ARTWAGON: VIRTUAL ART GALLERY

Developing a Dynamic Data Driven (Server-side) Web Application
By Using Advanced Technologies (AJAX, PHP)

Submitted in the partial fulfilment of the requirements for

the award of the degree of

Bachelor of Technology

In

Computer Science and Engineering

by

GOLTHI MADHU APPALA NARASIMHA(21761A05F6)
NAMBURI PRASANTH(21761A05H6)
BHEEMASETTI TEJA SRI (22765A0514)

Under the guidance of

Mrs. B. Usha Rani,

Sr. Assistant Professor, Dept. of CSE

Department of Computer Science and Engineering
Lakireddy Bali Reddy College of Engineering (Autonomous)
Accredited by NAAC & NBA (Under Tier - I)
Affiliated to JNTUK, Kakinada; ISO 9001:2015 Certified
2023-24

LAKIREDDY BALI REDDY COLLEGE OF ENGINEERING
(AUTONOMOUS)
Accredited by NAAC with 'A' Grade & NBA (Under Tier - I), ISO 9001:2015 Certified Institution
Approved by AICTE, New Delhi and Affiliated to JNTUK, Kakinada
L.B. REDDY NAGAR, MYLAVARAM, NTR DIST., A.P.-521 230.
hodcse@lbrce.ac.in, cseoffice@lbrce.ac.in, Phone: 08659-222 933, Fax: 08659-222931
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the **Server-Side Scripting Lab (20CS63**) project entitled "**ARTWAGON: VIRTUAL ART GALLERY**" is being submitted by **GOLTHI MADHU APPALA NARASIMHA (21761A05F6), NAMBURI PRASANTH (21761A05H6), BHEEMASETTI TEJA SRI (22765A0514)** in partial fulfilment for the award of B. Tech in Computer Science & Engineering to the **J**awaharlal **N**ehru **T**echnological **U**niversity **K**akinada is a record of bonafide work carried out by him/her under our guidance.

The results embodied in this Developing a Dynamic Data Driven (Server-side) Web Application By Using Advanced Technologies (AJAX, PHP) Project report have not been submitted to any other University or Institute for the award of any degree or diploma.

Project Guide
Mrs. B. Usha Rani,
Sr. Assistant Professor.

External Examiner

ACKNOWLEDGEMENT

I would like to thank **Mrs. B. Usha Rani,** Assistant Professor**,** CSE department for the encouraging and support in carrying out this Dynamic Data Driven (Server-side) Web Application.

I also take the privilege to record my thanks to **Dr. D. Veeraiah,** Professor, Head of the Department of CSE whose encouragement, cooperation and valuable support crown my success.

I express my thanks to the support given by management in completing my Web Application / Project. I also express my sincere gratitude & deep sense of respect to the Principal, **Dr.K. Appa Rao** for making us available all the required assistance and his support and inspiration to carry out this Dynamic Data Driven (Server-side) Web Application / Project in the Institute.

I am thankful to the teaching and non-teaching staff of CSE department for their direct as well as indirect help in my Dynamic Data Driven (Server-side) Web Application / Project.

I am elated to avail my selves to this opportunity to express my deep sense of gratitude to my parents.

ABSTRACT

ARTWAGON is a sophisticated web application that revolutionizes the art world by seamlessly connecting users with diverse artworks and artists through an immersive virtual gallery experience. This project aims to bridge the gap between art enthusiasts and creators, fostering a dynamic platform for appreciation, interaction, and communication within the art community. The core functionality of ARTWAGON revolves around providing users with a curated selection of artworks across various genres and mediums. From sculptures to oil paintings, users can explore an extensive collection of art pieces, each accompanied by detailed information about the artist and their work. The virtual gallery offers a captivating visual experience, allowing users to navigate through different exhibition rooms and discover new artworks with ease. One of the distinctive features of ARTWAGON is its emphasis on fostering communication between users and artists. Through the platform, users can engage with artists directly by sending responses and feedback about their work. This two-way interaction creates a collaborative environment where artists can receive valuable insights and appreciation from their audience, enhancing the overall experience for both parties. Furthermore, ARTWAGON provides artists with dedicated tools to manage their portfolios effectively. Artists can maintain separate tables in the database to organize and showcase their artworks, enabling seamless CRUD operations for efficient content management. This functionality empowers artists to showcase their creativity and connect with a wider audience through the platform. The project also includes an intuitive admin panel that allows administrators to monitor platform activity, track user engagement, and analyze data related to artists and artworks. This enables administrators to gain valuable insights into platform usage and make informed decisions to enhance the user experience further.

CONTENTS

INTRODUCTION

In the digital age, the art world is undergoing a profound transformation, with technology playing a pivotal role in reshaping how artworks are created, shared, and appreciated. Virtual art galleries have emerged as a dynamic platform that bridges geographical barriers and brings art enthusiasts closer to the vibrant world of creativity. In this context, the ARTWAGON Virtual Art Gallery project seeks to leverage the power of technology to create an immersive and interactive online platform that connects users with diverse artworks and artists. The primary objective of ARTWAGON is to provide users with a captivating virtual gallery experience where they can explore a wide range of artworks from the comfort of their homes. By curating an extensive collection of art pieces spanning various genres, styles, and mediums, ARTWAGON aims to cater to the diverse tastes and preferences of its users. From traditional paintings to contemporary sculptures, the platform offers a rich tapestry of artistic expression, inviting users to embark on a journey of discovery and appreciation.

Central to the ARTWAGON experience is the emphasis on fostering communication and engagement between users and artists. Unlike traditional galleries where the interaction between viewers and creators is limited, ARTWAGON enables users to connect directly with artists, providing a platform for dialogue, feedback, and collaboration. Through features such as response submissions and messaging tools, users can engage in meaningful conversations with artists, sharing their thoughts, insights, and appreciation for their work. Furthermore, ARTWAGON aims to empower artists by providing them with a comprehensive set of tools to showcase and manage their portfolios effectively. By allowing artists to maintain separate tables in the database for their artworks, the platform facilitates seamless content management, enabling artists to update, organize, and showcase their creations with ease.

MODULES

2.1 Main page:

The main.html serves as the primary landing page for the ARTWAGON Virtual Art Gallery, offering convenient navigation buttons for user login, artist login, and admin login. It plays a crucial role in directing users, artists, and administrators to their respective interfaces, facilitating seamless authentication and access control to different areas of the platform.

2.2 User login and signup:

signup_login.html provides essential forms for user registration and login, enabling users to create accounts and access the ARTWAGON platform securely. By offering a user-friendly interface for account management, this page simplifies the registration and login process, enhancing user experience and engagement with the virtual art gallery.

2.3 User index:

index.php serves as the central hub for users upon successful login, offering a wide range of options to explore artworks and interact with artists. Its functionality includes displaying artworks based on art types and artist names, facilitating user responses and comments, and providing access to the exhibition rooms for immersive art experiences.

2.4 Artist login and signup:

artist_signup_login.php is pivotal for artists to register and access their dedicated spaces within the ARTWAGON platform. It enables artists to create profiles, manage artworks, and engage with users through responses and comments. By establishing unique tables for each artist and facilitating secure login procedures, this module ensures personalized experiences for artists.

2.5 Artist index:

artist_index.php provides artists with a comprehensive dashboard to manage their portfolios and artworks effectively. It enables artists to view, update, and delete artworks, as well as upload new creations with descriptions and details. By empowering artists with intuitive tools for artwork management, this module fosters creativity and collaboration within the virtual art gallery.

2.6 Art display:

display_art.php dynamically presents artworks to users based on selected art types, enhancing the browsing experience and enabling users to discover diverse artworks. By filtering and displaying artworks in the bottom frame, this module streamlines the process of exploring and appreciating artworks within the ARTWAGON platform. display_art_by_name.php allows users to explore artworks specifically attributed to individual artists, providing a personalized viewing experience. By showcasing artworks based on artist names, this module promotes artist recognition and enables users to engage with specific artists' portfolios effectively.

2.7 Add art:

add_art.php facilitates artists in uploading new artworks to their portfolios, enriching the ARTWAGON platform with fresh and diverse content. By enabling artists to provide descriptions and details for each artwork, this module enhances the presentation and accessibility of artworks within the virtual art gallery.

2.8 Delete art:

delete_art.php empowers artists to manage their portfolios by allowing them to delete specific artworks as needed. By providing artists with control over their content, this module ensures the integrity and quality of artists' portfolios within the ARTWAGON platform.

2.9 Update details:

update_process.php and update_db.php work in conjunction to enable artists to update details and information for individual artworks within their portfolios. These modules streamline the process of modifying artwork details, ensuring accurate and up-to-date representations of artworks within the virtual art gallery.

2.10.1 Admin index:

admin.php and admin_index.php provide administrators with essential interfaces for managing platform activity, user accounts, and content. These modules offer comprehensive tools for monitoring and controlling platform operations, ensuring the smooth functioning and integrity of the ARTWAGON Virtual Art Gallery.

2.10.2 Exhibition room:

exhibition_room.php introduces users to the immersive experience of virtual exhibition rooms, where they can explore artworks with animations and interact with artists through comments and responses. By offering a unique and engaging platform feature, this module enhances user engagement and appreciation of artworks within the ARTWAGON platform.

2.10.3 Comment handling:

comment_handling.php facilitates communication between users and artists by managing user comments and responses within artist portfolios. This module ensures that user feedback and interactions are appropriately directed to the respective artists, fostering meaningful engagement and collaboration within the ARTWAGON Virtual Art Gallery. send_message.php enables users to send responses to artists' messages and communications, facilitating dialogue and interaction within the ARTWAGON platform. By providing a streamlined process for user-artist communication, this module enhances user experience and engagement with artists' portfolios.

2.10.4 Logout:

logout.php ensures secure session closure for users, artists, and administrators upon logout from the ARTWAGON platform. By terminating user sessions and maintaining platform security, this module safeguards user data and enhances the overall user experience within the virtual art gallery.

TECHNOLOGIES USED

i) <u>HTML:</u>

HTML stands for Hypertext Markup Language. It is the most widely used language to create web pages. HTML is a markup language used to simply ―mark-up a text document with tags that tell a web browser how to structure it to display. HTML is defined with the use of the Standard Generalized Markup Language (SGML), which is an International Standards Organization (ISO) standard notation for describing text- formatting languages. The addition of style sheets to HTML in the late 1990s advanced its capabilities to specify presentation details of the content in HTML document.

Basic Syntax:

- HTML is a descriptive language that helps convert ordinary text into hypertext by adding specialcodecalledtagsorelements.Thefundamentalsyntacticunits ofHTML aretags.
- TagstellthewebbrowserhowtodisplaythecontentintheHTMLdocument.
- The syntax of a tag is the tag's name surrounded by angle brackets (< and >). Most tags appear inpairs:anopeningtagandaclosingtag.Thenameofaclosingtagisthenameofitscorrespondingopeningtagwithaslashattachedtothebeginning.
- For example, if the tag's name is p, the corresponding closing tag is named /p, whateverappearsbetweena tagandits closingtagis the contentofthe tag.

3.1.1 Standard HTML Document Structure:

A HTML Document is mainly divided into two parts:

- HEAD: This contains the information about the HTML document. For Example, Title of the page, Meta Data, external link files etc.
- BODY: This contains everything you want to display on the Web Page.

```
<!DOCTYPE html>
<html>
<head>
<title>

</title>
</head>
<body>
- - - - - - - - - -
</body>
</html>
```

- The <!DOCTYPE html> tells the document is a HTML Document and its version.
- The <html> element is the root element of an HTML page
- The <head> element contains meta information about the HTML page
- The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The <body> element defines the document's body, and is a container for all the
- visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- And the HTML documents have the file extension name .html or .htm.

Example:

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
<h1>My First Heading</h1>
<p>My first paragraph.</p>
</body>
</html>
```

HTML Head Section:

The HTML <head> element is a container for the following elements: <title>, <style>, <meta>, <link>, <script> and <base>.

- The <title> element defines the title of the document. The title must be text-only, and it is shown in the browser's title bar or in the page's tab.
- The <style> element is used to define style information for a single HTML page
- The <link> element defines the relationship between the current document and and external resource. The <link> tag is most often used to link to external style sheets.
- The <meta> element is typically used to specify the character set, page description, keywords, author of the document, and viewport settings.
- The <script> element is used to define client-side Java Scripts.

HTML Body Section:

The actual contents of the web page are placed in the body section. It includes elements such as tables, paragraphs, lists, images, hyperlinks, headings, forms, etc. The <body> tag consists of the following attributes:

- bgcolor: specifies the background of the web page
- link: specifies the color of the unvisited link color
- alink: specifies the color of active link
- vlink: specifies the color of the visited link
- text: specifies the text color
- background: specifies the URL of an image which is to be set as background of the bodyAttributes
- Align is used to specify the alignment of the horizontal rule
- Size is used to specify the thickness (in terms of pixels) of the horizontal rule
- Width is used to specify the width of the horizontal ruler

3.1.2 Heading Tags:

Text is often separated into sections in documents by beginning each section with a heading. Larger sections sometimes have headings that appear more prominent than headings for sections nested inside them. In HTML, there are six levels of headings, specified by the tags <h1>, <h2>, <h3>, <h4>, <h5>, and <h6>, where <h1> specifies the highest-level heading, <h2> is below to the level of <h1>, <h3> is below to the level of <h2> and so on,
<h6> is the lowest level of heading.

3.1.3 Image Tag:

The HTML <img> tag is used to embed an image in a web page. The <img> tag has two required attributes:

- src - Specifies the path (URL) to the image
- alt - Specifies an alternate text for the image

Images are not inserted into a web page; images are just linked to web pages. When a web page loads; then the browser gets the image from the location specified in src attribute and inserts it into the page. If the browser cannot find the image, then it will display the text message specified to the alt attribute. The <img> tag is empty, it contains attributes only, and does not have a closing tag.

Usage:

<imgsrc="url" alt="alternate text">

3.1.4 Hyperlink in HTML:

HTML links are hyperlinks. You can click on a link and jump to another document. When you move the mouse over a link, the mouse arrow will turn into a little hand. Web pages can contain hyperlinks that take you directly to other pages and even specific parts of a given page. Hyperlinks allow visitors to navigate between Web sites or between the web pages of one single web site by clicking on words, phrases, and images. Hyperlinks can be created using text or images.

Linking Documents - The <a> Element:

A link is specified using the <a> element. This element is called anchor tag as well. Anything between the opening <a> tag and the closing </a> tag becomes part of the link, and a user can click that part to reach to the linked document.

Following is the simple syntax to use this tag:

```
<a href="Document URL" /> Link Phrase </a>
```

The href attribute is used to define the address of the document to be linked.

Targets within Documents:

If the target of a link is not at the beginning of a document, it must be some element within the document, in which case there must be some means of specifying it. The target element can include an id attribute, which can then be used to identify it in href attribute.

Following are most frequently used attributes for <a> tag:

- href: specifies the URL of the target of a hyperlink. Its value is any valid document URL, absolute or relative, including a fragment identifier or a JavaScript code fragment.

- target: specify where to display the contents of a selected hyperlink.

1. If set to "_blank" then a new window will be opened to display the loaded page.

2. If set to "_top" or "_parent" then same window will be used to display the loaded document

3. If set to "_self" then loads the new page in current window.

4. By default it is "_self".

3.1.5 Text Markup Tags:

- Markup means how the text content of an HMTL document is being formatted with the use of HMTL tags. Formatting describes layout and presentation details of the content in the document.
- Paragraphs: Text is normally organized into paragraphs in the body of the document. The <p> tag in HTML defines a paragraph. These have both opening and closing tags. So, anything mentioned within <p> and </p> is treated as a paragraph.

3.1.6 Line Breaks:

Sometimes text requires a line break without the preceding blank line. For this purpose, break tag <br/> is used. The break tag differs syntactically, as it has no content and therefore has no closing tag.

ii) <u>CSS</u>

The cascading style sheet (CSS) is a markup language used in web document for presentation purpose. The purpose of CSS is to separate web content from web presentation.

Advantages:

- By combining CSS with HTML document, considerable amount of flexibility into the content submission can be achieved.
- Separating out the style from actual contents help in managing large scale complex sites. So, CSS facilitates publication of contents in multiple presentation formats.
- If CSS is used effectively then global style sheet can be applied to a web document. This helps maintaining consistency in web document.
- If a small change needs to be done in the style of web content, then CSS makes it more convenient.

A Cascading Style Sheet is a collection of CSS style rules. Each style rule in the rule list has two parts:

1. Selector, which indicates the element or elements affected by rule

2. A list of property – value pairs

Usage:

selector { property : value ; property : value ;…}

Example: h1 { font – family : arial } □ This CSS rule applies to all <h1> elements

3.2.1 Types of CSS:

There are three levels of style sheets, from lowest to highest in order:

1. Inline Style Sheet

2. Embedded Style Sheet/Internal Style Sheet

3. External Style Sheet

Inline Style Sheet:

Inline style sheets are applied to single HTML Element. Document Level Style sheets apply to the whole body of the document. External Style sheets can apply to bodies of any number of documents. Inline style sheets have precedence over document level, which have precedence over external style sheets.

Internal Style Sheet:

Internal styles are styles that are written directly in the HTML tag on the document. The normal rules of CSS apply inside the style attribute. Each CSS statement must be separated with a semicolon ";" and colons appear between the CSS property and its value.

For example,

```
<p style="background: blue; color: white;">
A new background and font color with inline CSS
</p>
```

Embedded Style Sheet:

Embedded Style Sheets refer to embed style sheet information into an HTML document using the style element. Embedding the style sheet information within <style>...</style> tags which appears only in the head section of your document.

External Style Sheet:

In those times when we need to apply style to more than one web document, in such cases external style sheets can be used. The central idea of style sheet is defined in .css file. The style defined in .css file will be applied to all the web pages by using LINK tag.

External Style Sheet is specified using the LINK element within the HEAD element block to specify the URL location of the External Style Sheet. URL values may be relative or absolute.

Usage:

```
<link rel="stylesheet" type="text/css" href="[Style sheet URL]">
```

Below is the style sheet file (ex1.css):

```
body {background-color:tan;}

h1 {color:maroon;font-size:20pt;} hr {color:navy;} p {font-size:11pt;margin-left:15px;}
```

Below is the html file (.html):

```
<html>
<head>
<title>External CSS</title>
<link rel= ―stylesheet‖ type= ―text/css‖ href= ―ex1.css‖/>
</head>
<body>
<h1>This is header1</h1><br>
<p>this is paragraph</p><br>
</body>
</html>
```

3.2.2 CSS Selectors:

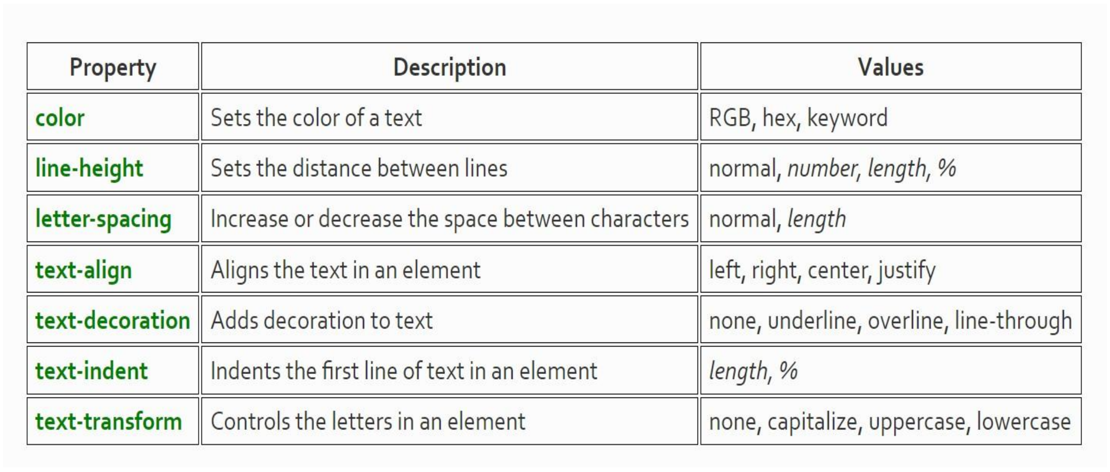
A selector specifies the elements to which the style information applies. The selector can have a variety of forms.

Simple Selector Forms: The simplest selector form is a single element name, such as h1, h2, p, etc… In this case the property values in the rule apply to all the occurrences of the named element.

- **Class Selectors:** Class selectors are used to allow different occurrences of the same tag to use different style specifications. A style class is defined in a style element by giving the style class a name, which is attached to the tag's name with a period.
- **Generic Selectors:** To specify the class of style specifications to the content of more than one tag, generic selectors are used. It is defined without a tag name in its selector. Without tag name, the name of the generic class begins with a period.
- **ID Selectors:** The ID selector is like the class selector but only the difference between the two is that class selector can be applied to more than one element whereas using the ID selector the style can be applied to one specific element.
- **Contextual Selectors:** Selectors can specify that the style should apply only to elements in certain positions in the document. The simplest form of contextual selector is the descendant selector. Element B is a descendant of element A if it appears in the content of A. And A is the ancestor of B. A particular element in the document can be selected by listing one or more ancestors of the element in the selector, with only white space separating the element names.

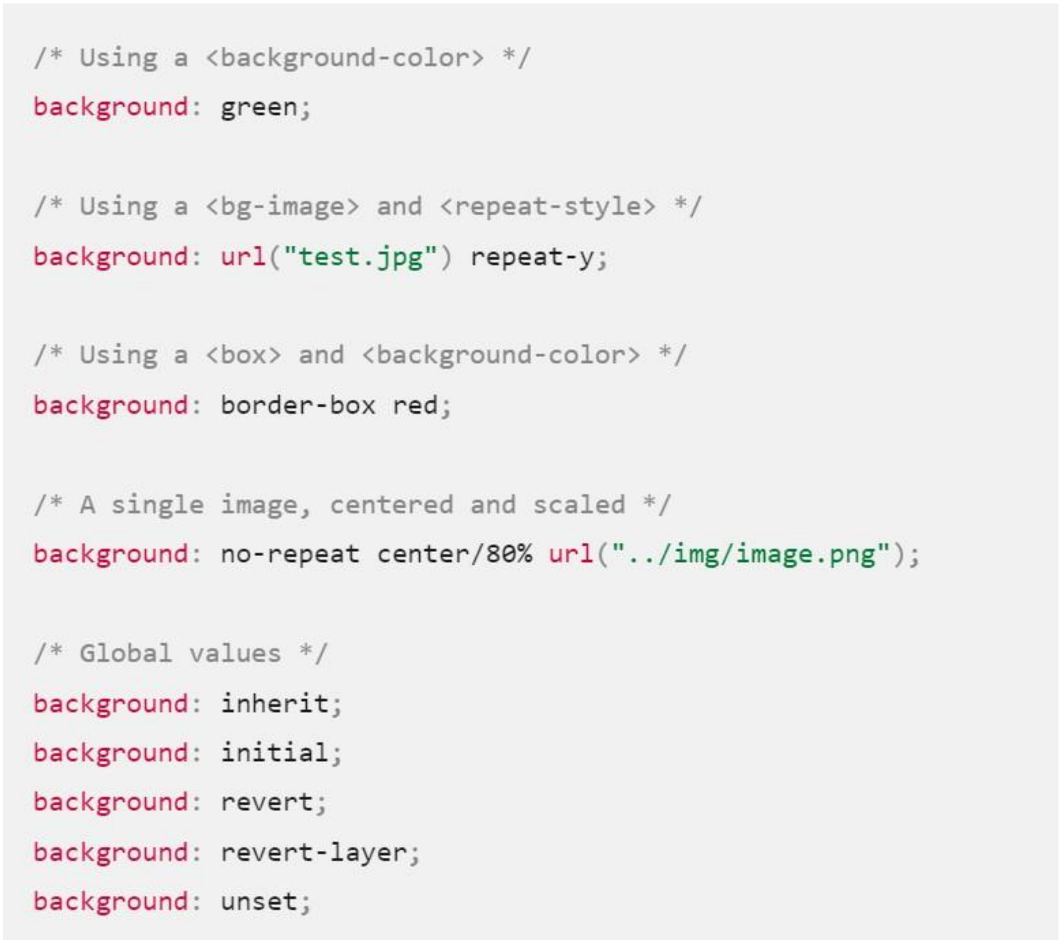
3.2.3 Properties:

Text Properties:



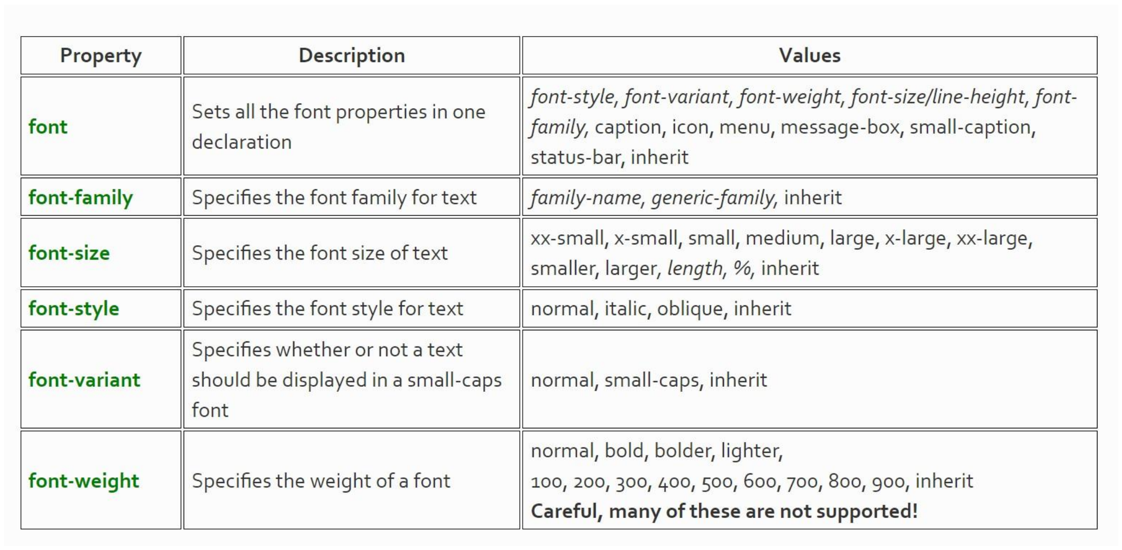
Property	Description	Values
color	Sets the color of a text	RGB, hex, keyword
line-height	Sets the distance between lines	normal, *number*, *length*, *%*
letter-spacing	Increase or decrease the space between characters	normal, *length*
text-align	Aligns the text in an element	left, right, center, justify
text-decoration	Adds decoration to text	none, underline, overline, line-through
text-indent	Indents the first line of text in an element	*length*, *%*
text-transform	Controls the letters in an element	none, capitalize, uppercase, lowercase

Background Properties:



```
/* Using a <background-color> */
background: green;

/* Using a <bg-image> and <repeat-style> */
background: url("test.jpg") repeat-y;

/* Using a <box> and <background-color> */
background: border-box red;

/* A single image, centered and scaled */
background: no-repeat center/80% url("../img/image.png");

/* Global values */
background: inherit;
background: initial;
background: revert;
background: revert-layer;
background: unset;
```

Font Properties:



Property	Description	Values
font	Sets all the font properties in one declaration	*font-style, font-variant, font-weight, font-size/line-height, font-family,* caption, icon, menu, message-box, small-caption, status-bar, inherit
font-family	Specifies the font family for text	*family-name, generic-family,* inherit
font-size	Specifies the font size of text	xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger, *length, %,* inherit
font-style	Specifies the font style for text	normal, italic, oblique, inherit
font-variant	Specifies whether or not a text should be displayed in a small-caps font	normal, small-caps, inherit
font-weight	Specifies the weight of a font	normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900, inherit **Careful, many of these are not supported!**

iii) <u>PHP</u>

3.3.1 Introduction to PHP

PHP, which stands for Hypertext Preprocessor, is a widely-used open-source server-side scripting language. Originally created by Danish-Canadian programmer Rasmus Lerdorf in 1994, PHP is now maintained by The PHP Development Team. It is specifically designed for web development and can be embedded into HTML, making it an integral part of web development for creating dynamic web pages and web applications.

key features and aspects of PHP:

Server-side scripting: PHP is primarily a server-side scripting language, meaning that the code is executed on the server before being sent to the client's browser. This allows for dynamic content generation, database interaction, and other server-side tasks.

Open source: PHP is open-source software, which means it is freely available for anyone to use, modify, and distribute. This has contributed to its widespread adoption and a vast community of developers who contribute to its development and maintenance.

Cross-platform compatibility: PHP is compatible with various operating systems, including Windows, Linux, macOS, and Unix-based systems. This makes it versatile and widely applicable for different web hosting environments.

Ease of use: PHP syntax is relatively easy to learn and understand, especially for those with prior experience in programming languages like C, Java, or Perl. Its syntax is similar to that of C and Perl, making it accessible to a broad range of developers.

Integration with databases: PHP provides built-in support for interacting with databases, with extensive support for database management systems such as MySQL, PostgreSQL, SQLite, and more. This makes it suitable for developing database-driven web applications.

Large ecosystem: PHP has a vast ecosystem of frameworks, libraries, and tools that streamline the development process and enhance productivity. Popular PHP frameworks include Laravel, Symfony, CodeIgniter, and Zend Framework, among others.

Security: While PHP itself is secure, developers need to follow best practices to ensure the security of their applications. This includes measures such as input validation, output escaping, secure database queries (e.g., prepared statements), and keeping PHP and server software up to date with security patches.

3.3.2 PHP GET and POST methods:

In PHP, the `$_GET` and `$_POST` superglobals are used to retrieve data sent to the server via HTTP GET and POST requests, respectively. These superglobals are associative arrays that contain key-value pairs of data sent from an HTML form or through URL parameters. Here's a brief overview of each:

`$_GET` Method:

The `$_GET` superglobal is used to retrieve data sent to the server using the HTTP GET method. Data sent using the GET method is appended to the URL as key-value pairs, visible in the address bar.
It is commonly used for retrieving data from a form with a method attribute set to "get", or for passing data between pages via URL parameters.

Example:

```
php
// Assuming URL: http://example.com/page.php?name=John&age=25
$name = $_GET['name']; // Retrieves value 'John'
$age = $_GET['age']; // Retrieves value '25'
```

$_POST Method:

The `$_POST` superglobal is used to retrieve data sent to the server using the HTTP POST method.
Data sent using the POST method is included in the body of the request and is not visible in the URL.
It is commonly used for submitting form data that may contain sensitive or large amounts of information.
Example:

```
php
// Assuming form submitted with method="post"
$username = $_POST['username']; // Retrieves value from input field with name="username"
$password = $_POST['password']; // Retrieves value from input field with name="password"
```

It's important to note that both `$_GET` and `$_POST` superglobals are arrays, and you can access their values using square brackets `[]` with the corresponding keys. Additionally, it's essential to validate and sanitize user input before using it in your application to prevent security vulnerabilities such as SQL injection and cross-site scripting (XSS) attacks.

3.3.3 PHP Cookie:

PHP cookie is a small piece of information which is stored at client browser. It is used to recognize the user.
Cookie is created at server side and saved to client browser. Each time when client sends request to the server, cookie is embedded with request. Such way, cookie can be received at the server side.

PHP setcookie() function

PHP setcookie() function is used to set cookie with HTTP response. Once cookie is set, you can access it by $_COOKIE superglobal variable.

Syntax:

bool setcookie (string $name [, string $value [, int $expire = 0 [, string $path
[, string $domain [, bool $secure = false [, bool $httponly = false]]]]]])

OR

Syntax:

setcookie(name, value, expire, path, domain, security);

Parameters: The setcookie() function requires six arguments in general
which are:

- Name: It is used to set the name of the cookie.
- Value: It is used to set the value of the cookie.
- Expire: It is used to set the expiry timestamp of the cookie after which the
- cookie can’t be accessed.
- Path: It is used to specify the path on the server for which the cookie will
- be available.
- Domain: It is used to specify the domain for which the cookie is available.
- Security: It is used to indicate that the cookie should be sent only if a
- secure HTTPS connection exists.

EXAMPLE PROGRAM

Cookie.php

```
<!DOCTYPE html>
<?php
 setcookie("Auction_Item", "Luxury Car", time() + 2 * 24 * 60 *
60);
?>
<html>
<body>
 <?php
 echo "cookie is created."
 ?>
 <p>
 <strong>Note:</strong>
 You might have to reload the
 page to see the value of the cookie.
 </p>
</body>
</html>
```

Output:

```
Cookie is created.
Note: You might have to reload the page to see the value of the cookie
```

Checking Whether a Cookie Is Set Or Not: It is always advisable to check whether a cookie is set or not before accessing its value. Therefore to check whether a cookie is set or not, the PHP isset() function is used. To check whether a cookie “Auction_Item” is set or not, the isset() function is executed as follows:

Example: This example describes checking whether the cookie is set or not.

EXAMPLE PROGRAM 2

```
<!DOCTYPE html>
<?php
 setcookie("Auction_Item", "Luxury Car", time() + 2 * 24 *
60 * 60);
?>
<html>
<body>
 <?php
 if (isset($_COOKIE["Auction_Item"]))
 {
 echo "Auction Item is a " .
$_COOKIE["Auction_Item"];
 }
 else
 {
 echo "No items for auction.";
 }
 ?>
 <p>
 <strong>Note:</strong>
 You might have to reload the page
 to see the value of the cookie.
 </p>
</body>
</html>
```

Output:

```
No items for auction.
You might have to reload the page to see the value of the cookie.
```

Deleting Cookies: The setcookie() function can be used to delete a cookie.
For deleting a cookie, the setcookie() function is called by passing the cookie name and other arguments or empty strings but however this time, the expiration date is required to be set in the past. To delete a cookie named “Auction_Item”, the following code can be executed.

Example: This example describes the deletion of the cookie value.

```
<!DOCTYPE html>
<?php
 setcookie("Auction_Item", "Luxury Car", time() + 2 * 24 *
60 * 60);
?>
<html>
<body>
 <?php
 setcookie("Auction_Item", "", time() - 60);
 ?>
 <?php
 echo "cookie is deleted"
 ?>
 <p>
 <strong>Note:</strong>
 You might have to reload the page
 to see the value of the cookie.
 </p>
</body>
</html>
```

Output:

```
Cookie is deleted.
Note: You might have to reload the page to see the value of the cookie
```

3.3.4 PHP Session

PHP session is used to store and pass information from one page to another temporarily (until user close the website).

PHP session technique is widely used in shopping websites where we need to store and pass cart information e.g. username, product code, product name, product price etc from one page to another.

PHP session creates unique user id for each browser to recognize the user and avoid conflict between multiple browsers. Or A session means the duration spent by a Web user from the time logged in to the time logged out—during this time the user can view protected content. Protected content means the information that is not open to everyone (like your e-mail inbox). The beauty of a session is that it keeps the login credentials of users until they log out, even if they move from one Web page to another, in the same Web service, of course.

PHP session_start() function

PHP session_start() function is used to start the session. It starts a new or resumes existing session. It returns existing session if session is created already. If session is not available, it creates and returns new session.

Syntax

bool session_start (void)

Example

session_start();

PHP $_SESSION

PHP $_SESSION is an associative array that contains all session variables. It is used to
set and get session variable values.

Example: Store information

1. $_SESSION["user"] = "Sachin";

Example: Get information

1. echo $_SESSION["user"];

PHP Session Example

session1.php

```
<?php
session_start();
?>
<html>
<body>
<?php
$_SESSION["user"] = "Sachin";
echo "Session information are set successfully.<br/>";
?>
<a href="session2.php">Visit next page</a>
</body>
</html>
```

session2.php

```
<?php
session_start();
?>
<html>
<body>
<?php
echo "User is: ".$_SESSION["user"];
?>
</body>
</html>
```

PHP Session Counter Example

sessioncounter.php

```
<?php
 session_start();

 if (!isset($_SESSION['counter'])) {
 $_SESSION['counter'] = 1;
 } else {
 $_SESSION['counter']++;
 }
 echo ("Page Views: ".$_SESSION['counter']);
?>
```

PHP Destroying Session

PHP session_destroy() function is used to destroy all session variables completely.

session3.php

```
<?php
session_start();
session_destroy();
?>
```

3.3.5 PHP Servers:

1.WAMP: (Windows, Apache, MySQL, PHP)

This server works only on Windows operating system. It is an open source platform and uses the Apache web server. It also uses the MySQL relational database management system and PHP object-oriented scripting language. The important part of WAMP is Apache that is used to run a web server on windows.

2.LAMP: (Linux, Apache, MySQL, and PHP)
It is an open source platform and works on the Linux operating system. It uses Apache web server, MySQL relational database management system, and PHP object-oriented scripting language. Since this platform has four layers, it can also be called a LAMP stack. It is highly secured working with Linux OS. The LAMP is easy to code with PHP.

3.MAMP: (Mac, Apache, MySQL, PHP)
The full form of MAMP stands for Mac, Apache, Mysql, and PHP. MAMP is an open source platform and it works on Mac operating system. As the above local server, MAMP uses Apache web server, Mysql relational database management system, and PHP object-oriented language. It gives you all the tools that you run WordPress on your machine, for the purpose of development and testing. You can install this in Mac or Windows-based PC.

4.XAMPP: (Cross-Platform, Apache, MySQL, PHP and Perl)
The full form of XAMPP stands for Cross-platform, Apache, MariaDB(Mysql), PHP and Perl. It is one of the simplest and light-weight local servers that is used to test your website locally. It is an open source platform.

3.3.6 phpMyAdmin:

phpMyAdmin is a free and open-source tool written in PHP, intended to handle the administration of MySQL or MariaDB databases over the web. It provides a graphical interface for users to perform various database management tasks, such as creating databases, tables, and executing SQL queries, without needing to use the command-line interface.

key features:

Web-Based Interface: phpMyAdmin is accessed through a web browser, allowing users to manage their databases from any location with internet access.

Database Management: Users can create, delete, and modify databases, tables, columns, indexes, and other database objects through the phpMyAdmin interface.
SQL Execution: phpMyAdmin enables users to execute SQL queries directly within the interface. Users can write custom SQL queries to retrieve, insert, update, or delete data from their databases.

Import and Export: It provides functionality for importing and exporting database files in various formats, including SQL, CSV, and XML. This allows users to transfer data between different database systems or backup their databases for safekeeping.

User Management: phpMyAdmin allows administrators to create and manage database users and their privileges. Users can be assigned specific permissions, such as read-only access or full administrative rights, to control access to database resources.

Database Maintenance: It offers tools for optimizing database tables, repairing corrupted tables, and performing other maintenance tasks to ensure the efficient operation of the database.

Multi-Language Support: phpMyAdmin supports multiple languages, making it accessible to users around the world. Users can customize the interface language to suit their preferences.

iv) AJAX

3.4.1 AJAX Introduction:

- AJAX = Asynchronous JavaScript and XML.
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.
- Classic web pages, (which do not use AJAX) must reload the entire page if the content should change.
- Examples of applications using AJAX: Google Maps, Gmail, Youtube, and Facebook tabs.

key components and concepts associated with AJAX:

Asynchronous Communication: AJAX enables asynchronous communication between the client (typically a web browser) and the server. This means that the client can send requests to the server and continue to interact with the web page without waiting for a response. When the server responds, the client can handle the response without disrupting the user's experience.

JavaScript: JavaScript is the primary language used to implement AJAX functionality on the client-side. JavaScript allows developers to send requests to the server, handle responses, and update the web page dynamically based on the retrieved data.

XMLHttpRequest (XHR) Object: The XMLHttpRequest object, commonly abbreviated as XHR, is a browser API that allows JavaScript to make HTTP requests to the server without reloading the entire web page. This object provides methods and properties for sending and receiving data asynchronously.

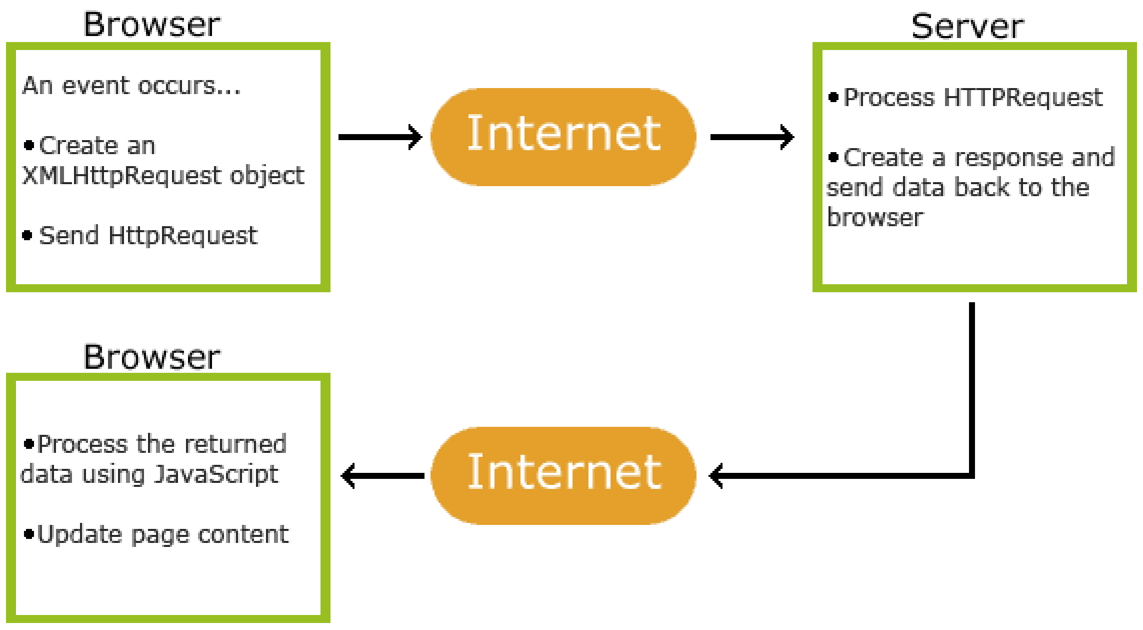
Server-Side Technologies: On the server-side, any web technology capable of processing HTTP requests and generating responses can be used in conjunction with AJAX. This includes server-side scripting languages like PHP, Python, Ruby, or server-side frameworks such as Node.js, Django, or Ruby on Rails.

Data Formats: While AJAX originally used XML (hence the name "Asynchronous JavaScript and XML") for data interchange, modern AJAX applications commonly use JSON (JavaScript Object Notation) due to its simplicity and lightweight nature. JSON is easy to parse and serialize using JavaScript, making it a preferred format for exchanging data between the client and server.

Dynamic Content Updating: One of the primary benefits of AJAX is its ability to update parts of a web page dynamically without requiring a full page reload. This enables developers to create more responsive and interactive user interfaces, where changes can be made to the content based on user interactions or data retrieved from the server.

Common Use Cases: AJAX is used in various web applications for features such as form submissions without page reloads, real-time updates (e.g., chat applications), auto-complete search suggestions, infinite scrolling, and more.

3.4.2 AJAX Working


Browser
An event occurs...
• Create an XMLHttpRequest object
• Send HttpRequest
Internet
Server
• Process HTTPRequest
• Create a response and send data back to the browser
Internet
Browser
• Process the returned data using JavaScript
• Update page content

1. An event occurs in a web page (the page is loaded, a button is clicked)
2. An XMLHttpRequest object is created by JavaScript
3. The XMLHttpRequest object sends a request to a web server
4. The server processes the request
   The server sends a response back to the web page
5. The response is read by JavaScript
6. Proper action (like page update) is performed by JavaScriptAJAX is Based on Internet Standards

AJAX is based on internet standards, and uses a combination of:

- XMLHttpRequest object (to exchange data asynchronously with a server)
- JavaScript/Document Object Model (DOM) (to display/interact with the information)
- CSS (to style the data)
- XML (often used as the format for transferring data)
- AJAX applications are browser- and platform-independent.

AJAX is a developer's dream, because you can:

- Update a web page without reloading the page
- Request data from a server - after the page has loaded
- Receive data from a server - after the page has loaded
- Send data to a server - in the background

IMPLEMENTATION

4.1 main.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Welcome to ARTWAGON</title>
</head>
<body>
    <div class="banner">
        <h1>Welcome to ARTWAGON: Virtual Art Gallery</h1>
    </div>
    <div class="container">
        <div class="button-container">
            <a href="signup_login.html" class="button">User Signup / Login</a>
            <a href="artist_signup_login.php" class="button">Artist Signup / Login</a>
            <a href="admin.php" class="button">Admin</a>
        </div>
        <br>
        <div class="about-text">
            <p>ArtWagon is your gateway to a virtual world of art, where creativity knows no
bounds. Immerse yourself in a diverse collection of artworks from talented artists around the
globe.
            Discover paintings, sculptures, digital art, and more, all curated to inspire and captivate
your imagination. Whether you're an artist seeking recognition or an art enthusiast in search of
inspiration, ArtWagon welcomes you to explore, connect, and create.
            At ArtWagon, we believe that art has the power to transform lives and connect people
across cultures and borders.</p>
        </div>
    </div>
```

```
</body>
</html>
```

4.2 signup_login.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ArtWagon: Login/Sign Up</title>
</head>
<body>
    <div class="container">
        <div class="form-container">
            <h2>Sign Up</h2>
            <form action="signup.php" method="post">
                <input type="text" name="name" placeholder="Name" required>
                <input type="email" name="email" placeholder="Email" required>
                <input type="tel" name="phone" placeholder="Phone Number" required>
                <input type="date" name="dob" required>
                <input type="password" name="password" placeholder="Password" required>
                <input type="password" name="confirm_password" placeholder="Confirm Password"
required>
                <button type="submit">Sign Up</button>
            </form>
        </div>
        <div class="form-container">
            <h2>Login</h2>
            <form action="login.php" method="post">
                <input type="email" name="email" placeholder="Email" required>
                <input type="password" name="password" placeholder="Password" required>
                <button type="submit">Login</button>
            </form>
        </div>
    </div>
    <p class="signup-text">Don't have an account? <a href="signup_login.html">Sign
Up</a></p>
</body>
</html>
```

4.3 index.php:

```
<?php
session_start();
?>
<!DOCTYPE html>
<html lang="en">
<body>
    <div class="header">
        <h1>ARTWAGON: VIRTUAL ART GALLERY</h1>
        <div class="navbar">
            <a href="index.php">Home</a>
            <div class="dropdown">
                <a href="#">Artists</a>
                <div class="dropdown-content">
                    <?php
                    // Your database connection parameters
                    $servername = "localhost";
                    $username = "root";
                    $password = "";
                    $dbname = "artg_users";

                    // Create connection
                    $conn = mysqli_connect($servername, $username, $password, $dbname);
                    // Check connection
                    if (!$conn) {
                        die("Connection failed: " . mysqli_connect_error());
                    }
                    // Query to get artist names
                    $sql = "SELECT name FROM artists_name";
                    $result = mysqli_query($conn, $sql);
                    // Check if there are artists
                    if (mysqli_num_rows($result) > 0) {
                        // Output data of each row
                        while ($row = mysqli_fetch_assoc($result)) {
                            echo '<a href="display_art_by_name.php?artist_name=' .
urlencode($row['name']) . '" target="bottom_frame">' . $row['name'] . '</a>';
                        }
                    } else {
                        echo "0 artists found";
                    }
                    // Close connection
                    mysqli_close($conn);
                    ?>
                </div>
            </div>
            <div class="dropdown">
                <a href="#">Art Types</a>
                <div class="dropdown-content">
                    <a href="display_art.php?art_type=SCULPTURES"
target="bottom_frame">SCULPTURES</a>
                    <a href="display_art.php?art_type=STREET%20ARTS"
target="bottom_frame">STREET ARTS</a>
                    <a href="display_art.php?art_type=CONCEPTUAL%20ARTS"
target="bottom_frame">CONCEPTUAL ARTS</a>
```

```
          <a href="display_art.php?art_type=OIL%20PAINTINGS"
target="bottom_frame">OIL PAINTINGS</a>
          <a href="display_art.php?art_type=SERIGRAPHS"
target="bottom_frame">SERIGRAPHS</a>
          <!-- Add more art types as needed -->
        </div>
      </div>
      <div class="dropdown">
        <a href="#">Exhibition Rooms</a>
        <div class="dropdown-content">
          <?php
          // Your database connection parameters
          $servername = "localhost";
          $username = "root";
          $password = "";
          $dbname = "artg_users";
          // Create connection
          $conn = mysqli_connect($servername, $username, $password, $dbname);
          // Check connection
          if (!$conn) {
            die("Connection failed: " . mysqli_connect_error());
          }
          // Query to get artist names
          $sql = "SELECT name FROM artists_name";
          $result = mysqli_query($conn, $sql);

          // Check if there are artists
          if (mysqli_num_rows($result) > 0) {
            // Output data of each row
            while ($row = mysqli_fetch_assoc($result)) {
                    echo '<a href="exhibition_room.php?artist_name=' . urlencode($row['name']) .
'" target="__blank">' . $row['name'] . '</a>';
                }
            } else {
                echo "0 artists found";
            }
            // Close connection
            mysqli_close($conn);
            ?>
        </div>
    </div>
    <a href="about.html" target="bottom_frame">About</a>
    <!-- Add more navigation links as needed -->
    <?php
    // Start the session
    // Check if user is logged in
    if(isset($_SESSION['email']) && isset($_SESSION['name'])) {
        // If logged in, display username and email and logout button
        echo '<div style="position: absolute; top: 4px; right: 3px; color: #fff;">';
        echo 'Welcome, </br>' . $_SESSION['name'] . ' </br> ';
        echo 'Email: ' . $_SESSION['email'] . ' </br> ';
        echo '<a href="logout.php" style="color: #fff; text-decoration: none;">Logout</a>';
        echo '</div>';
    }
    ?>
  </div>
</div>
<div class="bottom-frame">
  <iframe src="display_art.php" name="bottom_frame" frameborder="0"></iframe>
</div>
<!-- Display response and message form -->
```

```
    <div class="response">
        <?php
        // Display user information and logout link if logged in
        if(isset($_SESSION['email']) && isset($_SESSION['name'])) {
            // Connect to database
            $servername = "localhost";
            $username = "root";
            $password = "";
            $dbname = "artg_users";
            $conn = mysqli_connect($servername, $username, $password, $dbname);
            // Check connection
            if (!$conn) {
                die("Connection failed: " . mysqli_connect_error());
            }
            // Query to get response from users table
            $email = $_SESSION['email'];
            $response_query = "SELECT response,from_artist,art_id FROM users WHERE
email='$email'";
            $response_result = mysqli_query($conn, $response_query);
            if (mysqli_num_rows($response_result) > 0) {
                $response_row = mysqli_fetch_assoc($response_result);
                #echo 'Response: ' . $response_row['response'];
                #echo $response_row["from_artist"];
            }
            // Close connection
            mysqli_close($conn);
        }
        ?>
    </div>
    <div class="message-form">
```

```
        <?php echo 'Response from Artist: '. $response_row['from_artist']. ' is: ' .
$response_row['response']; ?>
        <form action="send_message.php" method="POST">
            <input type="hidden" name="from_art" value="<?php echo
$response_row['from_artist']; ?>">
            <input type="hidden" name="from_art_id" value="<?php echo $response_row['art_id'];
?>">
            <input type="text" name="message" placeholder="Enter your message">
            <input type="submit" value="Send Message">
        </form>
    </div>
</body>
</html>
```

4.4 artist_signup_login.php:

```
<!DOCTYPE html>
<html lang="en">
</head>
<body>
    <h2>Artist Signup/Login</h2>

    <!-- Signup Form -->
    <div>
        <center><h3>Signup</h3></center>
        <form id="registerForm" action="artist_signup.php" method="POST"
enctype="multipart/form-data">
            <input type="text" name="name" placeholder="Name" required><br>
            <input type="email" name="email" placeholder="Email" required><br>
            <input type="text" name="mobile" placeholder="Mobile" required><br>
            <input type="date" name="dob" placeholder="Date of Birth" required><br>
            <input type="password" name="password" placeholder="Password" required><br>
            <input type="submit" value="Signup">
        </form>
    </div>

    <!-- Login Form -->
    <div>
        <center><h3>Login</h3></center>
        <form id="loginForm" action="artist_login.php" method="POST">
            <input type="email" name="email" placeholder="Email" required><br>
            <input type="password" name="password" placeholder="Password" required><br>
            <input type="submit" value="Login">
        </form>
    </div>
</body>
</html>
```

4.5 artist_index.php:

```
<?php
// Start session
session_start();
// Check if user is logged in
if (isset($_SESSION['user_name'])) {
    $welcome_message = "Welcome, " . $_SESSION['user_name'] . "!";
    $user_email = $_SESSION['user_email'];
    $user_table_name = $_SESSION['user_table_name']; // Get the table name from session
} else {
    // Redirect to login page if user is not logged in
    header("Location: artist_login.php");
    exit();
}

// Function to escape special characters to prevent SQL injection
function escape_string($conn, $string) {
    return mysqli_real_escape_string($conn, $string);
}

// Establish database connection
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "artg_users";

$conn = mysqli_connect($servername, $username, $password, $dbname);
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// Fetch all image data from user's table
$sql_fetch_data = "SELECT * FROM $user_table_name";
$result_data = mysqli_query($conn, $sql_fetch_data);
?>

<!DOCTYPE html>
<html lang="en">
<body>
    <h1>Welcome to the Art Gallery</h1>
    <p>Email: <?php echo $user_email; ?></p>

    <h2>Artworks</h2>
    <table>
        <tr>
            <th>Art Name</th>
            <th>Art Image</th>
            <th>Art Type</th>
            <th>Description</th>
            <th>Date of Art</th>
            <th>Describal</th>
            <th>Price</th>
            <th>Comments</th>
            <th>Response</th>
            <th>Actions</th>
        </tr>
        <?php
        if (mysqli_num_rows($result_data) > 0) {
            while ($row = mysqli_fetch_assoc($result_data)) {
                echo '<tr>';
                echo '<td>' . $row['art_name'] . '</td>';
                echo '<td><img src="' . $row['art_image'] . '" alt="Art Image"></td>';
                echo '<td>' . $row['art_type'] . '</td>';
                echo '<td>' . $row['art_description'] . '</td>';
                echo '<td>' . $row['date_of_art'] . '</td>';
                echo '<td>' . $row['describal'] . '</td>';
                echo '<td>$' . $row['price'] . '</td>';
                echo '<td>' . $row['comments'] . '</td>';
                echo '<td>';
                echo '<form action="send_response.php" method="POST">';
                echo '<input type="hidden" name="from_user" value="' . escape_string($conn,
$row['from_user']) . '">';
                echo '<input type="hidden" name="artist_mail" value="' . $user_email . '">';
                echo '<input type="hidden" name="artist_id" value="' . $row['id'] . '">';
                echo '<input type="text" name="response" placeholder="Enter your response">';
                echo '<input type="submit" value="Send Response">';
                echo '</form>';
                echo '</td>';
                echo '<td>';
                echo '<form action="update_process.php" method="POST">';
                echo '<input type="hidden" name="art_id" value="' . $row['id'] . '">';
                echo '<input type="submit" value="Update">';
                echo '</form>';
                echo '<form action="delete_art.php" method="POST">';
                echo '<input type="hidden" name="art_id" value="' . $row['id'] . '">';
                echo '<input type="submit" value="Delete">';
                echo '</form>';
                echo '</td>';
                echo '</tr>';
            }
        } else {
            // No art data available
            echo '<tr><td colspan="10">No artworks available. Uploading soon!</td></tr>';
        }

        // Close connection
        mysqli_close($conn);
        ?>
    </table>

    <h2>Add New Art</h2>
    <form action="add_art.php" method="POST" enctype="multipart/form-data">
        <!-- Your form fields here -->
    </form>
</body>
</html>
```

4.6 display_art.php:

```
<!DOCTYPE html>
<html lang="en">
<body>
  <div class="header">
    <center><h1>ARTWAGON: VIRTUAL ART GALLERY</h1></center>
  </div>
  <div class="container">
    <div class="artworks">
      <?php
      // Check if art type is provided in the URL
      if(isset($_GET['art_type'])) {
        // Get the art type from the URL
        $art_type = $_GET['art_type'];
        $servername = "localhost";
        $username = "root";
        $password = "";
        $dbname = "artg_users";
        // Create connection
        $conn = mysqli_connect($servername, $username, $password, $dbname);
        // Check connection
        if (!$conn) {
          die("Connection failed: " . mysqli_connect_error());
        }
        // Query the artists_name table to get artist name and mobile number
        $sql = "SELECT name, mobile FROM artists_name";
        $result = mysqli_query($conn, $sql);
        echo "<center><h2>Artworks by Selected Art Type</h2></center>";

        if (mysqli_num_rows($result) > 0) {
          // Loop through each artist
```

```
            while ($row = mysqli_fetch_assoc($result)) {
                // Form the table name
                $table_name = preg_replace('/[^a-zA-Z0-9_]/', '', $row['name'] . "_" .
$row['mobile']);

                // Query the artist's table to retrieve artworks of the selected art type
                $art_query = "SELECT * FROM $table_name WHERE art_type = '$art_type'";
                $art_result = mysqli_query($conn, $art_query);

                if (mysqli_num_rows($art_result) > 0) {
                    // Display artworks
                    while ($art_row = mysqli_fetch_assoc($art_result)) {
                        echo '<div class="artwork">';
                        echo '<center><h3>' . $art_row['art_name'] . '</h3></center>';
                        echo '<img src="' . $art_row['art_image'] . '" alt="' . $art_row['art_name'] .
'">';
                        echo '<center><p>Artist: ' . $row['name'] . '</p></center>'; // Display artist's
name
                        echo '<center><p>Description: ' . $art_row['art_description'] .
'</p></center>'; // Display artwork description
                        // Display other attributes as needed
                        echo '</div>';
                    }
                } else {
                    echo "<p>No artworks found for artist: {$row['name']}</p>";
                }
            }
        } else {
            echo "No artists found.";
        }
        // Close connection
```

```
                mysqli_close($conn);
            }  else {
                echo '<marquee behavior="scroll" direction="left" scrollamount="19" style="width:
100%;">'; // Adjust scrollamount as needed
                // Array of art types and corresponding prefix for images
                $artTypes = ['SG', 'OILP', 'STA', 'Sculp', 'VCA'];
                // Loop through each art type
                foreach($artTypes as $type) {
                    // Loop through each image for the art type
                    for ($i = 1; $i <= 5; $i++) {
                        $imagePath = "images/{$type}{$i}.jpg"; // Forming the image path
                        // Add a crossed image at the beginning

                        echo '<img src="'.$imagePath.'" alt="Art Image" style="width: 200px; height:
250; margin-right: 50px;">'; // Adjust width, height, and margin as needed
                    }
                }
                echo '</marquee>';
            }
            ?>
        </div>
    </div>
</body>
</html>
```

4.7 display_art_by_name.php:

```
<!DOCTYPE html>
<html lang="en">
<body>
    <div class="header">
        <center><h1>ARTWAGON: VIRTUAL ART GALLERY</h1></center>
    </div>
```

```
<div class="container">
    <div class="artworks">
        <?php
        // Check if artist name is provided in the URL
        if(isset($_GET['artist_name'])) {
            // Get the artist name from the URL
            $artist_name = $_GET['artist_name'];

            // Your database connection parameters
            $servername = "localhost";
            $username = "root";
            $password = "";
            $dbname = "artg_users";

            // Create connection
            $conn = mysqli_connect($servername, $username, $password, $dbname);

            // Check connection
            if (!$conn) {
                die("Connection failed: " . mysqli_connect_error());
            }
            // Query the artists_name table to get artist's mobile number
            $sql = "SELECT mobile FROM artists_name WHERE name='$artist_name'";
            $result = mysqli_query($conn, $sql);
            if (mysqli_num_rows($result) > 0) {
                // Get the mobile number of the artist
                $row = mysqli_fetch_assoc($result);
                $mobile = $row['mobile'];

                // Form the table name
                $table_name = preg_replace('/[^a-zA-Z0-9_]/', '', $artist_name . "_" . $mobile);
            // Query the artist's table to retrieve artworks
            $art_query = "SELECT * FROM $table_name LIMIT 2"; // Limit to 2 artworks
            $art_result = mysqli_query($conn, $art_query);
            if (mysqli_num_rows($art_result) > 0) {
                // Display artworks
                while ($art_row = mysqli_fetch_assoc($art_result)) {
                    echo '<div class="artwork">';
                    echo '<img src="' . $art_row['art_image'] . '" alt="' . $art_row['art_name'] . '">';
                    echo '<div class="details">';
                    echo '<h3>' . $art_row['art_name'] . '</h3>';
                    echo '<p>Description: ' . $art_row['art_description'] . '</p>';
                    echo '<p>Describal: ' . $art_row['describal'] . '</p>';
                    echo '<p>Date of Art: ' . $art_row['date_of_art'] . '</p>';
                    echo '</div>'; // End .details
                    echo '</div>'; // End .artwork
                }
            } else {
                echo "<p>No artworks found for artist: {$artist_name}</p>";
            }
        } else {
            echo "Artist not found.";
        }

        // Close connection
        mysqli_close($conn);
    } else {
        echo "Artist name not provided.";
    }
    ?>
  </div>
</div>
```

```
</body>
</html>
```

4.8 delete_art.php:

```
<?php
// Start session
session_start();

// Check if user is logged in
if (!isset($_SESSION['user_name'])) {
    // Redirect to login page if user is not logged in
    header("Location: artist_login.php");
    exit();
}

// Check if art_id is provided
if (isset($_POST['art_id'])) {
    // Get the art_id from GET parameters
    $art_id = $_POST['art_id'];

    // Establish database connection
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "artg_users";

    $conn = mysqli_connect($servername, $username, $password, $dbname);
    if (!$conn) {
        die("Connection failed: " . mysqli_connect_error());
    }
    // Construct table name
    $user_table_name = $_SESSION['user_table_name'];

    // Delete the artwork from the database
    $sql_delete_art = "DELETE FROM $user_table_name WHERE id=$art_id";

    if (mysqli_query($conn, $sql_delete_art)) {
        echo "Artwork deleted successfully!";
    } else {
        echo "Error deleting artwork: " . mysqli_error($conn);
    }

    // Close connection
    mysqli_close($conn);
} else {
    echo "Art ID Not provided.";
}
?>
```

4.9 update_db.php:

```
<?php
// Start session
session_start();

// Check if user is logged in
if (!isset($_SESSION['user_name'])) {
    // Redirect to login page if user is not logged in
    header("Location: artist_login.php");
    exit();
}
// Check if form is submitted
```

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Get user table name from session
    $user_table_name = $_SESSION['user_table_name'];
    // Establish database connection
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "artg_users";

    $conn = mysqli_connect($servername, $username, $password, $dbname);
    if (!$conn) {
        die("Connection failed: " . mysqli_connect_error());
    }
    // Check if the required POST variables are set
    if (isset($_POST['art_id'], $_POST['art_name'], $_POST['art_type'],
$_POST['art_description'], $_POST['date_of_art'], $_POST['describal'], $_POST['price'],
$_POST['comments'])) {
        // Retrieve form data
        $art_id = $_POST['art_id'];
        $art_name = $_POST['art_name'];
        $art_type = $_POST['art_type'];
        $art_description = $_POST['art_description'];
        $date_of_art = $_POST['date_of_art'];
        $describal = $_POST['describal'];
        $price = $_POST['price'];
        $comments = $_POST['comments'];

        // Update the artwork in the database
        $sql_update_art = "UPDATE $user_table_name
                          SET art_name='$art_name', art_type='$art_type',
                              art_description='$art_description', date_of_art='$date_of_art',
                describal='$describal', price='$price', comments='$comments'
              WHERE id=$art_id";
    if (mysqli_query($conn, $sql_update_art)) {
        echo "Artwork updated successfully!";
    } else {
        echo "Error updating artwork: " . mysqli_error($conn);
    }
  } else {
    echo "Required POST variables are not set!";
  }
  // Close connection
  mysqli_close($conn);
} else {
  echo "Invalid request!";
}
?>
```

4.10.1 admin_index.php:

```
<!DOCTYPE html>
<html lang="en">
<body>
  <?php
  session_start();
  // Check if form is submitted
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Check if username and password are correct
    if ($_POST['username'] === 'admin' && $_POST['password'] === 'admin@123') {
        // Authentication successful, set session variables
        $_SESSION['loggedin'] = true;
    } else {
        // Authentication failed, redirect back to admin.php
        header("Location: admin.php");
        exit;
    }
  }
  // Check if the user is logged in
  if (!isset($_SESSION['loggedin']) || $_SESSION['loggedin'] !== true) {
    // User is not logged in, redirect to admin.php
    header('Location: admin.php');
    exit;
  }
  // Your database connection parameters
  $servername = "localhost";
  $username = "root";
  $password = "";
  $dbname = "artg_users";
  // Create connection
  $conn = mysqli_connect($servername, $username, $password, $dbname);
  // Check connection
  if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
  }
  // Fetch and display records from the users table
  $sql_users = "SELECT * FROM users";
  $result_users = mysqli_query($conn, $sql_users);
  echo "<h2>Users</h2>";
  echo "<table>";
  echo "<tr><th>ID</th><th>Name</th><th>Email</th><th>Mobile</th><th>DOB</th></tr>"
  while ($row_users = mysqli_fetch_assoc($result_users)) {
    echo "<tr>";
    echo "<td>{$row_users['id']}</td>";
    echo "<td>{$row_users['name']}</td>";
    echo "<td>{$row_users['email']}</td>";
```

```
        echo "<td>{$row_users['mobile']}</td>";
        echo "<td>{$row_users['dob']}</td>";
        echo "</tr>";
    }
    echo "</table>";
    // Fetch and display records from the artists_name table
    $sql_artists = "SELECT * FROM artists_name";
    $result_artists = mysqli_query($conn, $sql_artists);
    echo "<h2>Artists</h2>";
    while ($row_artists = mysqli_fetch_assoc($result_artists)) {
        echo "<h3>{$row_artists['name']}</h3>";
        echo "<table>";
        echo "<tr><th>ID</th><th>Art Name</th><th>Art Image</th><th>Art Type</th><th>Art
Description</th><th>Date of Art</th><th>Describal</th><th>Price</th></tr>";
        $table_name = preg_replace('/[^a-zA-Z0-9_]/', '',
"{$row_artists['name']}_{$row_artists['mobile']}");
        $sql_artist_details = "SELECT * FROM $table_name";
        $result_artist_details = mysqli_query($conn, $sql_artist_details);
        while ($row_artist_details = mysqli_fetch_assoc($result_artist_details)) {
            echo "<tr>";
            echo "<td>{$row_artist_details['id']}</td>";
            echo "<td>{$row_artist_details['art_name']}</td>";
            echo '<td><img src="' . $row_artist_details['art_image'] . '" alt="Art Image" style="max-
width: 100px; max-height: 100px;"></td>';
            echo "<td>{$row_artist_details['art_type']}</td>";
            echo "<td>{$row_artist_details['art_description']}</td>";
            echo "<td>{$row_artist_details['date_of_art']}</td>";
            echo "<td>{$row_artist_details['describal']}</td>";
            echo "<td>{$row_artist_details['price']}</td>";
            echo "</tr>";
        }
        echo "</table>";
    }
    // Close connection
    mysqli_close($conn);
    ?>
</body>
</html>
```

4.10.2 exhibition_room.php:

```
<!DOCTYPE html>
<html lang="en">
<body>
    <div class="header">
        <h1>ARTWAGON: VIRTUAL ART GALLERY</h1>
    </div>
    <div class="container">
        <div class="artwork-container">
            <div class="artwork-wrapper" id="artwork-wrapper">
            </div>
        </div>
    </div>
    <script>
    const artworksData = [
        <?php
        // Check if artist name is provided in the URL
        if(isset($_GET['artist_name'])) {
            // Get the artist name from the URL
            $artist_name = $_GET['artist_name'];

            // Your database connection parameters
            $servername = "localhost";
            $username = "root";
```

```
$password = "";
$dbname = "artg_users";
// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
// Query the artists_name table to get artist's mobile number
$sql = "SELECT mobile FROM artists_name WHERE name='$artist_name'";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) > 0) {
    // Get the mobile number of the artist
    $row = mysqli_fetch_assoc($result);
    $mobile = $row['mobile'];
    // Form the table name
    $name_mobile = preg_replace('/[^a-zA-Z0-9_]/', '', $artist_name . "_" . $mobile);
    // Query the database to fetch records from the specified table
    $sql = "SELECT * FROM $name_mobile";
    $result = mysqli_query($conn, $sql);

    // Check if there are records
    if (mysqli_num_rows($result) > 0) {
        // Output data of each row
        while ($row = mysqli_fetch_assoc($result)) {
            // Output art data
            echo json_encode($row) . ",";
        }
    } else {
        echo "No records found";
    }
```

```
        } else {
            echo "Artist not found.";
        }
        // Close connection
        mysqli_close($conn);
    } else {
        echo "Artist name not provided.";
    }
    ?>
];
const artworkWrapper = document.getElementById('artwork-wrapper');
let pauseAnimation = false; // Flag to control animation pause
function createArtworkElement(artworkData) {
const artwork = document.createElement('div');
artwork.classList.add('artwork');
artwork.innerHTML = `
    <img src="${artworkData.art_image}" alt="${artworkData.art_name}">
    <div class="text-overlay">
        <div class="text-content">
            <div>${artworkData.art_name}</div>
            <div>${artworkData.art_description}</div>
            <div>ID: ${artworkData.id}</div>
            <div>Art Type: ${artworkData.art_type}</div>
            <div>Date of Art: ${artworkData.date_of_art}</div>
            <div>Describal: ${artworkData.describal}</div>
            <div>Price: ${artworkData.price}</div>
        </div>
    </div>
    <form class="comment-form" action="comment_handling.php" method="POST">
        <div style="display: flex;">
            <input type="hidden" name="art_id" value="${artworkData.id}">
```

```
                <input type="hidden" name="art_name" value="${artworkData.art_name}">
                <input class="comment-input" type="text" name="comment" placeholder="Enter your
comment here" required>
                <input class="comment-submit" type="submit" value="Post Comment">
            </div>
        </form>
    `;

    // Attach event listeners for comment inputs
    const commentInput = artwork.querySelector('.comment-input');

    commentInput.addEventListener('focus', () => {
        pauseAnimation = true; // Pause animation
    });

    // Resume animation when cursor is removed from the comment box
    commentInput.addEventListener('blur', () => {
        pauseAnimation = false; // Resume animation
    });

    return artwork;
}
    function fadeInArtwork(artwork) {
        artwork.style.opacity = 1; // Set opacity to 1 for fade in effect
    }
    function startExhibition() {
        if (!pauseAnimation) {
            let currentIndex = 0; // Initialize currentIndex
            // Display the first artwork immediately
            const firstArtwork = createArtworkElement(artworksData[currentIndex]);
            artworkWrapper.appendChild(firstArtwork);
            fadeInArtwork(firstArtwork);
            // Start fading in and out artworks after a delay
            setInterval(() => {
                if (!pauseAnimation) {
                    currentIndex = (currentIndex + 1) % artworksData.length; // Update currentIndex
                    const currentArtwork = artworksData[currentIndex];
                    const artwork = createArtworkElement(currentArtwork);
                    artworkWrapper.innerHTML = ''; // Clear previous artworks
                    artworkWrapper.appendChild(artwork); // Append current artwork
                    fadeInArtwork(artwork); // Fade in the next artwork
                }
            }, 5000); // Change artwork every 5 seconds (5000 milliseconds)
        }
    }
    // Start the exhibition
    startExhibition();
</script>
</body>
</html>
```

4.10.3 comment_handling.php:

```
<?php
// Check if the form is submitted
session_start();
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // Retrieve data from the form
    $artId = $_POST['art_id'];
    $artName = $_POST['art_name'];
    $comment = $_POST['comment'];

    // Database connection parameters
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "artg_users";

    // Create connection
    $conn = new mysqli($servername, $username, $password, $dbname);

    // Check connection
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }

    // Retrieve name and mobile from artists_name table
    $artistNameQuery = "SELECT name, mobile FROM artists_name";
    $artistNameResult = $conn->query($artistNameQuery);

    if ($artistNameResult->num_rows > 0) {
        // Fetching each artist's name and mobile
        while ($row = $artistNameResult->fetch_assoc()) {
            // Form the table name as name_mobile
            $tableName = preg_replace('/[^a-zA-Z0-9_]/', '', $row['name'] . "_" . $row['mobile']);

            // Check if the table contains the artname
            $checkArtQuery = "SELECT * FROM $tableName WHERE art_name='$artName'";
            $result = $conn->query($checkArtQuery);
            $em = $_SESSION['email'];
            if ($result->num_rows > 0) {
                // Update the comment attribute in the corresponding table
                $updateQuery = "UPDATE $tableName SET comments='$comment', from_user='$em'
WHERE art_name='$artName'";
                echo "User is: ".$em;
                if ($conn->query($updateQuery) === TRUE) {
                    echo "Comment updated successfully for artist: " . $row['name'];
                } else {
                    echo "Error updating comment: " . $conn->error;
                }
                // Exit the loop if comment is updated
                break;
            }
        }
    } else {
        echo "No artists found.";
    }

    // Close connection
    $conn->close();
}
?>
```

4.10.4 send_message.php:

```
<?php
// Start session
session_start();

// Check if form data is received
if(isset($_POST['from_art']) && isset($_POST['message']) && isset($_POST['from_art_id'])) {
    // Form data
    $from_artist = $_POST['from_art'];
    #echo "From the ARTIST: ".$_POST['from_art'];
    #echo "From the ARTIST: ".$_POST['message'];
    $message = $_POST['message'];
    $art_id = $_POST['from_art_id'];
```

```php
echo "POST ID IS: ".$_POST['from_art_id'];
// Database connection parameters
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "artg_users";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// Query to search for name and mobile attributes from artists_name table
$search_query = "SELECT name, mobile FROM artists_name WHERE email='$from_artist'";
$search_result = mysqli_query($conn, $search_query);

if(mysqli_num_rows($search_result) > 0) {
    // Fetch name and mobile
    $row = mysqli_fetch_assoc($search_result);
    $artist_name = $row['name'];
    $mobile = $row['mobile'];

    // Form the table name
    $table_name = preg_replace('/[^a-zA-Z0-9_]/', '', $artist_name . "_" . $mobile);

    // Update comments field in the table
```

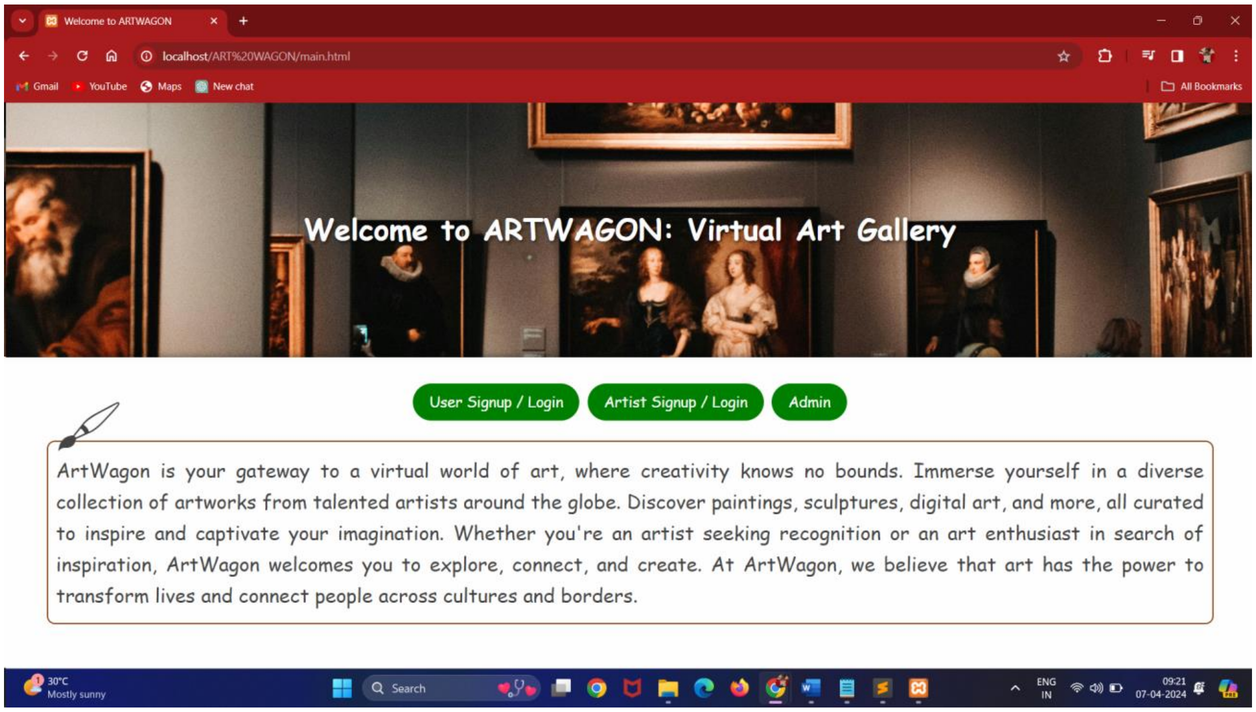
```
        $update_query = "UPDATE $table_name SET comments='$message' WHERE id
='$art_id'";
        echo "ID WHERE MESSG: ".$art_id;
        if(mysqli_query($conn, $update_query)) {
            echo "Message sent successfully!";
        } else {
            echo "Error updating record: " . mysqli_error($conn);
        }
    } else {
        echo "Artist not found!";
    }

    // Close connection
    mysqli_close($conn);
} else {
    echo "Form data not received!";
}
?>
```

4.10.5 logout.php:

```
<?php
// Start the session
session_start();

// Unset all of the session variables
$_SESSION = array();

// Destroy the session
session_destroy();

// Redirect to signup_login.html
header("Location: signup_login.html");
```

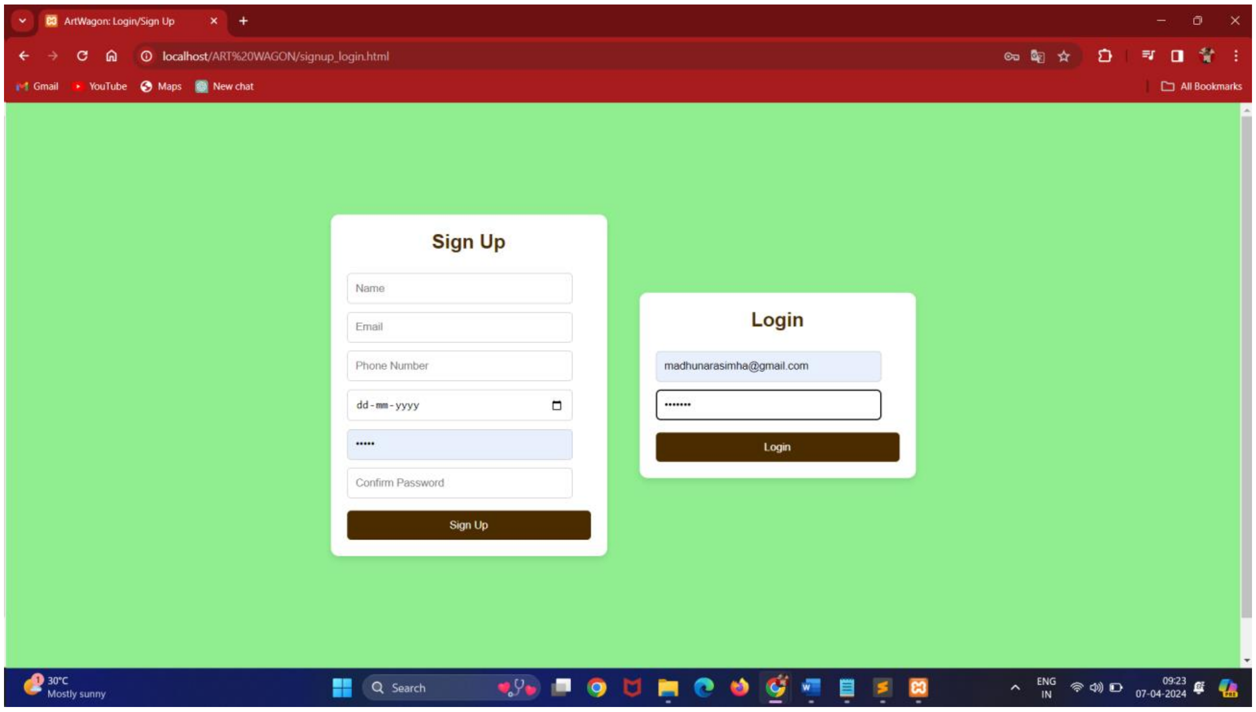
```
exit();
?>
```

SCREEN SHOTS

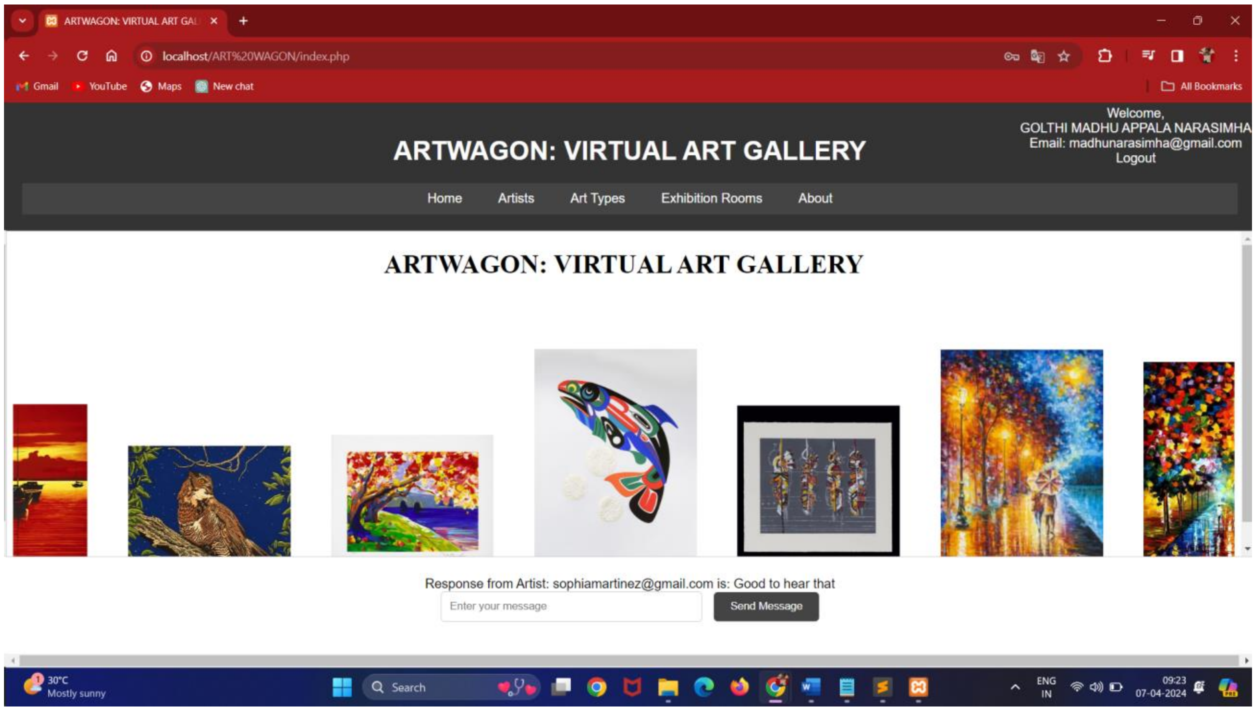

Welcome to ARTWAGON
localhost/ART%20WAGON/main.html
Welcome to ARTWAGON: Virtual Art Gallery
User Signup / Login
Artist Signup / Login
Admin

ArtWagon is your gateway to a virtual world of art, where creativity knows no bounds. Immerse yourself in a diverse collection of artworks from talented artists around the globe. Discover paintings, sculptures, digital art, and more, all curated to inspire and captivate your imagination. Whether you're an artist seeking recognition or an art enthusiast in search of inspiration, ArtWagon welcomes you to explore, connect, and create. At ArtWagon, we believe that art has the power to transform lives and connect people across cultures and borders.

Main Page



User Signup and Login Page



Welcome,
GOLTHI MADHU APPALA NARASIMHA
Email: madhunarasimha@gmail.com
Logout

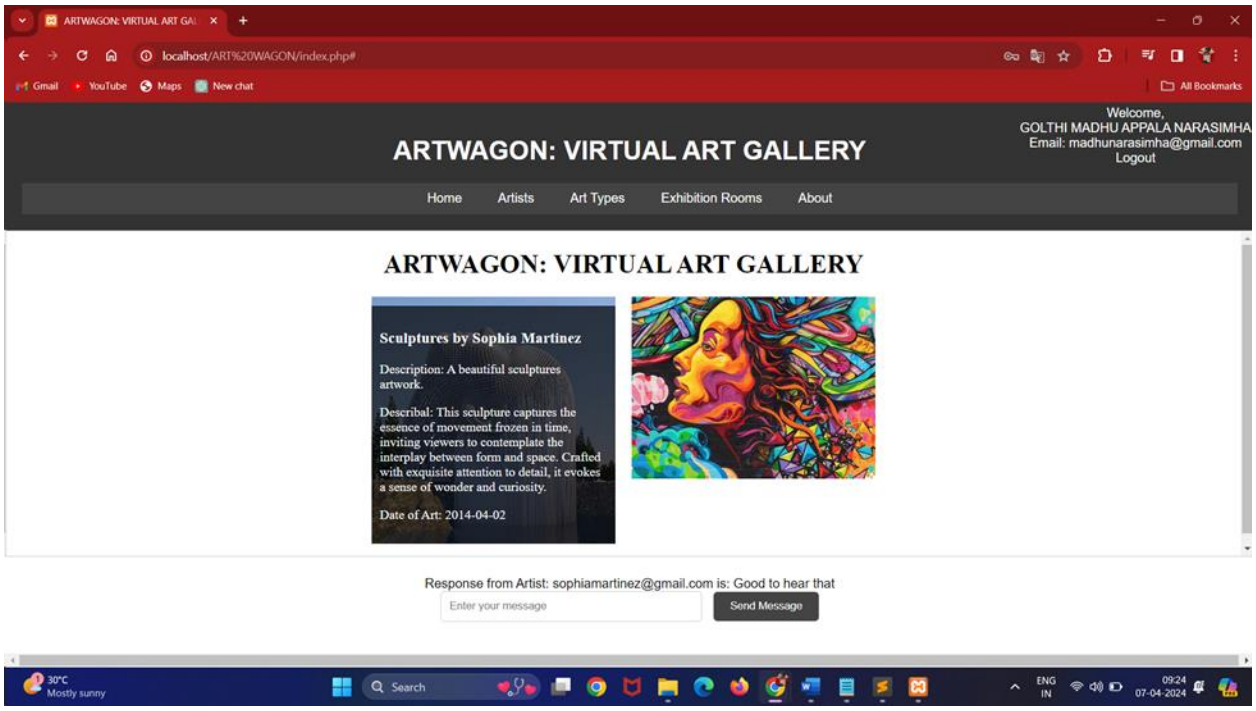
ARTWAGON: VIRTUAL ART GALLERY

ARTWAGON: VIRTUAL ART GALLERY

Response from Artist: sophiamartinez@gmail.com is: Good to hear that

Enter your message

Index Page



Welcome,
GOLTHI MADHU APPALA NARASIMHA
Email: madhunarasimha@gmail.com
Logout

ARTWAGON: VIRTUAL ART GALLERY

ARTWAGON: VIRTUAL ART GALLERY

Sculptures by Sophia Martinez

Description: A beautiful sculptures artwork.

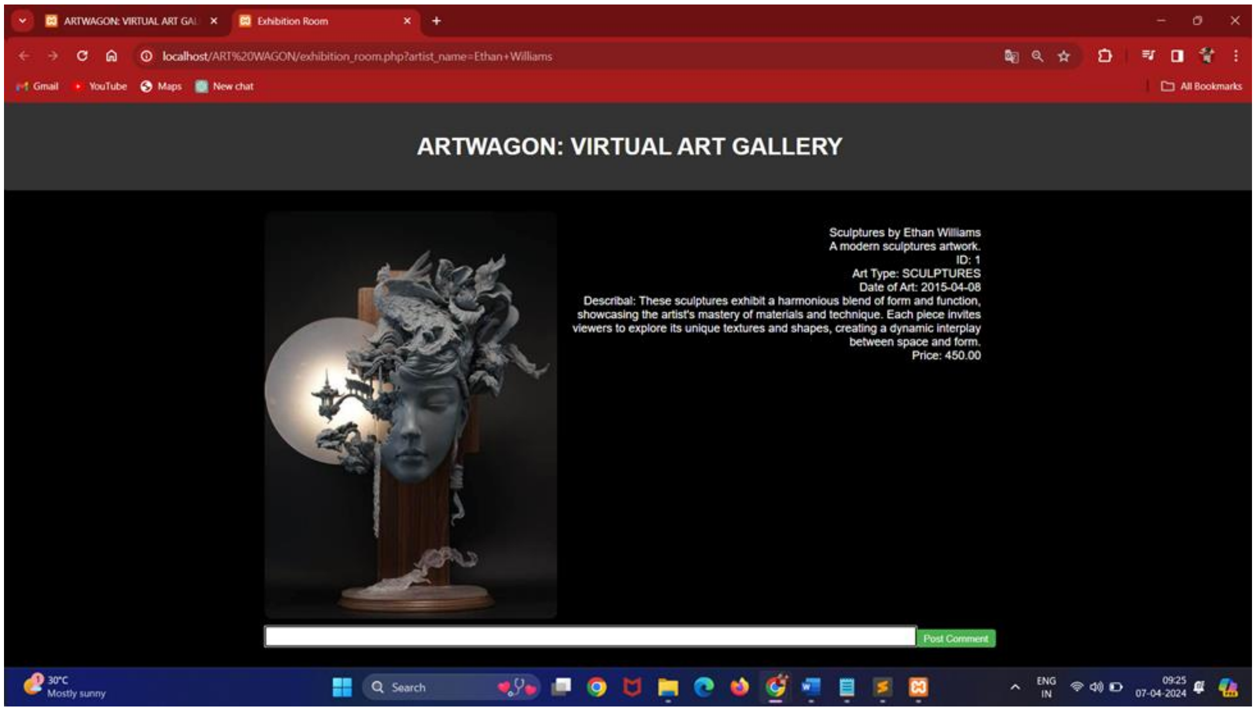
Describal: This sculpture captures the essence of movement frozen in time, inviting viewers to contemplate the interplay between form and space. Crafted with exquisite attention to detail, it evokes a sense of wonder and curiosity.

Date of Art: 2014-04-02

Response from Artist: sophiamartinez@gmail.com is: Good to hear that

Enter your message

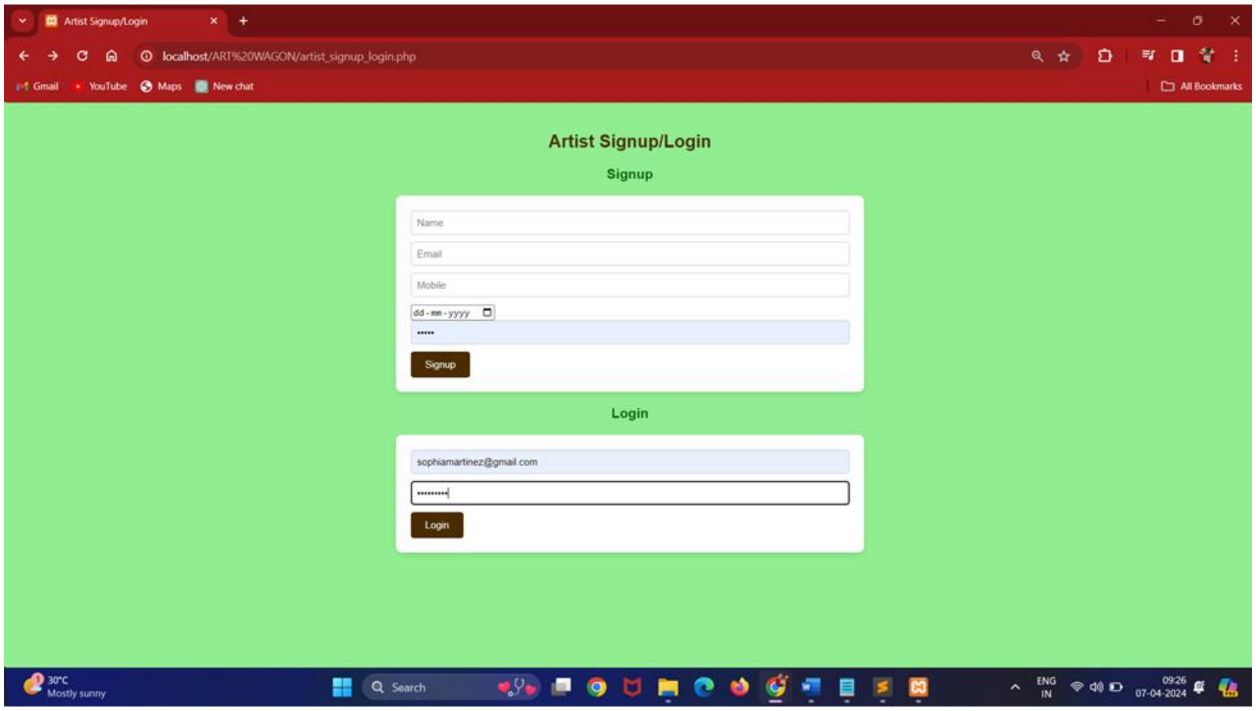
Art Display using Artist Name


ARTWAGON: VIRTUAL ART GAL
Exhibition Room
localhost/ART%20WAGON/exhibition_room.php?artist_name=Ethan+Williams
Post Comment

ARTWAGON: VIRTUAL ART GALLERY

Sculptures by Ethan Williams
A modern sculptures artwork.
ID: 1
Art Type: SCULPTURES
Date of Art: 2015-04-08
Describal: These sculptures exhibit a harmonious blend of form and function, showcasing the artist's mastery of materials and technique. Each piece invites viewers to explore its unique textures and shapes, creating a dynamic interplay between space and form.
Price: 450.00

Exhibition Room



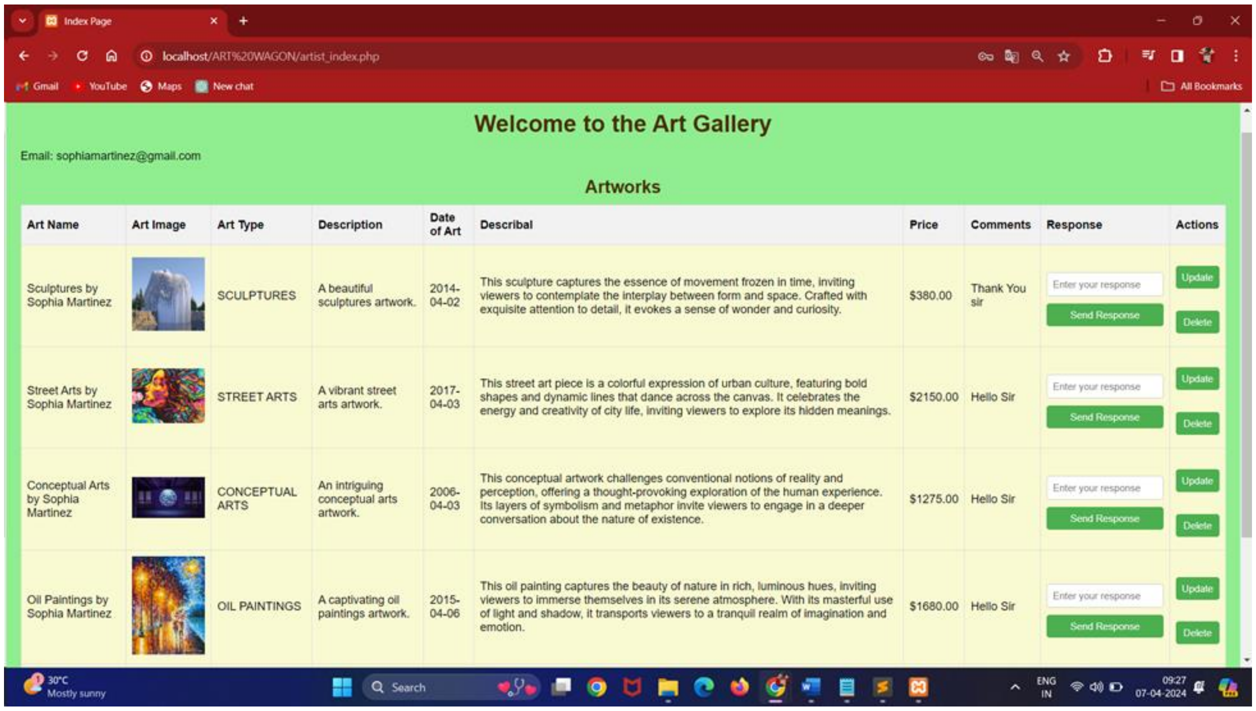
Artist Signup/Login

Signup

Login

sophiamartinez@gmail.com

Artist Signup and Login



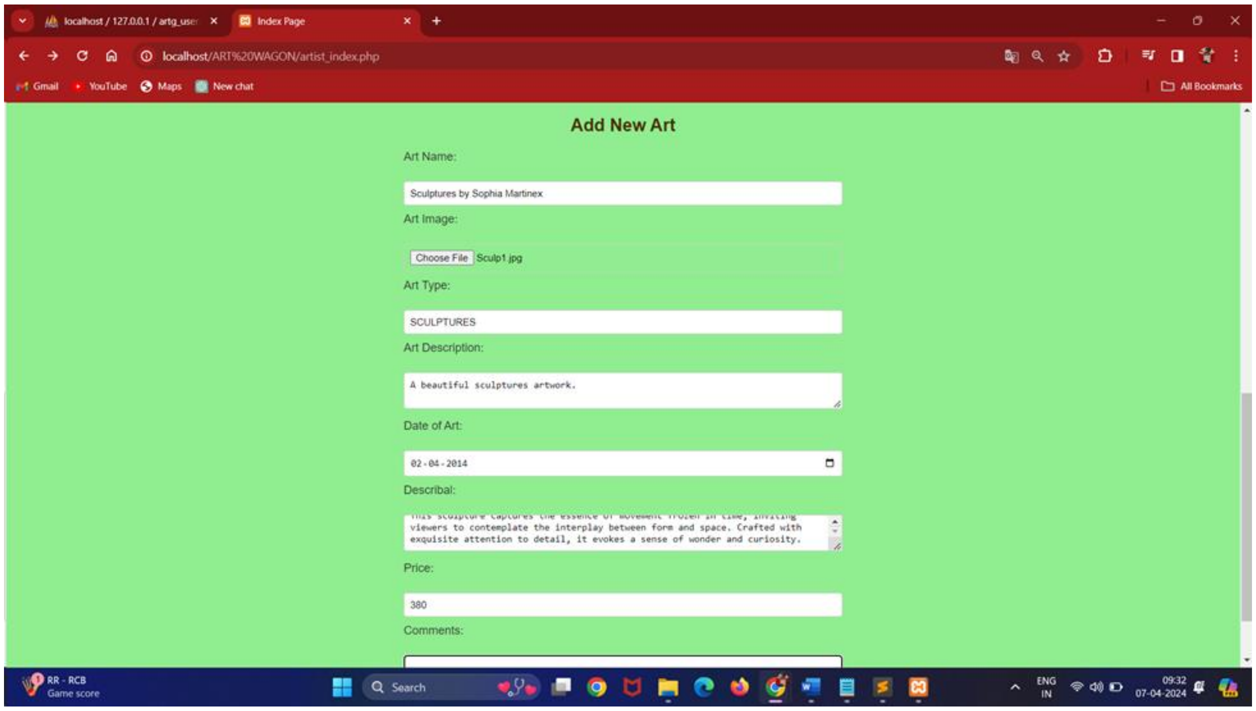
Welcome to the Art Gallery

Email: sophiamartinez@gmail.com

Artworks

Art Name	Art Image	Art Type	Description	Date of Art	Describal	Price	Comments	Response	Actions
Sculptures by Sophia Martinez		SCULPTURES	A beautiful sculptures artwork.	2014-04-02	This sculpture captures the essence of movement frozen in time, inviting viewers to contemplate the interplay between form and space. Crafted with exquisite attention to detail, it evokes a sense of wonder and curiosity.	$380.00	Thank You sir	Enter your response / Send Response	Update / Delete
Street Arts by Sophia Martinez		STREET ARTS	A vibrant street arts artwork.	2017-04-03	This street art piece is a colorful expression of urban culture, featuring bold shapes and dynamic lines that dance across the canvas. It celebrates the energy and creativity of city life, inviting viewers to explore its hidden meanings.	$2150.00	Hello Sir	Enter your response / Send Response	Update / Delete
Conceptual Arts by Sophia Martinez		CONCEPTUAL ARTS	An intriguing conceptual arts artwork.	2006-04-03	This conceptual artwork challenges conventional notions of reality and perception, offering a thought-provoking exploration of the human experience. Its layers of symbolism and metaphor invite viewers to engage in a deeper conversation about the nature of existence.	$1275.00	Hello Sir	Enter your response / Send Response	Update / Delete
Oil Paintings by Sophia Martinez		OIL PAINTINGS	A captivating oil paintings artwork.	2015-04-06	This oil painting captures the beauty of nature in rich, luminous hues, inviting viewers to immerse themselves in its serene atmosphere. With its masterful use of light and shadow, it transports viewers to a tranquil realm of imagination and emotion.	$1680.00	Hello Sir	Enter your response / Send Response	Update / Delete

Artist Index Page



Add New Art

Art Name:

Sculptures by Sophia Martinex

Art Image:

Choose File Sculp1.jpg

Art Type:

SCULPTURES

Art Description:

A beautiful sculptures artwork.

Date of Art:

02-04-2014

Describal:

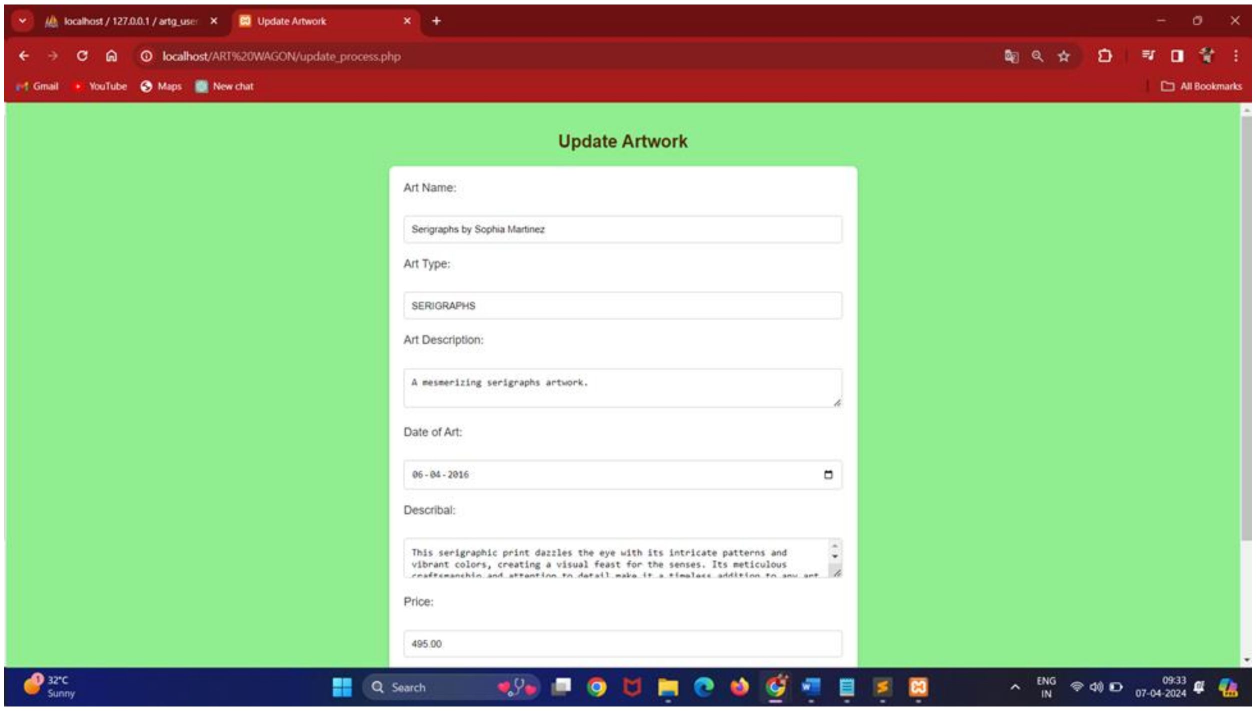
This sculpture captures the essence of movement frozen in time, inviting viewers to contemplate the interplay between form and space. Crafted with exquisite attention to detail, it evokes a sense of wonder and curiosity.

Price:

380

Comments:

Art Adding Form



localhost/ART%20WAGON/update_process.php

Update Artwork

Art Name:

Serigraphs by Sophia Martinez

Art Type:

SERIGRAPHS

Art Description:

```
A mesmerizing serigraphs artwork.
```

Date of Art:

06-04-2016

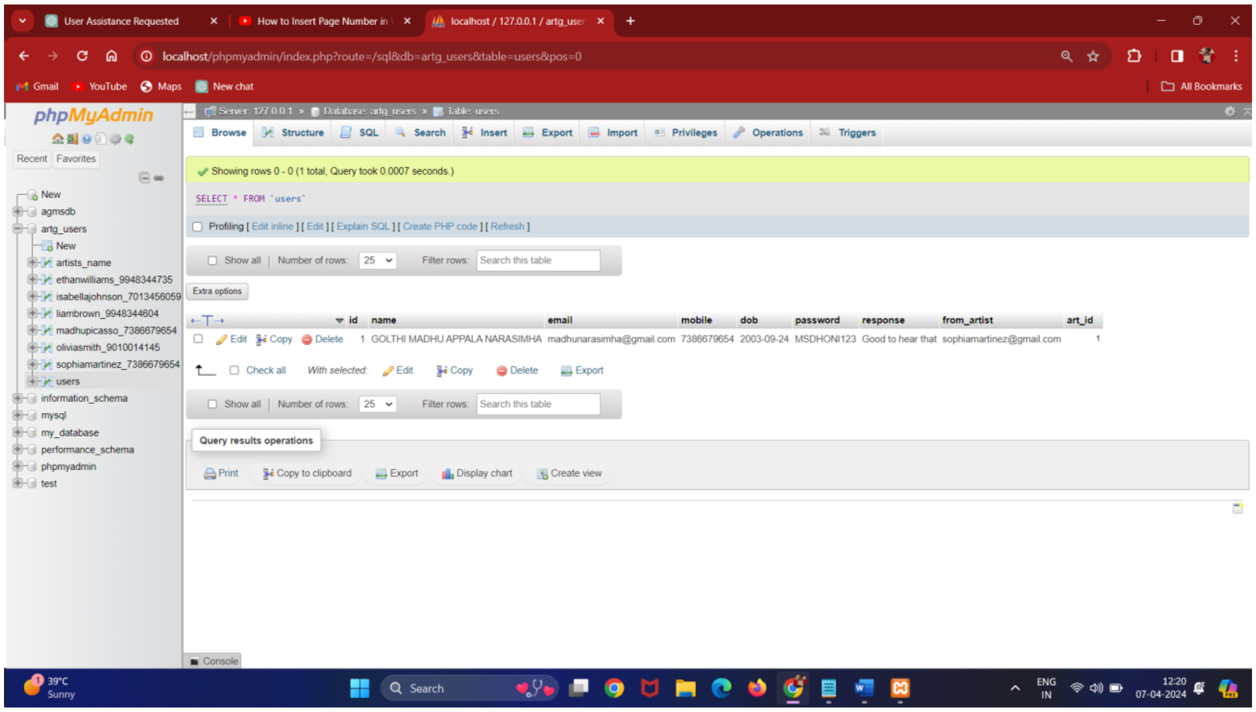
Describal:

```
This serigraphic print dazzles the eye with its intricate patterns and
vibrant colors, creating a visual feast for the senses. Its meticulous
craftsmanship and attention to detail make it a timeless addition to any art
```

Price:

495.00

Update Form Page



localhost/phpmyadmin/index.php?route=/sql&db=artg_users&table=users&pos=0

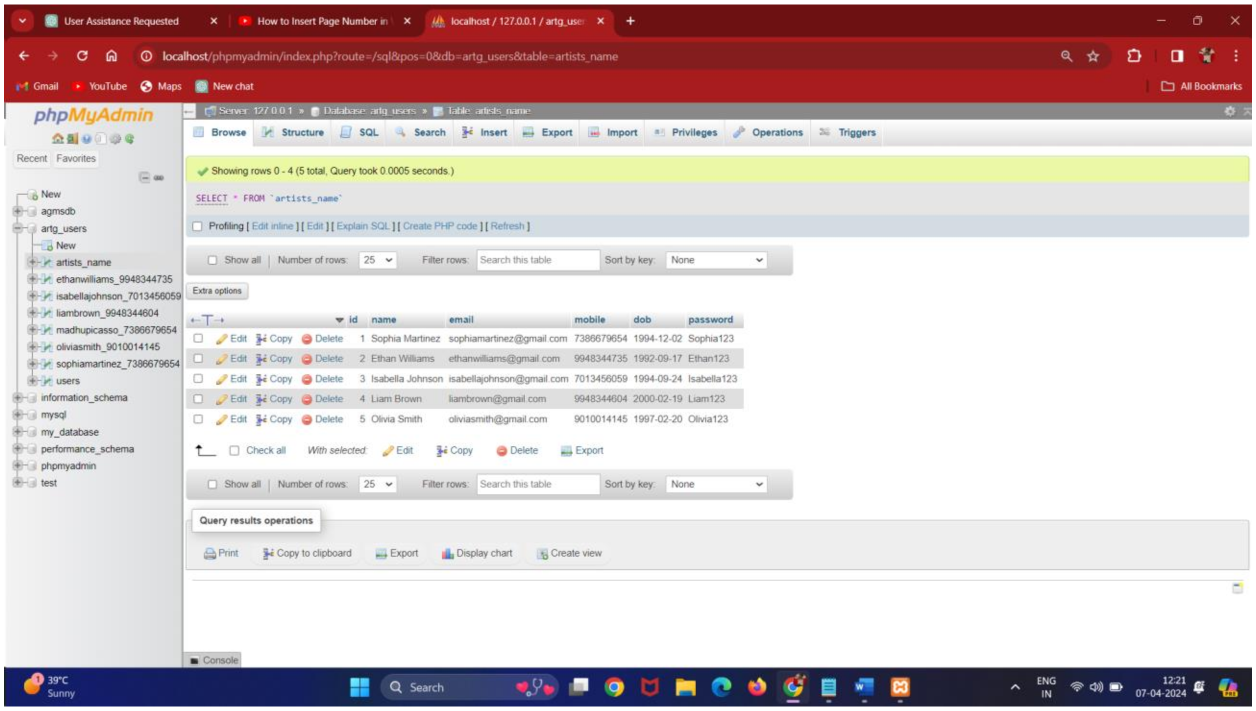
Server: 127.0.0.1 » Database: artg_users » Table: users

✓ Showing rows 0 - 0 (1 total, Query took 0.0007 seconds.)

```
SELECT * FROM `users`
```

id	name	email	mobile	dob	password	response	from_artist	art_id
1	GOLTHI MADHU APPALA NARASIMHA	madhunarasimha@gmail.com	7386679654	2003-09-24	MSDHONI123	Good to hear that	sophiamartinez@gmail.com	1

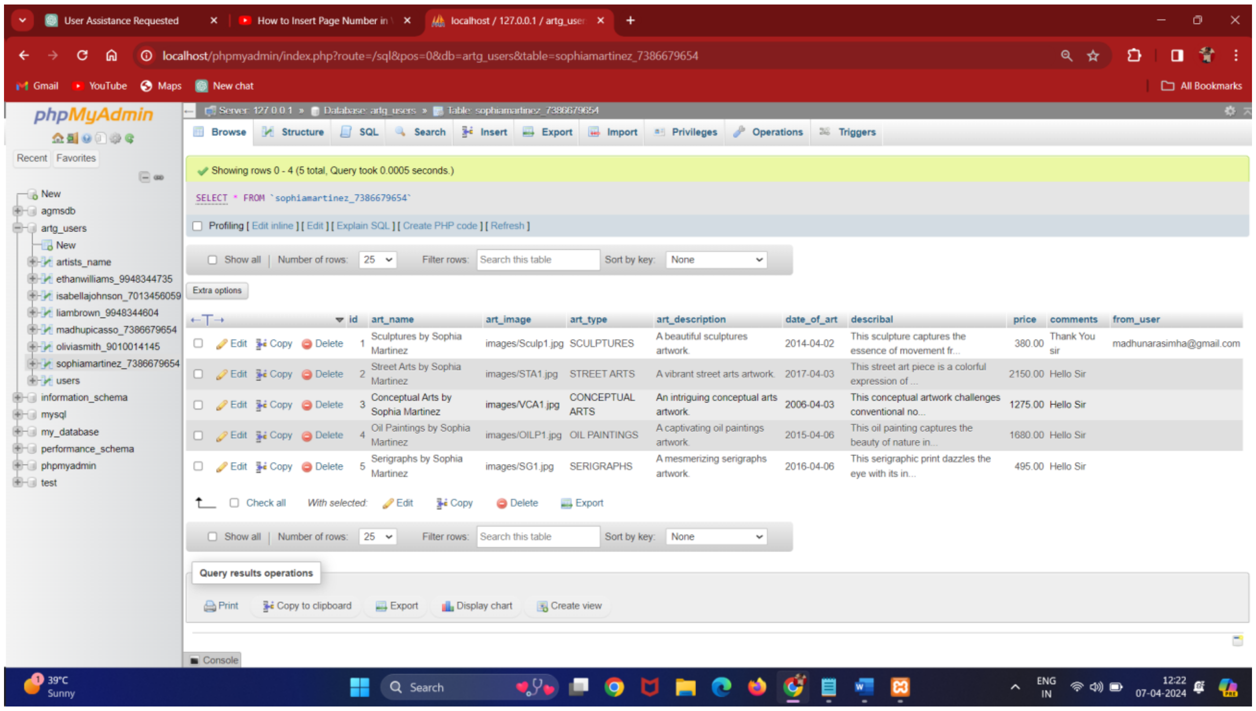
Users Table


Showing rows 0 - 4 (5 total, Query took 0.0005 seconds.)

```
SELECT * FROM `artists_name`
```

id	name	email	mobile	dob	password
1	Sophia Martinez	sophiamartinez@gmail.com	7386679654	1994-12-02	Sophia123
2	Ethan Williams	ethanwilliams@gmail.com	9948344735	1992-09-17	Ethan123
3	Isabella Johnson	isabellajohnson@gmail.com	7013456059	1994-09-24	Isabella123
4	Liam Brown	liambrown@gmail.com	9948344604	2000-02-19	Liam123
5	Olivia Smith	oliviasmith@gmail.com	9010014145	1997-02-20	Olivia123

Artists Name Table


Showing rows 0 - 4 (5 total, Query took 0.0005 seconds.)

```
SELECT * FROM `sophiamartinez_7386679654`
```

id	art_name	art_image	art_type	art_description	date_of_art	describal	price	comments	from_user
1	Sculptures by Sophia Martinez	images/Sculp1.jpg	SCULPTURES	A beautiful sculptures artwork.	2014-04-02	This sculpture captures the essence of movement fr...	380.00	Thank You sir	madhunarasimha@gmail.com
2	Street Arts by Sophia Martinez	images/STA1.jpg	STREET ARTS	A vibrant street arts artwork.	2017-04-03	This street art piece is a colorful expression of ...	2150.00	Hello Sir	
3	Conceptual Arts by Sophia Martinez	images/VCA1.jpg	CONCEPTUAL ARTS	An intriguing conceptual arts artwork.	2006-04-03	This conceptual artwork challenges conventional no...	1275.00	Hello Sir	
4	Oil Paintings by Sophia Martinez	images/OILP1.jpg	OIL PAINTINGS	A captivating oil paintings artwork.	2015-04-06	This oil painting captures the beauty of nature in...	1680.00	Hello Sir	
5	Serigraphs by Sophia Martinez	images/SG1.jpg	SERIGRAPHS	A mesmerizing serigraphs artwork.	2016-04-06	This serigraphic print dazzles the eye with its in...	495.00	Hello Sir	

Artists Arts Table

CONCLUSION

In conclusion, the ARTWAGON Virtual Art Gallery represents a dynamic and innovative platform that bridges the gap between artists and art enthusiasts, providing a seamless and immersive experience for all users. By leveraging advanced web technologies and intuitive interfaces, the platform facilitates effortless exploration and appreciation of diverse artworks across various genres and styles. Through features such as personalized artist portfolios, interactive exhibition rooms, and robust communication channels, ARTWAGON fosters meaningful connections between artists and users, fostering collaboration, dialogue, and creativity within the digital art community. The platform's commitment to user engagement, accessibility, and security ensures a rewarding and enriching experience for all participants. As the digital landscape continues to evolve, ARTWAGON remains dedicated to pushing boundaries and redefining the art viewing experience, empowering artists to showcase their talents, and enabling users to discover and connect with art in new and exciting ways. With its innovative features, user-centric design, and vibrant community, ARTWAGON stands as a testament to the transformative power of technology in the world of art and culture.

WEB REFERENCES

https://www.php.net/manual/en/features.http-auth.php

https://stackoverflow.com/questions/47050503/the-correct-way-to-use-sessions

https://www.youtube.com/watch?v=84IOtc05TuA

https://www.youtube.com/watch?v=n7qNyTuw5n8&t=22s