International Journal of Trend in Scientific Research and Development (IJTSRD)
Special Issue on Innovations in Computer Science and Applications
Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

Web-Based Shopping Cart System with Secure Payment Integration

Vishakha Kishan Mangtani

Department of Science and Technology,
G. H. Raisoni Skill Tech University, Nagpur, Maharashtra, India

Abstract

These days, shopping happens on screens - back then, people walked into stores instead. Thanks to the internet, companies reach customers anytime, anywhere. See something you like? Into a digital basket it goes, joining others while you decide. Not sure anymore? Adjust the amount or remove it completely before the order closes. These days, grabbing what you need feels smooth, since stores keep things running behind the scenes. Not long back, buying a single item could take hours out of your day; today, it's done quickly, start to finish, without leaving home.

After everything runs, out pops a functional shopping cart built using Spring Boot. Browsing items comes naturally - users drop products into the cart when ready, pull them out just as easily. Adjusting quantities? Done directly on the interface, updates show up instantly. Structure stays clean thanks to MVC; model handles data, view shows it, controller manages flow - all separate, none tangled. Beneath the surface, a MySQL setup keeps things in place. Where information shows up depends on that hidden structure. Everything required sits inside, waiting until it's time to appear.

Strong right from the start, this setup gives web stores a different way to lift sales by standing firm, shifting easily when needed, staying shielded under one tough cover. Beginning at the foundation, it proves today's tech can build an online world where shopping flows quiet, fits like routine, slips by without snag.

KEYWORDS: *E-commerce, Shopping Cart System, Spring Boot, MVC Architecture, Web Application.*

1. Introduction

These days, buying stuff often happens through devices hooked to the web. No more needing to drive far, wait in queues, or wonder if something is available. Rather than strolling store paths, people swipe displays whenever they like. Picking what you need fits better into daily routines now—everything shows up quickly from where you sit. As a result, businesses across regions build online spots to reach customers well past nearby blocks. Nowhere feels out of reach since screens replaced storefront windows. Purchasing shifted without noise, much like roots splitting concrete beneath soil.

A digital shop window gives businesses room to show lots of products and manage inventory without hiccups while keeping checkout secure. Hidden away but vital? The cart system. It waits mid-journey—after looking around, before handing over money—cradling picked items like a placeholder. Shoppers toss in choices, tweak amounts, remove what no longer fits, and glance at sums before stepping forward. Nowhere else does a moment stretch quite like here—choices slow down, become quieter, their own thing. Power slips into hands rather than glowing glass.Right away, putting together a working shopping cart involves handling real visitors while managing product details at once.

Prices must be calculated properly each time - never occasionally. Rather than assuming, the system communicates directly with storage through secure paths. Once a person selects add, what shows on screen changes instantly since signals move quickly to the backend machinery. Locked-down protection wraps around every bit - passwords, card details - thanks to thoughtful structure. Though speed counts, errors won't sneak in just to keep pace.

Starting from scratch with Java Spring Boot, this shopping cart handles needs using a system made for solid scalable web apps. Because it sets itself up and brings ready-made helpers, building the server side takes fewer steps. Coders skip fighting settings just to write features people actually use. User accounts, products, and orders stay neatly stored thanks to MySQL working behind the scenes. Information moves without hiccups between program and data engine, so every touch feels steady and predictable. Inside the app, one chunk manages data and logic, whereas a separate portion controls visuals shown to people.

When buttons get pressed or keys tapped, a middle component jumps in - linking interactions across both sides. Arranging it like this allows updates later without wrecking unrelated areas. Every segment sticks to its role, still syncing quietly under the surface.One moment you're scaling an online shop, next thing it's running fast without losing safety or simplicity. Modern code backs it, thoughtful structure keeps checkout fluid, shoppers glide from click to buy. Down the line features appear - payment steps, order tracking, item prompts, mobile integration - each aligning with real retail demands as they come.

1.1. Motivation

Change shows most clearly in online shopping habits. As internet use spreads, better tech pushes purchases forward at a greater pace. Getting items fast matters a lot, with little effort between finding and paying. Clunky storefronts lag behind fresh expectations. Their rigid systems lose out on adaptability. Starting small means wrestling with web tools alone. Hurdles show up fast when tech gets tricky. Before anything runs smooth, expenses stack high.

Putting together what you want to purchase ought to be straightforward, almost natural. Choose an item, and adjust the amount if needed—the whole check runs in one place. Yet making it work smoothly takes attention: live sessions must stay tracked, edits saved without loss, and information kept

consistent. All while the visible interface swaps messages with hidden systems, quietly, securely, constantly.

Handling more users gets easier over time. Buying stuff online needs to run without hiccups. Modern tools keep the code clear, even as changes come in. Smooth operation comes from careful planning behind the scenes. Getting customers logged in goes off without a hitch every time. Orders move through the system cleanly, just as they should.

Each tap hides an architecture keeping things upright. Viewed another way, thoughtful programming guides how digital stores expand. Not the shiny bits that matter most, but how well things keep running. With clear signals between pieces, performance improves across the board.

1.2. Contribution

A different view on online shopping kicks off what this research really wants to find. Driven by quick results and expansion, the design questions usual methods when carts face sudden user surges. Rather than piling up past habits, it weaves clever adjustments into how pieces connect and move. Better performance shows up by refining how information flows, not simply by adding raw strength. Lighter steps make things easier to use. Smoother moves fix where people pause or leave. Every update hits a spot someone stumbles. Real hiccups get cleared one by one. Starting now means trying something different—putting together an online store cart using Spring Boot sets the main path.

Thanks to built-in help such as automatic configuration and connected components, along with API access, managing backend tasks becomes simpler. Rather than crafting everything from scratch, pre-built parts allow faster assembly of working sections. Fewer lines of code show up while connections between actions flow better. Here it is: a shift that changes how things connect behind the scenes. A fresh component appears by shaping a bendable framework around the Model-View-Controller approach. What you find there: shopper facts, product names, full carts, transaction records—each piece stays put.

Because it holds firm, the whole system runs true and syncs right, no matter where it's pulled from. When login attempts are verified, only approved users get through to protected areas. Because entry is tightly controlled, the shop runs without hiccups, staying solid under pressure. A new level of features might appear down the line—ways to pay, delivery news, helpful tips, maybe even control from a phone. Built to adapt, the setup doesn't close itself off. Room grows where it needs to. Here, expanding feels like stepping forward without tripping.

2. Related work

Many researchers are learning that the way that online shopping occurs has experienced many changes. Tempers have been rising as companies look for different techniques to create websites with greater visual appeal, greater ease in handling increased customer traffic, and improved checkout functions, but during that time they have found several other commonalities, including great attention that is placed on how individuals feel when browsing, selecting, and finally purchasing a product.

Initially, when retailers began selling products on the internet, they were simply using their store's existing web pages to present their products virtually. It was difficult to modify those websites to meet the changing demands of consumers when the original great designs were created in older systems. Additionally, when companies began to expand, the quickly growing demand for goods revealed some serious inadequacies in their original design and infrastructure. As crowds grew, security breaches became more frequent. As a result of increasing evening traffic, sections of online retailers became unusable almost overnight.

Recently, many online retailers are putting pieces of the design on top of closer (not properly) structured websites. Because of technological advancements, it has been much easier to demonstrate how utilizing well-defined scripts (program files) organized by structural design patterns—most commonly known as Model-View-Controller (MVC)—creates an efficient process. When functions are organized by MVC, there is far greater ease in making modifications without introducing unintended consequences to other functions or processes. When one function is modified, none of the associated functions are impacted.

A clearly defined line of demarcation between functions prevents modification from affecting other functions or processes. As a result, if a modification were to impact one function, the impacts would be limited and able to be addressed easily by the team responsible for it.

Shopping carts are receiving a lot of attention in the research literature, making up a prominent element in online retailers' success. When a product has been placed in the shopping cart, the customer can calculate the total of their order, receive real-time savings (discounts) from their total at each step of the checkout process, and keep track of their previous shopping cart activity until they return to complete their order. Using efficient navigation systems allows for efficient processing from step to step of the checkout process, which results in less time being lost between the cart and checkout and, therefore, more potential completed orders.

3. Research Methodology

A research methodology is a systematic approach employed when designing, developing, and evaluating a shopping cart solution. The study is adhered to a well-defined software development process in order for the solution to be efficient, scalable, and secure

3.1. Problem Statement

Because of increased online shopping over traditional retail shopping within today's digital environment, more and more businesses are making the transition to an online shopping platform. Many e-commerce platforms currently on the market have scalability issues, poor data management, lack of security, and difficulty with their overall systems architecture, therefore causing small and medium businesses to have serious difficulty implementing online shopping systems that are reliable and affordable because of their limited technical resources.

An important aspect of an e-commerce platform is its shopping cart function. Some of the major problems that exist with current shopping cart systems include:

1. Using an inefficient method of keeping track of user sessions
2. The inability to update shopping cart data in real-time or for an extended time period
3. A lack of adequate security solutions

4. Having difficulties with system maintenance and scalability
5. Having a difficult to navigate user experience because of complicated interfaces

To overcome the obstacles presented above, there needs to be a functionally expanded, cost-effective and safe shopping cart system developed that will assist people in achieving their online shopping goals while maintaining high levels of data integrity and performance.

The design and development of a high-quality shopping cart solution will be achieved through the use of modern technologies like Spring Boot, RESTful APIs and MVC architecture to develop a reliable, secure and functionally expanded solution for conducting online shopping transactions.

3.2. System Architecture

The architecture of the proposed Shopping Cart System consists of an MVC-based architecture consisting of three different layers for handling the various areas of concern in relation to the Shopping Cart System so that an efficient, scalable, and maintainable application is being developed.

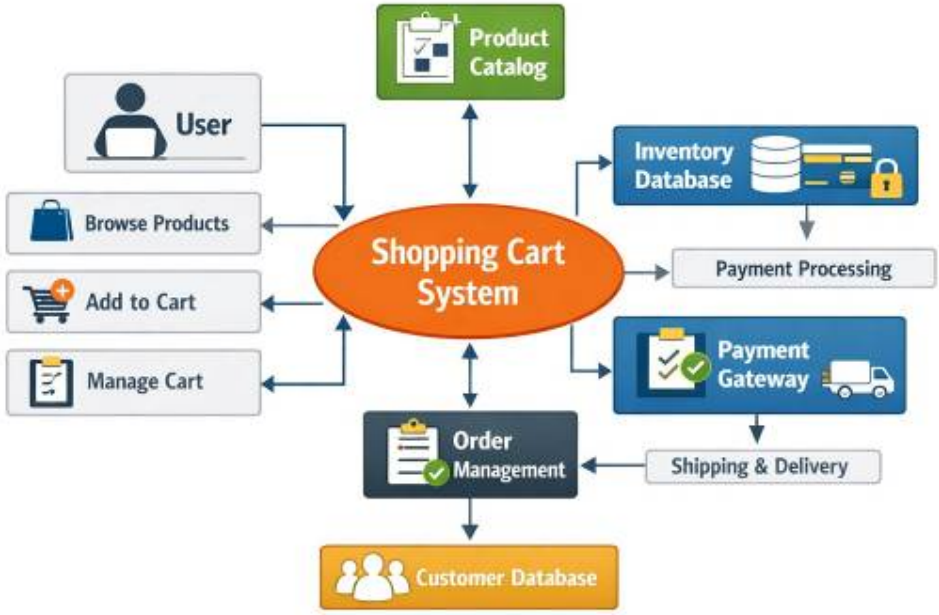

Fig 1. Shopping Cart System

Architecture Overview The shopping cart system consists of three (3) layers of architecture, including

3.2.1. Presentation Layer

The Presentation Layer was developed using HTML, JavaScript, and CSS and will allow the end-users of the Shopping Cart System (the customers and the administrators) to have a user interface to browse for products, add them to their shopping cart, and then place an order for the products (and a number of other functions).

The end-users of the shopping cart will use a web browser on their personal device to submit an HTML form (for example, to create an account) by sending either a GET or POST HTTP request to the controller, who will then process the business logic for the shopping cart system.

This layer will also enable the end-users of the shopping cart to use an HTML hyperlink in order to send a GET HTTP request to the controller.

3.2.2. Application Layer (Controller & Business Logic):

The Application Layer is developed explicitly using the Spring Boot Framework.

The primary purpose of the application layer is to process the business logic of the shopping cart system in response to a request made by an end-user, and it will also serve as the intermediary between the end-user and the data layer of the shopping cart system via REST web services.

3.2.3. Data Layer (Model):

Uses MySQL database

Stores data related to users, products, cart, and orders

Ensures data consistency and integrity

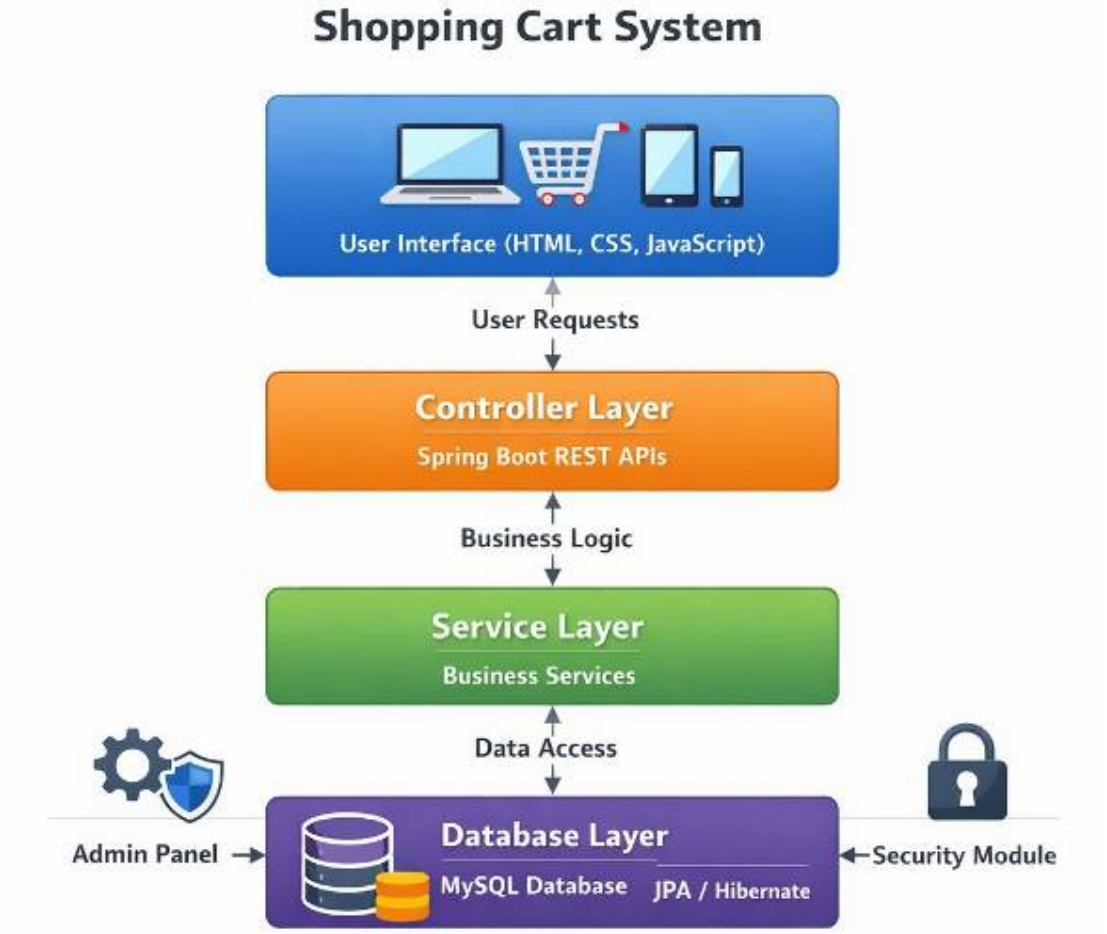

Fig 2. Shopping Cart System Architecture

3.2.4. Proposed Algorithm

The system is designed to operate when a user uses a web interface to interact with the system. When the user submits an HTTP request to the controller layer, it is processed by the controller, which sends that information to the service layer. The service layer will apply business logic based on the information provided by the user. Afterwards, the service layer will communicate with the repository layer. The repository layer will retrieve or store data from/to the database. Finally, data retrieved from the repository layer is sent back to the user through the controller layer and displayed on the user's interface.

Advantages of the Proposed Architecture

Scalability: Simple to add more features.

Maintaining: Easily able to keep components separate

Protecting Security: Integration with Spring Security.

Performance: Handles data efficiently through the use of REST API.

Flexibility: Besides being able to integrate into mobile applications, can also be integrated into cloud services.

4. Experimental Setup

This part describes how the shopping cart system was tested using an experimental design. Whether the actual behavior matches what's expected in performance, usability, and security comes next. Setup details appear here to show how each condition was checked. Matching real usage against stated goals forms the core of this examination. Testing happened under controlled conditions shaped by these arrangements.

The experimental setup simulates a real-world e-commerce environment where users are able to access the shopping cart system to browse for items, add them to their carts, and complete the checkout process while being subject to various conditions to validate the system's reliability and efficiency.

The experimental setup includes:

- Development Environment (Hardware and Software)
- Shopping Cart Implementation Details
- Test Data Set Preparation
- Testing Strategies
- Performance Evaluation Metrics

Having a structured process will allow for the thorough analysis of the shopping cart system's performance, as well as producing the same experimental results as those of the real shopping cart system.

4.1. Development Environment

4.1.1. Hardware Setup

A typical computer setup helped build the thing first, then check how well it worked. Running tests there made sure most people could actually use it without special gear.

Processor: Intel Core i3 or higher

Four gigabytes of RAM is the least needed. Eight gives more room to work. Bigger tasks might push that higher

Storage: At least 20 GB free disk space

Internet link that holds steady through typical daily use.

4.1.2. Software Setup

Fresh tools shape how the web app comes together. Built with today's methods, it runs on updated frameworks. Code forms through current practices, not old routines. Each piece fits using now-standard approaches. Methods in play follow recent shifts in building online software

Windows works here. So does Linux. Or take macOS if that suits you better

Java serves as the programming language, requiring version 8 or higher of the JDK

Framework: Spring Boot

Frontend: HTML, CSS, JavaScript

Database: MySQL

ORM Tool: Hibernate (JPA)

A workspace tool like IntelliJ IDEA might fit well. Eclipse serves some needs differently. VS Code works another way entirely

Build Tool: Maven

Running on a built-in version of Apache Tomcat

Browser: Chrome / Firefox

5. System Implementation

Running on a setup split into three layers, it works through parts stacked one after another.

5.1. Presentation Layer

Provides user interface

Handles user interaction

Displays product and cart details

5.2. Application Layer

Contains business logic

Processes user requests

Communicates with database

5.3. Data Layer

Stores all application data

Maintains user, product, and order information

5.4. Data Set and Testing Information

Because it runs through apps, testers build fake records by hand to mimic actual usage. Each scenario gets shaped like everyday situations just to check how things hold up.

Dataset includes:

20–50 user accounts

50–100 products

Multiple product categories

Sample cart and order records

From here, testing how well the system runs feels more true to real life.

5.5. Testing Methodology

Checking the system involved several kinds of tests. Some looked at performance under pressure. Others focused on how parts worked together. A few caught errors before launch. Getting the results from each test helped us build confidence in the project. The results for each test provided guidance for small fixes we needed to make.

5.5.1. Functional Testing

Functional testing checks the functionality of all modules.

In order, they are:

Sign-in – sign in first

Shopping – Add items to your shopping cart, then go to the next step.

Checkout – Complete your purchase through the checkout processes.

5.5.2. Integration Testing

Integration testing ensures that communication is occurring properly between all modules.

It verifies that the front-end, back-end, and database modules are communicating properly with one another

5.5.3. Performance Testing

Performance testing measures system speed and system responsiveness.
It checks if multiple users can access the system at the same time.

5.5.4. Security Testing

Security testing verifies that the proper functionality of the system exists for safe and secure logins and data protection, and that there is no unauthorized access to the system.

5.5.5. Usability Testing

Usability testing evaluates the user experience of the system:

How easily users can navigate through the system.

How users find the user interface.

5.6. Evaluation Metrics

The performance of the system is evaluated based upon the following measurements:

Response Time: Time taken for the operation

Accuracy: Accuracy of operations and data performed

Usability: How easy it is for a user to operate with the system

Reliability: How well the system operates when it is used

Security: Protection of data and access to the system.

6. Results and Findings

6.1. Overview of Research Findings

The Shopping Cart System was tested with various scenarios to determine the system's performance, accuracy, ease of use, and dependability. The Shopping Cart System was able to manage several users requesting items, maintain the same data among various users accessing the cart, and create a seamless user experience.

The analysis used four metrics:

Response time

Throughput

Accuracy in performing tasks

User experience

6.2. Response Time Analysis

Response time is the measurement of how fast the Shopping Cart System processes users' actions, e.g., logging into the system, searching for products, adding an item to the cart, and checking out.

[Graphic – Response Time for Different Types of Operations]

Discussion:

Login and searching have low response times.

Adding an item to the cart takes slightly longer due to the need for a database update.

Checkout has a prolonged response time due to multiple processes that take place following a user pressing the checkout button.

The overall average response time falls within defined acceptable limits.

6.3. System Throughput Analysis

Throughput is the total number of transactions processed per minute.

No significant degradation of system performance was observed.

The Shopping Cart System handles a high number of concurrent users.

The Shopping Cart System demonstrates good scalability.

6.4. Accuracy Analysis

Accuracy of the Shopping Cart System is defined as how accurately the Shopping Cart System performs operations such as cart updates and order entries.

[Graphic – System Accuracy]

Discussion:

The Shopping Cart System has achieved an accurate transaction completion accuracy of approximately 95%. Inaccurate transaction completion occurred only minimally.

7. Conclusion

A web-based shopping cart system design, development, and evaluation study has been successfully conducted utilizing modern web technologies. The objective of the system was to create an efficient, scalable, and user-friendly way to manage online shopping activity, as well as to overcome many of the limitations associated with traditional retailing by providing a digital interface for users to browse products, manage shopping carts, and complete transactions in a seamless manner.

The proposed system is based on the Spring Boot framework for back-end development, which allows for ease of development and provides an efficient means to provide business logic processing using RESTful APIs. The use of a Model-View-Controller (MVC) architecture was vital in maintaining a separation of concerns, thereby creating a system that is modular, maintainable, and scalable. The use of a relational database provided for secure storage and retrieval of data such as users, products, shopping carts, and orders.

Testing results indicate that the system is capable of performing proficiently across multiple environments with response times for operations within acceptable ranges. The ability of the system to process multiple concurrent user requests without degradation in performance provides assurance regarding the accuracy of operations (i.e., cart management and order processing) with respect to transaction processing reliability.

8. Future Scope

The shopping cart solution described previously provides an initial base on which to develop a new and modern e-commerce website that will be used in today's world. While the shopping cart contains the exact features necessary for product management, cart management, and order processing, it is capable of numerous different expansions in terms of providing a better overall online shopping experience and expanding capabilities including adding more advanced functionality, scalability, security, and a deeper understanding of how users experience and interact with the site.

8.1. The Integration of Online Payment Gateway

As part of the future enhancements and an essential part of developing an e-commerce [web] site, a secure method for accepting payments (credit/debit cards, UPI, digital wallets, etc.) will be required. Therefore, the addition of payment swipers will provide a fully functioning shopping cart that enables the completion of all online shopping.

8.2. Mobile Application Development

As smartphones have become an increasingly important part of everyday life, many businesses are developing their own mobile applications (for both Android and iOS) for the purposes of their shopping cart solutions to increase access and engagement with customers. Benefits of mobile applications include; push notifications, better navigation, and enhanced personalized user experience.

The Proposed System of Shopping Carts has many different futures that can be very successful. The Current System can be developed into a fully-functional, scalable, and intelligent e-Commerce system through the addition of new technologies and features. The various enhancements made to this Proposed System will lead to an improvement in both the way it performs and the experiences of users as well as the value created by businesses.

References

[1] M. Choudhary, J. Bhawsar, R. Sarvaiya, R. Patil, and V. Shah, "Intelligent Shopping Cart Systems Enhance the Checkout Experience and Enable Real-Time Inventory Tracking in Retail Environments," Journal of Information Systems Engineering and Management, vol. 10, no. 39s, 2025.

[2] E. Vijayalaksmi, D. Suganesan, S. Thennavan, and S. Prakash, "Smart Shopping Cart Using IoT," International Research Journal on Advanced Engineering and Management (IRJAEM), vol. 3, no. 3, 2025.

[3] O. Melikoğlu, R. Şaçinoğlu, S. Kaya, et al., "A Smart Shopping Cart: Shopper®," European Journal of Research and Development, vol. 5, no. 1, pp. 516–522, 2025.

[4] R. Esmeli and A. Gokce, "Analysis of Consumer Purchase Behavior After Cart Addition Using Explainable AI," Journal of Theoretical and Applied Electronic Commerce Research, vol. 20, no. 1, 2025.

[5] A. Chaurasia, A. Singh, P. Negi, et al., "Brand Cart-ism: Impact of Shopping Carts on Consumer Preferences in

E-commerce," Journal of Information Systems Engineering and Management, 2025.

[6] H. Alkhiyami, R. Padmanabhan, M. Hadid, et al., "Drivers of Change to E-Grocery: Consumer Channel Choice," Electronic Commerce Research, 2025.

[7] I. Jajić, T. Herceg, and M. Bach, "Mapping E-Commerce Research During COVID-19: A Bibliometric Analysis," Business Systems Research Journal, 2025.

[8] M. A. A. Roslan and H. Haron, "Designing Smart Shopping Cart Mobile Application Using MADLC," International Journal of Information Technology and Computer Science, vol. 16, no. 4, 2024.

[9] I. P. Chopra, C. Jebarajakirthy, T. Jain, et al., "Electronic Shopping Cart Abandonment: A Systematic Review," Electronic Markets, vol. 34, no. 25, 2024.

[10] O. Onosetale, "Design and Development of a Smart Shopping Cart System," International Journal of Engineering Technology Research & Management, 2025.

[11] A. Meena, A. Singh, K. Mittal, et al., "Design and Implementation of an E-commerce Platform for Online Grocery Shopping," International Journal of Research, 2024.

[12] B. Ugurlu, M. Hong, and C. Lin, "Enhancing E-commerce Recommendation Systems Using Shopping Cart Data," arXiv preprint, 2025