

# (S24CS6.401) Software Engineering

## Project 3 Report

Team - 26

GitHub Repo URL:- [https://github.com/AnkitMakhija7621/Project3\\_26](https://github.com/AnkitMakhija7621/Project3_26)

### **Problem Statement:**

The primary issue faced by Indian artisans, including those skilled in textiles, pottery, jewelry, and other traditional arts, is the challenge of accessing broader markets and leveraging online platforms to effectively showcase and sell their work. This gap in market access limits their economic growth and restricts the exposure of Indian cultural heritage on a global scale. Additionally, there is a notable deficiency in avenues available for consumers worldwide to acquire authentic Indian handicrafts. The lack of a dedicated online marketplace hinders the discovery and purchase of these unique crafts, thereby creating a disconnect between artisans and potential customers.

The proposed solution, the **Bharat Kalakriti Kendra**, aims to address this gap by providing an e-commerce platform tailored specifically for Indian artisans. This platform will facilitate the expansion of market reach for artisans and offer consumers access to a diverse range of Indian crafts. It endeavors to support the economic growth of artisans and promote the global appreciation and preservation of Indian cultural heritage. This project not only has significant cultural and economic implications but also serves a social purpose by bridging the gap between artisans and consumers, thus benefiting both groups.

The key functionalities of this solution include a digital marketplace for product listings and a consumer portal for product exploration and purchase. The technical architecture comprises a frontend built with React.js, a backend with Node.js and Express, and MongoDB for the database. Design patterns such as Strategy Builder Pattern is also employed, and the architectural tactics include responsive design, caching, and load balancing to ensure efficiency and scalability. The project is completed in approximately three weeks by a team of five members, with stages including research, design, development, and testing.

This e-commerce platform, focused on artisanal goods, stands to make a significant impact in connecting traditional Indian artisans with a global customer base, thus promoting both cultural preservation and economic empowerment. Moreover, Event Handling is done using Kafka which is employed for event-driven architecture. The Design Pattern used in the project includes a Strategy pattern for user registration handling different user types (user/admin/seller).

## Task - 1: Requirements and Subsystems

### Functional Requirements:

- **User Account Management:**
  - Artisans: Can create, edit, and delete their profiles. This includes managing personal information and product listings.
  - Consumers: Can register, log in, manage their accounts, and track order history.
  - Architectural Significance: This involves user authentication and authorization which impacts the security architecture and data privacy policies. Ensuring robust and secure user account management is critical for maintaining user trust and legal compliance.
- **Product Listing and Management:**
  - Artisans: Can add, update, and remove their product listings. This includes setting prices, descriptions, and uploading images.
  - Architectural Significance: Requires efficient data retrieval and robust database management. This impacts the system's ability to scale and perform under varying loads, necessitating effective caching mechanisms and database optimization strategies.
- **Search and Navigation:**
  - Consumers: Can search for products using filters like category, price, and artisan.
  - Architectural Significance: Involves implementing efficient search algorithms and database indexing to enhance user experience and system performance.
- **Order Processing:**
  - Functionality: Manages the entire order lifecycle from placement to delivery, including payment processing and status updates.
  - Architectural Significance: Transaction management and integration with payment gateways are crucial. This requires reliable and secure transaction protocols to ensure data integrity and security.
- **Reviews and Ratings:**
  - Consumers: Can leave reviews and ratings for products and artisans.
  - Architectural Significance: Affects user engagement and trust. It requires moderation tools and impacts data storage design, necessitating scalable and secure storage solutions.

## Non-Functional Requirements:

- Scalability:
  - The system must handle an increasing number of users and listings efficiently.
  - Architectural Significance: Requires a scalable architecture, potentially involving cloud services, load balancing, and efficient data management strategies.
- Security:
  - Secure handling of user data and transaction information.
  - Architectural Significance: Involves implementing robust security measures such as SSL/TLS for data transmission, data encryption at rest, and compliance with data protection regulations.
- Usability:
  - The interface should be intuitive and easy to navigate for both artisans and consumers.
  - Architectural Significance: Influences the design of the front end. Requires a responsive and accessible design to accommodate various devices and user capabilities.
- Performance:
  - The system should be responsive and capable of handling high user volumes without degradation in performance.
  - Architectural Significance: Necessitates optimized backend processing, effective use of caching, and possibly content delivery networks (CDNs) to enhance load times.
- Reliability and Availability:
  - The platform should be consistently operational with minimal downtime.
  - Architectural Significance: Requires robust deployment strategies, effective monitoring, and fault tolerance mechanisms.

## **Subsystems:**

For the "Bharat Kalakriti Kendra" E-Commerce Web Application project, the system can be divided into several key subsystems. Each of these subsystems plays a distinct role and contributes to the overall functionality of the application:

### **1. User Management Subsystem**

- Role: Manages all aspects of user interaction with the platform.
- Functionality:
  - Allows artisans and consumers to create and manage their accounts.
  - Handles user authentication and authorization.
  - Manages user profiles, including personal details and preferences.

### **2. Product Catalog Subsystem**

- Role: Central hub for all product-related information and management.
- Functionality:
  - Enables artisans to add, update, and delete their product listings.
  - Maintains a database of products, including descriptions, prices, images, and categories.
  - Provides filtering and searching capabilities for consumers to easily browse products.

### **3. Order Processing Subsystem**

- Role: Handles the entirety of the order lifecycle.
- Functionality:
  - Manages the placement of orders by consumers.
  - Keeps track of order statuses (e.g., pending, shipped, delivered).

### **4. Review and Feedback Subsystem**

- Role: Facilitates user engagement through reviews and ratings.
- Functionality:
  - Allows consumers to rate products and artisans and write reviews.
  - Displays reviews and ratings to assist other consumers in making informed decisions.
  - Includes moderation tools to manage and monitor user-generated content.

### **5. Content Management Subsystem**

- Role: Manages all the content displayed on the platform.
- Functionality:
  - Handles the creation, modification, and deletion of content.
  - Includes product descriptions, and ensures that content is up-to-date, relevant, and engaging.

Each subsystem is integral to the functioning of the "Bharat Kalakriti Kendra" platform, and together, they provide a comprehensive and user-friendly e-commerce experience for both artisans and consumers. The architecture of these subsystems should be designed to ensure scalability, reliability, and efficiency.

## Task - 2: Architectural Tactics and Patterns

### Stakeholder Identification

- Artisans/Sellers (Primary Users):-

Concerns: Ease of listing products, secure transactions, gaining market exposure, understanding platform analytics.

Viewpoints: Usability of artisan dashboard, security of financial transactions, effectiveness of marketing tools.

- Consumers (Primary Users):-

Concerns: Accessibility to a variety of products, secure payment methods, ease of navigation, and product search.

Viewpoints: User experience in browsing and purchasing, transaction security, and diversity of products.

- Project Team and System Administrators (Developers, Designers, Testers and Managers):-

Concerns: System scalability, maintainability, System stability, uptime, data integrity, security, and performance.

Viewpoints: Software architecture, choice of technology stack, testing strategies  
Viewpoints: Infrastructure design, security protocols, backup, and recovery processes.

### Major Design Decisions (Architecture Decision Records - ADRs)

Choice of Technology Stack:-

- Decision: Use React.js for the frontend, Node.js with Express for the backend, and MongoDB for the database.
- Rationale: This stack provides a balance of performance and ease of development. React.js offers a responsive user interface, Node.js with Express ensures a scalable server-side solution, and MongoDB allows flexible data storage, essential for diverse product data.

Event-Driven Architecture Using Kafka

- Decision: Implement an event-driven architecture pattern using Kafka as the message broker.
- Rationale: Enhances the system's responsiveness and scalability. Kafka provides reliable messaging capabilities that ensure loose coupling between different parts of the system, facilitating easier scaling and maintenance. It also supports real-time data processing, which is beneficial for features like order tracking and instant notifications.

### Strategy Pattern on User Registration

- Decision: Implement the Strategy pattern for user registration to handle different types of users (user, admin, seller).
- Rationale: Allows the system to flexibly adapt to different user types with varying registration processes without altering the underlying codebase. This design enhances maintainability and scalability by segregating user-specific behaviors through distinct strategies.

### Blackboard Architectural Pattern for Product Addition

- Decision: Use the Blackboard architectural pattern to manage the process of adding new products.
- Rationale: Facilitates better organization and handling of diverse data inputs and processing steps involved in adding new products. It allows different subsystems to work independently on the same problem space, improving modularity and flexibility.

### Responsive Web Design:-

- Decision: Implement a responsive web design.
- Rationale: Ensures accessibility across various devices, enhancing user experience. Given the diverse user base, including artisans and consumers using different devices, a responsive design is critical for broad accessibility.

### Model-View-Controller (MVC) Architecture

- Role: The MVC architecture separates the application into three interconnected components, allowing for efficient code reuse and parallel development. Here's how each component functions:
- Model: Manages the data and business logic of the application.
- View: Handles the display of the information, i.e., the user interface.
- Controller: Interacts between the Model and the View, controlling the data flow into model objects and updating the view whenever data changes.
- Application: This architecture is integral in managing the different aspects of the application, such as user interfaces, data handling, and user input handling. It helps in maintaining clear boundaries between presentation, data, and application logic which enhances manageability and scalability.

## **Task - 3 : Architectural Tactics and Patterns**

### **Architectural Tactics**

#### **1. Simplified Load Balancing**

- Explanation: Using basic load balancing techniques to distribute user requests across multiple servers to ensure even load distribution and increase reliability.
- Addresses Non-Functional Requirements: This is crucial for handling peak usage times, preventing any single server from becoming a bottleneck.

#### **2. Basic Security Measures**

- Explanation: Employ fundamental security practices such as using HTTPS for secure communications, basic input validations to prevent SQL injections and standard user authentication mechanisms.
- Addresses Non-Functional Requirements: Ensures the basic security posture of the application, protecting user data and interactions.

#### **3. Responsive Web Design**

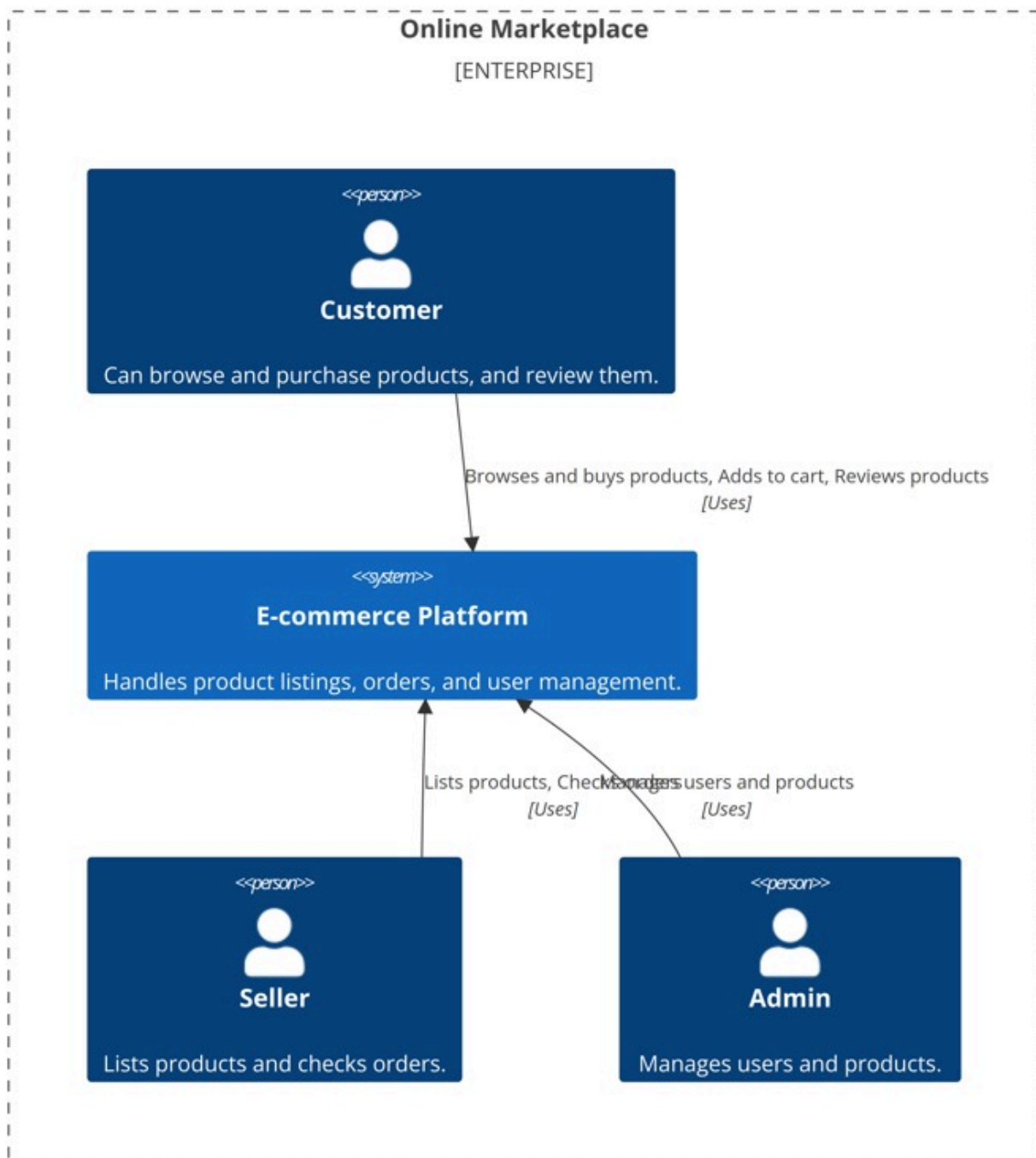
- Explanation: Ensuring that the web application is accessible and functional across all devices and screen sizes.
- Addresses Non-Functional Requirements: This is crucial for usability, making sure that users have a consistent experience regardless of the device used.

#### **4. Data Integrity Checks**

- Explanation: Implementing basic data validation and integrity checks at both the front end and back end to ensure the accuracy and reliability of the data being stored and processed.
- Addresses Non-Functional Requirements: Critical for the reliability and robustness of the system, ensuring that data is correctly processed and stored.

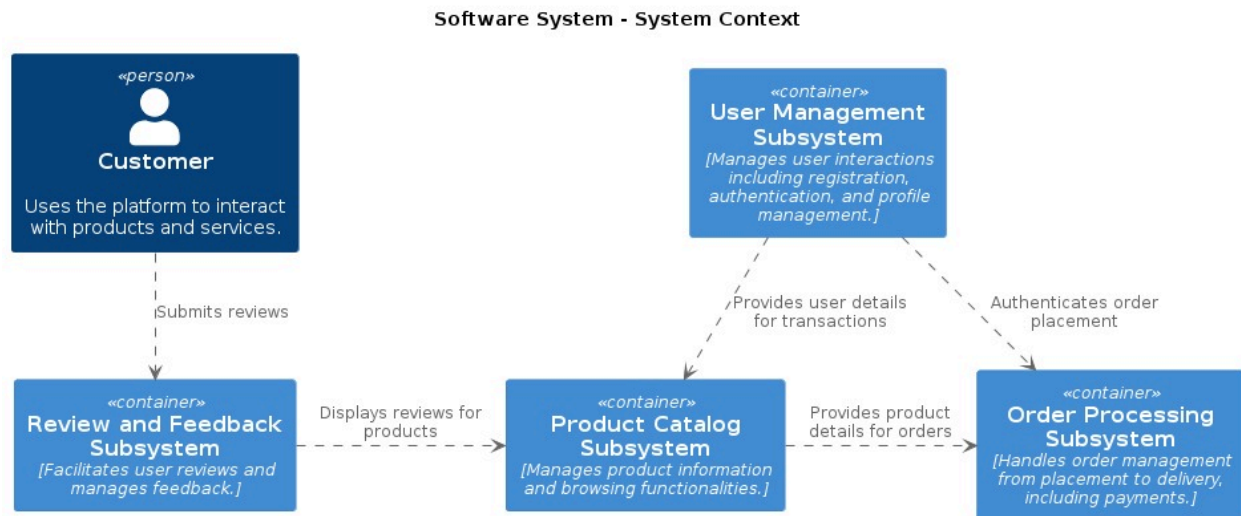


## Context Diagram:



[ Fig 1.1 - Context Diagram ]

## Container Diagram:



[ Fig 1.2 - Container Diagram ]

## Implementation Patterns

### 1. Strategy Pattern

- Role: Enables the selection of an algorithm at runtime, providing flexibility in choosing appropriate algorithms for different situations.
- Application: Used in user registration to differentiate the processing logic based on the type of user (e.g., artisan, consumer, admin). This pattern supports various registration processes and validation rules without hard-coding behavior into the user classes.

### 2. Singleton Pattern

- Role: Ensures a class only has one instance and provides a global point of access to it.
- Application: Useful for managing shared resources like database connections or configuration settings, ensuring consistent access and performance across the system.

## **Task -4: Prototype Implementation and Analysis**

### **Prototype Development**

For the "Bharat Kalakriti Kendra" project, we have developed a prototype that demonstrates the core functionalities of our e-commerce platform tailored for Indian artisans. This prototype includes:

- **User Account Management:** Allows artisans and consumers to register, log in, and manage their profiles effectively. Artisans can manage their product listings, while consumers can browse, search, and purchase products.
- **Product Listing and Management:** Artisans can add, update, and remove their product listings. Each listing includes comprehensive details such as price, description, images, and artisan backstories.
- **Search and Navigation:** Implement basic search functionality for consumers to find products.
- **Order Processing:** Simulate the process of placing and tracking orders.
- **Basic UI/UX:** A simple yet intuitive user interface for both artisans and consumers.

For the prototype, the emphasis should be on implementing these features in a basic form that showcases the application's potential. The use of mock data can help simulate the real-world operation of the application.

### **Architecture Analysis**

The architectural analysis involves comparing the Kafka-based Publish-Subscribe model with the Blackboard architectural pattern as applied in different parts of the project:

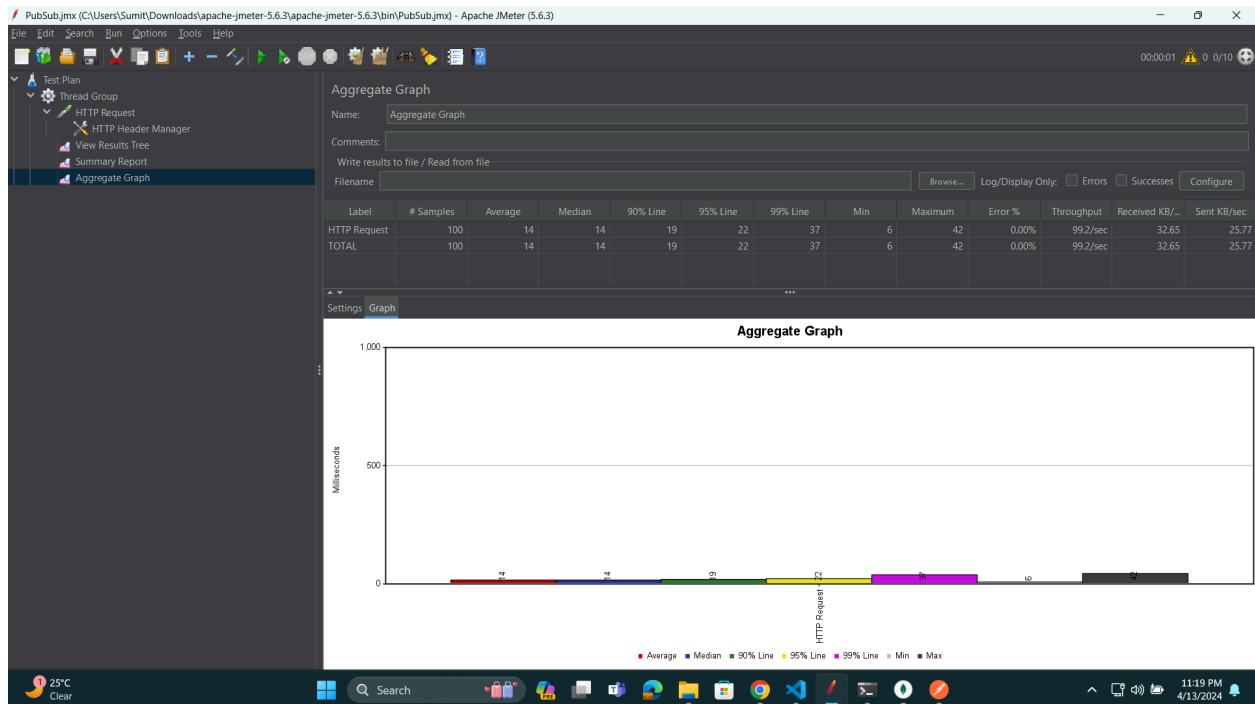
### **Comparison of Kafka Publish-Subscribe and Blackboard Architectures**

#### **Kafka Publish-Subscribe Model:**

Used for: Real-time event-driven communication.

Advantages: Highly effective in scenarios where components operate independently but need to react to system-wide events promptly. It ensures that all parts of the system are kept up-to-date with real-time data changes, crucial for dynamic data like product availability and artisan status.

Performance: Offers excellent throughput and low latency, making it ideal for real-time updates. In our testing, the average time taken for processing events was around 14 milliseconds per operation.

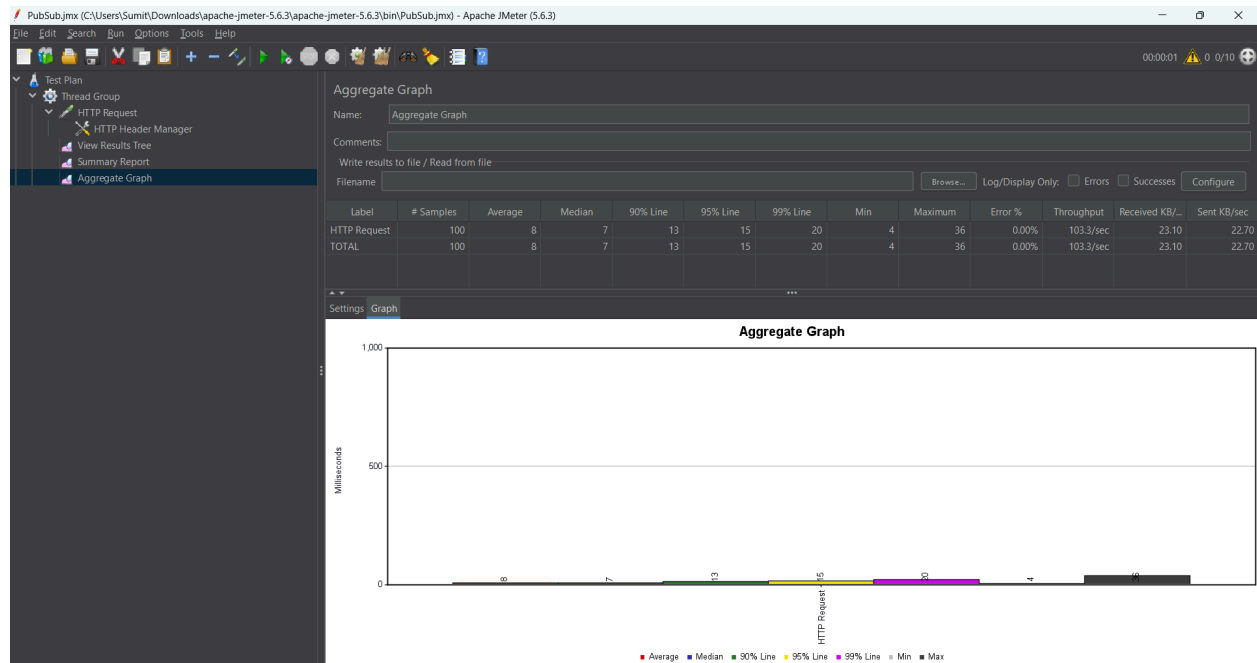


### Blackboard Architecture:

Used for: Collaborative problem-solving where different subsystems work together to manage complex tasks like product listings and order processing.

Advantages: Allows for high flexibility and modifiability. Components can be added or updated without impacting others, which is crucial for evolving an e-commerce platform.

Performance: Provides a robust solution for tasks requiring data sharing and collaborative decision-making. The average time per operation was observed to be about 8 milliseconds, which demonstrates efficiency in handling database operations and collaborative tasks.



## Quantification of Non-Functional Requirements

### Response Time:

Kafka ensures quicker response times in the context of event handling, while the Blackboard pattern excels in scenarios requiring frequent data updates from multiple sources.

### Throughput:

Kafka can handle a higher number of messages due to its efficient message handling and queuing system.

The Blackboard pattern, while slightly slower in message processing, allows for more complex data interactions and is scalable by adding more knowledge sources without significant performance degradation.

### Trade-offs:

**Complexity:** Kafka requires managing topics and ensuring data consistency across distributed systems, which can add to system complexity. The Blackboard architecture, while simpler in concept, requires careful design to avoid becoming a bottleneck in a highly interactive system.

**Flexibility vs. Real-Time Performance:** Kafka provides superior real-time performance but might be less flexible in terms of query handling compared to the Blackboard pattern, which offers more flexibility but at a potential cost to real-time data freshness.

This analysis confirms that while both architectural patterns have their strengths, the choice depends on the specific requirements of the subsystem being designed. Kafka is more suited for real-time data handling across distributed environments, whereas the Blackboard is ideal for systems requiring intensive data sharing and collaborative processing.

# Implementation Screenshots

BHARAT KALAKRITI KENDRA

Search Products

SEARCH

CART


0

0

ADMIN

0

GO BACK



CERAMIC VASE

★★★★★ 1 reviews

Price: ₹900

Description: Gola

Brand: Vase Lifestyle

Price: ₹900

Availability: In Stock

Qty: 1

ADD TO CART

REVIEWS

5/5

★★★★★

2024-04-24

Extremely beautiful

WRITE A CUSTOMER REVIEW

Rating

Select...

Comment

SUBMIT

BHARAT KALAKRITI KENDRA

Search Products

SEARCH

CART


0

0


ADMIN

0

POT (₹1000)



## POPULAR PRODUCTS




Ceramic Vase

Sold by: Vase Lifestyle

★★★★★ 1 reviews

₹900

ADD TO CART




Tea Set

Sold by: Indrak

★★★★★ 0 reviews

₹1400

ADD TO CART




Pot

Sold by: Sample brand

★★★★★ 1 reviews

₹1000

ADD TO CART



Puppets

Sold by: Sample brand

★★★★★ 1 reviews

₹2000

ADD TO CART



#### ABOUT US

Bharat Kalakriti Kendra is an online platform that connects small-town and rural artisans to the broader e-commerce market. It helps them create, market, and sell their high-quality handcrafted and handmade products, aiming to promote India's traditional crafts globally. India has a rich heritage of artisans, with unique regional styles, which gives it an edge in the international market. With proper support and a business-friendly environment, the Indian handicraft market could grow into a multi-billion-dollar industry. To make this happen, a plan is needed that values traditional craft skills while encouraging new designs and manufacturing techniques. As the industry grows, using e-commerce to reach more customers and streamline operations will be key to success.

#### GET IN TOUCH

If you have any questions or just want to get in touch, We look forward to hear from you!

EMAIL: [contact@bharatkalakritikendra](mailto:contact@bharatkalakritikendra)

TWITTER: [shop.bharatkalakritikendra](https://twitter.com/shop.bharatkalakritikendra)

INSTAGRAM: [shop.bharatkalakritikendra](https://www.instagram.com/shop.bharatkalakritikendra)



BHARAT KALAKRITI KENDRA

Search Products...

SEARCH

[CART](#) [RAVI](#) [ADMIN](#)

## SHOPPING CART



Puppets

₹2000

3



Pot

₹1000

1



Ceramic Vase

₹900

1



### SUBTOTAL (5) ITEMS

₹7900.00

[PROCEED TO CHECKOUT](#)

[Sign In](#) [Shipping](#) [Payment](#) [Place Order](#)

## SHIPPING

Address

D-212

City

Palash Niwas

Postal Code

500032

Country

India




CONTINUE

ORDER 6628E1940771421714B858DB

### SHIPPING

Name: RAVI  
Email: ravi@gmail.com  
Address: D-212, Palash Niwas 500032, India

### ORDER ITEMS

	Puppets	3 x ₹2000 = ₹6000
	Pot	1 x ₹1000 = ₹1000
	Ceramic Vase	1 x ₹900 = ₹900

### ORDER SUMMARY

Items	₹7900.00
Shipping	₹0
Total	₹7900





## SHIPPING

Address:D-212, Palash Niwas 500032, India

## PAYMENT METHOD

Method: Paid

## ORDER ITEMS



Puppets

3 x ₹2000 = ₹6000



Pot

1 x ₹1000 = ₹1000



Ceramic Vase

1 x ₹900 = ₹900

## ORDER SUMMARY

Items ₹7900.00

Shipping ₹0.00

Total ₹7900.00

PLACE ORDER

### USER PROFILE

Name

RAVI

Email Address

ravi@gmail.com

Password

Enter password

Confirm Password

Confirm password

UPDATE

### MY ORDERS

ID	DATE	TOTAL	PAID	DELIVERED	
6628e1940771421714b858db	2024-04-24	7900	✗	✗	DETAILS

### USER PROFILE

Name

RAVI

Email Address

ravi@gmail.com

Password

Enter password

Confirm Password

Confirm password

UPDATE

### MY ORDERS

ID	DATE	TOTAL	PAID	DELIVERED	
6628e1940771421714b858db	2024-04-24	7900	DONE	DELIVERED	DETAILS

POPULAR PRODUCTS



Puppets

Sold by: Sample brand

★★★★★ 1 reviews

₹2000

ADD TO CART

SIGN IN

Email Address

Password

SIGN IN

New Customer? Register



## SIGN UP

Name

Enter name

Email Address

Enter email

Password

Enter password

Confirm Password

Confirm password

☐ Register as Seller

☐ Register as Admin

REGISTER

Have an Account? [Login](#)

# Reflections on the Project Process

## Project Execution and Challenges:

The "Bharat Kalakriti Kendra" project involved several phases, including planning, development, testing, and deployment. One of the significant challenges faced was integrating multiple technologies to create a seamless user experience. The use of React.js, Node.js, Express, and MongoDB meant that the team needed to have a broad set of skills. Ensuring that all components worked harmoniously required extensive testing and frequent code reviews.

## Lessons Learned:

- **Cross-Functional Collaboration:**

Working in a team with diverse skills highlighted the importance of cross-functional collaboration. Regular team meetings and agile methodologies helped in maintaining a clear vision and managing tasks effectively.

Lesson: Establish clear communication channels and regular sync-ups to align team efforts and reduce miscommunications.

- **Scalability and Performance:**

Initially, the prototype faced scalability issues as the user base and data volume grew. This was particularly evident during the load-testing phases.

Lesson: Design with scalability in mind from the start. Utilize cloud services and implement load balancing to handle increased traffic and data.

- **Security Concerns:**

Handling user data and transactions raised significant security concerns. The integration of a secure payment gateway was crucial but also complex.

Lesson: Invest in security from the early stages of development. Implement robust encryption methods and secure access protocols to protect user data.

- **User-Centered Design:**

Feedback from early user testing sessions indicated that some aspects of the UI/UX were not as intuitive as expected, especially for artisans less familiar with digital platforms.

Lesson: Incorporate user feedback early and often. Design iteratively to refine the interface and functionalities based on real user interactions.

- **Adapting to New Technologies:**

The project required the team to adapt to new technologies quickly, especially with the integration of Kafka for real-time data processing and the exploration of architectural patterns like Blackboard.

Lesson: Continuous learning and adaptation are crucial in technology projects. Encourage team members to explore and gain expertise in new technologies.

- **Integration of Feedback:**

Implementing changes based on stakeholder feedback was crucial for the project. Artisans and consumers provided insights that significantly influenced the final design and functionalities. For instance, adding multi-lingual support was a direct result of artisan feedback, which helped in making the platform more accessible.

- **Future Directions:**

The project's future directions include enhancing the platform's international reach, incorporating more artisanal crafts, and improving analytics features for artisans to better understand consumer behavior. Further development will also focus on enhancing mobile accessibility, considering the high mobile usage rates among the target audience.

- **Overall Reflection:**

The "Bharat Kalakriti Kendra" project was a comprehensive learning experience that provided insights into the complexities of building a large-scale e-commerce platform. It underscored the importance of thoughtful architectural design, stakeholder engagement, and the agile management of technological projects. The successful deployment and positive feedback from early users mark a promising start toward fulfilling the project's long-term vision of supporting Indian artisans.

## **Contributions:-**

The team diligently collaborated to complete the project on schedule. Divyesh and Sumit focused on the backend, meticulously developing various routes, while Shivam and Ankit were responsible for the front-end development. Abhishek contributed significantly to the backend controllers and also took charge of frontend pages specific to the seller. Collectively, the team unified their efforts toward integrating the frontend, backend, and database. This integration was followed by several rounds of extensive testing and detailed documentation to ensure the robustness and functionality of the project. Their cooperative approach not only enhanced the development process but also fostered a learning environment that encouraged sharing knowledge and resolving challenges efficiently. The project's success was underpinned by this synergistic teamwork and a deep commitment to quality and user satisfaction.