IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 19, April 2026
Impact Factor: 8.2

ArtConnect: Cloud-Based Artwork Sharing Platform with Rule-Based Chatbot Assistance

Deepa M and Dr. S. Kamalakkannan
Department of Computer Applications School of Computing Sciences
VELS Institute of Science, Technology and Advanced Studies (VISTAS) Pallavarm, Chennai, India

Abstract: *ArtConnect is a cloud application that enables the exchange of artworks and provides artists with a tool for uploading, managing, and browsing their artworks online using a dynamic web interface. Users can also add artworks to a cart and manage selected items for potential purchase. The server-side is powered by Spring Boot, whereas the client-side leverages Thymeleaf along with HTML, CSS, and JavaScript. The database management system is based on MongoDB Atlas, whereas Cloudinary provides support for storing and delivering images. Rule-based chatbots developed using Flask enable users to perform routine actions such as registration, logging in, uploading artworks, and exploring art galleries. The application is deployed in the cloud using Render services.*

Keywords: Artwork sharing platform, Spring Boot, MongoDB Atlas, Cloudinary, Flask chatbot, cloud deployment, rule-based conversational assistant, web application architecture

I. INTRODUCTION

Digital platforms are currently popular in the sharing of artistic works among other benefits. Various applications designed specifically for sharing of artworks are available, which allow artists to upload their portfolios, explore artwork created by other users and socialize within a community of people engaged in creation. In addition, these online platforms eliminate geographical limitations and give access to art through clouds.

However, most of these platforms currently offer mostly image management capabilities without any specific guidelines for inexperienced users trying out the services for the first time. In such cases, users may find it difficult to create new accounts, upload their artworks, and explore galleries as there are no interactive features guiding them through the process.

ArtConnect is a web application designed to serve as an artwork sharing tool based on storage and assisted by a rule-based chatbot capable of providing interactive guidelines on how to use different aspects of the app, including signing in, browsing galleries, uploading art, and managing personal profile information.

The back-end of the ArtConnect application is developed using Spring Boot, whereas the front-end utilizes a combination of Thymeleaf along with HTML, CSS, and JavaScript. The user information and artwork metadata are stored in MongoDB Atlas, and Cloudinary serves as an efficient tool for storing and delivering images. A chatbot is also employed in the process of navigation and is implemented using Flask.

To summarize, ArtConnect provides a highly structured environment for artists to share their digital art pieces through the assistance of chatbots.

II. EXISTING SYSTEM

Traditional websites for sharing artwork provide basic functions such as uploading and showcasing art pieces as well as browsing the web gallery. This way an artist can create his/her own account, add pictures of their works of art and view those presented by other users. Such systems are extremely important for placing a portfolio online and reaching out to people outside one's local community.

However, modern platforms tend to prioritize storage and visualization over navigation support. New users can face problems while creating account, uploading images, and exploring galleries since no additional information is provided

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 19, April 2026
Impact Factor: 8.2

in the form of interactive assistance or instructions. The problem of inadequate communication support should also be addressed because users often has to go from one section of the website to another to find relevant information. This process can be especially problematic for beginners who have never managed an online gallery before.

In some instances, the inability of the website to capitalize on the benefits of cloud computing and image handling may affect scalability and performance in processing large images. These challenges emphasize the need for a cloud-hosted artwork-sharing websites that provides organized navigation and conversation assistance for users all over the world.

III. PROPOSED SYSTEM

The proposed solution ArtConnect is implemented in the form of a cloud-based platform dedicated to the sharing of artworks. The idea behind it is to provide artists with an easy and convenient way to upload, manage, and discover digital art using an intuitive web application. To achieve this goal, we need to combine structured artwork management with a rule-based chatbot providing assistance to the user while browsing.

Signing up and logging into one's account, the users will be able to perform such operations as uploading works of art, and browsing other artworks created by the users. As a result, we get a fully integrated solution for creating and exploring works of art. An important element of our implementation is the artwork upload and management module. In order to upload images to our service, we use Cloudinary. Instead of storing images directly on the server, we save references and metadata about artwork images in MongoDB Atlas, making access faster and easier.

A gallery module is implemented to facilitate interactions with the artworks provided by other artist's on the platform. The artwork can be explored by the users in a clean, well-organized interface. The users can peruse through various artworks using a special module meant for buyer's. There will be an option of viewing the details of the artwork and putting it into the cart. The user can then modify the cart by adding or removing the artwork as he pleases. To enhance user experience further, an assistance function implemented through a rule-based chatbot is offered. Such a chatbot, created with Flask framework, provides helpful tips and information about such issues as logging in, uploading artworks, navigating galleries, etc to assist first-time users. The application logic is implemented with Spring Boot framework, which takes care of such tasks as authentication, processing incoming requests, and communication between different components. MongoDB Atlas is responsible for storing user accounts and metadata of artworks, whereas Cloudinary helps with efficient image storage. The whole application is deployed to the cloud infrastructure provided by Render.

Overall, this application architecture allows creating a structured platform for sharing artworks in the cloud environment, where users can access all of the required functionality through chatting with the assistance bot.

IV. METHODOLOGY

The development process of the ArtConnect application involved number of distinct stages that included defining the needs of the users, developing the design of the system, implementing its elements, integrating components, and performing testing. Initially, we determined the core need of the users that was met with creating a platform for sharing artworks with user-friendly design features and using a chatbot to guide the users' navigation.

For the development stage, we identified the design strategy including a layered architecture where Spring Boot would serve as a framework for the back end where as the front end would be developed using Thymeleaf with HTML, CSSand JavaScript. The data storage would rely on MongoDB Atlas in the cloud environment while Cloudinary would be used for managing the images. Additionally, we implemented a rule-based chatbot using Flask aimed at assisting the users with performing specific actions in the app.

Finally, the integration of all the elements with their testing became crucial at this point of time.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 19, April 2026
Impact Factor: 8.2

V. SYSTEM ARCHITECTURE

The ArtConnect application has a modular architecture. There are four layers in the application, including the frontend, backend, database/storage, and chatbot. The interaction between all layers allows users to enjoy an integrated experience while interacting with their artworks.

The frontend layer comprises the interaction between users and the application using a web interface, where users can sign up and log in to their accounts, upload artworks, browse galleries. User interactions generate requests and responses processed by the application backend.

The backend layer runs the application's core logic, including authentication, request handling, routing, and communication between modules. The backend layer is implemented using Spring Boot, which acts as a controller responsible for communication between frontend, storage layer, and external services.

The backbone of the system is divided into two parts – MongoDB Atlas and Cloudinary. MongoDB Atlas is used as the cloud-based NoSQL database, storing all users' information, artwork meta-information, and data related to applications. Cloudinary takes care of the storage and management of artwork image files in high resolution, facilitating their delivery and reducing server load.

In relation to the chatbot, Flask provides the rule-based application that aids the user in common actions performed on the platform. The bot will provide solutions to questions related to logging in, uploading art pieces, navigating the gallery, etc., making navigation simpler for newbies.

The application is hosted in the cloud thanks to Render, making it more available, scalable, and reliable. The following figure (Figure 1) represents the structure of our architecture along with request handling.

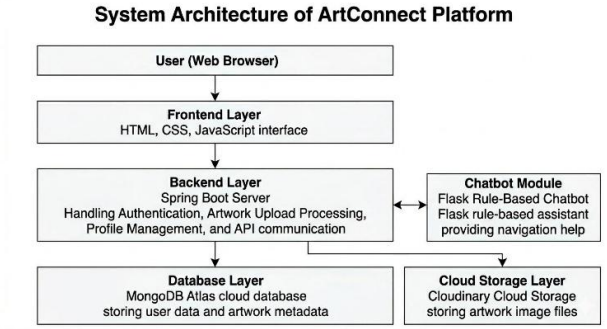

Fig. 1. System architecture of the ArtConnect platform

VI. IMPLEMENTATION

In the ArtConnect system, frontend and backend are combined into a single website. The combination involves interfaces that are easy to use along with the server logic, cloud services, and rules for a chatbot assistant who helps users navigate the system.

The frontend technologies include HTML, CSS, JavaScript, and Thymeleaf. In other words, users can register, log in, upload artworks, browse galleries. All this is made possible by Thymeleaf, which works as the server-side processor rendering templates dynamically according to the user requests and actions from the backend. In turn, Spring Boot is used to develop the backend part. Specifically, its application logic includes handling all types of users' actions on the website such as authentication, navigation across pages, processing uploaded files, interacting with the database, etc.

A module for buyers has also been incorporated to help improve user interaction with works of art. This module allows users to view available artwork pieces, browse more information on the same, and add selected items to their cart. Using the cart, users will be able to select or deselect selected items.

The data layer for our app uses the MongoDB Atlas, which is a cloud-based NoSQL database that contains user credentials, information on artworks, and metadata associated with all uploaded images. Atlas provides high flexibility when it comes to working with semi-structured data as well as scalable and reliable storage. In regard to storing large

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 19, April 2026
Impact Factor: 8.2

image files, we use Cloudinary. Instead of saving them inside the database, all files will be stored in the cloud environment and accessed via secure URLs optimized for delivery. The benefit of such arrangement is decreased load on our servers and increased performance.

The chatbot itself is written with the help of Flask and works in a rule-based mode. In other words, it uses pre-programmed responses to common questions related to how to use the website in terms of registration, artwork uploading, login, etc.

As for the deployment, it is done in the Render cloud environment where the application will run permanently, accessible remotely and hosted on scalable infrastructure.

VII. RESULTS AND DISCUSSION

ArtConnect was created as a fully operational web application that gives artists the capability to upload, organize, and browse their art pieces in one cohesive digital space. This application integrates various tools, including Spring Boot, Thymeleaf, MongoDB Atlas, Cloudinary, and a Flask chatbot assistant, to offer a seamless experience to its users.

Upon testing, the authentication module demonstrated itself as efficient in terms of registering and signing in users. All data related to each specific user were successfully saved into MongoDB Atlas, and access was provided accordingly when uploading artworks and accessing other personalized features. Thus, it was evident that backend services were able to provide a successful response to the user requests. Moreover, the artwork uploading process successfully collaborated with Cloudinary to ensure quick image uploads without lagging.

The URLs for the generated images were also successfully saved in the database, ensuring that there was no need to store images on the server. Thus, image availability was increased significantly through this method.

It can be stated that the buyer module functioned effectively since it allowed customers to put their selected art pieces in the cart and also edit and delete the items from it when required.

The chatbot assistant built using Flask provided the necessary responses to prearranged questions regarding the logging in process, uploading artworks, and the way of navigation across the whole platform. It contributed greatly to usability, particularly for those users visiting this website for the first time. The chatbot performed well during testing and facilitated access to the platform.

Hosting the web application on Render made it available and ensured it could be accessed online without problems. The cloud deployment allowed testing the availability of the system and the successful interaction between all the modules incorporated into it. The deployed application featured the integration between the frontend, backend services, databases, and chatbot.

To conclude, the aim of the project has been reached successfully as the designed platform provides a convenient way for the sharing and management of artwork digitally. Thus, it is evident how effectively the architecture suggested in this paper can be implemented into interactive artwork-sharing applications.

VIII. CONCLUSION AND FUTURE WORK

The paper highlights the development of the ArtConnect web application, which provides artists with the capability to upload, categorize, and browse artwork in a digital format using a graphical user interface. The architecture of the system incorporates the following technological components: Spring Boot is responsible for the backend operations; Thymeleaf, HTML, CSS, and JavaScript are used for the frontend, while MongoDB Atlas manages cloud-based data storage. Additionally, Cloudinary is employed for the efficient handling of images. Finally, the system utilizes a chatbot, which was developed in Flask, to support such functions as login, artwork upload, gallery browsing, and profile management.

The deployment of the web application in the cloud environment revealed its high accessibility and structured interaction capabilities. The uses of cloud technology combined with chatbot-driven navigation can be particularly helpful for new users of the artwork-sharing platform, as it ensures that they do not encounter any difficulties when exploring the system.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 19, April 2026 Impact Factor: 8.2

While the existing solution take care of all the basic need for collaboration and conversation, there are some improvements that can be implemented at a later stage. Improving the interaction of the chatbot using NLP technologies is one of the improvements worth implementing.

Other changes that should be made in order to improve the user experience include personalized artwork recommendations, better search capabilities, and automation of the image sorting process. The future prospects for the app include developing mobile versions and introducing social elements into the app.

The current system supports cart management but does not include payment or checkout functionality, which is considered for future enhancement.

ACKNOWLEDGMENT

My heartfelt thanks to my project guide for the unwavering support and guidance provided during this entire process. Also, my special thanks to the department of Computer Applications at Vels University for providing me with all the necessary resources. Last but not least, I thank my parents and friends for their encouragement.

REFERENCES

[1]. R. Johnson, “Spring Boot fundamentals for web application development,” Journal of Software Engineering, vol. 10, no. 2, pp. 45–52, 2020.
[2]. M. Grinberg, Flask Web Development: Developing Web Applications with Python, 2nd ed. Sebastopol, CA: O’Reilly Media, 2018.
[3]. MongoDB Inc., “MongoDB Atlas documentation,” Available: https://www.mongodb.com/atlas.
[4]. I. Sommerville, Software Engineering, 10th ed. Boston, MA: Pearson, 2016.
[5]. M. Fowler, “Patterns of enterprise application architecture,” Addison-Wesley Professional, 2002.
[6]. P. Dixit and R. Prakash, “Cloud-based web application deployment techniques,” International Journal of Computer Applications, vol. 182, no. 12, pp. 22–27, 2019.