International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 04 | April - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

AI ShopSmart: The Development of an AI-Powered E-Commerce Platform Built on the MERN Stack, Groq AI, Real-Time Analytics, and Gamification

Satasiya Kishan Vijaybhai[1]

[1]Department of Computer Science and Engineering, Parul University, Vadodara, Gujarat, India

Abstract -

Built on the MERN stack (MongoDB, Express.js, React, Node.js), ShopSmart AI is a production-ready, full-stack ecommerce web application whose whole design, development, and deployment are covered in this article. Groq AI (LLaMA 3.3 70B model) is integrated to provide intelligent features including a conversational AI chatbot, intelligent product search, budget planning assistant, review summarization, and AI-powered product comparison. The system has thorough ecommerce capability including product catalogue management, purchasing cart, order tracking, multi-step checkout, and an admin dashboard with JWT-based user authentication. Beyond conventional e-commerce, ShopSmart AI combines gamification features-daily spin wheel, loyalty points system with tier progression, and referral scheme with dark/light theme switching, animated UI with Framer Motion, PWA capabilities for offline access, and mobile-responsive bottom navigation. At no cost, the React frontend is hosted on Vercel while the backend REST API is deployed on Render.com. Through Nodemailer, email alerts are automated. System design, technology selection criteria, feature implementation specifics, UI/UX design choices, recorded known bug fixes, deployment process, and forward scope are all addressed in this paper.

Keywords: MERN Stack, Artificial Intelligence, Ecommerce, React, Node.js, MongoDB, Groq AI, Framer Motion, REST API, PWA, JWT Authentication, Gamification, Full-Stack Development

1. Introduction & Problem Statement -

1.1 Introduction

India's fast online commerce expansion has generated demand for smart, customized shopping platforms that go beyond only product listings. Conventional eCommerce sites lack AI-driven customization, engaging user experiences, and real-time smart assistance. By integrating the tried-and-true MERN stack with cutting-edge AI language models, ShopSmart AI was created to cover these gaps.

Inspired by industry giants such Flipkart and Amazon, ShopSmart AI is a thorough ecommerce platform modified to showcase contemporary full-stack development skills including AI integration, gamification, dark mode theming, real-time order tracking, and mobile PWA support - all implemented at zero infrastructure expense using free-tier cloud services.

1.2 Problem Statement

Current e-commerce systems are either too expensive to develop, lack artificial intelligence, or cannot keep consumers beyond the fundamental buy funnel engaged. Lack of access to advanced platforms mixing AI recommendations with contemporary UX patterns characterizes small and medium firms. This initiative shows how free and open-source tools can be used to construct a feature-complete, artificial intelligence driven e-commerce platform that makes sophisticated technology accessible to all.

1.3 Project Objectives

The main goals of this project are to develop a production-ready MERN stack ecommerce platform including full CRUD capabilities, incorporate free artificial intelligence (Groq API) for chatbot, search, recommendations, and analysis, implement gamification including loyalty points, spin wheel, and referral system; to attain dark/light theming, seamless animations, and mobile responsiveness; to deploy fully on free cloud infrastructure (Render + Vercel + MongoDB Atlas); and to build a comprehensive admin dashboard featuring order management and statistics.

1.4 Scope

Built with React 18 and Framer Motion animations, a frontend, a backend using a Node.js/Express REST API, a MongoDB Atlas (cloud) database, AI integration via the Groq API with the LLaMA 3.3 70B model, email alerts using Nodemailer, and cloud deployment. It emphasizes Indian e-commerce uses, Indian phone numbers, INR currency, and a regional coupon system. Developed to be a verifiable, production-quality application, the platform emphasizes cutting-edge artificial intelligence features together with current web design methods. Each architectural decision was taken with zero infrastructure cost in mind without thereby lowering user experience or feature quality.

2. System Architecture & Technology Stack -

2.1 Architecture Overview

ShopSmart AI uses a three-tier system that includes a client, application, and data layer. The front end, built with React, connects to the application layer through HTTP/HTTPS. This application layer is powered by Express.js and handles REST API requests. It then communicates with the data layer, which is MongoDB Atlas. The backend layer uses external services such as Gmail SMTP and the Groq AI API to secure server operations and protect API keys. The flow of data follows this order: MongoDB Atlas Database, REST API (Express/Render), Client (React/Vercel). The external APIs used are Gmail SMTP (Nodemailer) and Groq AI (LLaMA 3.3 70B).

2.2 Technology Stack

Layer	Technology	Version	Purpose
Frontend	React	18.x	UI component library
Animations	Framer Motion	10.x	Page transitions & micro-animations
Routing	React Router	v6	SPA client-side routing
HTTP Client	Axios	1.x	API communication
Backend	Node.js	18+	JavaScript runtime

Layer	Technology	Version	Purpose
Framework	Express.js	4.x	REST API framework
Database	MongoDB Atlas	7.x	Cloud NoSQL database
ODM	Mongoose	7.x	MongoDB object modeling
Authentica-tion	JWT + bcryptjs	Latest	Secure auth tokens
AI Engine	Groq API	Latest	LLaMA 3.3 70B model
Email	Nodemailer	6.x	SMTP email service
Deployment FE	Vercel	Free	React hosting + CDN
Deployment BE	Render.com	Free	Node.js hosting

2.3 Project Directory Structure

The project is organized as: shopsmart/ containing .env (environment variables), package.json (root scripts), server/ with index.js (Express entry + MongoDB connect), models/ (User.js, Product.js, Order.js), middleware/auth.js (JWT protect middleware), and routes/ (auth, products, orders, ai, cart, email); and client/ with public/ (mani-fest.json, sw.js for PWA) and src/ containing App.js (all 14 routes with AnimatePresence), context/ (AppContext.js, ThemeContext.js), hooks/ (useVoiceSearch, useRecentlyViewed), components/ (16 reusable components), and pages/ (14 full-page components).

3. Backend Development - Node.js + Express -

3.1 Server Setup & Middleware

The Express server is initialized with dotenv configuration as the first statement - critical for environment variable access throughout the application. CORS middleware is configured to accept requests only from the CLIENT_URL environment variable, providing security in production. The server runs on port 5000 by default and connects to MongoDB Atlas before beginning to listen for requests.

3.2 REST API Endpoints

Route	Method	Auth?	Description
/api/auth/register	POST	No	Register new user, returns JWT
/api/auth/login	POST	No	Login, returns JWT + user data
/api/products	GET	No	List products with filters/pagina-tion
/api/products/seed	GET	No	Seed 16 demo products (dev only)

Route	Method	Auth?	Description
/api/products/:id	GET	No	Single product details
/api/products/:id/reviews	POST	Yes	Add product review
/api/orders	GET	Yes	Get user orders
/api/orders	POST	Yes	Create new order
/api/orders/:id	PATCH	Yes	Cancel order (placed/confirmed only)
/api/cart	GET/POST	Yes	Get/update user cart
/api/ai/chat	POST	No	Groq AI chatbot response
/api/ai/search	POST	No	AI-powered product search
/api/email/order-confirm	POST	No	Send order confirmation email
/api/health	GET	No	Backend health check

3.3 Data Models

Three primary mongoose schemas are the building blocks of the data layer. The user schema holds the authentication data (hash of password using bcryptjs), roles (user/admin), and user information. The product schema is defined with the following properties: emojis, category, brand name, ratings, an array of reviews, stock, and badges. The order schema includes an array of items, address details, payment mode, order status, and delivery estimate. A cart schema exists in a one-to-one relation with a user and uses the upsert feature..

3.4 AI Integration - Groq API

The Groq AI plugin uses the https package that comes along with the Node.js package to avoid using other npm plugins. In this way, we have zero dependencies where native HTTP calls are made towards the Groq AI compatible endpoint. The model used here is called “llama-3.3-70b-versatile,” which is said to offer GPT-4 level reasoning at zero cost with limitations of making 14,400 requests a day. Moreover, the system prompt makes it clear that ShopSmart is a shopping assistant for India, using INR as currency.

4. Frontend Development - React + Framer Motion -

4.1 Application Architecture

In the React app, there are two global contexts that have been created using Context API, namely AppContext, which handles the state management of the cart, wishlist, authentication, cart count, and total calculation; and ThemeContext, which handles the dark or light mode, including saving data in localStorage. Additionally, there is AnimatePresence that includes all the routes for smooth transitions using Framer Motion.

4.2 All 14 Application Routes

Route Path	Component & Description
/	HomePage - Hero, categories, flash sale, AI budget planner, recently viewed
/products	ProductsPage - Grid with filters, sorting, search, infinite scroll
/product/:id	ProductDetailPage - Gallery, reviews, AI Q&A, price drop alert
/cart	CartPage - Items, quantity control, coupon system
/checkout	CheckoutPage - 3-step: address, payment, review + confetti animation
/orders	OrdersPage - Order history with cancel functionality
/track/:id	OrderTrackingPage - Animated timeline, delivery countdown timer
/profile	ProfilePage - 4 tabs: profile, orders, wishlist, addresses
/auth	AuthPage - Login and register with JWT authentication
/wishlist	WishlistPage - Saved products grid
/loyalty	LoyaltyPage - Points, tiers, rewards redemption
/spin	SpinWheelPage - Canvas-based daily spin wheel
/referral	ReferralPage - Referral codes, sharing, tracking
/compare	ProductComparison - Side-by-side with AI verdict

4.3 State Management - AppContext

The AppContext utilizes React's useReducer hook along with the localStorage method for storing cart and wishlist information. The available methods of manipulation within the cart include ADD_TO_CART (that adds one more item if the product is already present in the cart), REMOVE_FROM_CART, UPDATE_QTY, CLEAR_CART, TOGGLE_WISHLIST, SET_USER, and LOGOUT. CartCount and cartTotal are computed properties, which are generated based on the contents of the cart array. An essential fix for a bug includes initializing the cart within a try-catch clause and verifying that the cart is an array using Array.isArray.

4.4 Page Transition System

All paths are enclosed within the PageTransition component, which utilizes the AnimatePresence function of Framer Motion with mode="wait". Every single path animates itself by setting its initial state from opacity:0 to 1, y:18 to 0, and scale:0.99 to 1 within 300 milliseconds with the cubic-bezier easing effect. The key passed to the AnimatePresence function is the location.pathname.

4.5 Critical Framer Motion Implementation Note

CSS 'background' shorthand is not supported by Framer Motion as it contains the values for position, repeat, and attachment besides others. One must always use 'backgroundColor' for animating colors in Framer. Using 'background' in combination with whileHover will cause an error. None of the hover events use 'whileHover' at all.

5. AI Integration - Groq LLaMA 3.3 70B -

5.1 AI Model Selection Rationale

Groq was selected among other options including OpenAI, Google Gemini, and Anthropic Claude owing to the following advantages: availability of a fully free version providing 14,400 requests daily, unmatched speed in inference (up to 300+ tokens/sec through special LPU chips), signup without a need for a credit card, and the llama-3.3-70b-versatile is at the level of reasoning quality of GPT-4. Previously made attempts with Google Gemini were not successful because of quota restrictions.

5.2 AI Features Implemented

Feature	Endpoint	Description
AI Chatbot	/api/ai/chat	Persistent chat with ShopSmart AI persona, responds in INR, Indian context
Smart Search	/api/ai/search	Natural language product search with AI suggestions
Budget Planner	/api/ai/chat	3-step wizard: budget → category → AI recommends 3 products
Review Summarizer	/api/ai/chat	Summarizes all product reviews into pros/cons format
Product Q&A	/api/ai/chat	Answer customer questions about specific products
Comparison Verdict	/api/ai/chat	Compares 2-3 products and picks winner with justification
AI Size Guide	/api/ai/chat	Recommends clothing size based on measurements
AI Negotiator	/api/ai/chat	Playful price negotiation game with AI

5.3 Groq API Implementation Details

There is no reliance on the groq npm library whatsoever, opting instead for Node.js' native https library for no extra dependencies. The request includes an expertly designed system prompt to set up the ShopSmart AI character and a user prompt with the question. Answers are restricted to 300 tokens for speed and conciseness. Full error handling includes JSON parse errors, problems with the API keys, and lack of access to the model.

5.4 AI Chatbot UI Design

ChatBot is displayed as a purple floating button located on bottom-right (z-index 500) and differentiates from the green Live Chat widget on bottom-left (z-index 490). Markdown style formatting for bold and italic messages allows users to receive more sophisticated answers. The typing effect consists of three bouncing dots, animated via Framer Motion library y keyframes. The conversation history is kept inside component state to allow multi-turn conversation with the use of previous messages by the AI.

6. Feature Implementation Details -

6.1 Authentication System

User registration hashes passwords with bcryptjs (10 salt rounds) before storage, ensuring passwords are never stored in plaintext. Login verifies the hash and returns a JWT signed with JWT_SECRET with a 24-hour expiry. The frontend stores the token in localStorage and attaches it via Axios request interceptor as 'Bearer <token>' in Authorization headers. Protected API routes use middleware that verifies the JWT and attaches the decoded user object to req.user.

6.2 Product Catalog & Filtering

Products support filtering by category (electronics, fashion, home, books, sports, beauty), badge (sale, new, trending, bestseller), price range, and full-text search. Sorting supports price-asc, price-desc, rating, and newest. Pagination is implemented with page/limit query parameters for efficient data loading. The seed endpoint creates 16 demo products complete with emojis, categories, brands, ratings, and stock levels.

6.3 Coupon System

Six coupon codes provide different discount types: SAVE10 (10% off, no minimum), FLAT200 (₹200 flat off, minimum ₹999), NEWUSER (20% off), WELCOME (₹100 off, minimum ₹499), SUPER50 (50% off, minimum ₹4,999), and FREESHIP (free shipping). The CouponSystem component dynamically shows eligible coupons based on the current cart total, preventing application of invalid codes.

6.4 Loyalty & Gamification System

There are four membership levels which offer loyalty programs for users based on their loyalty points. These include Bronze, Silver, Gold, and Platinum. Points can be earned by making purchases, writing reviews, inviting others to sign up, and celebrating birthdays. In the reward program, there are 6 rewards for earning a certain number of points. For instance, a person will receive ₹50 off for 500 points, and a Premium Badge for 5,000 points. The spin wheel is based on HTML5 Canvas API technology, offering one spin per day.

6.5 Order Tracking & Email Notifications

OrderTrackingPage features a 6-step animated timeline consisting of the following steps: Placed, Confirmed, Packed, Shipped, Out for Delivery, Delivered. The vertical line representing the progress moves to the current stage of the order using the animation engine of Framer Motion library. The Countdown Timer component calculates the number of days, hours, and minutes left until the expected delivery date and updates the information every minute. Nodemailer library with Gmail SMTP sends three different HTML emails..

6.6 PWA Implementation

The manifest.json file contains information regarding the app name, icons, theme color (4f46e5), and the standalone mode that enables an app-like experience during installation of the app on mobile devices. The service worker (sw.js) caches the files while installing and serves the cached files when fetching data for offline capabilities. It should be noted that API requests made on /api/ are not to be cached, thus ensuring fresh data access.

7. Database Design - MongoDB -

7.1 MongoDB Atlas Configuration

The MongoDB Atlas M0 Cluster has 512 MB of data stored on a shared cloud infrastructure within the Singapore region. The connection string follows the mongodb+srv protocol to enable replica set discovery automatically. The database shopsmart should be included before any query string parameter within the connection string. The network access setting allows access from all IPs, i.e., 0.0.0.0/0.

7.2 Data Schema Design

Collec-tion	Key Fields	Notes
users	_id, name, email, password (hashed), role (user/admin), wishlist[], createdAt	Password never returned in API re-sponses
products	_id, name, brand, category, price, originalPrice, emoji, bgColor, ratings, numRe-views, reviews[], stock, badge, description	16 seeded demo products with full metadata
orders	_id, user (ref), items[], ship-pingAddress{}, payment-Method, orderStatus, total, trackingId, estimatedDeliv-ery, createdAt	Status enum: placed/con-firmed/packed/shipped/out_for_de-livery/delivered/cancelled
carts	_id, user (ref), items[{prod-uct, qty}]	One cart per user, upserted on change

7.3 Order Status State Machine

The states of an order follow a linear state machine pattern, meaning that the status of an order follows logic to move from one stage to another. An order can be cancelled only when it is either placed or confirmed. It is only possible to cancel an order using the PATCH endpoint when the order's current status is checked against its validity. The transition from one status to another occurs automatically through the admin dashboard.

7.4 Data Integrity & Security

Best practices in the security of handling sensitive information are always adhered to, whereby passwords are hashed through bcryptjs with 10 salts before they are stored, without ever being returned in an API response. JWT tokens are secured through a server-secret password with a lifetime of only one day, thus eliminating the possibility

of a stolen token. MongoDB injections are prevented by validating input through Mongoose's schema, before any database operation.

8. UI/UX Design System -

8.1 CSS Variables - Dark/Light Mode

The entire design system uses CSS custom properties toggled by the data-theme attribute on the document root element. This allows instant theme switching without React re-renders, providing a seamless user experience.

CSS Variable	Light Mode	Dark Mode
--surface	#f7f7ff	#0f0f1a
--card	#ffffff	#1a1a2e
--text	#0d0d1a	#f1f5f9
--text-muted	#6b7280	#94a3b8
--accent	#4f46e5	#4f46e5 (same)
--border	#e5e7eb	#2d2d4e
--input-bg	#ffffff	#16162a
--success	#10b981	#10b981 (same)
--danger	#ef4444	#ef4444 (same)

8.2 Typography System

Two fonts from Google Fonts have been used. The first font is the Syne with weights 700 and 800 and has been used for all the headings, logos, prices, and brand names. This choice has been made because of its geometric, modern design. The second font is the DM Sans with weights 400, 500, and 600, and it has been used for all the body text, buttons, inputs, and user interface labels. This choice has been made because of its good legibility and proportional design.

8.3 Animation Design Philosophy

The Framer Motion library deals with three kinds of animations. Page transition animations include opacity, translateY, and scale to achieve seamless route switching. Micro-animations for components include whileHover scale transformations and whileTap scale shrinking animations to provide users with tactile interaction. Background animations utilize requestAnimationFrame canvas drawing techniques for particles, shooting stars, and ripple effects triggered by clicking in Animated-Background.js. Ambient animations include floating emojis, which employ CSS keyframe animations that randomly delay and position. The custom cursor consists of two components: one small dot that tracks precisely and another large ring that trails with linear interpolation.

8.4 Component Library Overview

This platform contains 16 reusable elements:

Navbar (fixed header, featuring logo, search, theme switcher, cart, user profile),

Footer (footer content, newsletter signup, modals for FAQ/returns/privacy policy),

AIChat (floating chatbot),

ProductCard (grid element, featuring cart, wish list, ratings),

BottomNav (mobile only, automatically hides on scroll),

AnimatedBackground (Canvas particle effects),

FlashSaleTimer (animated digit countdown),

CouponSystem (drop-down, featuring coupon validation),

SkeletonCard (loading states),

Confetti (celebration effect on successful order),

RecentlyViewed (horizontal scrollable strip),

AiBudgetPlanner (3-step budget planner),

PriceDropAlert (email subscription to price drops),

Float-ingEmojis (ambient**9. Deployment & DevOps -**

9.1 Zero-Cost Deployment Architecture

Service	Platform	Configuration
React Frontend	Vercel (free)	Root: client/, Build: npm run build, Output: build/
Express Backend	Render.com (free)	Build: npm install, Start: node server/index.js
MongoDB Database	MongoDB Atlas (free M0)	512MB, shared cluster, Singa-pore region
Domain	Vercel subdomain	https://shopsmart-ai.vercel.app (free forever)

9.2 Environment Variables Management

The environment variables hold all secret data in both Render and Vercel environments without being pushed into GitHub. Files in .gitignore include .env, node_modules/, and client/build/ folders. An important condition is that the value of CLIENT_URL in the Render environment must be the same as the deployment URL of Vercel for CORS to work, while REACT_APP_API_URL in the Vercel environment must equal the backend Render URL.

9.3 CI/CD Pipeline

GitHub is the code repository for Source Control. Each time a new commit is pushed to the master branch, it results in a re-build of both Render backend (approximately 3 mins) and Vercel frontend (approximately 2 mins). No further steps are necessary to manually deploy after the initial set-up. The client/package.json proxy value forwards

requests to the localhost:5000 which allows for seamless local development similar to how the app functions in production. Important note: Free-tier of Render shuts down after 15 mins of inactivity. Initial request after sleep takes 30–60 seconds to wake up.

10. Testing & Bug Fixes -

10.1 Known Bugs Identified and Fixed

Bug	Cause	Fix Applied
rgba background not animatable	Framer Motion cannot animate CSS background shorthand	Use backgroundColor or plain CSS transitions with onMouseEnter/Leave
React child object error	Broken ternary expression creates object instead of number	Corrected ternary: star===5?65:star===4?20:star===3?10:star===2?3:2
Invalid style prop: lastChild	lastChild:{borderBottom:'none'} is not valid React inline style	Use index-based conditional: i < arr.length-1 ? border : 'none'
Cart reduce not a function	Corrupted localStorage: non-array value crashes Array.reduce()	Wrap initialization in try-catch with Array.isArray() check
Groq model decommissioned	llama3-8b-8192 was sunset by Groq in 2024	Updated to llama-3.3-70b-versatile (better quality, still free)
Products seed 404 error	/:id route matched 'seed' as an ID before /seed route was registered	Move /seed route definition ABOVE /:id route in products.js
dotenv undefined error	dotenv.config() called after other require() statements	require('dotenv').config() must be the absolute FIRST line

Bug	Cause	Fix Applied
Spin wheel not rendering	useEffect ran before canvas mounted; rotation math incorrect	Use useCallback for draw(); fix rotation: endRot = startRot + delta aligning winner center to top pointer

10.2 Testing Approach

Manual end-to-end testing was conducted on all user flows: sign-up, log-in, filtering through products using all filters, shopping cart functionalities, complete purchase process, purchase status with animation, chat with AI using all eight user flows, spinning wheel with validation of winner, redemp-tion of points, and all admin dashboard functionalities. JavaScript issues were captured in real-time using browser DevTools console. Validation of backend API endpoints was conducted using URL bar in browser for GET requests and browser console fetch() function for POST requests.

11. Results & Performance Analysis -

11.1 Feature Completion Summary

Category	Features Built	Status
Core Ecommerce	Product catalog, cart, checkout, orders, order tracking, profile, admin dashboard	Complete ✓
Authentication	JWT register/login, bcrypt hashing, protected routes, role-based admin access	Complete ✓
AI Features	Chatbot, smart search, budget planner, review summarizer, product comparison verdict	Complete ✓
Gamification	Loyalty tiers, daily spin wheel, referral program, coupon system	Complete ✓
UI/UX	Dark/light mode, page transitions, custom cursor, animated background, confetti	Complete ✓
Mobile	Bottom navigation, responsive design, PWA manifest + service worker	Complete ✓
Email	Order confirmation, shipping updates, welcome email (HTML templates)	Complete ✓
Performance	Skeleton loading, lazy images, recently viewed, infinite scroll hook	Complete ✓

Category	Features Built	Status
Deployment	Vercel (FE) + Render (BE) + MongoDB Atlas - zero infrastructure cost	Complete ✓

11.2 Platform Metrics

Metric	Count
Total Pages (Routes)	14 full-page components
Reusable Components	16 components
Custom React Hooks	3 (useVoiceSearch, useRecent-lyViewed, useInfiniteScroll)
Backend API Endpoints	16 endpoints across 6 route files
MongoDB Collections	4 (users, products, orders, carts)
CSS Design Tokens	10 CSS custom properties
Coupon Codes	6 working discount codes
Loyalty Rewards	6 redeemable reward options
Spin Wheel Prizes	8 prize slots
Email Templates	3 HTML email templates
Product Categories	6 (electronics, fashion, home, books, sports, beauty)
Total Lines of Code	8,000+ lines across all files

11.3 AI & Deployment Performance

Groq API, paired with LLaMA 3.3 70B, is able to respond within 800-1500 ms for common ecommerce queries, which is well within acceptable UX levels. Free tier support of up to 14,400 requests per day for the free plan is more than enough to cater to demo and minimal traffic levels. Vercel CDN provides delivery of React build from edge locations around the globe with under 100 ms for static assets. MongoDB Atlas M0 cluster provides 50-200 ms query latencies for common product listings..

12. Conclusion & Future Scope -

12.1 Conclusion

The ShopSmart AI project proves that it is possible to create a modern, AI-enabled, and feature-rich e-commerce website using entirely free, open source software. In combination with Groq AI, the MERN stack provides a production-level user experience equal to commercial-grade platforms. The development process included all stages of the software development life cycle, starting from the system architecture design to implementing features, debugging code, and deploying on the cloud infrastructure.

Some of the technical highlights are as follows: no dependency integration of the Groq API using built-in Node.js libraries, theme creation system based on CSS variables with instant dark and light mode toggle, winner prediction spin wheel using the Canvas API with accurate mathematics, integration of Framer Motion animations avoiding non-animatable CSS properties, and CI/CD pipeline with GitHub, Vercel, and Render.

12.2 Skills Demonstrated

The project is an example of full-stack JavaScript development (MERN Stack), including the creation of REST APIs, AI API implementation and prompt generation for ecommerce applications; React state management, custom hooks, and Context API implementations; MongoDB database architecture and setting up a database on Atlas; cloud deployment along with handling environment variables and CORS setup; UI/UX design using CSS variables and Framer Motion for animations; debugging and fixing problems in React and Node.js applications; and implementing email automation with HTML template design and SMTP setup.

12.3 Future Scope

The proposed system could be improved further by implementing the following features: (1) Razorpay/Stripe integration for handling actual transactions; (2) real-time capabilities such as live stock information and order update notifications through WebSocket communication; (3) intelligent recommendations utilizing collaborative filtering based on purchase history; (4) social media login using Google OAuth and OTP-based phone verification; (5) a multi-vendor marketplace system that includes seller account creation and revenue sharing; (6) analytics such as heatmap reporting, conversion funnel evaluation, and A/B testing setup; (7) Redis for caching high-volume APIs related to product listings; (8) test automation using Jest and Cypress for unit tests and end-to-end tests, respectively; (9) migration from free to paid tier at Render (cost of $7/month) to avoid cold startup times; and (10) international shipment cost calculator API.

13. References

1. MongoDB Inc. (2024). MongoDB Atlas Documentation. https://www.mongodb.com/docs/atlas/

2. Meta AI. (2024). LLaMA 3.3 Model Card - Open Foundation and Fine-Tuned Chat Models. https://ai.meta.com/llama/

3. Groq Inc. (2024). Groq API Documentation - OpenAI-Compatible Endpoints. https://console.groq.com/docs/

4. Vercel Inc. (2024). Vercel Deployment Documentation. https://vercel.com/docs/

5. Render Inc. (2024). Render Web Services Documentation. https://render.com/docs/web-services

6. Facebook Open Source. (2024). React 18 Documentation. https://react.dev/

7. Framer. (2024). Framer Motion API Reference. https://www.framer.com/motion/

8. Mongoose. (2024). Mongoose ODM Documentation v7. https://mongoosejs.com/docs/

9. Express.js. (2024). Express 4.x API Reference. https://expressjs.com/en/api.html

10. Auth0. (2023). JSON Web Tokens Introduction. https://jwt.io/introduction/

11. Nodemailer. (2024). Nodemailer Documentation. https://nodemailer.com/about/

12. MDN Web Docs. (2024). Canvas API Reference. https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API

13. MDN Web Docs. (2024). Progressive Web Apps (PWA) Guide. https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps

14. Flanagan, D. (2020). JavaScript: The Definitive Guide (7th Edition). O'Reilly Media.

15. Banks, A. & Porcello, E. (2020). Learning React (2nd Edition). O'Reilly Media.