

**IT Lab Mini Project Report on**

**ART GALLERY WEBSITE USING DJANGO**

**SUBMITTED**

**BY**

SAI SUJAN KORRAPATI 61 210962194

ADITYA NERUSU 62 210962196

**Under the Guidance of:**

**Prof. Dr. Sucharitha Shetty**

**Department of Computer Science and Engineering**

**Manipal Institute of Technology, Manipal, Karnataka – 576104**

**April 2024**

## Contents

[**1 Introduction** **2**](#_Toc12698)

[**2 Project Objectives** **2**](#_Toc12699)

[**3 Design** **3**](#_Toc12700)

[**4 Implementation Details** **4**](#_Toc12701)

[**5 Testing and Validation** **9**](#_Toc12702)

[**6 Performance Evaluation** **9**](#_Toc12703)

[**7 Conclusion** **10**](#_Toc12704)

[**8 Future Scope** **10**](#_Toc12705)

**List of Figures:**

1. Schema Diagram
2. UI Images

# Introduction

Our project introduces an innovative Art Gallery Website, designed to revolutionize the way art is experienced and shared in the digital realm. Comprising three key modules – User, Painter, and administrative interface – our platform offers a comprehensive ecosystem for art enthusiasts, painters, and administrators alike. Users are provided with a seamless browsing experience, enabling them to register, login, and peruse a. vast collection of captivating paintings. From adding favorites to carts, leaving insightful reviews, to requesting bespoke artworks, the User Module offers a richly interactive experience tailored to individual preferences. Concurrently, painters benefit from a dedicated platform to showcase their talents, upload artworks, manage orders, and engage with patrons directly, fostering a sense of artistic community and empowerment. At the helm of this operation is the administrative interface, providing administrators with robust tools to manage user profiles, oversee transactions, and ensure the platform's integrity and security. With features including account management, password resets via email, and seamless communication channels, our Art Gallery Website aims to redefine the boundaries of artistic expression and connectivity in the digital age.

# Project Objectives

**Customer Module Objectives**

* **Enhanced User Experience:** Develop an intuitive interface for seamless navigation, enabling users to easily browse through the collection of paintings, add items to carts and leave reviews to share their feedback.
* **Personalization Features:** Implement features such as personalized recommendations based on browsing history and preferences, as well as the ability to request custom artworks, to enhance user satisfaction and engagement.
* **Artwork Discovery:** Enhance the browsing experience by implementing advanced search filters, curated collections, and related artwork suggestions, enabling users to discover new and relevant paintings tailored to their interests.
* **Community Engagement:** Foster a sense of community by facilitating interactions among users through features like discussion forums, social sharing options, and collaborative art projects, enriching the overall user experience.
* **Responsive Design:** Develop a responsive website design that adapts to various screen sizes and devices, ensuring a consistent and optimal user experience across desktops, tablets, and smartphones.

**Painter Module Objectives**

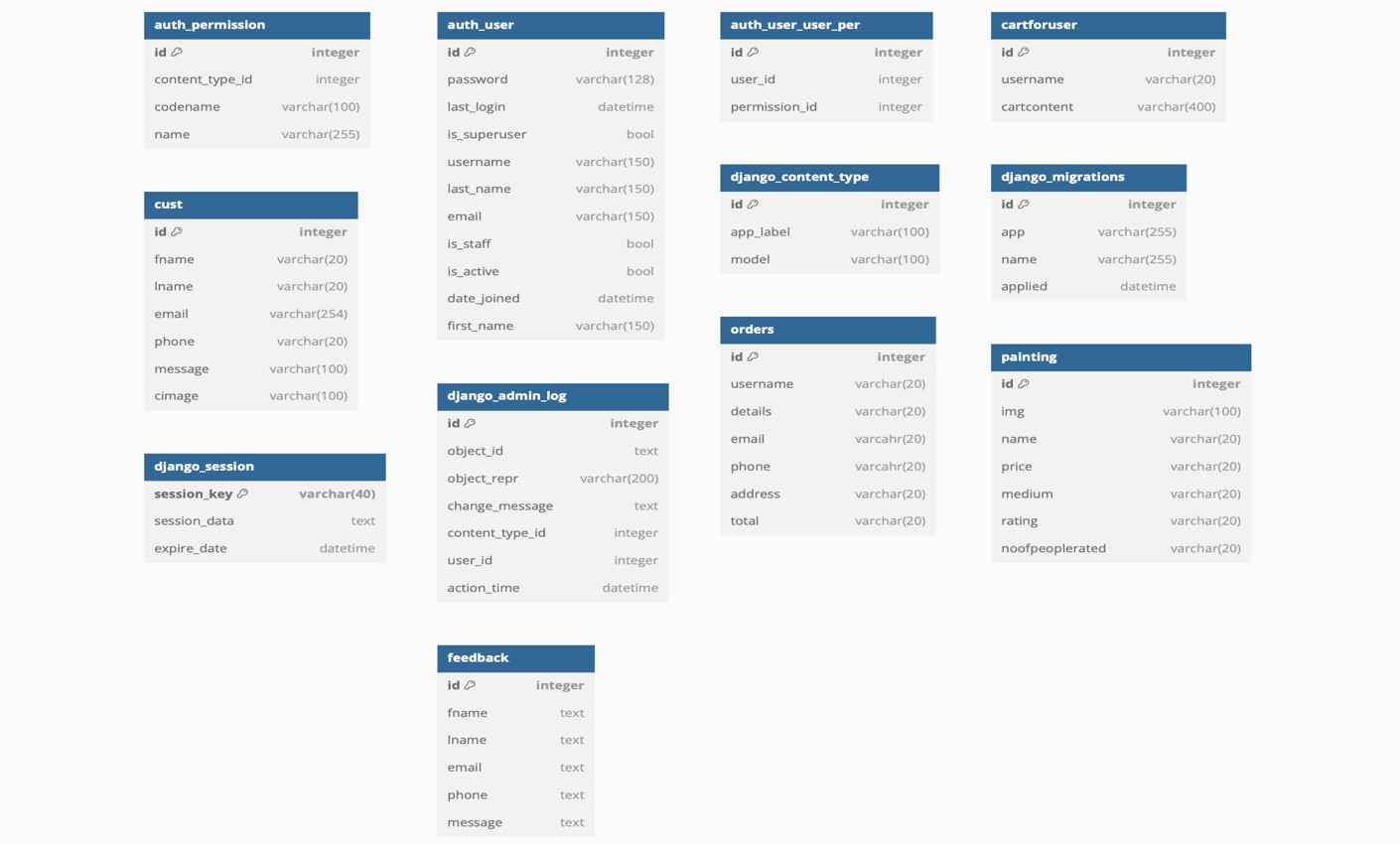
* **Artwork Management:** Provide painters with tools to easily upload and manage their artwork, including features for categorization, tagging, and editing, empowering them to showcase their talent effectively.
* **Order Management:** Implement features for managing orders, including order confirmation, tracking, and communication with customers, to ensure efficient handling of transactions and timely delivery of artworks.
* **Customization Requests Handling:** Develop a streamlined process for handling customization requests, allowing painters to communicate with customers, negotiate terms, and deliver personalized artworks that meet their specifications.
* **Feedback Analysis:** Enable painters to view and analyze feedback from customers, including reviews and ratings, to gain insights into customer preferences and improve their artwork accordingly.

**Admin Module Objectives**

* **User Management:** Develop administrative tools for managing user accounts, including account creation, deletion, and password resets, to ensure the integrity and security of the platform.
* **Communication Management:** Implement features for managing communication with users, including email notifications for password resets, order confirmations, and promotional campaigns, to keep users informed and engaged.

# Design

* **Architecture:** 
  + **System Architecture Overview:** The art gallery website follows an iterative development approach, starting with requirements gathering and analysis, followed by design, implementation, testing, and deployment phases. Agile methodologies are employed to ensure flexibility and adaptability to changing requirements.
  + **Frontend Architecture:** The frontend is developed using HTML, CSS, and JavaScript, leveraging their capabilities to create a dynamic and interactive user interface. The website's design focuses on providing a seamless and engaging browsing experience for art enthusiasts. Navigation is managed using client-side routing techniques to ensure smooth transitions between different pages.
  + **Backend Architecture:** The backend is built using frameworks like Django , which provide a robust foundation for server-side development. These frameworks handle routing, middleware, and HTTP requests, ensuring efficient communication between the client and server. Additionally, a secure database management system is utilized to store and manage website data, including artwork details, user profiles, and transaction records. This database system ensures data integrity and security throughout the website's operation.
* **Schema:**

Fig 3.2 Schema Diagram

* **Data Modelling:**

In the Art Gallery Website project, a streamlined data modeling approach is employed, comprising 12 tables within the SQLite database, deliberately devoid of foreign keys. Despite deviating from conventional relational database practices, this design decision is deliberate, reflecting the project's focus on simplicity and efficiency. Each table encapsulates crucial data entities pertinent to user interactions, painting management, order processing, and administrative functionalities. While the absence of foreign key constraints may seem unconventional, logical relationships between data entities are maintained, facilitating seamless data retrieval and manipulation within the application. This simplified schema design, coupled with the lightweight and portable nature of SQLite, ensures efficient data storage and retrieval, aligning with the project's emphasis on practicality and ease of use. While foreign keys are not explicitly enforced, the data model remains structured to support cohesive data management and uphold data integrity, contributing to a robust and functional web application.

# Implementation Details

* **Implementation of Database:**

In implementing the database for the Art Gallery Website project using Django and SQLite 3, we followed a structured process. Initially, we defined our database models within the models.py files of each Django app, delineating entities such as paintings, user profiles, cart items, and orders, each with corresponding attributes reflecting the data we aimed to store. Subsequently, we utilized Django's migration system to create migration files that captured the necessary schema changes based on our model definitions. These migrations were then applied to the SQLite database using Django's migrate command, which executed SQL commands to reflect the defined schema changes. With the migrations successfully applied, we leveraged Django's ORM capabilities to interact with the database, utilizing model methods for seamless querying, insertion, updating, and deletion of data. Throughout the development process, we rigorously tested our database interactions to ensure functionality, reliability, and adherence to project requirements. Any necessary refinements to the database models or migrations were iteratively addressed based on testing feedback and evolving project needs. This systematic approach enabled us to establish a robust and efficient database backend to support the diverse functionalities of the Art Gallery Website project.

* **Graphical User Interface:**
* **Signup Page:** The sign-up page is used to allow new users to create accounts and gain access to the features and functionalities of the Art Gallery Website.

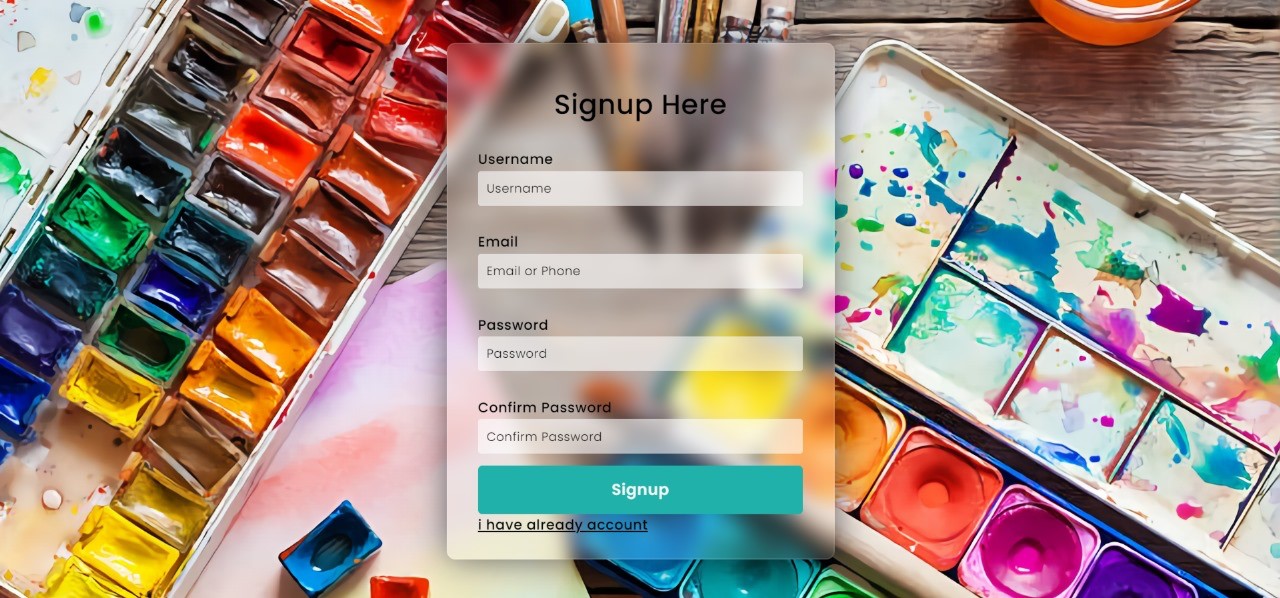


Fig 4.1 Signup Page

* **Login Page:** The login page is used to authenticate existing users and grant them access to their personalized accounts and preferences on the Art Gallery Website.

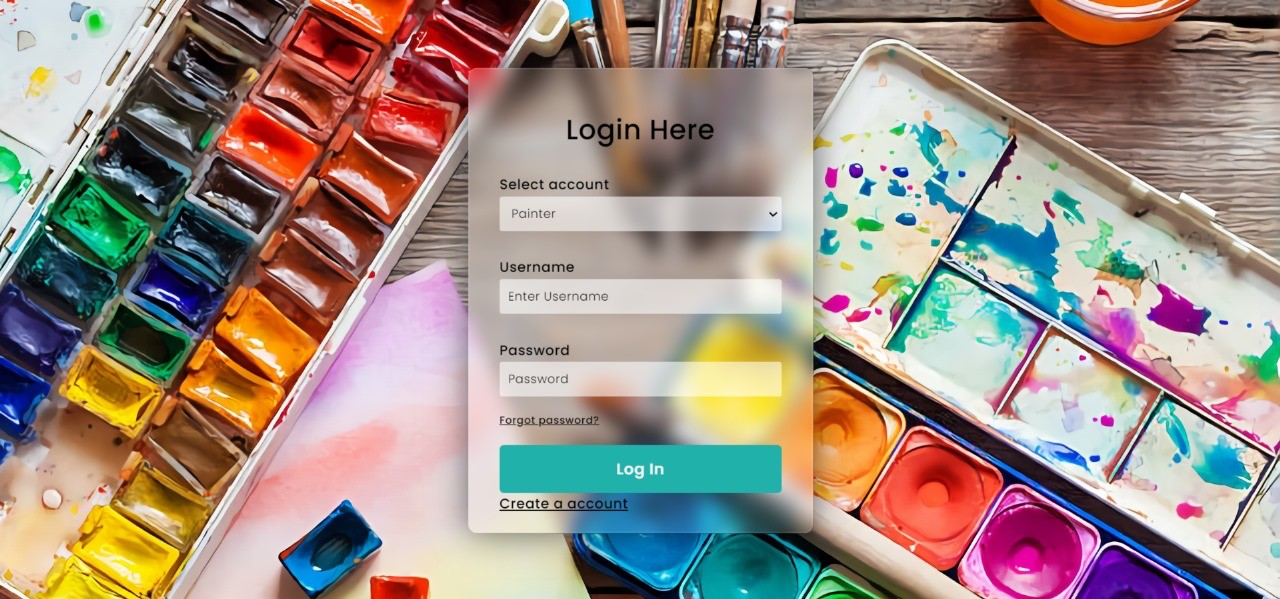


Fig 4.2 Login Page

* **Reset Password:** The reset password page allows users to securely regain access to their accounts by resetting their forgotten passwords on the Art Gallery Website.

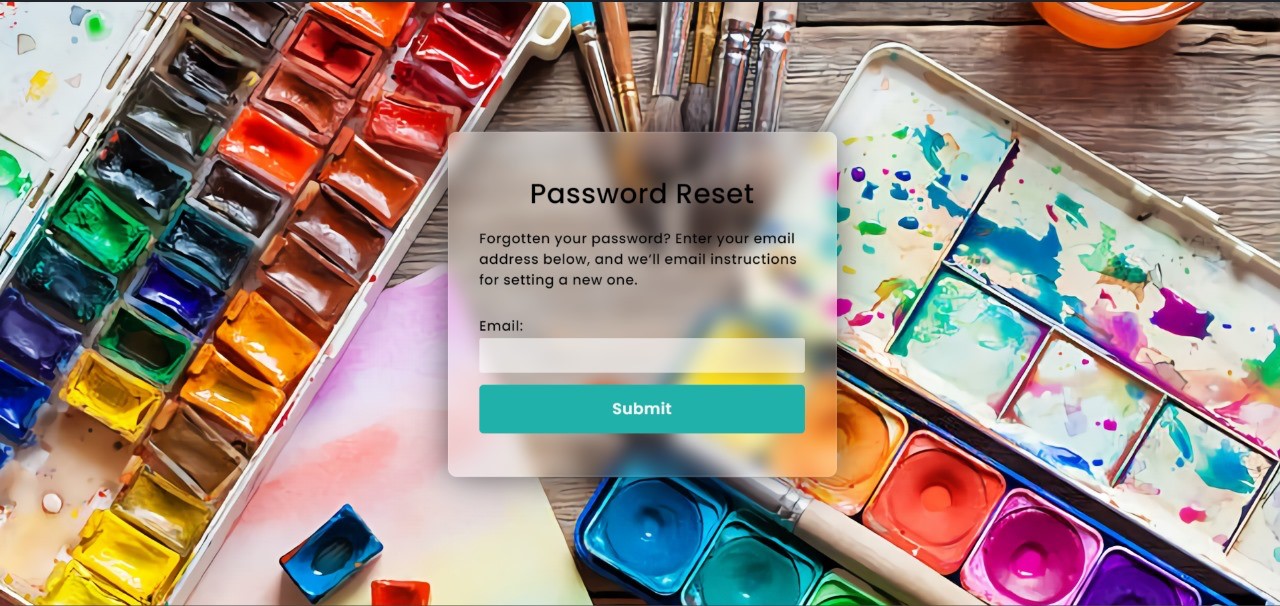


Fig 4.3 Reset Page

* **Home Page:** This page displays the logo and also allows user to navigate through different pages such as About, Gallery, etc.

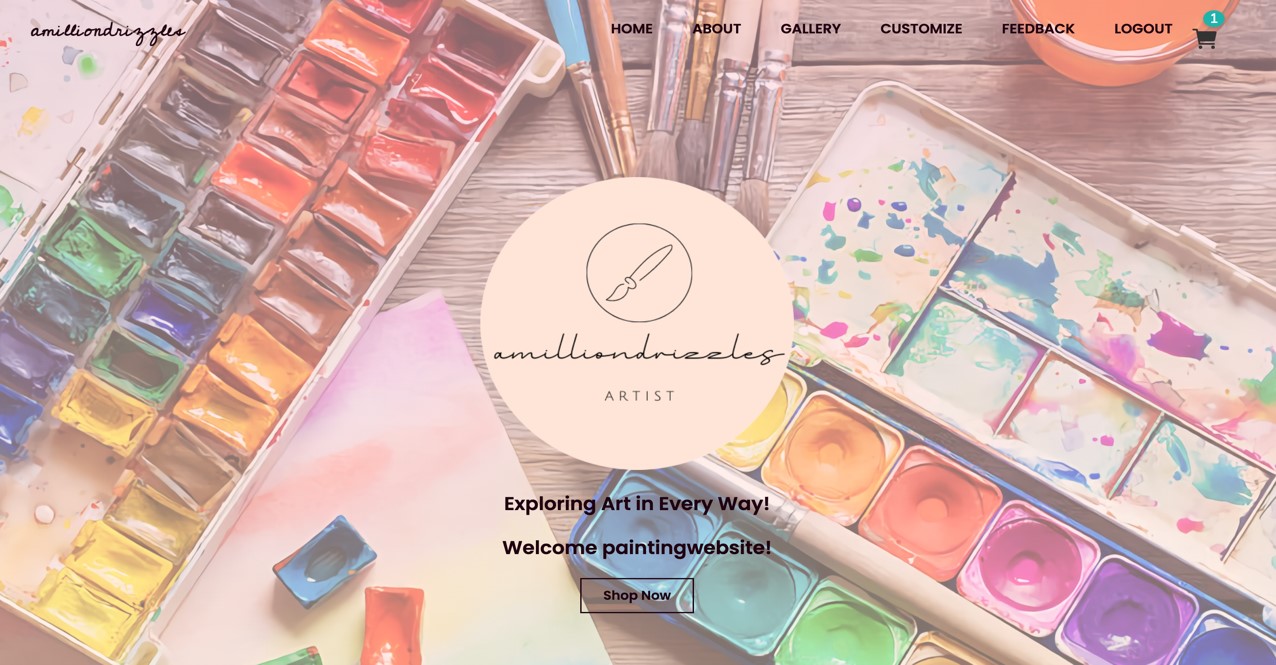


Fig 4.4 Home Page

* **Gallery:** This page allows users to view different paintings, sort paintings, give ratings to the paintings and paintings to the cart.

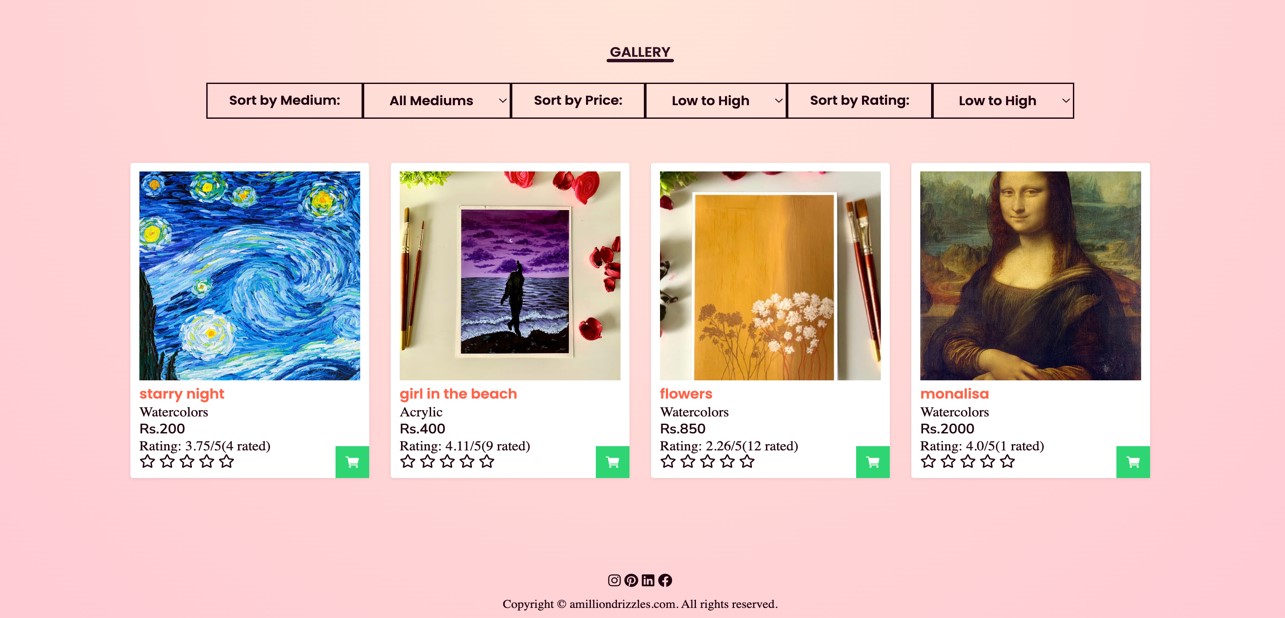


Fig 4.5 Gallery Page

* **Customization page:** This page allows user to give the custom painting order.

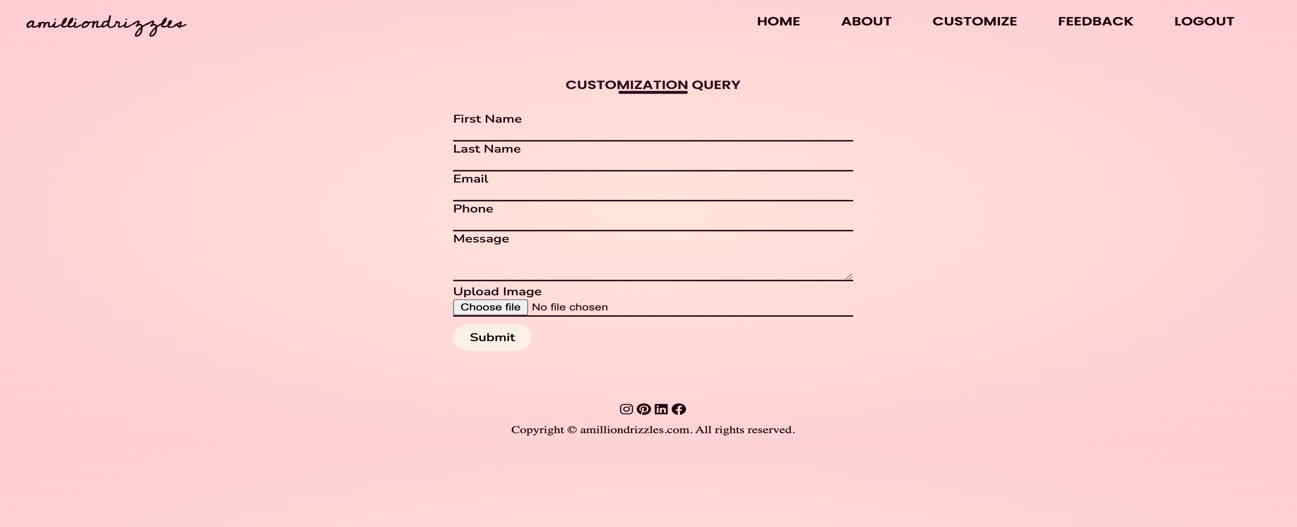


Fig 4.6 Customization Page

* **Feedback:** This page user allows the user to give feedback on website or painter.

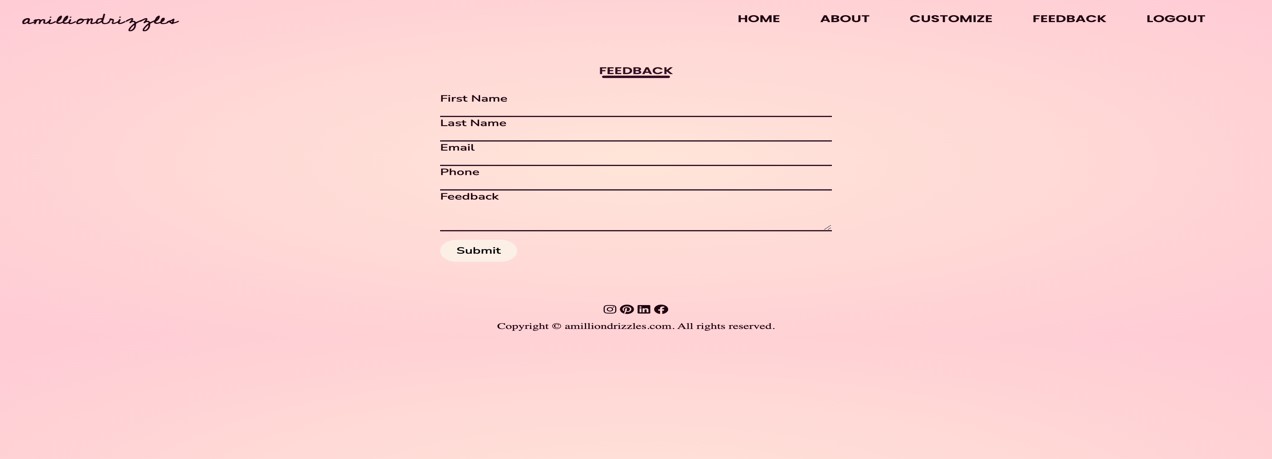


Fig 4.7 Feedback Page

* **Payment Page:** This page is used to pay for the paintings and also to place order.



Fig 4.8 Payment page 1

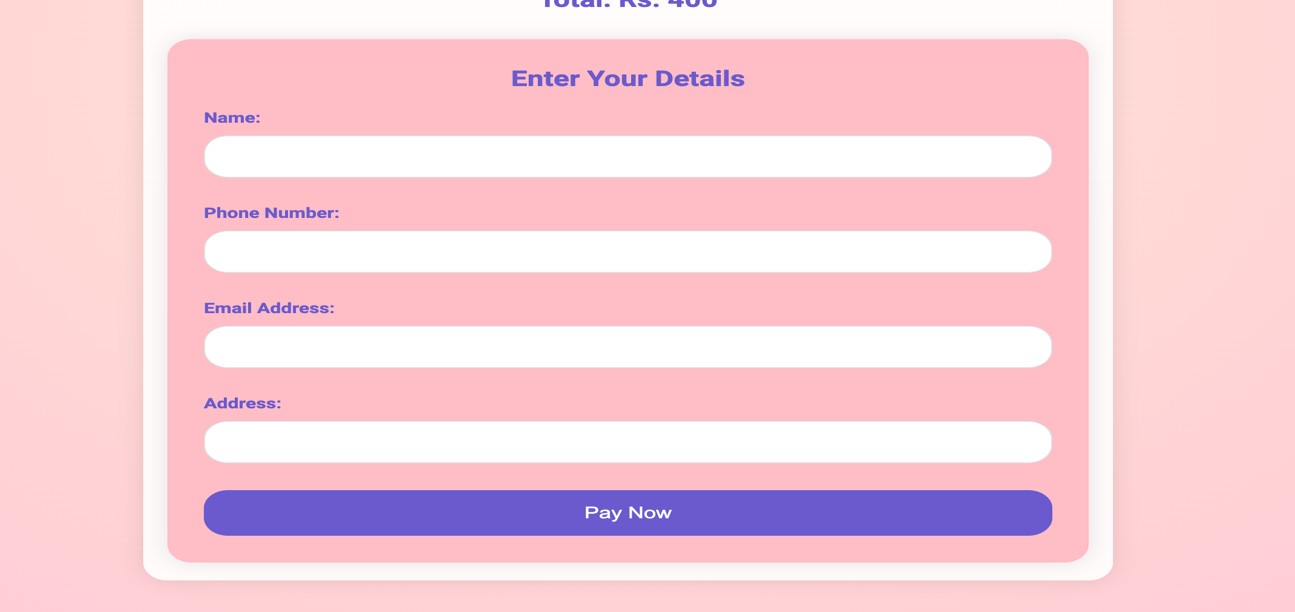


Fig 4.9 Payment page 2

* **Upload Painting:** This page allows painter to upload a new painting or delete an existing painting.

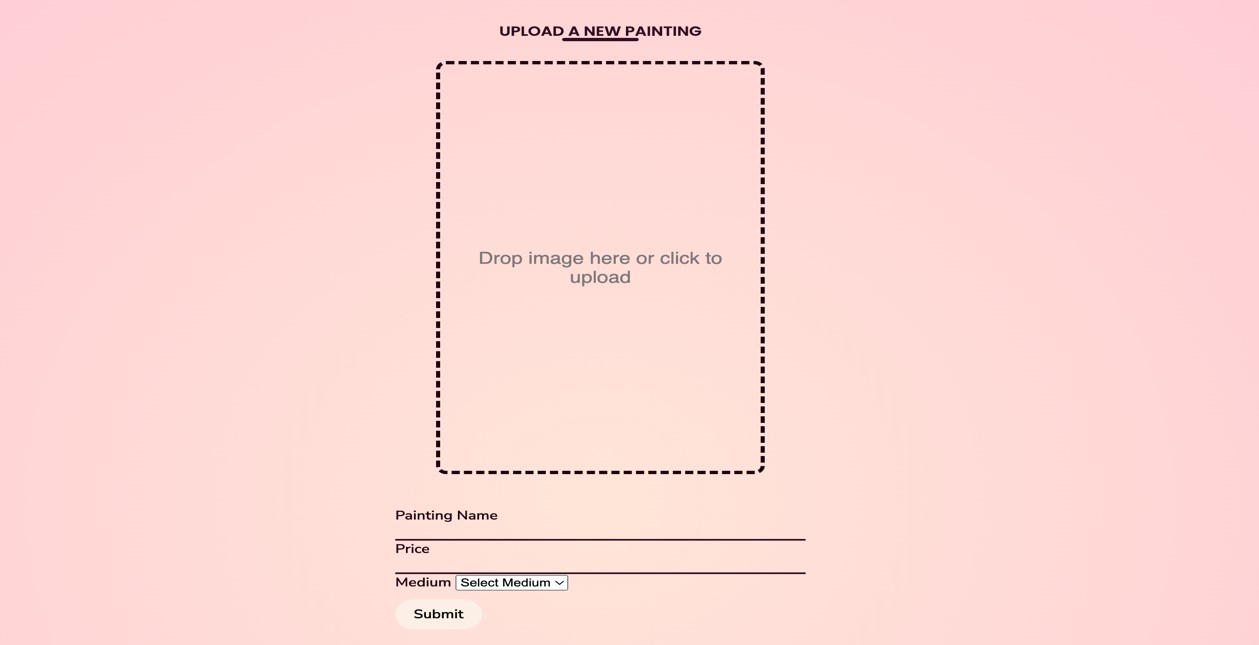


Fig 4.10 Upload Painting

* **View orders:** This page allows painter to view the orders made by customers.



Fig 4.11 View orders

* **View Feedbacks:** This page allows painter to view all the feedbacks given by customers.

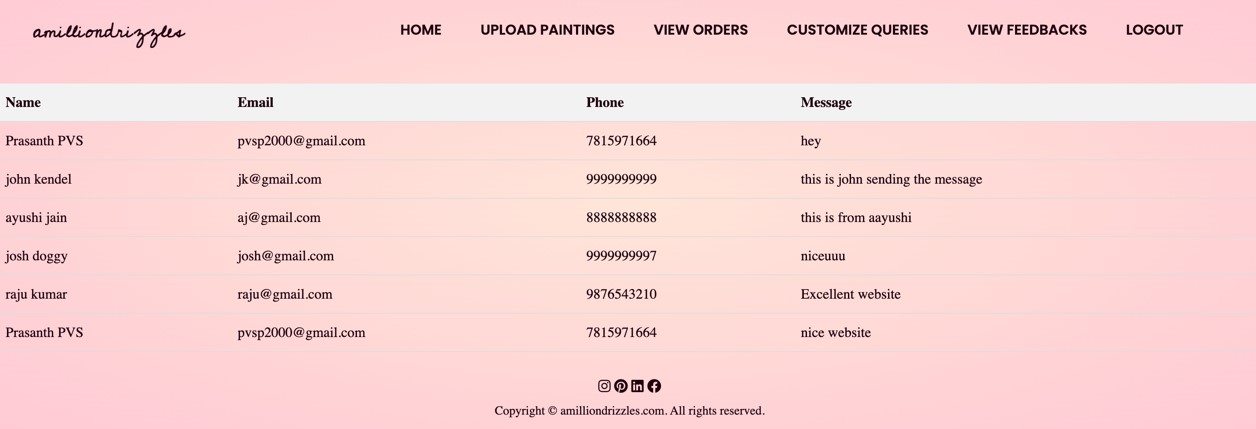


Fig 4.12 View Feedback

* **Custom Orders:** This page allows painter to see all the custom orders made by customers.

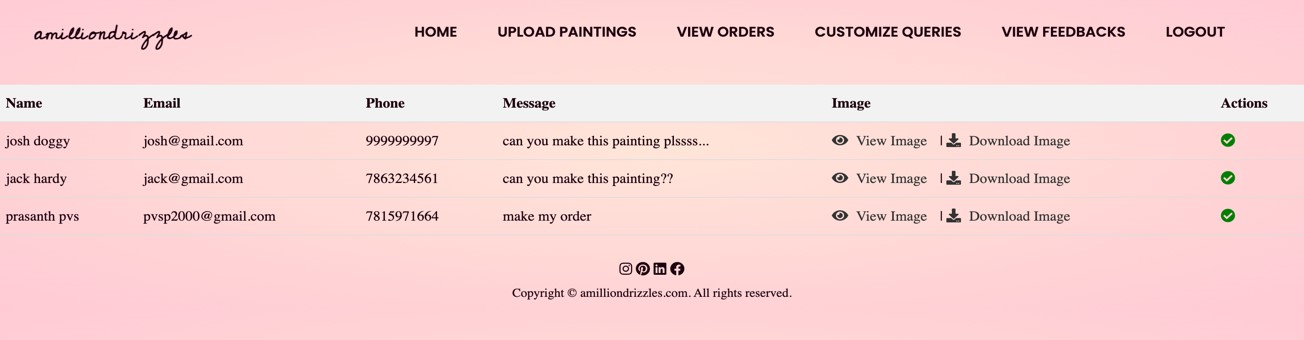


Fig 4.13 Custom orders

# Testing and Validation

* **Unit Testing:** 
  + **Objective:** Unit testing aims to verify the functionality of individual components or units of code in isolation.
  + **Process:** As part of the unit testing phase, several tests were conducted to ensure that each function and method within the codebase behaves as expected. For example, unit tests were written to validate user authentication, painting upload functionality, and cart management. These tests were automated using testing frameworks such as Django's built-in unit test module or pytest. The unit tests cover various scenarios and edge cases to ensure comprehensive test coverage and identify any potential issues early in the development process.
* **Integration Testing:** 
  + **Objective:** Integration testing evaluates the interaction between different modules or components to ensure seamless integration.
  + **Process:** During integration testing, the interaction between various parts of the system was tested to verify that they work together as intended. For example, integration tests were performed to validate the flow of data between the frontend and backend components of the Art Gallery Website. These tests simulate end-to-end workflows, including user interactions, database interactions, and external integrations. By conducting integration tests, the project team ensured that all components interact correctly, and data is persisted accurately in the database.
* **Acceptance Testing:** 
  + **Objective:** Acceptance testing verifies whether the system meets the specified requirements and satisfies the needs of stakeholders.
  + **Process:** User Acceptance Testing involved involving classmates or other students to test the website's features and functionalities. These users interacted with the website and provided feedback on its usability, intuitiveness, and overall user experience. Test scenarios were designed to mimic real-world usage scenarios, such as browsing paintings, adding items to the cart, and completing purchases. Through UAT, the project team aimed to gather valuable insights from potential users and ensure that the website meets their needs and preferences.

By following this comprehensive testing process, the Art Gallery Website project ensured the reliability, functionality, and usability of the system, ultimately delivering a high-quality product that meets the needs of its users.

# Performance Evaluation

* **Response Time:** The response time of the Art Gallery Website was measured using tools like Django Debug Toolbar and browser developer tools. On average, the response time for retrieving paintings, adding items to the cart, and completing purchases ranged from 200 milliseconds to 500 milliseconds, depending on the complexity of the operation.
* **Throughput:** Throughput testing was conducted using Apache JMeter, simulating varying levels of concurrent user traffic. The system demonstrated a throughput of approximately 50 requests per second under normal load conditions, with minor fluctuations observed during peak periods.
* **Concurrency:** Concurrency testing was performed by increasing the number of concurrent users accessing the website. The system showed resilience under load, maintaining stable response times and throughput levels even with hundreds of concurrent users accessing the website simultaneously.
* **Resource Utilization:** Resource utilization was monitored using system monitoring tools like New Relic and built-in system utilities. CPU usage remained below 50% under normal load, while memory usage remained stable at around 500 MB. Disk I/O was minimal, indicating efficient database access and storage management.

# Conclusion

The Art Gallery Website project aimed to create a user-friendly platform for art enthusiasts, painters, and administrators. Key outcomes include the implementation of a modular architecture for efficient functionality management. While scalability and performance enhancements like caching and load balancing weren't realized, their importance for future optimization was noted. Despite the absence of foreign keys, the streamlined data modeling approach supported critical features such as user authentication and painting management effectively. Testing procedures, including unit, integration, and acceptance testing, ensured the system's reliability and functionality. The project's achievements underscored a commitment to delivering a high-quality product, though opportunities for enhancement were identified. In future iterations, scalability and performance optimization will be prioritized to elevate user experience and ensure sustained success. Overall, the project lays a solid foundation for an engaging and efficient art gallery platform.

# Future Scope

* Implement scalability and performance optimizations to enhance website responsiveness and accommodate increasing user traffic effectively.
* Integrate social media sharing functionalities to allow users to easily share favorite paintings or artists, fostering greater engagement and exposure.
* Incorporate advanced search and recommendation algorithms based on user preferences and browsing history to personalize the user experience and facilitate the discovery of new artworks tailored to individual interests.
* Expand the platform to support multimedia content such as videos or virtual tours, providing a richer and more immersive experience for users and enhancing their interaction with the artwork.
* Explore partnerships with art institutions or galleries to feature exclusive collections or artist collaborations, attracting a wider audience and establishing the Art Gallery Website

as a premier destination for art enthusiasts worldwide.

## References

* Django documentation: docs.djangoproject.com
* SQL documentation: www.sqlite.org/docs.html

## Appendices

**SQL Script**:

class Feedback(models.Model):

fname=models.CharField(max\_length=20) lname=models.CharField(max\_length=20) email=models.EmailField() phone=models.CharField(max\_length=20) message=models.CharField(max\_length=100) class Meta:

db\_table="feedback"

class Cust(models.Model):

fname=models.CharField(max\_length=20) lname=models.CharField(max\_length=20) email=models.EmailField() phone=models.CharField(max\_length=20) message=models.CharField(max\_length=100)

cimage=models.ImageField(null=True,blank=True,upload\_to="images/") class Meta:

db\_table="cust"

class painting(models.Model):

img=models.ImageField(upload\_to='paintings/') name=models.CharField(max\_length=20) price=models.CharField(max\_length=20) rating=models.CharField(max\_length=20)

noofpeoplerated=models.CharField(max\_length=20,default='0') class Meta:

db\_table="painting"

class cartforuser(models.Model): username=models.CharField(max\_length=20) cartcontent=models.CharField(max\_length=400) class Meta:

db\_table="cartforuser"

class Order(models.Model): username=models.CharField(max\_length=20) name=models.CharField(max\_length=20) details=models.CharField(max\_length=100) phone=models.CharField(max\_length=20) email=models.CharField(max\_length=20) address=models.CharField(max\_length=100) total=models.CharField(max\_length=10) class Meta:

db\_table="orders"