Visvesvaraya Technological University Belagavi, Karnataka- 590018

A
Mini Project Report
On

"ART GALLERY MANAGEMENT SYSTEM"

Submitted in partial fulfilment of the requirements for the DBMS Laboratory with mini project (18CSL58) course of the 5th semester

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted by,

SANJANA S
1JS20CS142

YASHASVI G M
1JS20CS187

Under the guidance of

Mrs. K S Rajeshwari
Assistant Professor, Dept of CSE
JSSATE, Bengaluru

Mrs. Pooja H
Assistant Professor, Dept of CSE
JSSATE, Bengaluru

JSS ACADEMY OF TECHNICAL EDUCATION, BENGALURU

Department of Computer Science and Engineering

2022-2023

JSS MAHAVIDHYAPEETHA, MYSURU

JSS Academy of Technical Education
JSS Campus, Uttarahalli, Kengeri main road, Bengaluru – 560060

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the mini-project work entitled "**ART GALLERY MANAGEMENT SYSTEM**" is a Bonafide work carried out by **Ms. SANJANA S (1JS20CS142)** and **Ms. YASHASVI G M (1JS20CS187)** in partial fulfilment for the Database Management Systems Laboratory with Mini Project (18CSL58) of 5th semester **Bachelor of Engineering in Computer Science and Engineering** of the **Visvesvaraya Technological University, Belgaum** during the academic year 2022-2023. It is certified that all corrections and suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Mrs. K S Rajeshwari	**Mrs. Pooja H**	**Dr. P B Mallikarjuna**
Assistant Professor,	Assistant Professor,	Associate Professor and Head,
Dept of CSE	Dept of CSE	Dept of CSE
JSSATE, Bengaluru	JSSATE, Bengaluru	JSSATE, Bengaluru

NAME OF THE EXAMINERS	SIGNATURE WITH DATE
1) ………………………………	1) ……………………………
2) ………………………………	2) ……………………………

ABSTRACT

This report is meant to serve as a guide for our project – **Art Gallery Management System**. It not only describes our project but also states its importance. This report describes the technical requirements of our project and also input and output views which would interact with the application. The material present in this report has been designed so well that it will be in good use to the user.

Online Art Gallery is an e-commerce application software and it is very helpful for the art lovers and others who wants to know the venue where this kind of arts will be sold. This application helps the customers to search for their arts and paintings as well as can place order for the selected pieces. The user can also get information about the art exhibition so they can visit those exhibitions.

There will be two types of user's roles in the website, the first one is the customer who can search and discover the art. If they want to buy any art they have to register and login to buy the art. The second one is the admin who has to login to post the artworks of the artist for exhibiting and selling on the website. The admin also specifies authority and restrictions on the users. He looks into the feedback of the customer and accounts of the artworks sold.

ACKNOWLEDGEMENT

We express my humble pranamas to His Holiness Jagadguru **Sri Sri Sri Shivarathri Deshikendra Mahaswamiji** who has showered their blessings on us for framing our career successfully.

We express our sincere thanks to our beloved principal, **Dr. Bhimasen Soragaon** for having supported us in our academic endeavours.

We are also indebted to **Dr. P B Mallikarjuna**, Head of Department of Computer Science and Engineering for the facilities and support extended towards us.

We are thankful to the resourceful guidance, timely assistance and graceful gesture of our guide **Mrs. K S Rajeshwari**, Assistant Professor, Department of Computer Science and Engineering, and **Mrs. Pooja H**, Assistant Professor, Department of Computer Science and Engineering, who has helped us in every aspect of our project work.

And last but not the least, we would be very pleased to express our heart full thanks to **all the teaching and non-teaching staff of CSE department** and **our friends** who have rendered their help, motivation and support.

The completion of any project involves the efforts of many people. We have been lucky enough to have received a lot of help and support from all quarters during the making of this project so with gratitude, we take this opportunity to acknowledge all those who have given guidance and encouragement helped us emerge successful.

SANJANA S
1JS20CS142

YASHASVI G M
1JS20CS187

LIST OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

1.1 Introduction to DBMS

Database Management Systems (DBMS) are software systems used to store, retrieve, and run queries on data. A DBMS serves as an interface between an end-user and a database, allowing users to create, read, update, and delete data in the database.

DBMS manage the data, the database engine, and the database schema, allowing for data to be manipulated or extracted by users and other programs. This helps provide data security, data integrity, concurrency, and uniform data administration procedures. DBMS offer many benefits over traditional file systems, including flexibility and a more complex backup system. Database management systems can be classified based on a variety of criteria such as the data model, the database distribution, or user numbers.

There is a wide range of database software solutions, including both enterprise and open-source solutions, available for database management.

Here are some of the most popular database management systems:

Oracle

Oracle Database is a commercial relational database management system. It utilizes enterprise-scale database technology with a robust set of features right out of the box. It can be stored in the cloud or on-premises.

MySQL

MySQL is a relational database management system that is commonly used with open-source content management systems and large platforms like Facebook, Twitter, and You-Tube.

SQL Server

Developed by Microsoft, SQL Server is a relational database management system built on top of structured query language (SQL), a standardized programming language that allows database administrators to manage databases and query data.

1.2 Introduction to SQL

SQL which is an abbreviation for **Structured Query Language** is a language to request data from a database, to add, update, or remove data within a database, or to manipulate the metadata of the database.

Sometimes SQL is characterized as **non-procedural** because procedural languages generally require the details of the operations to be specified, such as opening and closing tables, loading and searching indexes, or flushing buffers and writing data to file systems. Therefore, SQL is designed at a higher conceptual level of operation than procedural languages.

Commonly used statements are grouped into the following categories:

Data Query Language (DQL)

[1] SELECT-Used to retrieve certain records from one or more tables.

Data Manipulation Language (DML)

[1] INSERT - Used to create a record

[2] UPDATE - Used to change certain records.

[3] DELETE - Used to delete certain records.

Data Definition Language (DDL)

[1] CREATE - Used to create a new table, a view of a table, or other object in database.

[2] ALTER - Used to modify an existing database object, such as a table.

[3] DROP - Used to delete an entire table, a view of a table or other object in the database.

Data Control Language (DCL)

[1] GRANT - Used to give a privilege to someone

[2] REVOKE - Used to take back privileges granted to someone.

SQL was one of the first commercial languages to use Edgar F. Codd’s relational model. The model was described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks". Despite not entirely adhering to the relational model as described by Codd, it became the most widely used database language.

1.3 Project Introduction

The topic of our mini project is Online Art Gallery Management. This would be an online website where people buy artworks. Our website consists of all kinds of paintings, fine arts, photography, and many kinds of artworks. The system is specially designed for the analysis of the data that is stored about the sales of purchase and art images. The "Online Art Gallery" has been developed to override the problems prevailing in the practicing of manual system. Moreover, this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

This website contains homepage, about us, customer registration, contact us, admin and customer login. The first page is home page which gives the summary of the website whatever the website contains. Second page contains the about page which contains the information of the website. Third page is the contact us page where customer can contact the website for further details regarding the artworks and the admin shall look into it. From home page we can navigate other pages such as admin and customer login. The purpose of Online Art Gallery is to automate the existing manual system by the help of computerized equipment and full-fledged computer software, fulfilling requirements, so that their valuable data can be stored for a longer period with easy access and manipulating the same.

Where the admin of the website maintains the records of the customer in book. The data of the artworks are stored securely and users can access data only when they register to the website first. Art Gallery brings you the opportunity to view online art exhibitions at our Online Art Gallery. We bring you the details of all the art exhibitions to be held in the forthcoming show. The Online Art Gallery is updated daily, so the user can and buy the latest collection of contemporary art online from any where in the world. You can view and buy the latest Indian Contemporary Art collection available at their exhibitions and also at their online gallery website. This application helps the end-users to search their arts and paintings and can place order for the selected pieces. The end-user can also get the information about the art exhibition and the respective address, so that they can to those exhibitions.

1.4 Objective of the project

[1] To manage the details of gallery, exhibition, artwork and artist. It manages all the sales and inventory in the gallery. The purpose of the project is to build and application program to reduce the manual work.

[2] To tracks all the details about the sales of the artwork, the customer that bought it, etc. It manages the information about the artwork. Provides an information and description of the artworks left, thereby increasing the efficiency of managing the gallery. The organisation can maintain a computerized record of the artwork present in the gallery.

[3] To help in the utilization of the resources in an effective manner. It maintains a list of all the customers and the various artwork that they have bought and the money that have invested in each.

[4] To maintains the record of exhibitions and various sales made during it. The objective of developing such computerized system is to reduce the paper work and safe of time in art gallery database management, thereby increasing the efficiency and decreasing the work load.

[5] To develop such computerized system is to reduce the paper work and safe of time in art gallery database management, thereby increasing the efficiency and decreasing the work load.

Chapter 2

DESIGN

2.1 ENTITY TYPES

An entity type is defined as a collection of entities, which share a common definition in terms of their attributes. Each entity type is assigned a name for its subsequent identification. It describes the type of the information that is being mastered. An entity type typically corresponds to one or several related tables in database.

There are two types of entities:

1. **Strong entity:**

A strong entity in DBMS is an independent table that doesn't rely on any other tables for its existence. A primary key uniquely identifies each row in the table, and foreign keys are used to relate this table to other tables. NULL values are not allowed in the primary key columns.
For example, 'art info' entity contains art_id as a primary key and hence it is a strong entity.

2. **Weak Entity:**

A weak entity type is a dependent table that relies on another table for its existence; it hasno meaningful attributes of its own except for the foreign key from the parent table. For a weak entity type to exist, it must have some relationship with another (parent) table and must not contain any primary key, otherwise, it wouldn't appear in the database.
From Figure 2.2, we can observe that there are no weak entities in the database as all the entities contain a primary key. Hence all the entities in the database are strong entities.

2.2 ENTITY SET

An entity set is a collection or set of all entities of a particular entity type at any point in time. The type of all the entities should be the same. All **entities** can be **distinctly identified** in an entity set. This is because all the entities have a different set of value for some set of attributes. In our database there exists entity sets in the table categories like (1, 'Black sand beach', 'Painting', 90000, 10000, ' ..\img\painting1.jpg', 'Artist: Shina Choi')

2.3 ATTRIBUTES

Attributes are characteristics or properties of an entity in a database. They are used to describe the entity and are typically represented as columns in a table. Each attribute has a name and a data type, such as text, numbers, date, etc.

In our database the table ‘art info’ contains attributes such as (art id, art name, art type, art price, art discount, art file, artist name, unique code)

2.3.1 TYPES OF ATTRIBUTES

Simple Attribute: An attribute that cannot be further subdivided into components is a simple attribute.
In our database, customer id is simple attribute.

Composite Attribute: An attribute that can be split into components is a composite attribute.
In our database, ‘customer Fname’ and ‘customer Lname’ are composite attributes.

Single-Valued Attribute: The attribute which takes up only a single value for each entity instance is a single-valued attribute.
In our database, ‘Art id’ is single valued.

Multi-Valued Attribute: The attribute which takes up more than a single value for each entity instance is a multi-valued attribute.
In our database, there exists a multivalued attribute that is ‘Event Category’.
Event Category= {painting, fine art}

Derived Attribute: An attribute that can be derived from other attributes is derived attributes.

Stored attribute: The stored attributes are those attributes which doesn’t require any type of further update since they are stored in the database.

For example, the Age and Birth date attributes of a person, the value of Age can be determined from the current (today’s) date and the value of that person’s Birth date The Age attribute is hence called a **derived attribute** Birth date attribute is called a **stored attribute**.

Complex attribute: Those attributes, which can be formed by the nesting of composite and multi-valued attributes, are called complex attribute. These attributes are rarely used in DBMS (Data Base Management System). That’s why they are not so popular. composite and multivalued attributes can be nested arbitrarily arbitrary nesting by grouping components of a composite attribute between parentheses () and separating the components with commas, and by displaying multivalued attributes between braces {}. Such attributes are called complex attributes.

NULL Value Attributes: In some cases, a particular entity may not have an applicable value for an attribute.

2.4 RELATIONSHIP TYPES

One-to-one: A relationship in which one record in a table is related to one and only one record in another table.
One Customer can make one payment.

One-to-many: A relationship in which one record in a table is related to multiple records in another table.
One admin can manage many arts.

Many-to-many: A relationship in which multiple records in a table are related to multiple records in another table.
Many customers can confirm many bookings.

Self-referencing: A relationship in which a table is related to itself.

Hierarchical: A relationship in which one record in a table is related to one or more records in the same table, creating a parent-child relationship.

Associative: A relationship in which two or more tables are related through an additional table, also known as a join table or junction table.

2.5 STRUCTURAL CONSTRAINTS

Structural constraints are rules and limitations that are placed on the design and structure of a database in order to maintain its integrity and consistency. These constraints are used to define the relationships between tables and columns, and to ensure that data is entered and stored in a consistent and predictable manner.

2.6 MIN MAX NOTATIONS

In an Entity-Relationship (ER) diagram, the "min" and "max" notation is used to indicate the minimum and maximum cardinality of a relationship between two entities. Cardinality refers to the number of instances of one entity that can be associated with each instance of another entity.

2.7 ER-TO-RELATIONAL MAPPING

Step 1: For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E.

Step 2: For each weak entity type W in the ER schema with owner entity type E, create a relation R, and include all simple attributes (or simple components of composite attributes) of W as attributes. In addition, include as foreign key attributes of R the primary key attribute(s)of the relation(s) that correspond to the owner entity type(s).

Step 3: For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. Choose one of the relations, say S, and include the primary key of T as a foreign key in S. Include all the simple attributes of R as attributes of S.

Step 4: For each regular binary 1:N relationship type R identify the relation (N) relation S. the primary key of T as a foreign key of S. Simple attributes of R map to attributes of S.

Step 5: For each binary M:N relationship type R, create a relation S. Include the primary keys of participant relations as foreign keys in S. Their combination will be the primary key for S. Simple attributes of R become attributes of S.

Step 6: For each multi-valued attribute A, create a new relation R. This relation will include an attribute corresponding to A, plus the primary key K of the parent relation (entity type or relationship type) as a foreign key in R. The primary key of R is the combination of A and K.

Step 7: For each n-ary relationship type R, where n>2, create a new relation S to represent R. Include the primary keys of the relations participating in R as foreign keys in S. Simple attributes of R map to attributes of S. The primary key of S is a combination of all the foreign keys that reference the participants that have cardinality constraint > 1. For a recursive relationship, we will need a new relation.

2.8 ER DIAGRAM

[1] An entity-relationship model (ER Model) describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types.

[2] An entity may be defined as a thing capable of an independent existence that can be uniquely identified. An entity is an abstraction from the complexities of a domain.

[3] Attributes are drawn as ovals and are connected with a line to exactly one entity or relationship set.

[4] An entity relationship model, also called an entity-relationship (ER) diagram, is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems.

[5] Cardinality constraints are expressed as follows:

[1] A double line indicates a participation constraint, totality or subjectivity: all entities in the entity set must participate in at least one relationship in the relationship set.

[2] An arrow from entity set to relationship set indicates a key constraint, i.e. injectivity: each entity of the entity set can participate in at most one relationship in the relationship set.

[3] A thick line indicates both, i.e. bijectivity: each entity in the entity set is involved in exactly one relationship.

[4] An underlined name of an attribute indicates that it is a key: two different entities or relationships with this attribute always have different values for this attribute.

The Figure 2.1 explains the structure of the database, i.e., it depicts the number, type and name of the entities, the number, type and name of the attributes in each entity, the relationship between the entities, and the constraints.

2.9 ENTITY RELATIONSHIP DIAGRAM

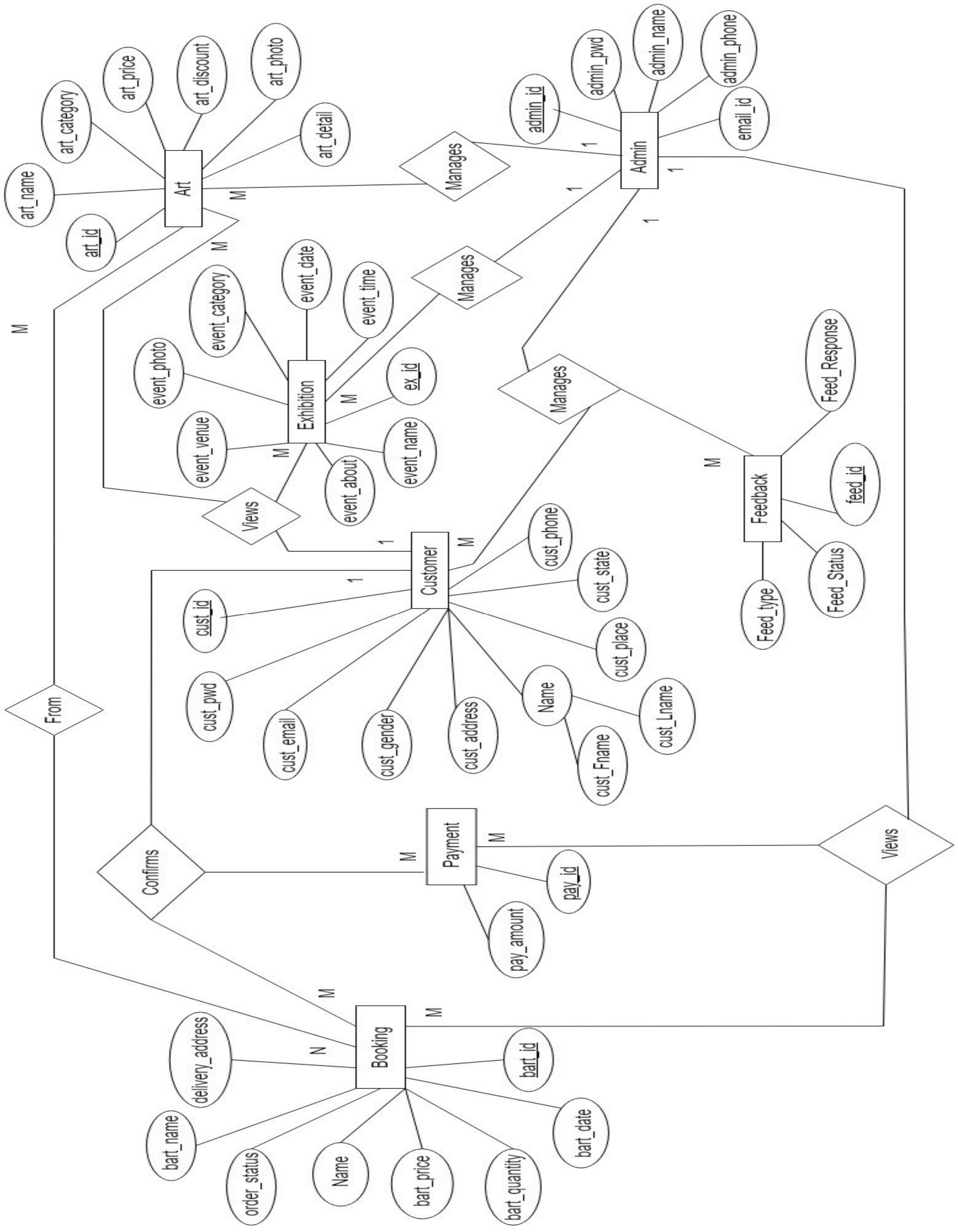


Figure 2.1 Entity Relationship Diagram

2.10 RELATIONAL SCHEMA

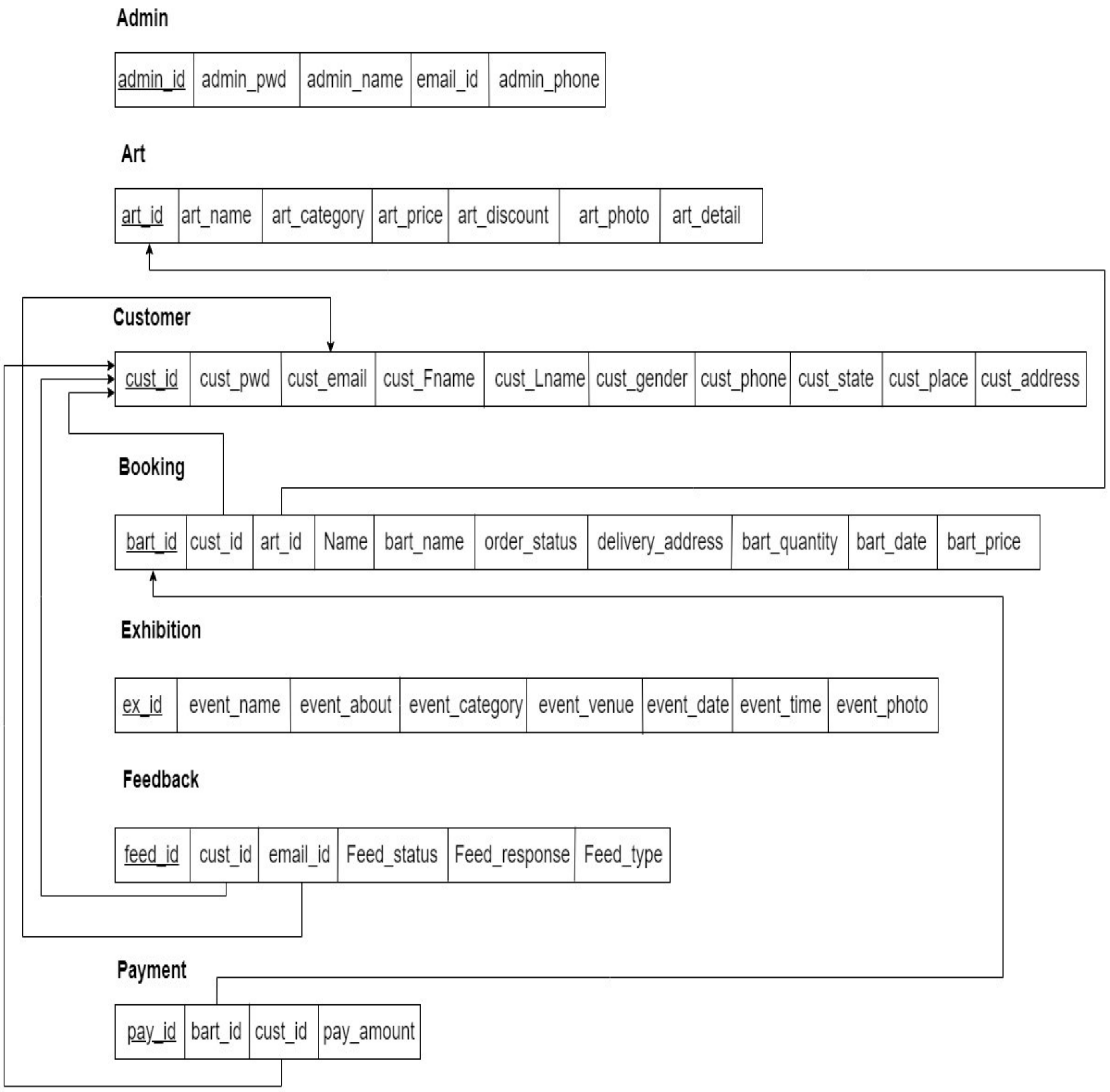


Figure 2.2 Relational Schema

The Figure 2.2 shows the relational schema diagram of our database i.e Art Gallery Management system. It dipicts the database structure logically. It uses tables to show the relationship between the entities. Each table should have atleast one primarey key.

2.11 KEYS IDENTIFICATION

2.11.1 KEYS IN DBMS

A key in DBMS is an attribute or a set of attributes that help to uniquely identify a tuple (or row) in a relation (or table). Keys are also used to establish relationships between the different tables and columns of a relational database. Individual values in a key are called key values. For example, every unique identification number is used to identify candidates in an educational institute. These can also help find all the available details maintained on the server about the candidate, such as their address, passport number, or phone number are keys unique to each candidate. Keys are imperative for analysing and identifying data types.

2.11.2 WHY ARE KEYS REQUIRED?

A key is used in the definitions of various kinds of integrity constraints. A table in a database represents a collection of records or events for a particular relation. Now there can be thousands and thousands of such records, some of which may be duplicated. There should be a way to identify each record separately and uniquely, i.e. no duplicates. Keys allow us to be free from this hassle. A key could either be a combination of more than one attribute (or columns) or just a single attribute. The main motive of this is to give each record a unique identity.

2.11.3 TYPES OF KEYS

Primary Key:

Primary key is a column of a table or a set of columns that helps to identify every record present in that table uniquely. There can be only one primary Key in a table. Also, the primary Key cannot have the same values repeating for any row. Every value of the primary key must be different with no repetitions. Amid many details, a primary key is the most significant one to understand what are keys and what is primary key in DBMS. Figure 2.2 shows primary key attributes in relational schema diagram.

Example

[1] In Admin Table - admin_id is the primary key.
[2] In Art Table - art_id is the primary key.
[3] In Customer Table - cust_id is the primary key.
[4] In Booking Table - bart_id, cust_id, art_id are the primary keys.
[5] In Exhibition Table - ex_id is the primary key.
[6] In Feedback Table - feed_id, cust_id are the primary keys.
[7] In Payment Table - pay_id, bart_id, cust_id are the primary keys.

Foreign Key:

Foreign Key is used to establish relationships between two tables. A foreign key will require each value in a column or set of columns to match the Primary Key of the referential table. Foreign keys help to maintain data and referential integrity. They are essential for maintaining a difference between two entities that might be linked with the same information but do not share similar information. Figure 2.2 shows foreign key attributes in relational schema diagram.

Example

[1] In the Booking Table we have the foreign keys cust_id which is from the table Customer and art_id which is from the table Art.

[2] In the Feedback Table we have the foreign keys cust_id from the table Customer and email_id from the table Customer itself.

[3] In the Payment table we have the foreign keys bart_id from the table Booking and cust_id from the table Customer.

Super Key:

Super Key is the set of all the keys which help to identify rows in a table uniquely. This means that all those columns of a table than capable of identifying the other columns of that table uniquely will all be considered super keys. Super Key is the superset of a candidate key. The primary Key of a table is picked from the super key set to be made the table's identity attribute.

Candidate Key:

Candidate keys are those attributes that uniquely identify rows of a table. The Primary Key of a table is selected from one of the candidate keys. So, candidate keys have the same properties as the primary keys. There can be more than one candidate keys in a table. There can be more candidate keys than just one for any table, but they can never be empty.

Unique Key:

Unique Key is a column or set of columns that uniquely identify each record in a table. All values will have to be unique in this Key. A unique Key differs from a primary key because it can have only one null value, whereas a primary Key cannot have any null values.

Chapter 3
SYSTEM REQUIREMENTS

3.1 FRONT END

HTML: HTML (Hypertext Markup Language) is the standard language used to create web pages. It uses a system of tags and attributes to structure and format the content of a webpage, including text, images, and links. HTML documents are viewed in web browsers and are rendered into a visual representation of the page, which can include text, images, videos, and interactive elements. HTML is the foundation of all websites and is used in conjunction with other languages such as CSS and JavaScript to create dynamic and interactive web pages.

CSS: CSS (Cascading Style Sheets) is a language used to control the presentation and layout of HTML documents. It allows developers to separate the presentation of a webpage from its structure and content, defined in HTML. With CSS, you can control the colours, fonts, spacing, and overall layout of a webpage, as well as add visual effects such as hover states, animations, and transitions. CSS can be written in separate files or included in the same document as the HTML, and it can be applied to individual elements or groups of elements on a webpage. It's a powerful tool that allows developers to create visually appealing and consistent designs across multiple web pages and devices.

JAVASCRIPT: JavaScript is a programming language that is primarily used to create interactive front-end web applications and dynamic website content. It is a scripting language that runs in web browsers and enables developers to create interactive elements such as forms, animations, and other dynamic features. JavaScript is also commonly used on the back-end, through technologies such as Node.js, to create server-side applications. It is a widely-used and versatile language that is supported by all modern web browsers and is easy to learn for both beginners and experienced programmers.

JQUERY: JQuery is a fast, small, and feature-rich JavaScript library. It makes HTML document traversal and manipulation, event handling, and animation much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

AJAX: Ajax (short for Asynchronous JavaScript and XML) is a set of web development techniques used for creating interactive and responsive web applications. It allows for the creation of dynamic web pages without the need for a page refresh. This is achieved by using JavaScript to make asynchronous requests to a server, and updating only the relevant parts of the page with the new data. This allows for a smoother, faster user experience and enables web developers to create more complex and interactive applications. jQuery provides a convenient API for making Ajax requests, which makes it easy to use in a web development project.

BOOTSTRAP: Bootstrap is a free and open-source front-end development framework that helps developers create responsive, mobile-first web pages and web applications. It is a collection of HTML, CSS, and JavaScript components and tools that are designed to be used in conjunction with the Bootstrap CSS framework. Bootstrap provides a consistent, easy-to-use set of classes and styles that can be used to create common web design elements such as forms, buttons, navigation bars, and more. It also includes JavaScript plugins for common web development tasks such as creating modals, carousels, and other interactive elements. One of the benefits of Bootstrap is that it makes it easy to create responsive designs that look and function well on a wide range of devices, from desktop computers to smartphones.

3.2 Back End

PHP v8.0.7: Dynamic and interactive websites. PHP is executed on the server, and the results are sent to the browser as plain HTML. This allows PHP to handle tasks such as reading and writing to files, sending and receiving cookies, and creating and manipulating databases. PHP can be integrated with a variety of web development technologies, including HTML, CSS, and JavaScript. It is often used in conjunction with other web development technologies such as MySQL (a popular open-source database management system) to create dynamic and interactive web applications. PHP is also often used for creating RESTful web services, that can be consumed by other systems. PHP is widely supported by web hosting providers, making it easily accessible and easy to run on a variety of platforms, including Windows, Linux, and MacOS. It's one of the most popular server-side scripting language and has a large community support and many frameworks built on top of it, like Laravel, CodeIgniter, Cake PHP and many more.

MYSQL DATABASE: MySQL is a popular open-source relational database management system (RDBMS) that is used to store, organize, and retrieve data in a structured manner. It is widely used in web applications to store and manage data such as user information, website content, and other data. MySQL uses a structured query language (SQL) to manage the data in the databases, which allows for tasks such as creating tables, inserting and updating data, and retrieving data based on specific criteria. MySQL supports various data types including integers, floating-point numbers, strings and date and time, and it also supports advanced features such as stored procedures, triggers, views, and indexes which are used to optimize the performance of data retrieval. MySQL can be used on a variety of platforms, including Windows, Linux, and MacOS. It's widely supported by a large community and has a wide range of tools and libraries for management, administration and monitoring. It's also known for its high performance and scalability. It's often used in conjunction with other web development technologies such as PHP to create dynamic and interactive web applications.

3.3 Web Server

XAMPP v3.3.0: XAMPP is a free, open-source, and cross-platform web server solution that allows developers to create and test web applications on their local machines. XAMPP stands for Cross-Platform (X), Apache (A), MariaDB (M), PHP (P) and Perl (P). It is a package of software components that includes Apache web server, MariaDB (MySQL) database, PHP, and Perl programming languages. This bundle allows developers to set up a local web server environment with the necessary components to run PHP and MySQL-based web applications without the need for an internet connection. XAMPP is easy to install and use, and it is particularly popular among developers working on WordPress, Joomla, and other PHP-based web applications. With XAMPP, developers can test their code and see how it behaves on a live web server, without having to upload it to a remote web server. This can save time and money, and it also allows developers to work offline. my local webserver that has a PHP Version 8.0.7

3.4 Operating System

Minimum Windows 7

1.5 FUNCTIONAL REQUIREMENTS

Functional Requirements describe the service that the Art Gallery management system must offer, they are subdivided into three access levels: Admin Mode and Customer Mode

Admin Side:

Home Page
List of customers
Modify customer bookings
List of artworks
Add new artworks
Modify artworks
Manage feedbacks
List of exhibition events
Add exhibition events
Modify exhibition events
View Booking details
View Payment Details
Login and Logout

Customer Side:

Customer Register
Welcome page
View Artworks
Booking artworks
View booking details
Cancel booking
Giving feedbacks
View payment details
View upcoming exhibition events
Login and Logout

3.6 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements specify criteria that can be used to judge the operation of a systemas a whole rather than specific behaviours. They describe emergent properties like security, performance, and availability and, unlike the functional requirements that can be worked around, are essential to fulfil for a usable system. The estimation of whether the product fulfilsthe non-functional requirement or not usually reduces to a Boolean answer: yes or no.

For an online art gallery management system, the most important non-functional requirements include security, performance, usability, and availability.

Security:

Art Gallery Management Systems are notorious for being subject to malicious attacks, so security is the major requirement for the system. Unauthorized access to the data is not permissible. The data must be backed up daily and stored in a secured location, at a distance from different facilities of the system. The system also must employ firewall software as a defence against network attacks.

Accept only secure passwords that have sufficient length and non-alphabetic characters, and block login attempts after several unsuccessful trials. Artworks are only viewed when a customer registers first.

Performance:

The art gallery management system is a multi-client system that must reach response time targets for each of the clients during simultaneous calls and must be able to run a target number of bookings per second without failure. The system must effectively utilize the hardware and energy resources to minimize operational costs.

Usability:

The system must provide different graphical interfaces for customers and admins. All system interfaces must be user-friendly and simple to learn, including helping hints and messages and intuitive workflow, especially in a client interface: the client must be able to fastlearn and use the interface without prior knowledge of banking terminology or rules. The interfaces must automatically adjust to devices with different screen sizes, and allow to changetypeface size and colour scheme to improve readability.

Reliability:

The application should be highly reliable and it should generate all the updated information in correct order.

Availability:

The system must be available during all the hours. The cash on delivery service and booking service must be available round-the-clock with minimal maintenance times, reaching 99.999% availability time per year.

Software Requirements:

Software requirements specification (SRS) is the description of the software system that is going to be developed, it is made at the latest phase of analysis, after the functional and non- functional requirements.

1.6 METHODS USED IN BACK END (PHP)

```
Function add_exhibition_events()
{
$sql = "INSERT INTO exhibition_events (event_name, event_about, event_category, event_venue, event_date, event_time, event_photo) VALUES ('$a11', '$a12', '$a13', '$a14', '$a15', '$a16','$target_path')"; // this is an insert query for add product
if (mysqli_query($con, $sql))
{
echo "<script>alert('Your Exhibition & Events has been added!');
 window.location.assign('add_exhibitionevents.php') </script>";
 }
 else {
 echo "Error: " . $sql . "<br>" . mysqli_error($con);
 } }
```

```
Function add_product() {
$sql = "INSERT INTO art_info (art_name, art_category, art_price, art_discount, art_photo, art_detail) VALUES ('$a11', '$a12', '$a13', '$a14', '$target_path', '$a16')"; // this is an insert query  for add product
```

```
if (mysqli_query($con, $sql)) {
echo "<script>alert('Your Product has been added!');
window.location.assign('add_product.php') </script>";
} else {
echo "Error: " . $sql . "<br>" . mysqli_error($con);
}       }

Function delete_customer_detail()
{
$sql = "select * from customer_info"; // this is query for fetching all customer details.
        $run = mysqli_query( $con, $sql );
        echo "<table class='divform'>
        <tr>
        <th>customer Id</th>
        <th>customer password</th>
        <th>customer Email</th>
        <th>customer FirstName</th>
        <th>customer LastName</th>
        <th>customer sex</th>
        <th>customer Phone</th>
        <th>customer state</th>
        <th>customer Place</th>
        <th>customer Address</th>
        <th>Delete</th>
        </tr>";
        while ( $result = mysqli_fetch_array( $run ) ) { // this is function for fetching the data as
an array.
          echo "<tr>
        <td>$result[0]</td>
        <td>$result[1]</td>
        <td>$result[2]</td>
        <td>$result[3]</td>
        <td>$result[4]</td>
        <td>$result[5]</td>
        <td>$result[6]</td>
        <td>$result[7]</td>
        <td>$result[8]</td>
        <td>$result[9]</td>
        <td><a class='button special-red' href='delete_customer_now.php?id=$result[0]'
class='btn btn-danger'>delete</a></td>
         </tr>";
        } }

Function update_exhibition() {
$up = "update exhibition_events set  event_name='$a',  event_about='$b',
event_category='$c'  event_venue='$d',  event_date='$e',event_time='$f',
event_photo='$target_path', event_photo_disp='$target_path_disp' where ex_id='$r'"; // this
query for updating Exhibition & Events Details1
if ( mysqli_query( $con, $up ) ) {
              echo " <script> alert('Exhibition & Events Details Updated Successfully');
window.location.assign('manage_exhibitionevents.php') </script>";
} else {
echo "Exhibition & Events Details Updated Occure Error";
}}

Function delete_exhibition_events() {
$up="delete from exhibition_events where ex_id='$a'"; // this is query for removing
exhibition_events detail.
$run=mysqli_query($con, $up);
        if($run){
              if(file_exists($g)){
                    unlink($g);
              }
              echo " <script>confirm('Exhibition and Events Details Successfully Removed');
window.location.assign('manage_exhibitionevents.php') </script>" }
        else{
              echo "not ok"; }
}
```

3.7 CLASS DIAGRAM

A class diagram is a type of diagram in software engineering that describes the structure of a system by showing the classes, attributes, and relationships between them. The Figure 3.1 describes a class diagram for our database system.

The class diagram has seven classes: Customer_info, Booking_info, Feedback_info, Payment_info, Art_info, Exhibition_events, Admin_info. Each class has a set of attributes, such as id, name, and status, and a primary key.

Additionally, there are several foreign key relationships between the classes. For example, the bart_id attribute in the booking_info class is a primary key that references the pay_id attribute in the payment_info class. Similarly, the cust_id attribute in the feedback_info class is a foreign key that references the id attribute in the cust_info class.

<u>**Class Diagram of our database**</u>

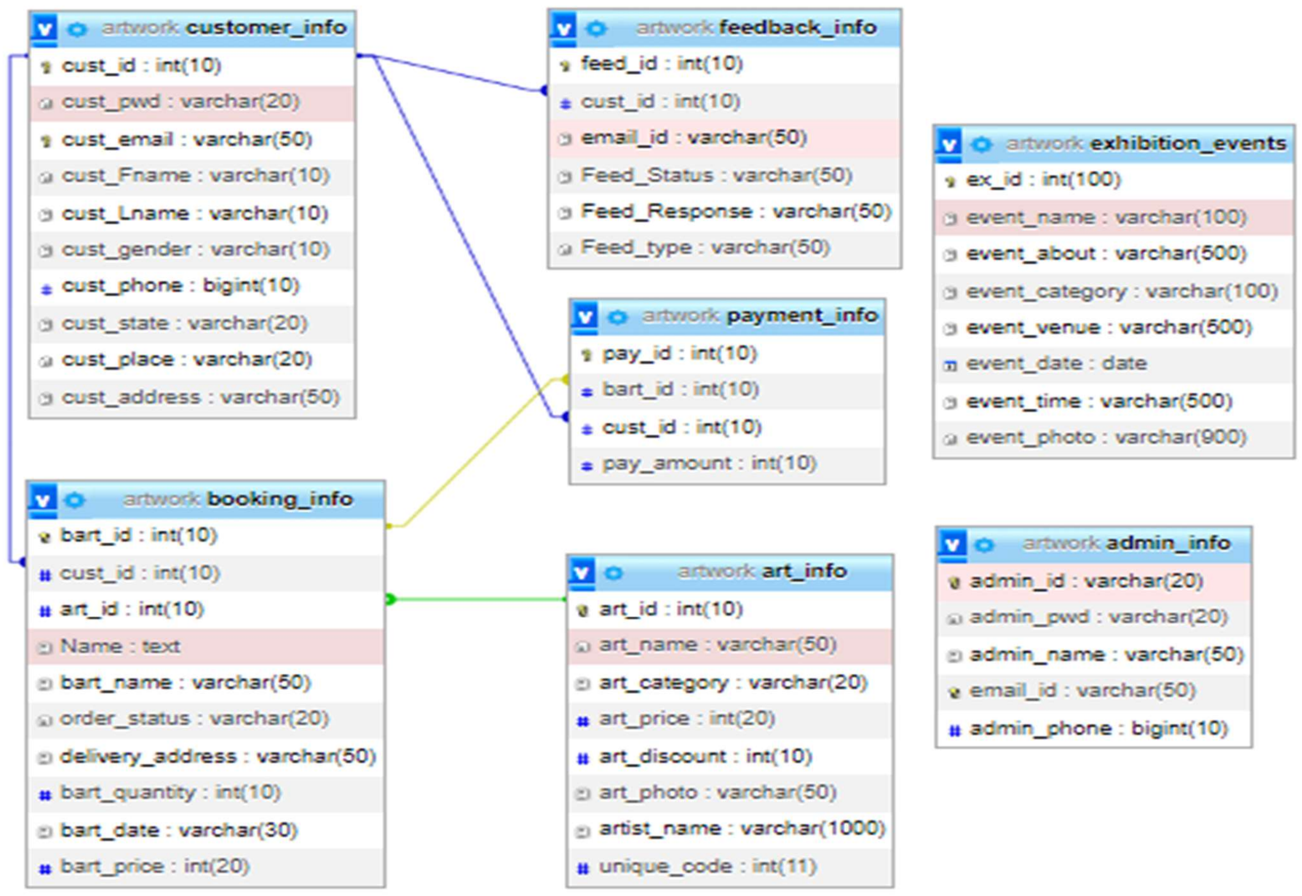


Figure 3.1 Class Diagram

Chapter 4
IMPLEMENTATION

4.1 CODE SNIPPETS

CUSTOMER REGISTER

```
<section id="One" class="wrapper style3"  style="margin-top: -2%; ">
<div class="inner">
                    <header class="align-center">
                                        <p>Customer Registration</p>
                                        <h2>Customer Registration</h2>
                    </header></div>
                    <section id="two" class="wrapper style2">
                              <div class="inner">
                                        <div class="box">
                                                  <div class="content">
</section><form action="" method="post" name="f1">
          <div class="form-group">
                    <label for="first">First Name:</label>
                    <input type="text" class="form-control" id="first" name="first"
placeholder="Enter First Name" required>
          </div>
          <div class="form-group">
                    <label for="last">Last Name:</label>
                    <input type="text" class="form-control" id="last" name="last"
placeholder="Enter Last Name" required>
          </div>
          <div class="form-group">
                    <label for="email">Email:</label>
                    <input type="email" class="form-control" id="email" name="mail"
placeholder="Enter email" required>
          </div>
          <div class="form-group">
                    <label for="pwd">Password:</label>
                    <input type="password" class="form-control" id="pwd" name="pass"
placeholder="Enter password" required>
          </div>
          <div class="form-group">
                    <label for="sel1">Sex:</label>
                    <select class="form-control" id="sel1" name="sex" required>
                              <option value="" hidden>------Select------</option>
                              <option value="Male">Male</option>
                            <option value="Female">Female</option> </select></div>
          <div class="form-group">
                    <label for="mob">Mobile Number:</label>
                    <input type="text" class="form-control" id="mob" name="num"
placeholder="Enter Mobile Number" required pattern="\d*" oninput="javascript: if
(this.value.length > this.maxLength) this.value = this.value.slice(0, this.maxLength);"
    type = "number"
    maxlength = "10">
          </div>
          <div class="form-group">
                    <label for="add">Complete Address:</label>
                    <input type="text" class="form-control" id="address" name="add"
placeholder="Enter Address" required>
          </div>
          <div class="form-group">
                    <label for="st">State:</label>
                    <input type="text" class="form-control" id="st" name="stat"
placeholder="Enter State" required>
          </div>
          <div class="form-group">
                    <label for="pl">Country:</label>
                    <input type="text" class="form-control" id="pl" name="place"
placeholder="Enter Country" required>
          </div>
      <b/>
      <b/>
<button type="submit" class="button special" name="sub" >Register</button>
</form>
</div></div></div></section>
      <!--End this is form for customer registration-->
<?php
if ( isset($_POST['sub'])){
$fn=$_POST['first'];
$ln=$_POST['last'];
$mail=$_POST['mail'];
$pass=$_POST['pass'];
$sex=$_POST['sex'];
$num=$_POST['num'];
$add=$_POST['add'];
$state=$_POST['stat'];
$place=$_POST['place'];
$sql = "INSERT INTO customer_info (cust_pwd, cust_email, cust_Fname, cust_Lname,
cust_sex, cust_phone, cust_state, cust_place, cust_address) VALUES ('$pass', '$mail', '$fn',
'$ln', '$sex', '$num', '$state', '$place', '$add')"; //this is query for registration
if (mysqli_query($con, $sql))
            {
  $customerid=mysqli_insert_id($con);
          $sql1= "select * from customer_info where cust_email='".$mail."' and
cust_pwd='".$pass."'"; //This  is query for login.
  $result = mysqli_query($con, $sql1);
  if (mysqli_num_rows($result) > 0) {
    // output data of each row
    if($row = mysqli_fetch_assoc($result))
{
$_SESSION["uid"]=$row["cust_id"];
$_SESSION["name"]=$row["cust_Fname"];
}
          echo "<script>window.location.assign('customer/')</script>";
}
}
  else {
   echo "<script>alert('This Email Id has used. Please use other Email ID');
window.location.assign('index.php?page=customer_register')</script>";
}
}
mysqli_close($con);
  ?>
```

ADMIN LOGIN PAGE

```
<!--this is form for admin Login-->
<section id="One" class="wrapper style3"  style=" margin-top: -2%; ">
                              <div class="inner">
                                      <header class="align-center">
                                              <p>Admin Login</p>
                                              <h2>Admin Login</h2>
                                              </header>
                              </div> </section>
<section id="two" style="margin-top: -5%;" >
                              <div class="inner">
                                      <div class="box">
                                              <div class="content">
<section>
<div>
<div class="container divform" style="margin-top: 3em;">
  <h2><span class="glyphicon glyphicon-log-in"></span> Admin Login</h2>
  <form action="index.php?page=admin_login" method="post" name="f1">
<div class="form-group"> <label for="id">Admin Id / Username:</label>
```

```
<input type="text" class="form-control" id="id" name="id" placeholder="Enter Id">
            </div>
            <div class="form-group">
                      <label for="pwd">Password:</label>
                      <input   type="password"   class="form-control"   id="pwd"   name="pass"
placeholder="Enter password">
            </div>
            <br>
            <button type="submit" class="button special" name="sub">Login</button>
  </form>
  </div></div></div></div></section>
<!--End this is form for admin Login-->
<?php
if ( isset( $_POST[ 'sub' ] ) ) {
  $id = $_POST[ 'id' ];
  $pas = $_POST[ 'pass' ];
  $sql= "select * from admin_info where admin_id='".$id."' and admin_pwd='".$pas."'"; //This
is query for login.
  $result = mysqli_query($con, $sql);
  if (mysqli_num_rows($result) > 0) {
   // output data of each row
   $row = mysqli_fetch_assoc($result);
 $_SESSION["admin"]=$row["admin_name"];
$_SESSION["adminid"]=$row["admin_id"];
 header('Location:admin/');
} else {
    echo  "<script>  window.alert('Your  input  is  invalid!  Please  Enter  the  Correct  Id  &
Password!'); window.location.assign('admin/') </script>";
}}
mysqli_close($con);
?>
```

BOOKING BY CUSTOMER

```
<?php include("header/header.php");?>
<?php
if($_SESSION["uid"] == "" || $_SESSION["uid"]==NULL)
{
header('Location:../customer/');
}
else{
  $cid = $_SESSION["uid"];
  $cn=$_SESSION["name"];
}?>
```

```
<?php
$aid=$_GET['id'];
$run_art=mysqli_query($con,"select * from art_info where art_id=$aid"); //
$art_result=mysqli_fetch_array($run_art);
$_SESSION['aart'] = $aid;?>
<section>
  <!--this is form for booking-->
<div class="back-img back-img1">
<div class="container divform">
  <h2>Booking</h2>
  <form action="" method="post" name="f1">
        <div class="form-group">
              <label for="a1">Customer Id:</label>
              <input type="number" class="form-control" id="a1" name="custid"
placeholder="Enter Id" readonly value="<?php echo $cid;?>" >
        </div>
        <div class="form-group">
              <label for="a2">Art_Id:</label>
              <input type="number" class="form-control" id="a2" name="arid" readonly
value="<?php echo $aid;?>">
        </div>
        <div class="form-group">
              <label for="a3"> Name:</label>
              <input type="text" class="form-control" id="a3" name="cname" readonly
required value="<?php echo $cn;?>">
        </div>
        <div class="form-group">
              <label for="a4"> Art_Name:</label>
              <input type="text" class="form-control" id="a4" name="aname" readonly
value="<?php echo $art_result[1];?>">
        </div>
        <div class="form-group">
              <label for="a5">Order_Status:</label>
              <input type="text" class="form-control" id="a5" name="os" readonly
value="Available"></div>
        <div class="form-group">
              <label for="a6">Delivery_Address:</label>
              <input type="text" class="form-control" id="a6" name="da"
placeholder="Enter Address" required>
        </div>
        <div class="form-group">
              <label for="a7">Quantity:</label>
              <input type="number" class="form-control" id="a7" name="quan"
placeholder="Enter Quantity" required></div>
```

```
<div class="form-group">
                    <label for="a8">Price(Rs.):</label>
                    <input type="number" class="form-control" id="a8" name="price" readonly
value="<?php echo $art_result[3];?>">
            </div>
            <button type="submit" class="btn btn-primary" name="sub">Submit</button>
  </form>
</div></div><?php include("../footer/footer.php"); ?>
</section>
<!--End this is form for booking-->
<?php
if ( isset( $_POST[ 'sub' ] ) ){
  $a11=$_POST['custid'];
  $a12=$_POST['arid'];
  $a13=$_POST['cname'];
  $a14=$_POST['aname'];
  $a15=$_POST['os'];
  $a16=$_POST['da'];
  $a17=$_POST['quan'];
  $a18=($a17 * $_POST['price']);
  $sql = "INSERT INTO booking_info (cust_id, art_id, Name, bart_name, order_status,
delivery_address, bart_quantity, bart_date,bart_price) VALUES ('$a11', '$a12', '$a13', '$a14',
'$a15','$a16', '$a17', NOW(), '$a18')"; //this is query for booking
  if (mysqli_query($con, $sql)) {
            $current_baid = mysqli_insert_id($con);
            $_SESSION['baid'] =$current_baid;
 echo "<script>alert('Your Booking has been successfully! Proceed To Payment..');
  window.location.assign('paymentmode.php')</script>";
  } else {
   echo "Error: " . $sql . "<br>" . mysqli_error($con);
} }
mysqli_close($con);
?>
```

PAYMENT DETAILS OF CUSTOMER

```
<div class="back-img back-img1">
  <?php include("header/header.php");?>
  <div class="container-fluid" style="margin-top: 31px;">
    <h2>View Payment Details</h2>
  </div>
  <div class="container-fluid" style="background-color: white”;>
    <div class="row" style="margin-left: 10px;">
    <?php $a=$_GET['id']; $sql="select * from payment_info where cust_id=$a";// display the
payment details.
```

```
$run=mysqli_query($con,$sql);
        echo "<table class='table table-bordered'>
        <tr>
        <th>Payment Id</th>
        <th>Booking Art Id</mmth>
        <th> Customer Id</th>
        <th>Amount</th>
        </tr>";
        while($result=mysqli_fetch_array($run))
        {
            echo "<tr>
            <td>$result[0]</td>
        <td>$result[1]</td>
        <td>$result[2]</td>
        <td>$result[3]</td>
            </tr>";
        }
        echo "</table>";
        mysqli_close($con);
        ?>
  </div></div> </div>
  <?php include("../footer/footer.php");?>
```

PAYMENT MODE USED

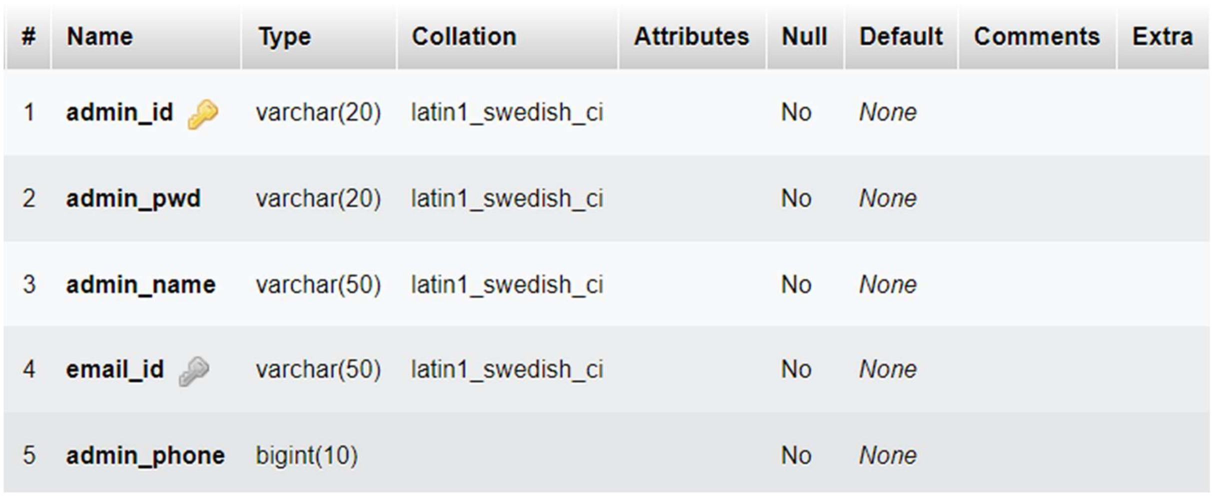
```
<?php include("header/header.php"); ?>
<?php
  if($_SESSION["uid"] == ""  ||  $_SESSION["uid"]==NULL)
  {
  header('Location:index.php?page=customer_login');
  }
  else{
        $cbid = $_SESSION['baid'];
        $run_bart=mysqli_query($con,"select * from booking_info where bart_id=$cbid");
        $art_result1=mysqli_fetch_array($run_bart);
  }
  ?>
<!--this is form for payment-->
<div class="back-img back-img1">
  <div class="container divform">
    <h2>Payment Information</h2>
    <form action="" method="post" name="f1">
      <div class="form-group">
        <label for="a11">Booking Art Id:</label>
```

```
<input type="number" class="form-control" id="a11" name="bart" placeholder="Enter Id"
readonly value="<?php echo $art_result1[0];?>" >
        </div>
        <div class="form-group">
          <label for="a1">Customer Id:</label>
          <input type="number" class="form-control" id="a1" name="custid1" placeholder="Enter
Id" readonly value="<?php echo $art_result1[1];?>" >
        </div>
        <div class="form-group">
          <label for="a2">Art_Id:</label>
          <input type="number" class="form-control" id="a2" name="arid" readonly
value="<?php echo $art_result1[2];?>">
        </div>
        <div class="form-group">
          <label for="a3"> Name:</label>
          <input type="text" class="form-control" id="a3" name="cname" readonly required
value="<?php echo $art_result1[3];?>">
        </div>
        <div class="form-group">
          <label for="a4"> Art_Name:</label>
          <input type="text" class="form-control" id="a4" name="aname" readonly  value="<?php
echo $art_result1[4];?>">
        </div>
        <div class="form-group">
          <label for="a5">Order_Status:</label>
          <input type="text" class="form-control" id="a5" name="os" readonly value="<?php
echo $art_result1[5];?>">
        </div>
        <div class="form-group">
          <label for="a6">Delivery_Address:</label>
          <input type="text" class="form-control" id="a6" name="da" readonly
placeholder="Enter Address" required value="<?php echo $art_result1[6];?>">
        </div>
        <div class="form-group">
          <label for="a7">Quantity:</label>
          <input type="number" class="form-control" id="a7" name="quan" readonly
placeholder="Enter Quantity" required value="<?php echo $art_result1[7];?>">
        </div>
        <div class="form-group">
          <label for="a8">Date:</label>
          <input type="text" class="form-control" id="a8" name="date" readonly value="<?php
echo $art_result1[8];?>">
        </div>
        <div class="form-group"> <label for="a9">Price(Rs.):</label>
```

```
<input type="text" class="form-control" id="a9" name="price" readonly value="<?php echo
$art_result1[9];?>">
      </div>
      <div class="form-group">
        <label for="a10">Mode:</label>
        <select class="form-control" name="sel" id="a10" required>
          <option value="">---Select--</option>
          <option value="Cash On Delivery">Cash On Delivery</option>
        </select>
      </div>
      <button type="submit" class="btn btn-primary" name="sub">Proceed To
Payment</button>
    </form>
  </div>
</div>
<?php include("../footer/footer.php"); ?>
<!--End this is form for payment-->
<?php
  if ( isset( $_POST[ 'sub' ] ) ){
        $a=$_POST['bart'];
        $b=$_POST['custid1'];
        $c=$_POST['price'];
  $run_pay=mysqli_query($con, "insert into payment_info (bart_id, cust_id, pay_amount)
values('$a', '$b', '$c')"); // this is query for payment to insert database.
        if($run_pay){
                echo "<script> window.alert('Congratulation! You will get your product at
home in working day');
        window.location.assign('../customer/')
  </script>";
        }
        else{
                echo "<script> window.alert('Please try again later.');
        window.location.assign('../customer/')
  </script>";
        }
  }
  mysqli_close($con);
        ?>
```

4.2 TABLE DESCRIPTION

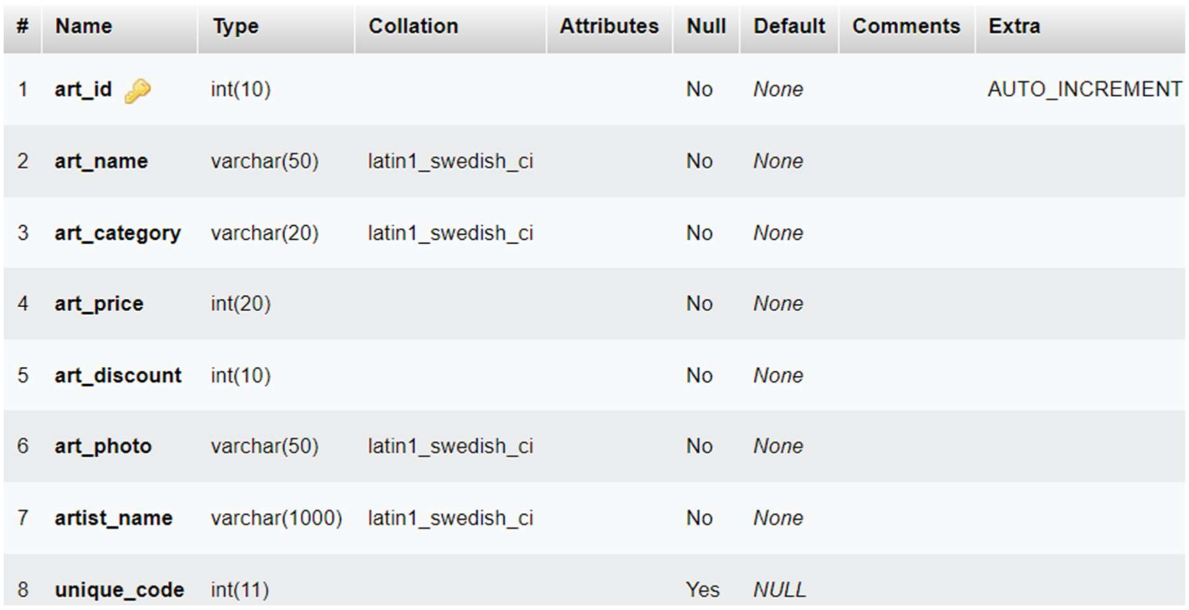
Admin Table: Figure 4.1 provides description about the details of admin i.e admin id, admin password, admin name, email id, admin phone.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**admin_id**	varchar(20)	latin1_swedish_ci		No	*None*		
2	**admin_pwd**	varchar(20)	latin1_swedish_ci		No	*None*		
3	**admin_name**	varchar(50)	latin1_swedish_ci		No	*None*		
4	**email_id**	varchar(50)	latin1_swedish_ci		No	*None*		
5	**admin_phone**	bigint(10)			No	*None*		

Figure 4.1 admin table

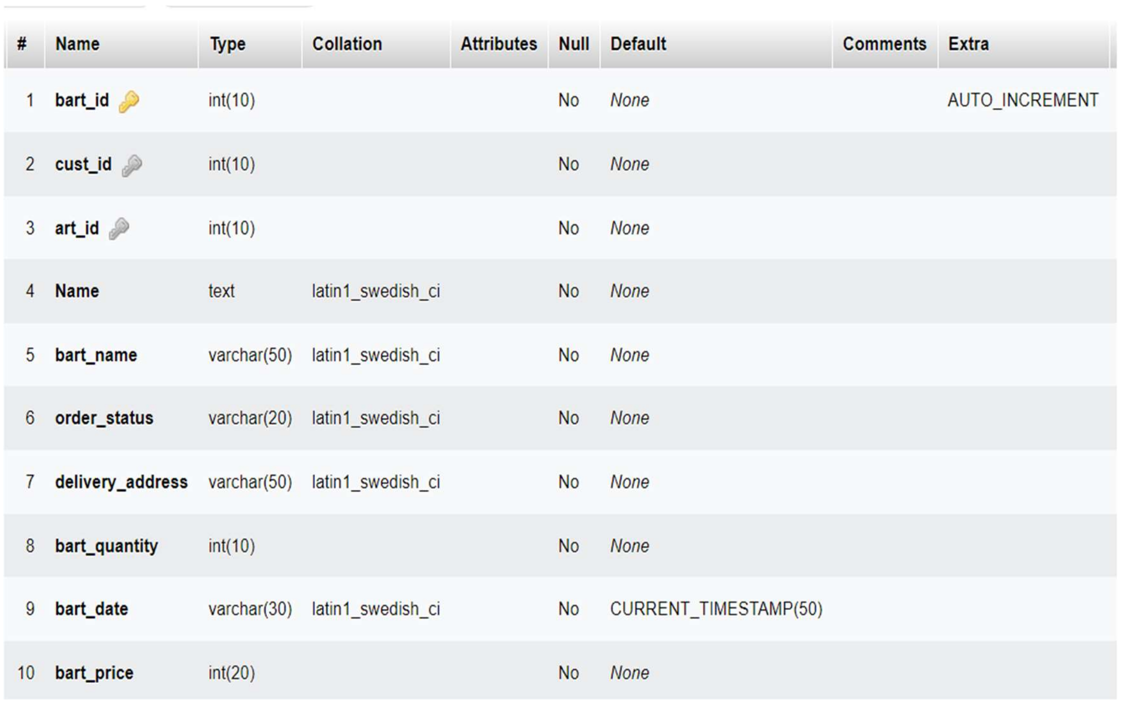
Art_info Table: Figure 4.2 provides description about the details of artworks such as art id which is a primary key, art name, art category, art price and discount with artist name. ‘art_photo’ is an attribute where we can upload the artwork.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**art_id**	int(10)			No	*None*		AUTO_INCREMENT
2	**art_name**	varchar(50)	latin1_swedish_ci		No	*None*		
3	**art_category**	varchar(20)	latin1_swedish_ci		No	*None*		
4	**art_price**	int(20)			No	*None*		
5	**art_discount**	int(10)			No	*None*		
6	**art_photo**	varchar(50)	latin1_swedish_ci		No	*None*		
7	**artist_name**	varchar(1000)	latin1_swedish_ci		No	*None*		
8	**unique_code**	int(11)			Yes	*NULL*		

Figure 4.2 Art_info table

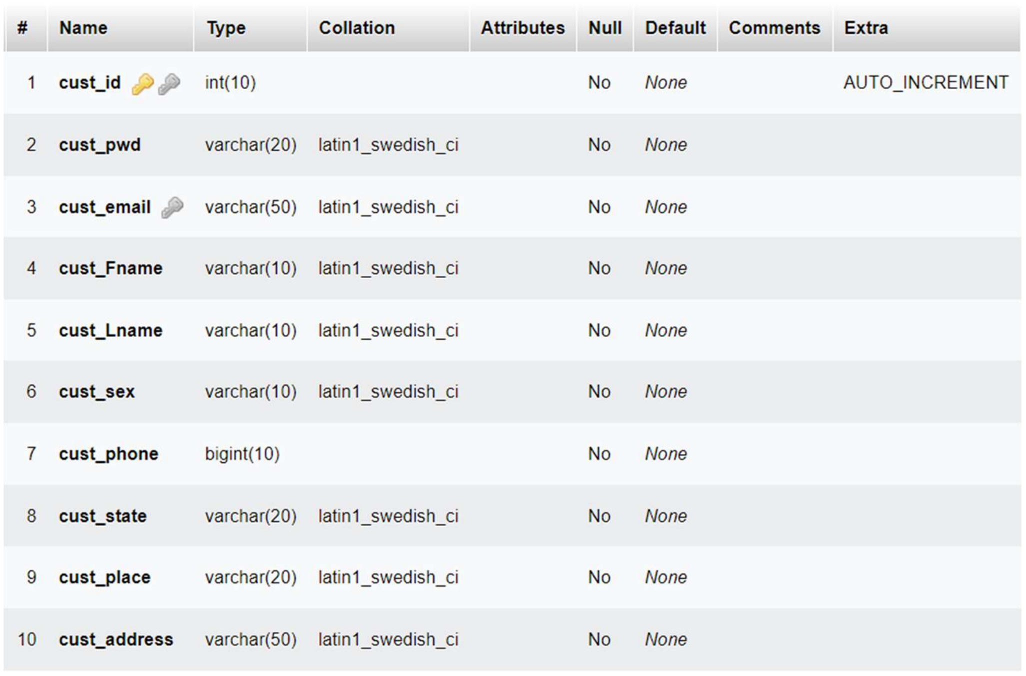
Booking_info Table: Figure 4.3 shows the decription of attributes about booking details of the artwork made by the customer



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	bart_id	int(10)			No	*None*		AUTO_INCREMENT
2	cust_id	int(10)			No	*None*		
3	art_id	int(10)			No	*None*		
4	Name	text	latin1_swedish_ci		No	*None*		
5	bart_name	varchar(50)	latin1_swedish_ci		No	*None*		
6	order_status	varchar(20)	latin1_swedish_ci		No	*None*		
7	delivery_address	varchar(50)	latin1_swedish_ci		No	*None*		
8	bart_quantity	int(10)			No	*None*		
9	bart_date	varchar(30)	latin1_swedish_ci		No	CURRENT_TIMESTAMP(50)		
10	bart_price	int(20)			No	*None*		

Figure 4.3 Booking_info table

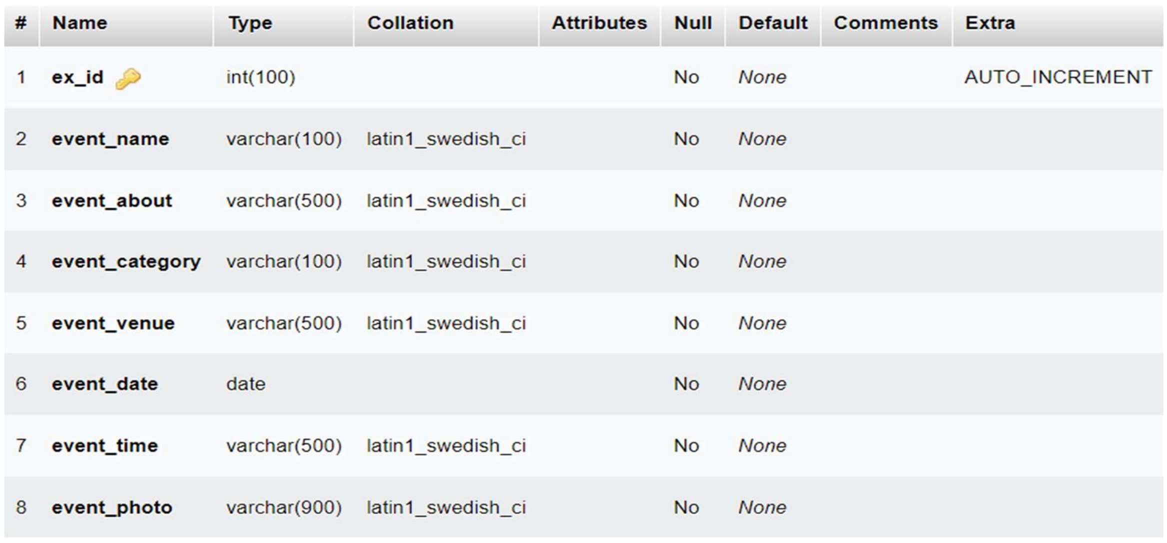
Customer_info Table: Figure 4.4 shows description about the attributes that tells about the customer details.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	cust_id	int(10)			No	*None*		AUTO_INCREMENT
2	cust_pwd	varchar(20)	latin1_swedish_ci		No	*None*		
3	cust_email	varchar(50)	latin1_swedish_ci		No	*None*		
4	cust_Fname	varchar(10)	latin1_swedish_ci		No	*None*		
5	cust_Lname	varchar(10)	latin1_swedish_ci		No	*None*		
6	cust_sex	varchar(10)	latin1_swedish_ci		No	*None*		
7	cust_phone	bigint(10)			No	*None*		
8	cust_state	varchar(20)	latin1_swedish_ci		No	*None*		
9	cust_place	varchar(20)	latin1_swedish_ci		No	*None*		
10	cust_address	varchar(50)	latin1_swedish_ci		No	*None*		

Figure 4.4 Customer_info table

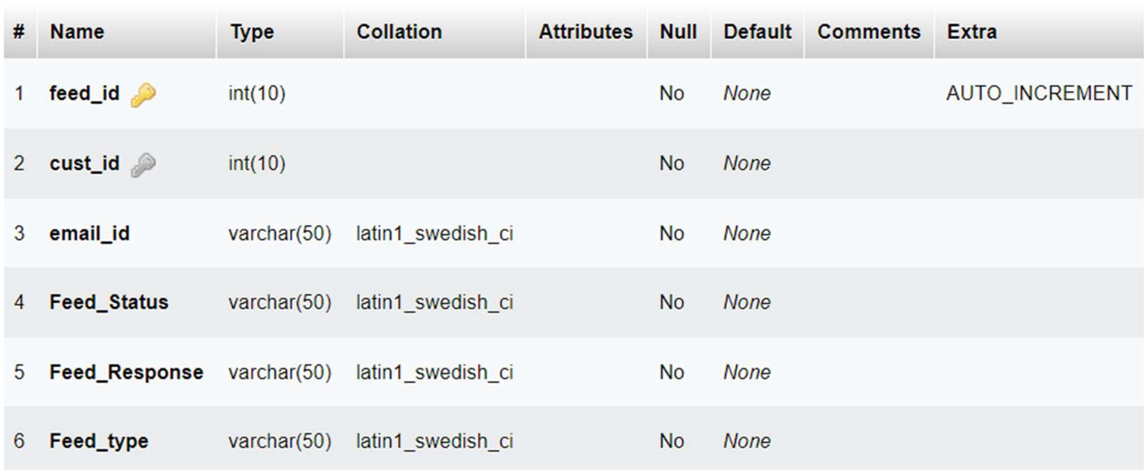
Exhibition_events Table: Figure 4.5 shows description of attributes about the details of upcoming exhibition events.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**ex_id**	int(100)			No	*None*		AUTO_INCREMENT
2	**event_name**	varchar(100)	latin1_swedish_ci		No	*None*		
3	**event_about**	varchar(500)	latin1_swedish_ci		No	*None*		
4	**event_category**	varchar(100)	latin1_swedish_ci		No	*None*		
5	**event_venue**	varchar(500)	latin1_swedish_ci		No	*None*		
6	**event_date**	date			No	*None*		
7	**event_time**	varchar(500)	latin1_swedish_ci		No	*None*		
8	**event_photo**	varchar(900)	latin1_swedish_ci		No	*None*		

Figure 4.5 Exhibition_events table

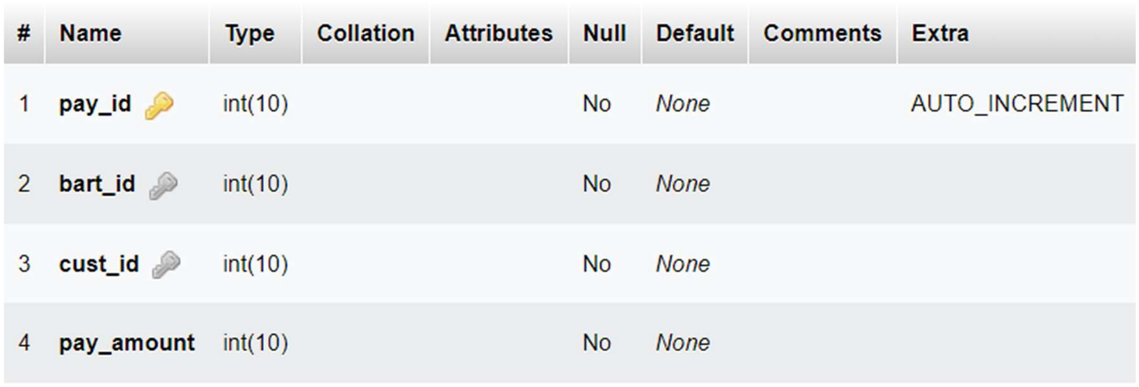
Feedback_info Table: Figure 4.6 shows the description of attributes about the feedback details.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**feed_id**	int(10)			No	*None*		AUTO_INCREMENT
2	**cust_id**	int(10)			No	*None*		
3	**email_id**	varchar(50)	latin1_swedish_ci		No	*None*		
4	**Feed_Status**	varchar(50)	latin1_swedish_ci		No	*None*		
5	**Feed_Response**	varchar(50)	latin1_swedish_ci		No	*None*		
6	**Feed_type**	varchar(50)	latin1_swedish_ci		No	*None*		

Figure 4.6 Feedback_info table

Payment_info Table: Figure 4.7 shows the description of attributes about the payment details.



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**pay_id**	int(10)			No	*None*		AUTO_INCREMENT
2	**bart_id**	int(10)			No	*None*		
3	**cust_id**	int(10)			No	*None*		
4	**pay_amount**	int(10)			No	*None*		

Figure 4.7 Payment_info table

4.3 CREATION OF TABLES

Table art_info

```
  CREATE TABLE `admin_info` (
 `admin_id` varchar(20) NOT NULL,
 `admin_pwd` varchar(20) NOT NULL,
 `admin_name` varchar(50) NOT NULL,
 `email_id` varchar(50) NOT NULL,
 `admin_phone` bigint(10) NOT NULL
  );
```

Table art_info

```
  CREATE TABLE `art_info` (
 `art_id` int(10) NOT NULL,
 `art_name` varchar(50) NOT NULL,
 `art_category` varchar(20) NOT NULL,
 `art_price` int(20) NOT NULL,
 `art_discount` int(10) NOT NULL,
 `art_photo` varchar(50) NOT NULL,
 `artist_name` varchar(1000) NOT NULL,
 ‘unique_code’ int(4) NOT NULL );
```

Table booking_info

```
  CREATE TABLE `booking_info` (
 `bart_id` int(10) NOT NULL,
 `cust_id` int(10) NOT NULL,
 `art_id` int(10) NOT NULL,
 `Name` text NOT NULL,
 `bart_name` varchar(50) NOT NULL,
 `order_status` varchar(20) NOT NULL,
 `delivery_address` varchar(50) NOT NULL,
 `bart_quantity` int(10) NOT NULL,
 `bart_date` varchar(30) NOT NULL DEFAULT 'CURRENT_TIMESTAMP(50)',
 `bart_price` int(20) NOT NULL );
```

Table customer_info

```
  CREATE TABLE `customer_info` (
 `cust_id` int(10) NOT NULL,
 `cust_pwd` varchar(20) NOT NULL,
 `cust_email` varchar(50) NOT NULL,
 `cust_Fname` varchar(10) NOT NULL,
 `cust_Lname` varchar(10) NOT NULL,
 `cust_sex` varchar(10) NOT NULL,
 `cust_phone` bigint(10) NOT NULL,
 `cust_state` varchar(20) NOT NULL,
 `cust_place` varchar(20) NOT NULL,
 `cust_address` varchar(50) NOT NULL
  );
```

Table exhibition_events

```
  CREATE TABLE `exhibition_events` (
 `ex_id` int(100) NOT NULL,
 `event_name` varchar(100) NOT NULL,
 `event_about` varchar(500) NOT NULL,
 `event_category` varchar(100) NOT NULL,
 `event_venue` varchar(500) NOT NULL,
 `event_date` date NOT NULL,
 `event_time` varchar(500) NOT NULL,
 `event_photo` varchar(900) NOT NULL
  );
```

Table feedback_info

```
  CREATE TABLE `feedback_info` (
 `feed_id` int(10) NOT NULL,
 `cust_id` int(10) NOT NULL,
 `email_id` varchar(50) NOT NULL,
 `Feed_Status` varchar(50) NOT NULL,
 `Feed_Response` varchar(50) NOT NULL,
 `Feed_type` varchar(50) NOT NULL );
```

Table Payment_info

```
CREATE TABLE `payment_info` (
  `pay_id` int(10) NOT NULL,
  `bart_id` int(10) NOT NULL,
  `cust_id` int(10) NOT NULL,
  `pay_amount` int(10) NOT NULL
)
```

4.4 INSERTION OF TUPLES

1. Insertion into ‘admin_info’ table

```
 INSERT INTO `admin_info’
 VALUES ('admin', 'admin', 'Sanjana S', ‘sanj18218@gmail.com', 7338590055);
```

2. Insertion into ‘art_info’ table

```
INSERT INTO `art_info` VALUES (1, 'Black sand beach', 'Painting', 90000, 10000, ' ..\img\painting1.jpg', 'Artist: Shina Choi');
INSERT INTO `art_info` VALUES (2, 'The Indian Door', 'Painting', 15000, 17000, ' ..\img\painting2.jpg', 'Artist:  Ram Onkar');
INSERT INTO `art_info` VALUES (3, 'Ganesha Pencil Carving', 'Pencil Carving', 6999, 7500, '..\img\pc6.png', 'Artist:Bhuvan Patel');
INSERT INTO `art_info` VALUES (4, 'Awakening Light', 'Mandala', 55500,57999, '..\img\mandala7.jpg', 'Artist: Raghav Joshi');
INSERT INTO `art_info` VALUES (5, 'The Vintage Car', 'Photography', 2000, 3500, '..\img\p3.jpg', 'Artist: Paul Williams');
INSERT INTO `art_info` VALUES (6, 'Zen Meditation Mandala', 'Mandala', 30000, 33000, '..\img\mandala4.jpg', 'Artist: Preeti Kashyap'),
INSERT INTO `art_info` VALUES (7, 'The Blue Eyes', 'Fine Art', 39000, 40000, '..\img\fineart5.jpg', 'Artist: Krishna Sharma');
INSERT INTO `art_info` VALUES (8, 'Jaymahal Palace ', 'Photography', 3299, 3500, '..\img\p2.jpg', 'Artist: Prateek J T');
INSERT INTO `art_info` VALUES (9, 'The City of New York', 'Painting', 39999, 41999, '..\img\painting4.jpg', 'Artist: Nischith Gowda');
```

3. Insertion into ‘booking_info’ table

```
INSERT INTO `booking_info`
VALUES (16, 9, 14, 'Yashasvi', 'Seven Wonders', 'Available', 'bangalore', 2,'2023-01-19
16:51:59', 59998);
```

```
INSERT INTO `booking_info`
VALUES (17, 9, 2, 'Yashasvi', 'The Indian Door', 'Available', 'bangalore', 1, '2023-01-19
19:24:06', 15000);
```

```
INSERT INTO `booking_info`
VALUES (18, 10, 1, 'Sneha', 'Black sand beach', 'Available', 'bangalore', 1, '2023-01-19
22:04:29', 9000);
```

4. Insertion into ‘customer_info’ table

```
INSERT INTO `customer_info
VALUES (9, 'yash', '1js20cs187@gmail.com', 'Yashasvi', 'GM', 'Female', 8892235305,
'karnataka', 'india', 'bangalore');
```

```
INSERT INTO `customer_info
VALUES (10, 'sneha', '1js20cs161@gmail.com', 'Sneha', 'M', 'Female', 1234567890, 'karnataka',
'india', 'bangalore');
```

5. Insertion into ‘exhibition_events’ table

```
INSERT INTO `exhibition_events`
VALUES (100, 'India Art Festival - 2023', 'Group Event', 'Mandala , Folk', 'New Delhi', '2021-
02-16', '11:00 AM', '..\img\eventimages\iaf.png');
```

```
INSERT INTO `exhibition_events`
VALUES (101, 'The Museum of Art and Photography', 'Group Event', 'Photography , Pencil
Carving', 'M G Road, Bangalore', '2023-03-05', '12:00 PM', '..\img\eventimages\map.png');
```

6. Insertion into ‘payment_info’ table

```
INSERT INTO `payment_info` VALUES (15, 15, 9, 2000);
INSERT INTO `payment_info` VALUES (16, 16, 9, 59998),
INSERT INTO `payment_info` VALUES (17, 18, 10, 9000);
```

4.5 COMMANDS

```
Indexes for table `admin_info`
ALTER TABLE `admin_info`
ADD PRIMARY KEY (`admin_id`),
ADD UNIQUE KEY `Login_id` (`email_id`);

Indexes for table `art_info`
ALTER TABLE `art_info`
ADD PRIMARY KEY (`art_id`);

Indexes for table `booking_info`
ALTER TABLE `booking_info`
ADD PRIMARY KEY (`bart_id`),
ADD KEY `cust_id` (`cust_id`),
ADD KEY `art_id` (`art_id`),
ADD KEY `cust_id_2` (`cust_id`);

Indexes for table `customer_info`
ALTER TABLE `customer_info`
ADD PRIMARY KEY (`cust_id`),
ADD UNIQUE KEY `cust_email` (`cust_email`),
ADD UNIQUE KEY `cust_id_4` (`cust_id`),
ADD KEY `cust_id` (`cust_id`),
ADD KEY `cust_id_2` (`cust_id`),
ADD KEY `cust_id_3` (`cust_id`);

Indexes for table `exhibition_events`
ALTER TABLE `exhibition_events`
ADD PRIMARY KEY (`ex_id`);

Indexes for table `feedback_info`
ALTER TABLE `feedback_info`
ADD PRIMARY KEY (`feed_id`),
ADD KEY `cust_id` (`cust_id`);

AUTO_INCREMENT for table `payment_info`
```

```
ALTER TABLE `payment_info`
MODIFY `pay_id` int(10) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=15;
```

Constraints for table `booking_info`

```
ALTER TABLE `booking_info`
ADD CONSTRAINT `booking_info_ibfk_1` FOREIGN KEY (`art_id`) REFERENCES
`art_info` (`art_id`) ON DELETE CASCADE ON UPDATE CASCADE,
 ADD CONSTRAINT `booking_info_ibfk_2`FOREIGN KEY(`cust_id`)REFERENCES
`customer_info` (`cust_id`) ON DELETE CASCADE ON UPDATE CASCADE;
```

Constraints for table `feedback_info`

```
ALTER TABLE `feedback_info`
ADD CONSTRAINT `feedback_info_ibfk_1` FOREIGN KEY (`cust_id`) REFERENCES
`customer_info` (`cust_id`) ON DELETE CASCADE ON UPDATE CASCADE;
```

Constraints for table `payment_info`

```
ALTER TABLE `payment_info`
ADD CONSTRAINT `payment_info_ibfk_1` FOREIGN KEY (`cust_id`) REFERENCES
`customer_info` (`cust_id`) ON DELETE CASCADE ON UPDATE CASCADE,
ADD CONSTRAINT `payment_info_ibfk_2` FOREIGN KEY (`bart_id`) REFERENCES
`booking_info` (`bart_id`) ON DELETE CASCADE ON UPDATE CASCADE;
 COMMIT;
```

4.6 QUERIES

```
$sql1= "select * from customer_info where cust_email='".$mail."' and cust_pwd='".$pass."'";
```

This is a query for customer register login.

```
$sql= "select * from admin_info where admin_id='".$id."' and admin_pwd='".$pas."'";
```

This is a query for admin login.

```
$run=mysqli_query($con, "select * from art_info where art_category='$sr'");
$run=mysqli_query($con, "select * from art_info");
//This is a query to display the products.
```

```
$sql="select * from payment_info where cust_id=$a";  //This is a query to display the payment details.
```

```
$run_bart=mysqli_query($con,"select * from booking_info where bart_id=$cbid");

$run=mysqli_query($con, "select * from  exhibition_events");
// all display Exhibition and Events Details

$sql = "select * from booking_info where cust_id=$r";
// display all details of booking

$up="delete from booking_info where bart_id=$a";
 // this is query for cancelling of booking

$run_art=mysqli_query($con,"select * from art_info where art_id=$aid");
 // this is a query to do booking of the product by customer.

$run_cust=mysqli_query($con,"select * from customer_info where cust_id=$cid");
//this is a query for the customer to give feedback

$sql = "select * from customer_info"; // this  is query for fetching all customer

$sql = "select * from customer_info"; // this  is query for fetching all customer  details. details.

$sql = "select * from booking_info"; // this is query for displaying  all booking details

$s = mysqli_query( $con, "select * from art_info where art_id=$r" );
// this is query for fetching detail of product who is updating.

$up = "update exhibition_events set   event_name='$a',  event_about='$b',    event_category='$c',
event_venue='$d',              event_date='$e',event_time='$f',             event_photo='$target_path',
event_photo_disp='$target_path_disp' where ex_id='$r'";
// this query for updating Exhibition & Events Details1

$s = mysqli_query( $con, "select * from exhibition_events where ex_id=$r" );
// this is query for fetching detail of product who is updating.

$s=mysqli_query($con, "select * from feedback_info where feed_id=$r");
// this is query for displaying feedback who is modifying.

$sql="select * from feedback_info"; // this query for displaying all feedback

$sql="select * from exhibition_events"; // this is query for displaying all Exhibition & Events
Details
```

$s=mysqli_query($con, "select * from art_info where art_id=$r"); // this query for displaying art record who is removing

$up="delete from art_info where art_id='$a'"; // this is query for removing art detail.

$sql="select * from art_info"; // this is query for displaying all product

$s=mysqli_query($con, "select * from feedback_info where feed_id=$r"); // this is query for displaying data who is removing.

$up="delete from feedback_info where feed_id='$a'"; // this is query for deleting feedback record by admin.

$s = mysqli_query($con, "select * from customer_info where cust_id=$r"); // this query for displaying info who is removing.

$up = "delete from customer_info where cust_id='$a'"; // this query for deleting for customer details.

$sql = "select * from customer_info"; // this is query for fetching all customer details.

$s=mysqli_query($con, "select * from exhibition_events where ex_id=$r"); // this query for displaying art record who is removing

4.7 TRIGGERS

```
CREATE TRIGGER ‘insert_art_trigger’ BEFORE INSERT ON ‘art_info’
FOR BEGIN
IF (SELECT unique_code FROM art_info
   WHERE unique_code=NEW.unique_code) THEN
   SIGNAL SQLSTATE '45000'
   SET MESSAGE_TEXT = 'art already exists';
   END IF;
END
```

This is a trigger, named insert_art_trigger. In this, the artworks are are given a unique code by the admin. While adding new artworks, if the unique code of the artwork matches with that of the previous added artworks, then the trigger alerts saying that artwork already exists.

Chapter 5

RESULTS

5.1 FRONT END

5.1.1 HOME PAGE

Figure 5.1 and 5.2 shows the home page of the Art Gallery, which shows the slider of five pictures giving the description about the art gallery website. 'Art Club' is the name of website. It includes two logins for customer and admin. It also shows customer register where new users can register and view the application.

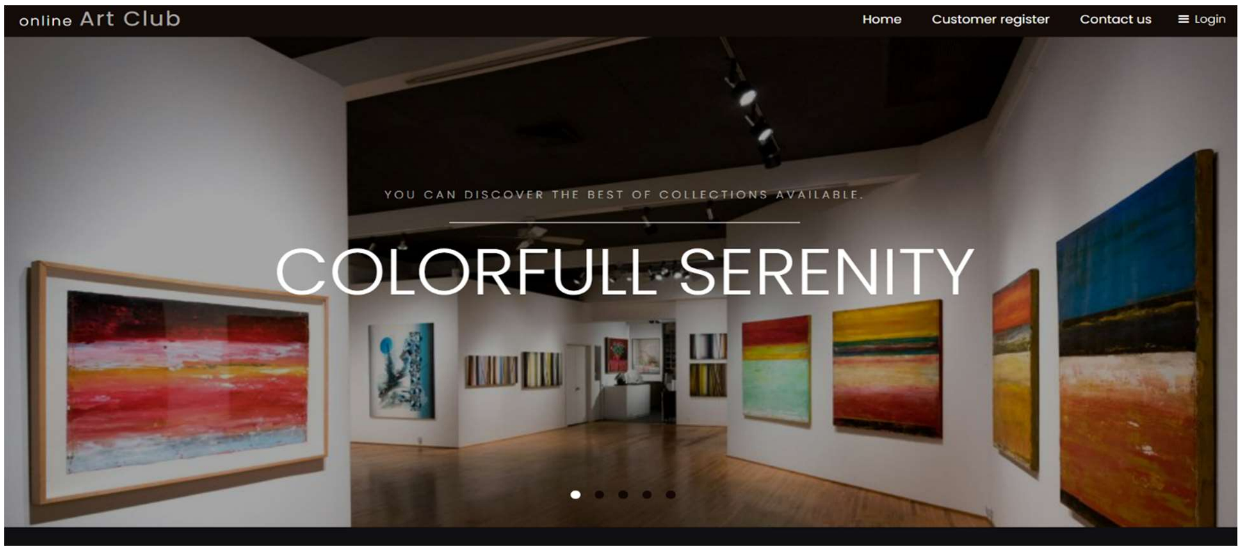


Figure 5.1 Home page

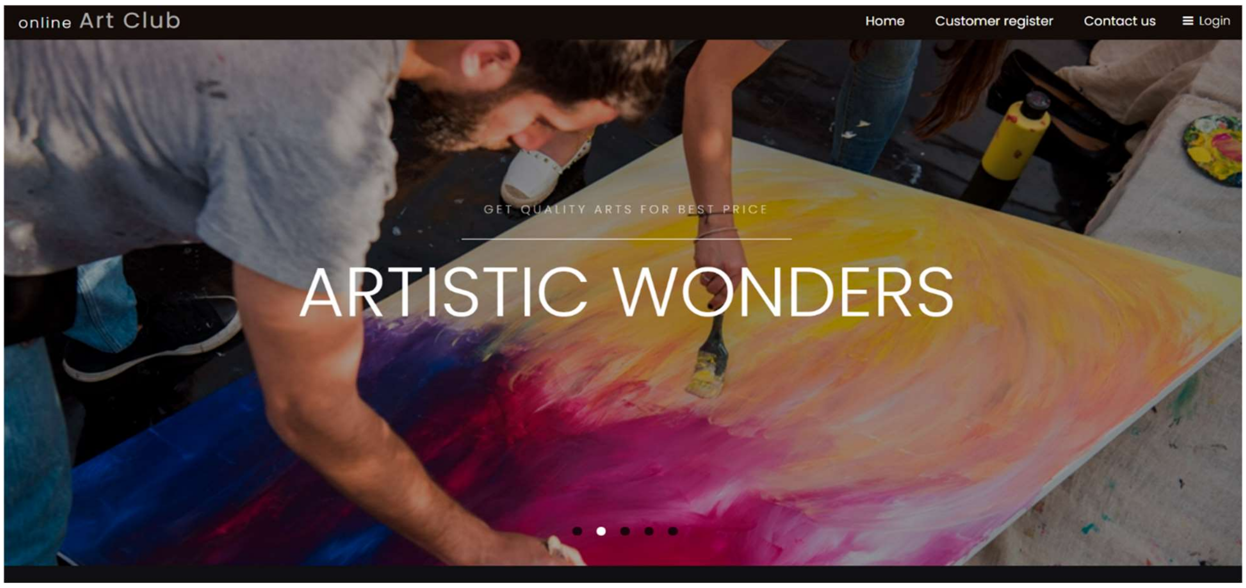
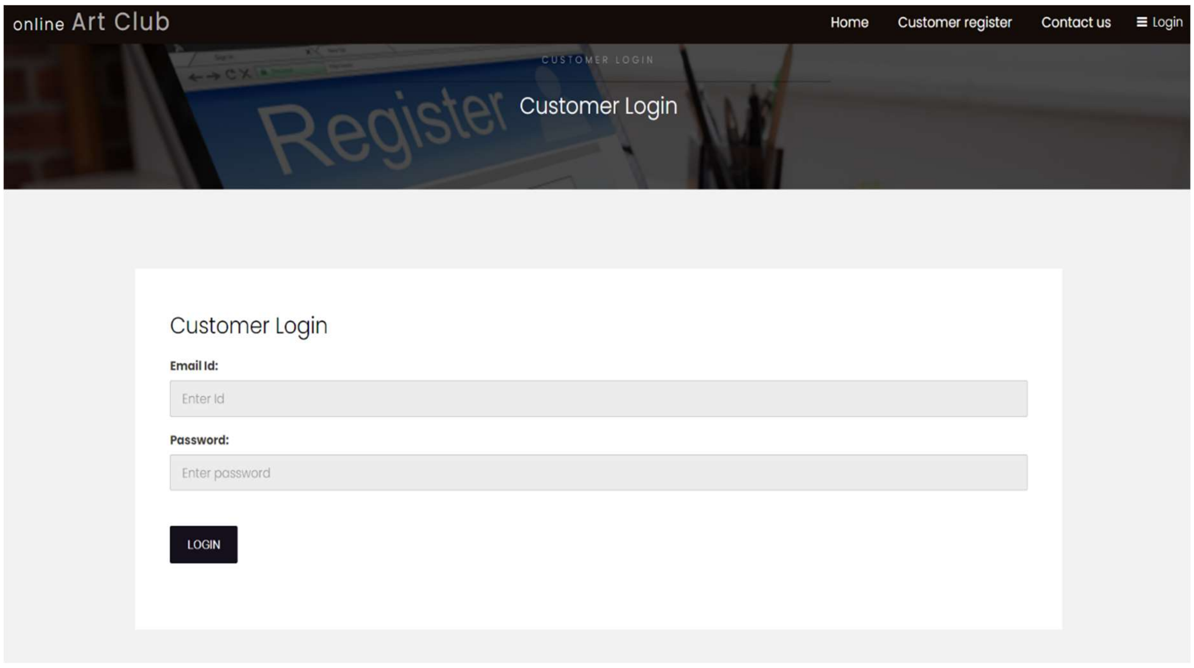


Figure 5.2 Home Page (slider pictures)

5.1.2 CUSTOMER SIDE

Figure 5.3 shows the customer login page. It is only then the customer can view the artworks and can place the order. In customer page, we can navigate through cancel booking page, feedback page, and can also view the upcoming exhibition event's details. Figure 5.4 shows the artworks page.

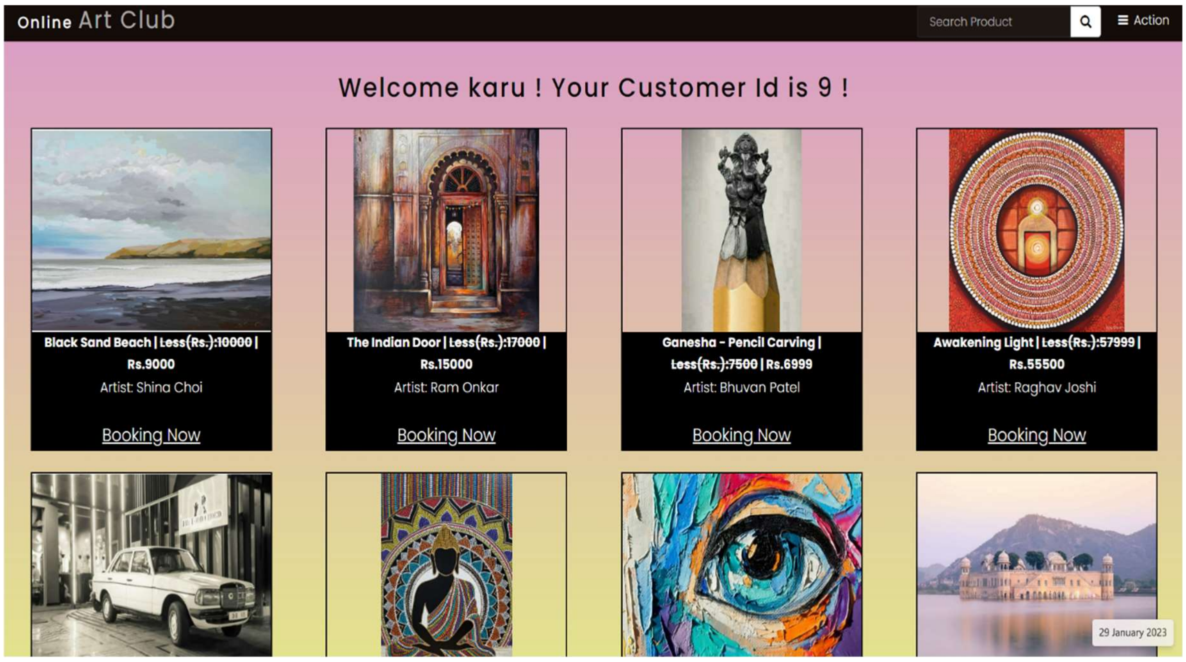


Customer Login

Email Id: Enter Id

Password: Enter password

Figure 5.3 Customer login



Welcome karu ! Your Customer Id is 9 !

Black Sand Beach | Less(Rs.):10000 | Rs.9000
Artist: Shina Choi

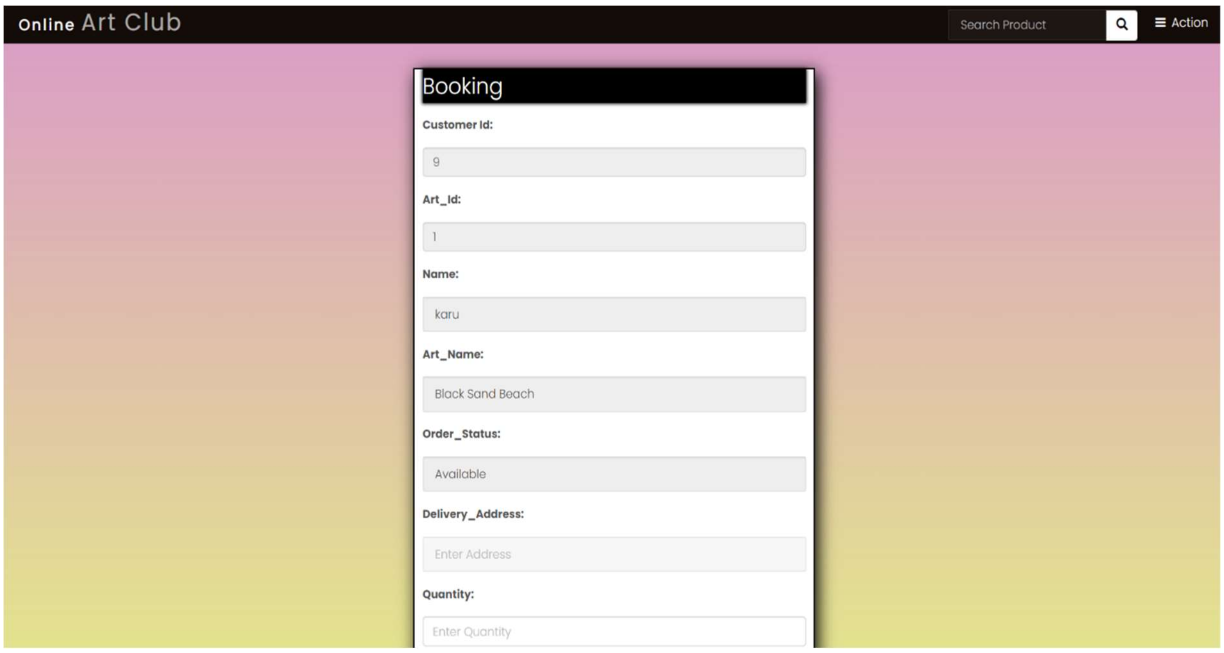
The Indian Door | Less(Rs.):17000 | Rs.15000
Artist: Ram Onkar

Ganesha - Pencil Carving | Less(Rs.):7500 | Rs.6999
Artist: Bhuvan Patel

Awakening Light | Less(Rs.):57999 | Rs.55500
Artist: Raghav Joshi

Figure 5.4 Artworks

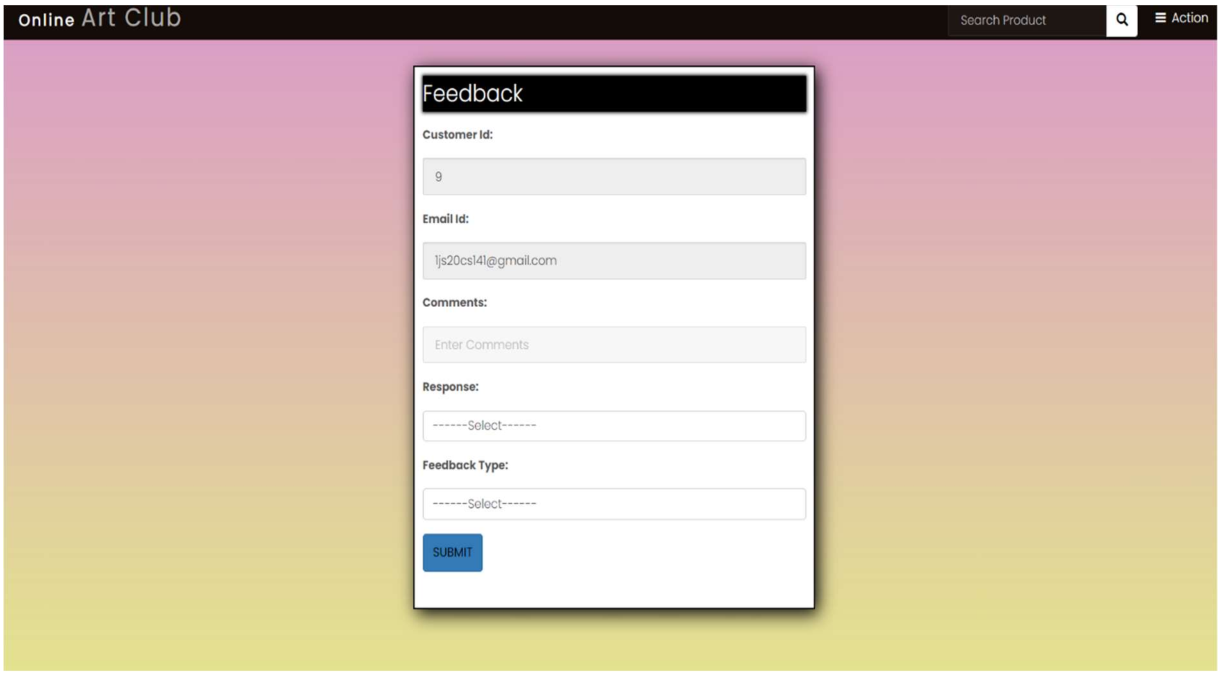
Figure 5.5 shows the booking of artworks by customer. The form takes down the details of customer such as customer id, art name, art id, quantity, delivery address etc.



Customer Id:	9
Art_Id:	1
Name:	karu
Art_Name:	Black Sand Beach
Order_Status:	Available
Delivery_Address:	Enter Address
Quantity:	Enter Quantity

Figure 5.5 customer booking

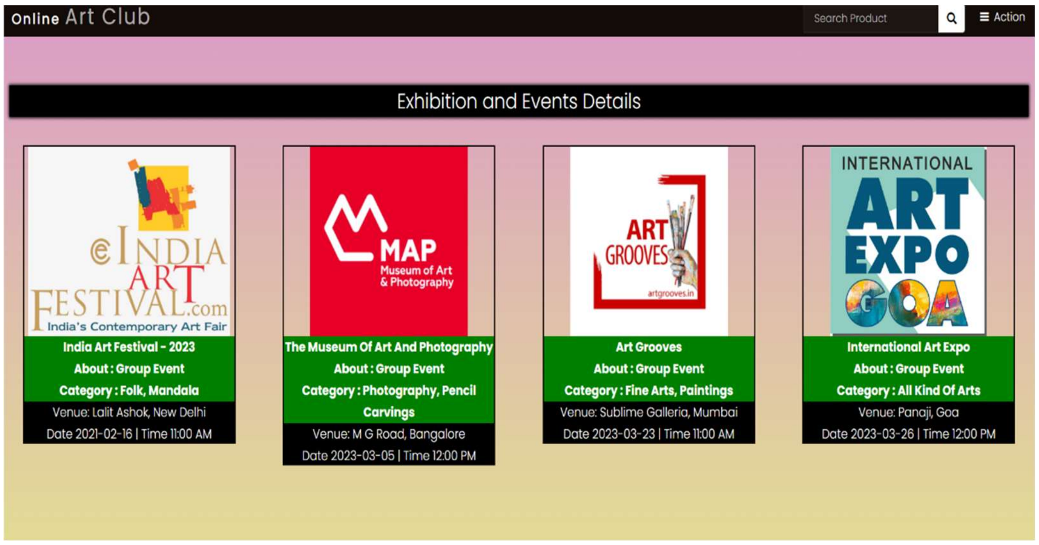
Figure 5.6 shows the feedback page that is provided to the customer. The customer can share their views and opinions about the artworks, website, their services etc. The feedback page takes the email id of the customer. This feedback is viewed by the admin and then the admin can manage those feedbacks.



Customer Id:	9
Email Id:	1js20cs141@gmail.com
Comments:	Enter Comments
Response:	------Select------
Feedback Type:	------Select------

Figure 5.6 Customer feedback

Figure 5.7 shows the list of upcoming exhibition event's details.



India Art Festival - 2023
About : Group Event
Category : Folk, Mandala
Venue: Lalit Ashok, New Delhi
Date 2021-02-16 | Time 11:00 AM

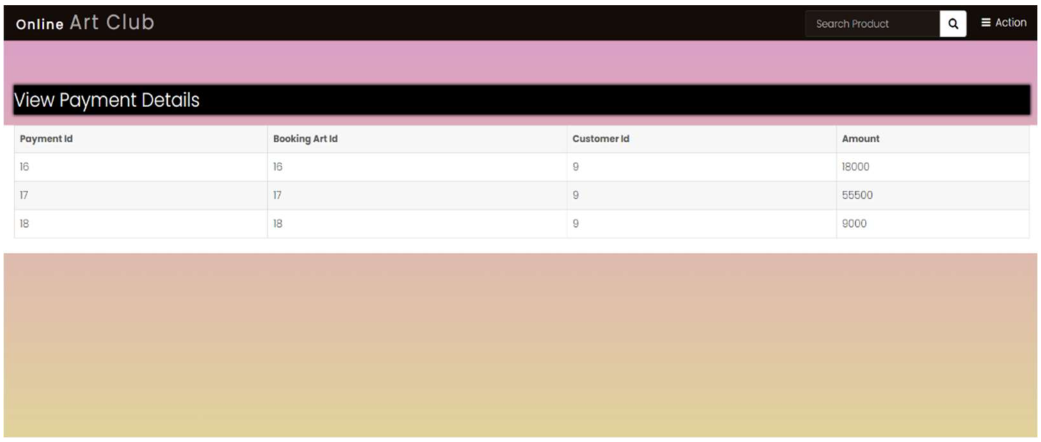
The Museum Of Art And Photography
About : Group Event
Category : Photography, Pencil Carvings
Venue: M G Road, Bangalore
Date 2023-03-05 | Time 12:00 PM

Art Grooves
About : Group Event
Category : Fine Arts, Paintings
Venue: Sublime Galleria, Mumbai
Date 2023-03-23 | Time 11:00 AM

International Art Expo
About : Group Event
Category : All Kind Of Arts
Venue: Panaji, Goa
Date 2023-03-26 | Time 12:00 PM

Figure 5.7 Exhibition event details

Figure 5.8 displays payment details of booking made by the customer.



View Payment Details

Payment Id	Booking Art Id	Customer Id	Amount
16	16	9	18000
17	17	9	55500
18	18	9	9000

Figure 5.8 Payment details

Figure 5.9 shows cancel page where customer can cancel their booking if they wish to.

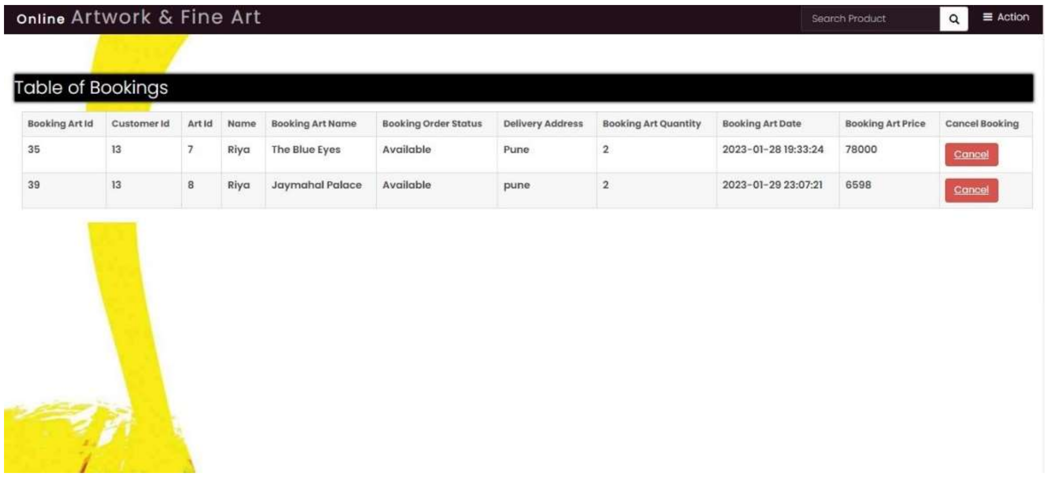


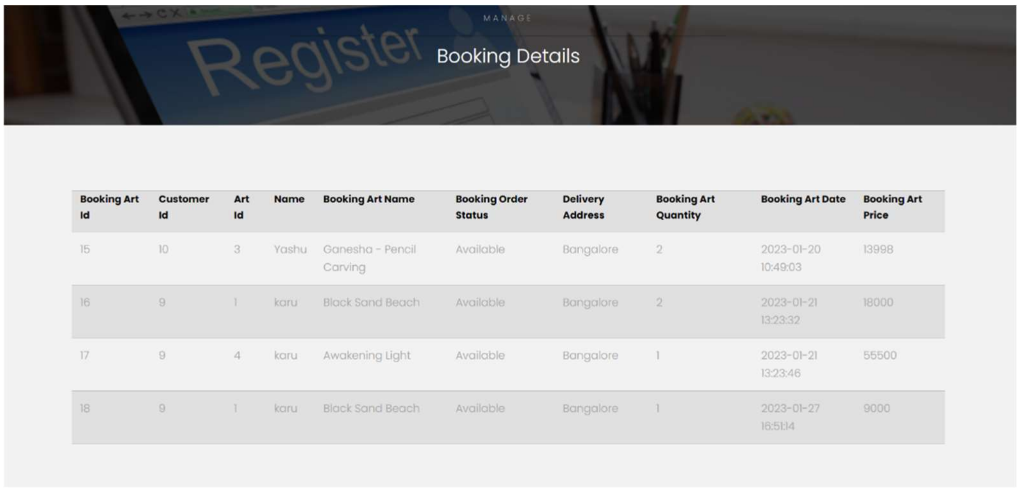
Table of Bookings

Booking Art Id	Customer Id	Art Id	Name	Booking Art Name	Booking Order Status	Delivery Address	Booking Art Quantity	Booking Art Date	Booking Art Price	Cancel Booking
35	13	7	Riya	The Blue Eyes	Available	Pune	2	2023-01-28 19:33:24	78000	Cancel
39	13	8	Riya	Jaymahal Palace	Available	pune	2	2023-01-29 23:07:21	6598	Cancel

Figure 5.9 Cancel booking

5.1.3 ADMIN SIDE

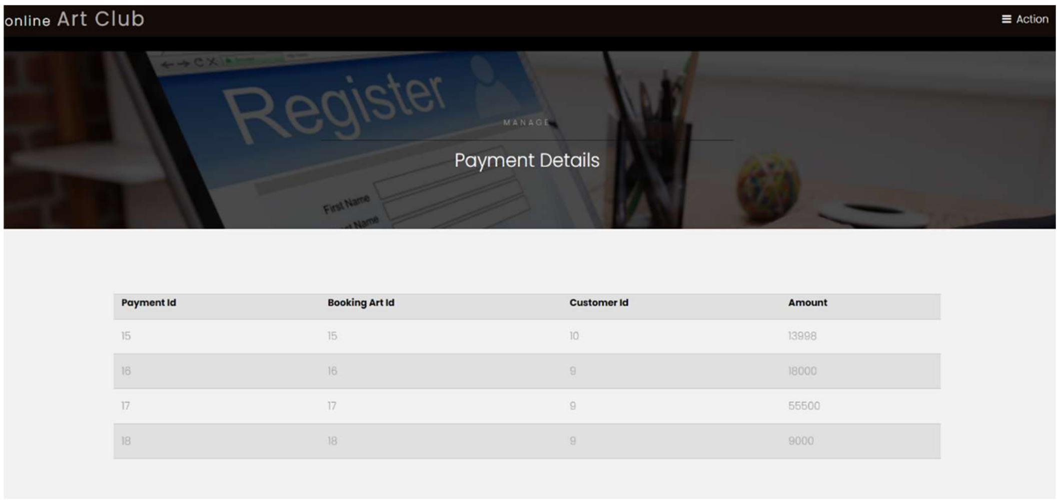
Figure 5.10 shows the admin side where he can view the bookings made by the customers.



Booking Art Id	Customer Id	Art Id	Name	Booking Art Name	Booking Order Status	Delivery Address	Booking Art Quantity	Booking Art Date	Booking Art Price
15	10	3	Yashu	Ganesha - Pencil Carving	Available	Bangalore	2	2023-01-20 10:49:03	13998
16	9	1	karu	Black Sand Beach	Available	Bangalore	2	2023-01-21 13:23:32	18000
17	9	4	karu	Awakening Light	Available	Bangalore	1	2023-01-21 13:23:46	55500
18	9	1	karu	Black Sand Beach	Available	Bangalore	1	2023-01-27 16:51:14	9000

Figure 5.10 Booking Details

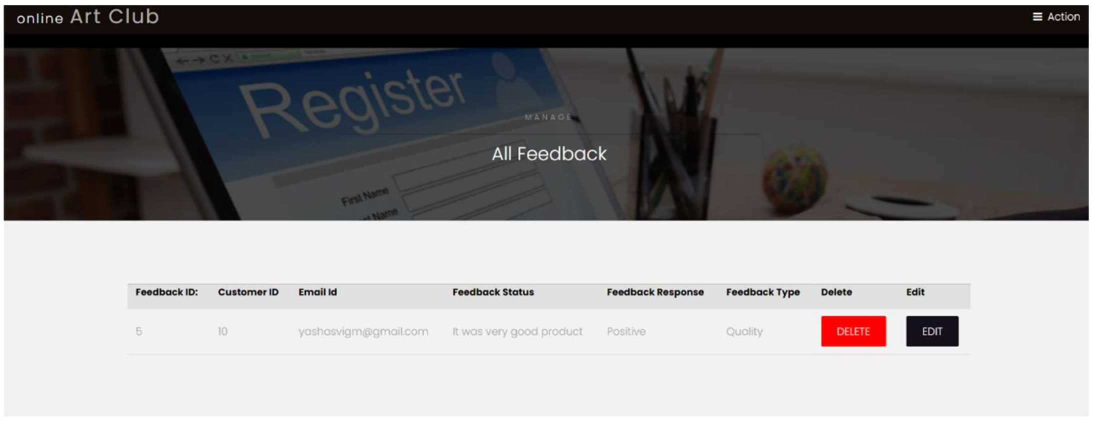
Figure 5.11 shows the payment details to the admin side made by the customers.



Payment Id	Booking Art Id	Customer Id	Amount
15	15	10	13998
16	16	9	18000
17	17	9	55500
18	18	9	9000

Figure 5.11 Payment details

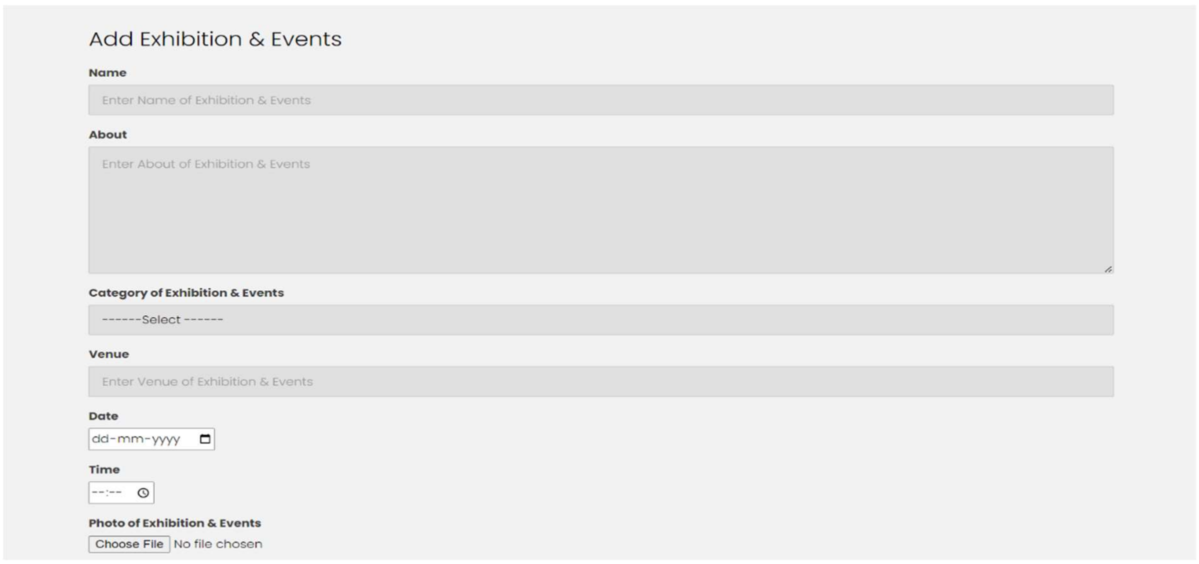
Figure 5.12 shows feedback page managed by the admin.



Feedback ID:	Customer ID	Email Id	Feedback Status	Feedback Response	Feedback Type	Delete	Edit
5	10	yashasvigm@gmail.com	It was very good product	Positive	Quality	DELETE	EDIT

Figure 5.12 Feedback Page

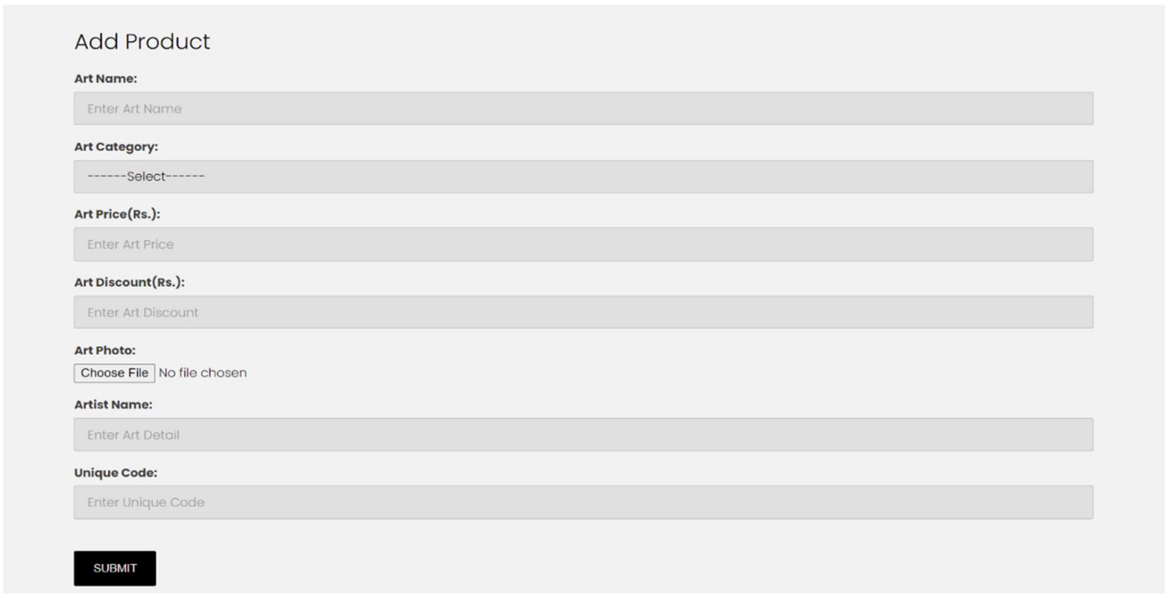
Figure 5.13 shows adding exhibition events page.


Add Exhibition & Events

Name	Enter Name of Exhibition & Events
About	Enter About of Exhibition & Events
Category of Exhibition & Events	------Select ------
Venue	Enter Venue of Exhibition & Events
Date	dd-mm-yyyy
Time	--:--
Photo of Exhibition & Events	Choose File No file chosen

5.13 Add exhibition and events

Figure 5.14 shows the adding artworks page.


Add Product

Art Name:	Enter Art Name
Art Category:	------Select------
Art Price(Rs.):	Enter Art Price
Art Discount(Rs.):	Enter Art Discount
Art Photo:	Choose File No file chosen
Artist Name:	Enter Art Detail
Unique Code:	Enter Unique Code

Figure 5.14 Add Artworks Page

5.2 BACK END

The Figure 5.15 describes how the database is created and managed at the back end using PHP server-side scripting language.

From Figure 5.15 we can observe that the database created is named as ‘artwork’ and it contains thefollowing tables:

Admin_info: This table shows information about the details of admin who manages the account.

Customer_info: This table contains information about the details of the customer.

Art_info: This tables contains information about the booking details of the artwork purchased by the customer.

Booking_info: This table contains information about the booking details of the artworks made by the customer.

Exhibition_info: This table contains information about the upcoming details of the exhibition events.

Feedback: This table contains information about the feedback page.

Payment_info: This table contains information about the payment details made by the customer for purchasing the artworks.

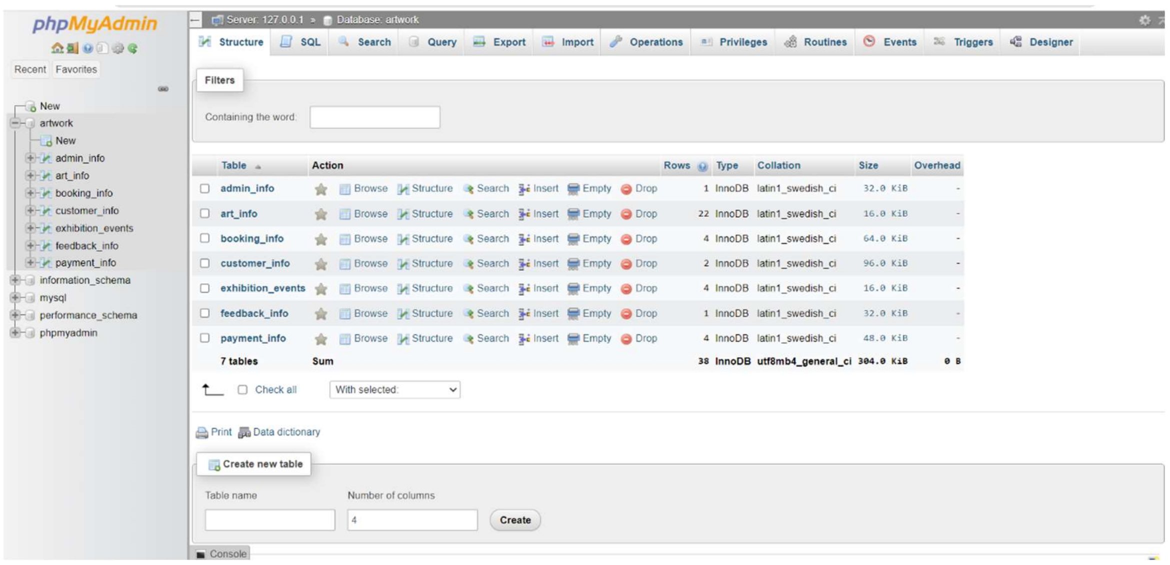


Table	Action	Rows	Type	Collation	Size	Overhead
admin_info	Browse Structure Search Insert Empty Drop	1	InnoDB	latin1_swedish_ci	32.0 KiB	-
art_info	Browse Structure Search Insert Empty Drop	22	InnoDB	latin1_swedish_ci	16.0 KiB	-
booking_info	Browse Structure Search Insert Empty Drop	4	InnoDB	latin1_swedish_ci	64.0 KiB	-
customer_info	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	96.0 KiB	-
exhibition_events	Browse Structure Search Insert Empty Drop	4	InnoDB	latin1_swedish_ci	16.0 KiB	-
feedback_info	Browse Structure Search Insert Empty Drop	1	InnoDB	latin1_swedish_ci	32.0 KiB	-
payment_info	Browse Structure Search Insert Empty Drop	4	InnoDB	latin1_swedish_ci	48.0 KiB	-
7 tables	Sum	38	InnoDB	utf8mb4_general_ci	304.0 KiB	0 B

Figure 5.15 a view of the database in phpMyAdmin (back end)

In Figure 5.15 we can see various options like

SQL: can be used to run SQL queries in the backend

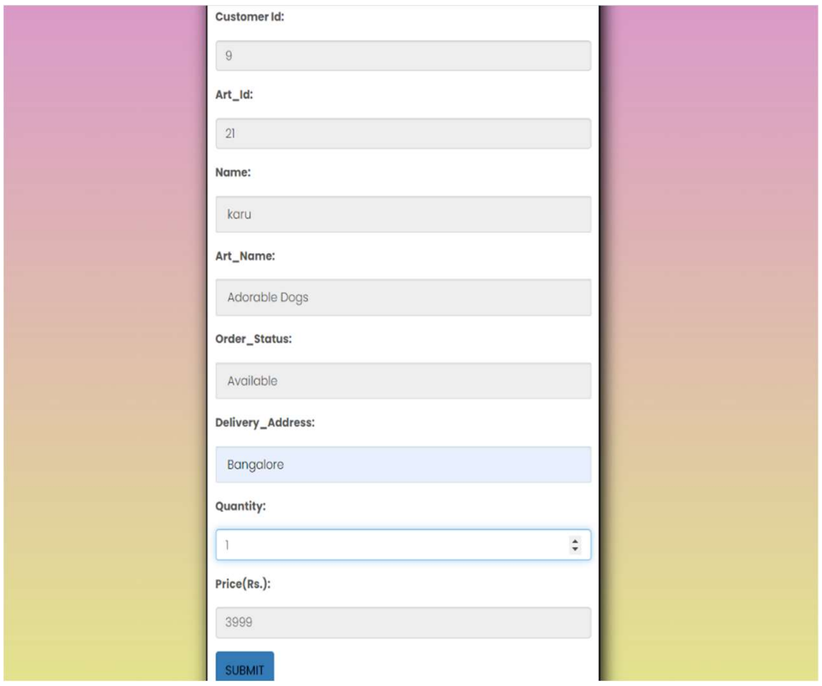
Insert: can be used to insert tuples into an existing table

Export: is used to export SQL code of the current table

Import: is used to import SQL code of a table or database

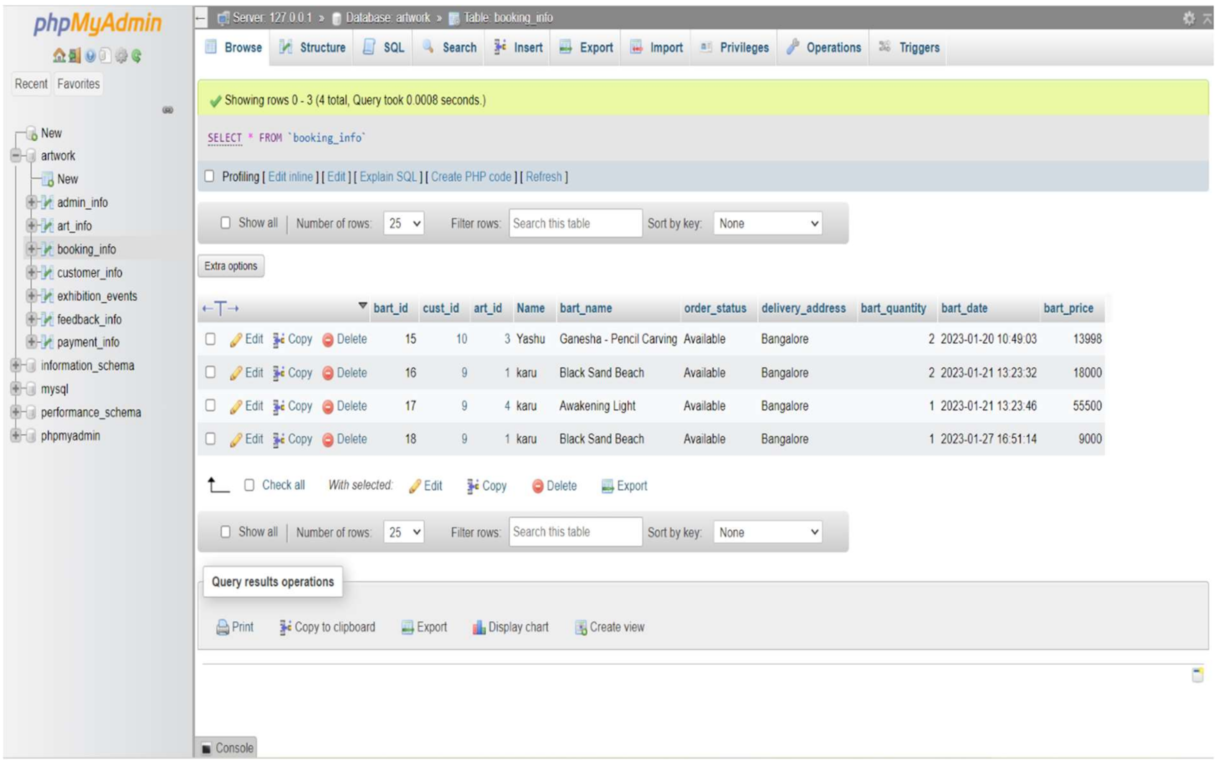
Triggers: can be used to create triggers to place restrictions on the type of data entering the table

From Figure 5.16 we can observe that when new values are inserted at the front end the samevalues are being inserted at the back end to the table Booking_info. When you click on the submit button of booking page which is in the front-end side, the booking details are automatically generated in the back end side of PHP. Fig 5.17 shows booking details being inserted automatically on the back-end side.



Customer Id:	9
Art_Id:	21
Name:	karu
Art_Name:	Adorable Dogs
Order_Status:	Available
Delivery_Address:	Bangalore
Quantity:	1
Price(Rs.):	3999

Figure 5.16 booking page in front end side



phpMyAdmin
Recent Favorites
New
artwork
New
admin_info
art_info
booking_info
customer_info
exhibition_events
feedback_info
payment_info
information_schema
mysql
performance_schema
phpmyadmin
Server: 127.0.0.1 » Database: artwork » Table: booking_info
Browse Structure SQL Search Insert Export Import Privileges Operations Triggers

Showing rows 0 - 3 (4 total, Query took 0.0008 seconds.)

```
SELECT * FROM `booking_info`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Extra options

bart_id	cust_id	art_id	Name	bart_name	order_status	delivery_address	bart_quantity	bart_date	bart_price
15	10	3	Yashu	Ganesha - Pencil Carving	Available	Bangalore	2	2023-01-20 10:49:03	13998
16	9	1	karu	Black Sand Beach	Available	Bangalore	2	2023-01-21 13:23:32	18000
17	9	4	karu	Awakening Light	Available	Bangalore	1	2023-01-21 13:23:46	55500
18	9	1	karu	Black Sand Beach	Available	Bangalore	1	2023-01-27 16:51:14	9000

Check all With selected: Edit Copy Delete Export
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Query results operations
Print Copy to clipboard Export Display chart Create view
Console

Figure 5.17 booking details being inserted automatically on the back-end side

Similarly all the other details are automatically inserted in the respective database tables of back-end side when new values are added in the front end side. Not only the insertion of values happens, but even the modification such as update and deletion takes place in same manner.

Chapter 6

CONCLUSION

SUMMARY

A database was created for a market that can use it for keeping track on art gallery Galleries are divided into many art galleries. Galleries have different names, locations, etc. Each gallery will have different exhibitions and each exhibition will have a start and end date. The galleries will have different artist displaying their artwork. The model can also be adapted to meet other purposes and thus be used for other projects. The database structure is quite simple, which makes it easy for also other programmers to understand it. In conclusion, a database is a far more efficient mechanism to store and organize data than spreadsheets it allows for a centralized facility that can easily be modified and quickly shared among multiple users. Having a web based front end removes the requirement of users having to understand and use a database directly, and allows users to connect from anywhere with an internet connection and a basic web browser. It also allows the possibility of queries to obtain information for various surveys. Due to the number of users reading and modifying student data in the department, it is an ideal use for such a system.

FUTURE ENHANCEMENTS:

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:

[1] We can add printer in future.
[2] We can give more advance software for Online Art Gallery including more facilities.
[3] We can host platform on online servers to make it accessible worldwide.
[4] Integrate multiple load balancers to distribute the loads of the system.
[5] Create the master and slave database structure to reduce the overdue of the database queries.
[6] Implement the backup mechanisms for taking backup of codebase and database on regular basis on different servers.

The above-mentioned points are the enhancements which can be done to increase the applicability and usage of this mini project. Here we can maintain the records of the arts and orders. Also, as it can be seen that nowadays the players are versatile i.e so there is a scope for introducing a method to maintain all the arts, orders order update, customer, as well as the details regarding the upcoming exhibition events.

LIMITATIONS:

Though the software presents a broad range of options to its users, some intricate options could not be covered into it; partly because of the logistic and partly due to the lack of sophistication. Paucity of time was also a major constraint, thus it was not possible to make the software foolproof and dynamic. Excel support has not been developed for arts, order due to some criticality. The transactions are made through only cash on delivery mode and no online payment mode. Offline reports of Arts, Order Updates cannot be generated due to the batch execution. Considerable efforts have made the software easy to operate even for the people not related to the field of computers but it is acknowledged that a layman may find it a bit problematic at the first instance.

Chapter 7

REFERENCES

BOOK REFERNCES

[1] Fundamentals of Database Systems, Ramez Elmasri and Shamkant B. Navathe, 7th Edition, 2017,Pearson.

[2] Database management systems, Ramakrishnan, and Gehrke, 3rd Edition, 2014, McGraw Hill.

[3] Silberschatz Korth and Sudharshan, Database System Concepts, 6th Edition, McGrawHill, 2013.

WEB REFRENCES

For Front End Code and CSS styling

[1] https://www.w3schools.com/html

[2] https://www.stackoverflow.com

[3] https://www.tutorialspoint.com

[4] https://developer.mozilla.org/

For MySQL references

[1] https://www.youtube.com

[2] https://www.udemy.com

For connecting web server

[1] https://www.wampserver.com/en/