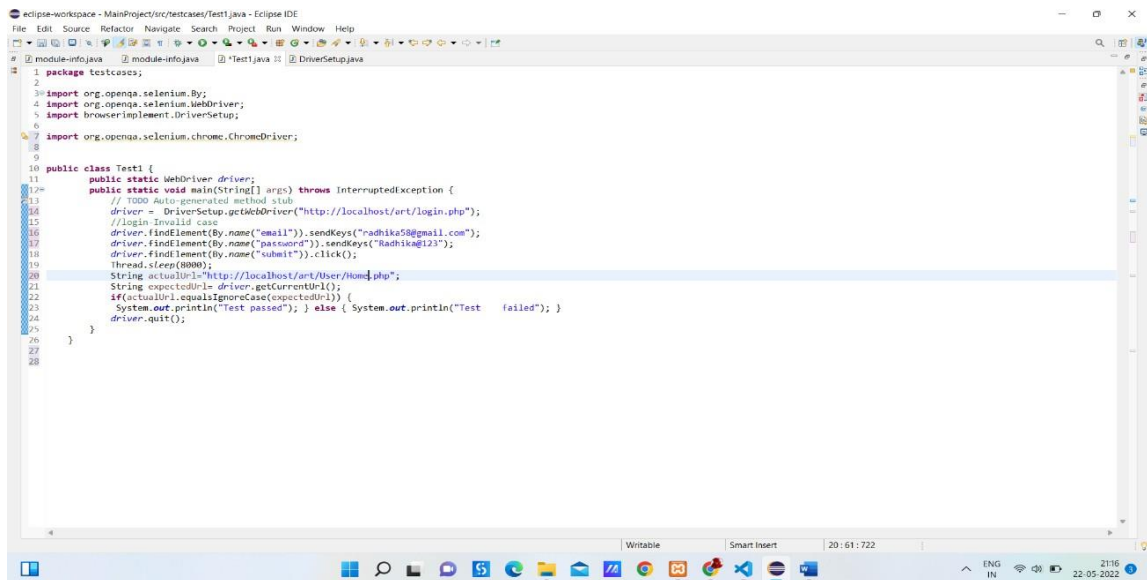
## 5.2.6 TEST CASE

Test Case				
Project Name: Art Gallery Management System				
Login Test Case				
Test Case ID: Fun_1		Test Designed By: Midhi V Mohan		
Test Priority (Low/Medium/High): High		Test Designed Date: 22-05-2022		
Module Name: Login Screen		Test Executed By: Sona Maria Sebastian		
Test Title: Verify login with valid email and password		Test Execution Date: 23-05-2022		
Description: Test the Login Page				
Pre-Condition: User has valid email id and password				
Test Step	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
Navigation to Login Page		Login Page should be displayed	Login page displayed	Pass
Provide Valid Email Id	Email radhika58@g mail.com	User should be able to Login	User Logged in and navigated to Home page	Pass
Provide Valid Password	Password: Radhika@123			
Click on Login button				
Provide Invalid Email Id or Password	Email Id: hii@gmail. Com  Password: Hii23456	User should not be able to Login	Message for enter valid email id or password displayed	Pass
Provide Null Email Id or Password	Email Id: null Password: null			
Click on Button				
Post-Condition: User is validated with database and successfully login into account. The Account session details are logged in database				



## CODE

```
-----  
package testcases;  
  
import org.openqa.selenium.By;  
import org.openqa.selenium.WebDriver;  
import browserimplement.DriverSetup;  
  
import org.openqa.selenium.chrome.ChromeDriver;  
  
public class Test1 {  
    public static WebDriver driver;  
    public static void main(String[] args) throws InterruptedException {  
        // TODO Auto-generated method stub  
        driver = DriverSetup.getWebDriver("http://localhost/art/login.php");  
        //login-Invalid case  
        driver.findElement(By.name("email")).sendKeys("radhika58@gmail.com");  
        driver.findElement(By.name("password")).sendKeys("Radhika@123");  
        driver.findElement(By.name("submit")).click();  
        Thread.sleep(8000);  
        String actualUrl="http://localhost/art/User/Home.php";  
        String expectedUrl= driver.getCurrentUrl();  
        if(actualUrl.equalsIgnoreCase(expectedUrl)) {  
            System.out.println("Test passed"); } else { System.out.println("Test   failed"); }  
        driver.quit();  
    }  
}
```



The screenshot shows the Eclipse IDE with a Java file named `Test1.java` open. The code is a Selenium test case for a login page. It imports `org.openqa.selenium.*` and `org.openqa.selenium.chrome.ChromeDriver`. The `main` method uses `WebDriver` to navigate to `http://localhost/art/login.php`, find the email and password fields, enter the credentials `radhika58@gmail.com` and `Radhika@123`, and click the submit button. It then checks the current URL against the expected URL `http://localhost/art/User/Home1.php` and prints the result.

```
1 package testcases;
2
3 import org.openqa.selenium.*;
4 import org.openqa.selenium.WebDriver;
5 import org.openqa.selenium.chrome.ChromeDriver;
6
7 import org.openqa.selenium.chrome.ChromeDriver;
8
9 public class Test1 {
10     public static WebDriver driver;
11     public static void main(String[] args) throws InterruptedException {
12         // TODO Auto-generated method stub
13         driver = DriverSetup.getWebDriver("http://localhost/art/login.php");
14         //login invalid case
15         driver.findElement(By.name("email")).sendKeys("radhika58@gmail.com");
16         driver.findElement(By.name("password")).sendKeys("Radhika@123");
17         driver.findElement(By.name("submit")).click();
18         Thread.sleep(8000);
19         String actualUrl = "http://localhost/art/User/Home1.php";
20         String expectedUrl = driver.getCurrentUrl();
21         if(actualUrl.equalsIgnoreCase(expectedUrl)) {
22             System.out.println("Test passed"); } else { System.out.println("Test failed"); }
23         driver.quit();
24     }
25 }
26
27
28
```

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Starting ChromeDriver 101.0.4951.41 (93c720db8323b3ec10d056025ab95c23a31997c9-refs/branch-heads/4951@{#504}) on port 60761
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
May 18, 2022 12:40:16 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
May 18, 2022 12:40:16 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Found exact CDP implementation for version 101
Test passed
```

**Test Case 2**

Project Name: Art Gallery Management System					
Login Test Case					
Test Case ID: edit Event			Test Designed By: MIDHI V MOHAN		
Test Priority(Low/Medium/High):High			Test Designed Date: 20-05-2022		
Module Name: Login Screen					
Test Title: edit event details			Test Execution Date: 21-05-2022		
Description: Login to system and edit event information, if some error occurs, test will fail					
Pre-Condition :User has valid user name and password					
Step	Test Step	Test Data	Expected Result	Actual Result	Status(Pass/Fail)
1	Navigation to Login Page		Login Page should be	Login page displayed	Pass
2	Provide Valid name	Name: admin	User should be able to Login	Logged in and navigated to Admin Dashboard with records	Pass
3	Provide Valid Password	Password: admin123			
4	Click on Login button				
5	Provide event informations	Input event details	User will be redirected to dashboard	User will be redirected to dashboard	Pass
7	Click on edit button				
8	Provide invalid informations	Input invalid event details	User will be redirected to dashboard	User will be stay on that page showing error message	Pass
9	Click on edit button				
Post-Condition: User is validated with database and successfully login into account. The Account session details are logged in database					

**Code**

```

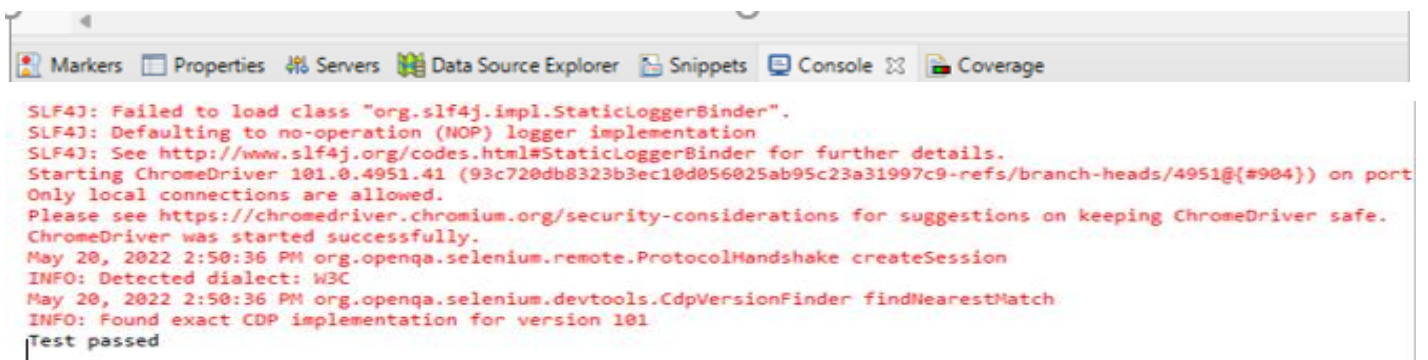
package testcases;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import chromedriver.DriverSetup;

public class EditEvent {
public static WebDriver driver;

```

```
public static void main(String[] args) {  
    driver = DriverSetup.getWebDriver("https://midhi.000webhostapp.com/ admin/login.php");  
    driver.findElement(By.name("login")).click();  
    driver.findElement(By.name("userName")).sendKeys("admin");  
    driver.findElement(By.name("password")).sendKeys("admin123");  
  
    driver.findElement(By.name("submitButton")).click();  
    driver.get("https://midhi.000webhostapp.com/admin/events.php");  
  
    driver.findElement(By.name("eventtitle")).sendKeys("Kalasuthram");  
    driver.findElement(By.name("Artist")).sendKeys("Samuel Vargheese");  
    driver.findElement(By.name("Event Schedule")).sendKeys("2022-04-06");  
    driver.findElement(By.name("svebtn")).click();  
  
    String actualUrl="https://midhi.000webhostapp.com/admin/manage_event.php";  
  
    String expectedUrl= driver.getCurrentUrl();  
    if(actualUrl.equalsIgnoreCase(expectedUrl)) {  
        System.out.println("Test passed");  
    } else {  
        System.out.println("Test failed");  
    }  
}
```



The screenshot shows the Selenium IDE console with the following logs:

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".  
SLF4J: Defaulting to no-operation (NOP) logger implementation  
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.  
Starting ChromeDriver 101.0.4951.41 (93c720db8323b3ec10d056025ab95c23a31997c9-refs/branch-heads/4951@{#904}) on port  
Only local connections are allowed.  
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.  
ChromeDriver was started successfully.  
May 20, 2022 2:50:36 PM org.openqa.selenium.remote.ProtocolHandshake createSession  
INFO: Detected dialect: W3C  
May 20, 2022 2:50:36 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch  
INFO: Found exact CDP implementation for version 101  
Test passed
```

## **CHAPTER 6**

### **IMPLEMENTATION**

## 6.1 INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover.

The implementation state involves the following tasks:

- ☐ Careful planning.
- ☐ Investigation of system and constraints.
- ☐ Design of methods to achieve the changeover.

## 6.2 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to

ensure that the resistance does not build up, as one has to make sure that:

- The active user must be aware of the benefits of using the new system.
- Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won't take place.

### **6.2.1 User Training**

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer-based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

### **6.2.2 Training on the Application Software**

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the data entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy

### **6.2.3 System Maintenance**

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

## **CHAPTER 7**

### **CONCLUSION AND FUTURE SCOPE**



## **7.1 CONCLUSION**

The current system working technology is old fashioned and there is no usage of commonly used technologies like internet, digital money. The proposed system introduces facility for customer to buy art online and view all information. Provides various advantages such as exhibition events, sales etc. To buy paintings online people prefer the service of reliable and trusted web galleries it is more convenient and practical way of spending money on art relics. In a nutshell, online art galleries are the future scope of exhibiting the art works in the market. Gradually it will be more and more improved and grow throughout the world. At the last I should say that technology is the future of everything so as soon the artists adapt it will be beneficial for art world.

## **7.1 FUTURE SCOPE**

- The proposed system is designed in such a way that the payment should be done in online mode.
- Customers can able to do advanced search options
- Data security can be enhanced.

## **CHAPTER 8**

## **BIBLIOGRAPHY**

**REFERENCES:**

- Gary B. Shelly, Harry J. Rosenblatt, “*System Analysis and Design*”, 2009.
- Roger S Pressman, “*Software Engineering*”, 1994.
- PankajJalote, “*Software engineering: a precise approach*”, 2006.
- James lee and Brent ware Addison, “Open-source web development with LAMP”, 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

**WEBSITES:**

- [www.w3schools.com](http://www.w3schools.com)
- [www.jquery.com](http://www.jquery.com)
- <http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf>
- [www.agilemodeling.com/artifacts/useCaseDiagram.html](http://www.agilemodeling.com/artifacts/useCaseDiagram.html)

## **CHAPTER 9**

## **APPENDIX**

## 9.1 Sample Code

```

login.php
<?php
session start ();

?>
<!DOCTYPE html>
<html style="font-size: 16px;">
<head>

<title>Login</title>
<link rel="stylesheet" href="nicepage.css" media="screen">
<link rel="stylesheet" href="Login.css" media="screen">
<script class="u-script" type="text/javascript" src="jquery.js" defer=""></script>
<script class="u-script" type="text/javascript" src="nicepage.js" defer=""></script>

<link id="u-theme-google-font" rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Roboto:100,100i,300,300i,400,400i,500,500i,700,700i,900,900i|Open+Sans:300,300i,400,400i,600,600i,700,700i,800,800i">
<style>
    input[type=submit]{
background-color: #0000FF;
border: none;
color: white;
padding: 16px 32px;
text-decoration: none;
margin: 4px 2px;
cursor: pointer;
    }

</style>

</head>
<body class="u-body">
<header class="u-clearfix u-header u-palette-1-base u-header" id="sec-7e32">
<a href="https://nicepage.com" class="u-image u-logo u-image-1" data-image-width="131" data-image-height="150">

</a><nav class="u-menu u-menu-dropdown u-off canvas u-menu-1">
    <div class="menu-collapse" style="font-size: 1rem; letter-spacing: 0px; font-weight: 700; text-transform: uppercase;">

        </div>
        <div class="u-custom-menu u-nav-container">
            <ul class="u-nav u-spacing-30 u-unstyled u-nav-1"><li class="u-nav-item"><a href="Home.html" style="padding: 10px 0px;">Home</a>
                </li><li class="u-nav-item"><a href="About.html" style="padding: 10px 0px;">About</a>
                </li><li class="u-nav-item"><a href="blog/blog.html" style="padding: 10px 0px;">Blog</a>
            
```

```

</li><li class="u-nav-item"><a href="contact.html" style="padding: 10px 0px;">Contact</a>
    </li><li class="u-nav-item"><a href="signup.php" style="padding: 10px 0px;">Signup</a>
    </li><li class="u-nav-item"><a href="login.php" style="padding: 10px 0px;">Login</a>
    </li></ul>

</div>

</nav><h4 class="u-text u-text-default u-text-1">
    <span style="font-size: 0.8rem;">Artspace</span>
    <span style="font-size: 1.5rem;"></span>

    </h4></header>
    <section class="u-align-left u-clearfix u-image u-shading u-section-1" src="" data-image-
width="150" data-image-height="100" id="sec-2c82">
        <div class="u-clearfix u-sheet u-sheet-1">
            <div class="u-align-center u-container-style u-gradient u-group u-radius-50
u-shape-round u-group-1">
                <div class="u-container-layout u-container-layout-1">

                    <br><br>
                    <form name="form1" action="logincheck.php" method="POST">

                        <h1 style="font-family: Bodoni MT Black;">Login</h1>

                            <div class="end u-form-spacing-30 u-form-vertical " source="custom" style="padding:
10px;">
                                <div class="mail">
                                    <div class="u-form-group u-form-name">
                                        <input type="text" name="uemail" class="u-input u-input-rectangle u-radius-43 u-white form-control"
placeholder="Email" autocomplete="off"/>

                                    </div>
                                    <div class="u-form-group u-form-password">
                                        <input type="password" id="txtpass" name="txtpass" pattern="(?!.*\d)(?!.*[a-z])(?!.*[A-
Z]).{8,}" class="u-input u-input-rectangle u-radius-43 u-white form-control"
placeholder="Password" required>
                                        </div>

                                <div class="u-align-center u-form-group ">

                                    <input type="submit"
value="Login" name="save" onclick="ValidateEmail(document.form1.uemail)"/>

                                </div>
                                <a href="signup.php"><div>Register </div></a>

                                    </div>
                                    <div>

                                        <br>

                                    </form>

```

```
</div>
</div>
</div>
</section>
<script>

    function ValidateEmail (input Text)
    {
    var mailformat = "[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-z]{2,4}$";
    if (inputText.value.match(mailformat))
    {
    alert ("You have entered a valid email address!");
    document. form1. uemail. focus();
    return true;
    }
    else
    {
    alert("You have entered an invalid email address!");
    document.form1. uemail. focus ();
    return false;
    }
    }

var myInput = document.getElementById("txtpass");
var letter = document.getElementById("letter");
var capital = document.getElementById("capital");
var number = document.getElementById("number");
var length = document.getElementById("length");

myInput.onkeyup = function() {

    var lowerCaseLetters = /[a-z]/g;
    if(myInput.value.match(lowerCaseLetters)) {
    letter.classList.remove("invalid");
    letter.classList.add("valid");
    } else {
    letter.classList.remove("valid");
    letter.classList.add("invalid");
    }
    var upperCaseLetters = /[A-Z]/g;
    if(myInput.value.match(upperCaseLetters)) {
    capital.classList.remove("invalid");
    capital.classList.add("valid");
    } else {
    capital.classList.remove("valid");
    capital.classList.add("invalid");
    }
    var numbers = /[0-9]/g;
    if(myInput.value.match(numbers)) {
    number.classList.remove("invalid");
    number.classList.add("valid");
    } else {
    number.classList.remove("valid");
    number.classList.add("invalid");
    }
```

```

}

if(myInput.value.length >= 8) {
    length.classList.remove("invalid");
    length.classList.add("valid");
} else {
    length.classList.remove("valid");
    length.classList.add("invalid");
}
}
</script>

<footer class="u-align-center u-clearfix u-footer u-palette-1-base u-footer" id="sec-1d2c"><div class="u-clearfix u-sheet u-sheet-1">
    <p class="u-small-text u-text u-text-variant u-text-1">Sample text. Click to select the text box. Click again or double click to start editing the text.</p>

    </div></footer>

</body>
</html>

```

### Product.php

```

<?php include './Header.php';
include '../dbcon.php';
?>

<fieldset>
    <legend align="center">ART DETAILS</legend>
    <form name="" method="post" action="ProductDetailsCode.php" enctype="multipart/form-data">
        <center> <table>

        <tr>
            <td>ART CATEGORY</td>
            <td><select name="ddlcategory" required="">
                <option value="">.....</option>
                <?php
                    $sql="SELECT * FROM `artcategory`;
                    $exe= mysqli_query($con,$sql);
                    if(!$exe)
                    {
                        die('Error'.mysqli_error());
                    }
                    $count=mysqli_num_rows($exe);
                    if($count>0)
                    {
                        while ($row= mysqli_fetch_array($exe))
                        {

                            ?>
                            <option><?php echo $row[1]?></option>
                            <?php
                                }

```



```

        }
        ?>
    </select>
</td>
</tr>
<tr><td>NAME</td>
    <td>
        <input type="text" name="txtname" placeholder="" value="" pattern="^[A-Z a-z]+$"
        title="Use alphabets Only" required=""/>
    </td></tr>
</td>
</tr>

<tr><td>UNIT PRICE</td>
    <td>
        <input type="number" name="txtPrice" placeholder="" value="" required="" min="1"/>
    </td></tr>
</td>
<tr><td>IMAGE</td>
    <td><input type="file" name="img" row="10" cols="30"required=""/>
    </td></tr>

    <tr><td></td><td><input type="submit" value="save" class="btn btn-success"
    style="width:100px"/></td></tr>
</table></center>
</form>

<?php include './Footer.php'; ?>

```

### ProductDetailsCode.php

```

<?php
include './dbcon.php';
$productcategory=$_POST['ddlcategory'];
$name=$_POST['txtname'];

$txtPrice=$_POST['txtPrice'];
$i=$_FILES["img"]["name"];

move_uploaded_file($_FILES["img"]["tmp_name"],"pic/".$_FILES["img"]["name"]);
$sql=mysqli_query($con,"INSERT INTO `art` ( `name`, `image`, `unit_price`, `status`) VALUES
('$name','$i','$txtPrice','Active')");

if(!$sql)
{
    die('error creating insertion'.mysqli_error());
}
echo '<script type="text/javascript">
alert("Product Saved..");
window.location="Product.php";
</script>';

?>

```

**View\_events.php**

```

    <?php include 'db_connect.php' ?>
    <?php
    if(isset($_GET['id'])){
    $qry = $conn->query("SELECT e.*,u.name as aname FROM events e inner join users
    u on u.id = e.artist_id where e.id= '$_GET[id]');
    foreach($qry->fetch_array() as $k => $val){
    $$k=$val;
    }
    }
    ?>
<style type="text/css">
.imgs{
    margin: .5em;
    max-width: calc(100%);
    max-height: calc(100%);
}
.imgs img{
    max-width: calc(100%);
    max-height: calc(100%);
    cursor: pointer;
}

</style>
<div class="container-field">
<div class="col-lg-12">
<div class="card">
<div class="card-body">
<div class="row">
<div class="col-md-12">

    <?php
    $images = array();
    if(isset($id)){
    $fpath = 'assets/uploads/event_'.$id;
    $images= scandir($fpath);
    }
    $i = 1;
    foreach($images as $k => $v):
    if(!in_array($v,array('.', '..'))):
    $active = $i == 1 ? 'active' : '';

    ?>
    
    </div>

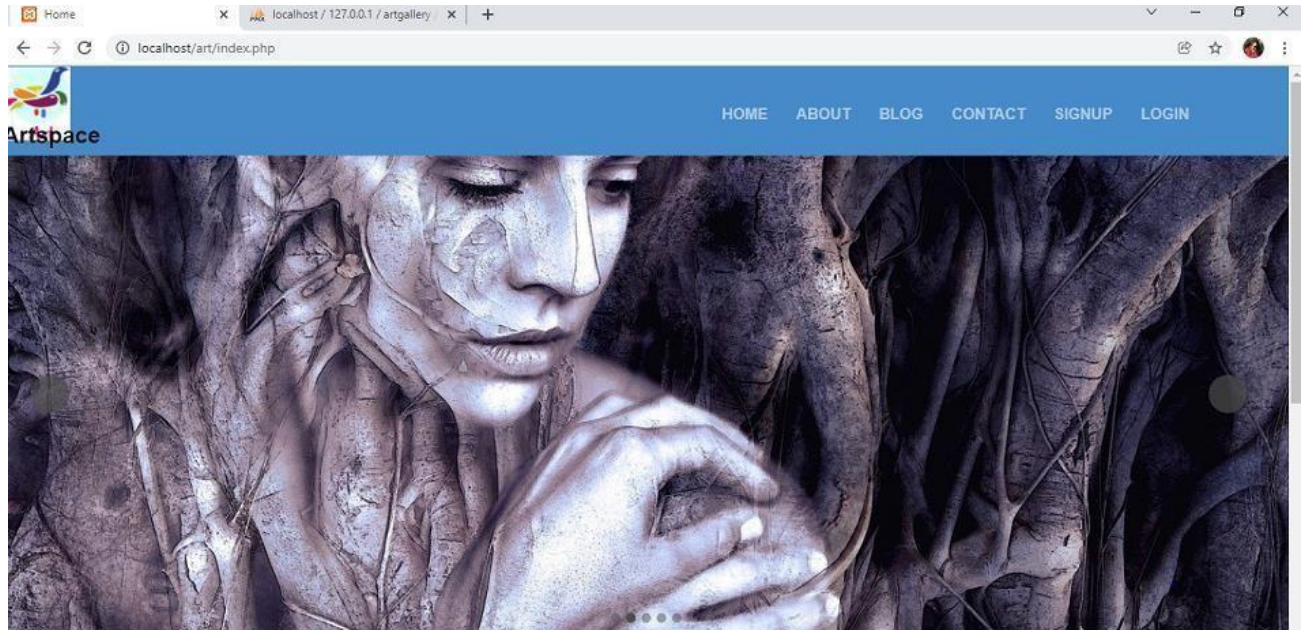
```

---

```
<?php
    $i++;
    else:
        unset($images[$v]);
    endif;
endforeach;
?>
</div>
</div>
</div>
<div class="col-md-12" id="content">
<h4 class="text-center"><b><?php echo ucwords($title) ?></b></h4>
<hr class="divider">
<center><small><?php echo ucwords($aname) ?></small></center>
<br>
<p><b><i class="fa fa-calendar"></i>
<?php echo date("F d, Y h:i A",strtotime($event_datetime)) ?></b></p>
<?php echo html_entity_decode($content); ?>
</div>
</div>
</div>
</div>
</div>
</div>
```

## 9.2 Screen Shots

### Home page



### Add Art page

ArtSpace Admin ▾

- Home
- Orders
- Sale
- Art Category
- Add Arts
- Arts
- Artist List
- Events
- Users

### ART DETAILS

ART CATEGORY

NAME

UNIT PRICE

IMAGE  No file chosen

## View, Edit, Delete Art page

ArtSpace

Admin

Home

Orders

Sale

Art Category

Add Arts

**Arts**

Artist List

Events

Users

List of Arts

+ New Entry

Show 10 entries

Search:

SNo:-	Art Title	Description	Artist	Status	Action
1	Mixed Colours	Attractive	Ravi Verma	Unpublished	View Edit Delete
2	Lady In Colours	Beautiful Lady	Williams	Published	View Edit Delete
3	Seashore	Mixed Colours	Anjana	Published	View Edit Delete
4	Mixed Designs	Different Styles	Samuel Varghuese	Published	View Edit Delete

Add Events page (Edit, View, Delete)

ArtSpace

Admin

Home

Orders

Sale

Art Category

Add Arts

Arts

Artist List

**Events**

Users

List of Events

+ New Entry

Show 10 entries

Search:

SNo:-	Schedule	Event	Artist	Action
1	Apr 06, 2022 06:21 AM	Kalasuthram	Samuel Varghuese	View Edit Delete
2	Apr 15, 2022 06:36 AM	VI Monisha Exhibition & Events	Ravi Verma	View Edit Delete
3	Apr 18, 2022 09:02 AM	Dubai Expo	Anjana	View Edit Delete
4	Apr 29, 2022 09:03 AM	International Art Events 2022	Ravi Verma	View Edit Delete
5	Apr 15, 2022 12:06 PM	Treasure Expo	Ram Sharma	View Edit Delete

**Arts For Sale**

ArtSpace

Admin ▾

Home

Orders

**\$ Sale**

Art Category

Add Arts

Arts

Artist List



Events

Users

List of Arts for Sale + New

Show 10 ▾ entries

Search:

SNo:- ▲	Date ▲	Image ▲	Cavas Information ▲	Price ▲	Status ▲	Action ▲
1	May 10,2022 11:51 AM		<b>Title: Dancing Theme</b> Artist: <b>Ram Sharma</b> Description: <i>Attracting the Beauty Of Everyone.</i>	199.00	<b>For Sale</b>	<a>Edit</a> <a>Delete</a>
2	May 04,2022 12:26 PM		<b>Title: Lady In Colours</b> Artist: <b>Williams</b> Description: <i>Beautiful Lady</i>	250.00	<b>For Sale</b>	<a>Edit</a> <a>Delete</a>