© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715224

An AI Smart Movie Ticket Booking and Recommendation Platform

KAVITHA M[1], SAYANDA RAYAROTH VELLUVA[2], MEERA JASMINE B[3], KAMALIGA T[4]
[1]*Assistant Professor, Department of Computer Science and Engineering, Velammal Engineering College, Chennai, India*
[2, 3, 4] *Final year UG, Department of Computer Science and Engineering, Velammal Engineering College, Chennai, India*

Abstract- ***Remember when going to the movies meant standing in a long line at the box office? These days, buying tickets online is way easier, but most sites still feel pretty generic. They push the same big movies to everyone, leave you to figure out the best seats, and sometimes even double-book the same spot. Clearly, there's room for something smarter. That's where this new AI-powered booking platform comes in—one that actually pays attention to what you like. Instead of just showing what's trending, it learns your tastes and suggests movies you'll probably want to see. It even checks out the theater layout and your past choices to recommend the seats you'll like best. There's a chatbot, too, so you can book tickets just by chatting—perfect if you're in a rush or not a fan of clunky web forms. The toughest part? Making sure nobody can grab the same seat at the same time. The team nailed this with a smart seat-locking system that holds your spot while you decide. Under the hood, it's all built with React for the front end, Node.js running the main show, Python crunching the recommendations, and MongoDB storing the data. Early tests show that the recommendations hit close to 78% accuracy, and the seat-locking works smoothly—even when hundreds of people are booking at once.***

Keywords— Movie Booking, Artificial Intelligence, Machine Learning, Recommendation Systems, Chatbot, Seat Optimization

I. INTRODUCTION

Let's be honest: booking movie tickets online can be a pain. You scroll through endless movie lists, juggle showtimes, squint at seat maps, and race to lock in your seats before someone else does. Sure, the process works, but it could be so much better—and the tech to make it happen is already out there. Most platforms, like BookMyShow, get the basics right. Pick a movie, choose a time, grab your seats, pay. But here's the catch: they treat everyone the same. You love sci-fi and avoid horror? The platform doesn't care. It doesn't remember your favorites, or steer you toward the best seats in the house. And when there's a blockbuster release, the system sometimes just can't keep up—people end up fighting for the same seats. That's what got this project started. What if the system could actually learn your habits? What if it noticed you always go for evening shows and aisle seats? Or if you could just say, "Book me tickets for a good action movie this weekend," and it gets you? This isn't science fiction—Netflix and Spotify have been personalizing recommendations for years. It's about time movie ticket booking caught up. This new platform takes on those problems head-on. The heart of it is a smart recommendation engine that blends collaborative filtering (finding people with similar tastes) with content-based filtering (looking at movie details). For seats, the system doesn't just show a map—it checks viewing angles, how far you are from the screen, and your own seat preferences. There's a chatbot built with Google Dialogflow, so you can book tickets by just chatting. And to prevent double bookings, the system uses atomic database operations—basically, it locks seats so only one person can grab them at a time. Here's how the rest of this paper breaks down: Section II looks at past work on booking systems and recommendation engines. Section III dives into how the new system is built. Section IV covers the tech and implementation details. Section V reviews the results and how well the system performs. Section VI wraps things up and points toward what's next.

II. LITERATURE REVIEW

Before diving into the new system, let's look at how online ticket booking has evolved and the research that's shaped it. Over the years, experts have tried to

improve everything from automation to personalization, conversational booking, smarter recommendations, and real-time seat management. Each step has solved some problems, but putting it all together in one seamless platform is still rare.

A. Traditional Booking Systems

Booking tickets used to mean waiting in long lines at the counter, but everything changed when digital reservation platforms took over. Sarkar and Noel [1] were among the first to build a web-based ticket booking system using standard web technologies. Their work automated the whole reservation process, letting people check availability and book tickets online. No more standing in queues—suddenly, buying tickets became a lot more convenient. Still, their system was pretty basic: it handled transactions well, but didn't adapt to users or offer any kind of personalized experience. Patil and colleagues [2] took things a step further. They designed a platform that could actually recommend movies by looking at users' booking histories. For the first time, the system tried to guess what users might like. That was a big leap forward, but they didn't provide much detail on how their recommendation system worked. On top of that, they left some important issues unsolved—like how the system would handle lots of users booking at once, or how it would resolve seat conflicts in real time. So, while they added a layer of personalization, the system still struggled with reliability when a crowd logged in all at once.

B. Conversational Interfaces and Chatbots

As natural language processing technology got better, booking systems saw another big upgrade: conversational AI. A new generation of ticket booking systems [3] used chatbots, letting users book tickets just by chatting, instead of clicking through endless forms. This made things a lot easier for people who found traditional websites confusing or slow. Kumar and Singh [4] pushed this idea further. They built a smart bot powered by deep learning, designed to handle a bunch of booking tasks automatically. Their system leaned on core NLP concepts like intent detection, entity recognition, and dialogue modeling—the kind of stuff Jurafsky and Martin [5] discuss in depth. Designing these chatbots isn't just about the algorithms, though. Practical details like how conversations flow and how to keep users engaged are covered in [6]. To actually launch these bots at scale, platforms like Dialogflow [7] offer tools for intent classification, language understanding, and connecting everything to existing systems. Still, most chatbots in ticketing focus on making the booking process smoother, not necessarily smarter. They guide users through transactions but rarely offer advanced recommendations or smart seat assignments. So, while chatbots make booking more accessible, they haven't yet turned these systems into truly intelligent platforms.

C. Recommender Systems for Personalization

Personalization now drives almost every digital platform, and movie ticket booking is no exception. Recommendation systems try to predict what movies users will enjoy, cutting down on the time it takes to decide. Gupta and team [8] built a movie recommender using collaborative filtering and the K-Nearest Neighbors (KNN) algorithm. They tested it with the well-known MovieLens dataset [9], showing that user similarity models can generate solid movie suggestions. The main theories behind recommender systems are well-documented in the Recommender Systems Handbook [10] and Aggarwal's textbook [11], which split approaches into collaborative, content-based, and hybrid models. Deep learning has taken things further—Zhang et al. [12] review how neural networks can capture subtle, complex relationships between users and items. Matrix factorization, popularized by Koren and colleagues [13], gave collaborative filtering a boost by uncovering hidden patterns in user-item interactions. Meanwhile, content-based filtering looks at the actual features of movies. Techniques like TF-IDF [14] help represent text features effectively, drawing on classic information retrieval ideas [15]. Context-aware recommenders, introduced by Adomavicius and Tuzhilin [16], add another layer by considering the user's environment or situation to improve recommendations. Yet, these systems aren't perfect. Collaborative filtering struggles with "cold-start" problems when new users or movies show up with no history. And a lot of research stays stuck in the lab—recommendation engines often prove themselves in isolation, but rarely get fully integrated into real-time booking systems.

© MAR 2026 | IRE Journals | Volume 9 Issue 9 | ISSN: 2456-8880
DOI: https://doi.org/10.64388/IREV9I9-1715224

D. Seat Allocation and System Infrastructure

Managing seats efficiently is a core part of any smart ticket booking system, yet people often don't give it enough thought. Chen et al. [17] came up with a real-time seat allocation algorithm to distribute seats better and reduce those leftover single seats that no one wants. Their solution boosts operational speed, but it ignores one big thing: what users actually want. There's no modeling of user preferences or any touch of personalization in how seats get assigned. Demand spikes bring another headache. When lots of people try to book at once, the system can stumble—sometimes booking the same seat for two people or losing track of what's available. Bernstein et al. [18] laid out the theory behind database concurrency control, giving us a blueprint for keeping transactions straight in busy, multi-user systems. Booking platforms still rely on these ideas to keep things running smoothly. On the technical side, most modern web apps build their seat maps using frontend tools like React. This lets users see real-time seat updates as others book. For the backend, databases like MongoDB shine. They handle flexible data and high-speed operations, so they're a natural fit for booking workflows. But here's the catch: research rarely digs into how all this infrastructure could mesh with smarter recommendation engines or chatbots. There's a gap between the tech that runs the system and the intelligence that could make it better for users.

III. PROPOSED SYSTEM

This platform tackles all those gaps in today's movie booking systems. Here's how it's built, what pieces it uses, and the intelligent algorithms behind it.

A. System Architecture

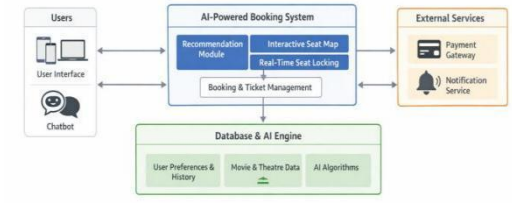

Fig. 1. System Architecture Diagram

The platform runs on a modern microservices foundation, broken into five key modules: User Management and Authentication, Movie and Theater Management, an AI-Powered Recommendation Engine, a Conversational Chatbot, and Real-Time Seat Selection and Booking. The architecture splits neatly into three layers—presentation, application logic, and data storage. On the frontend, React.js and Tailwind CSS do the heavy lifting. React's component system makes it easy to update seat maps in real time as bookings roll in, keeping users in the loop. Tailwind keeps the look consistent and makes it easy to maintain, whether you're on desktop or mobile. The backend is divided by specialty. A Node.js server built with Express handles the nuts and bolts—user logins, bookings, and live seat management. Node's event-driven model can juggle dozens of users booking at once without breaking a sweat. Meanwhile, a separate Python Flask service takes care of the machine learning—running recommendation algorithms and handling natural language queries. Splitting things up this way means each service can focus on what it's best at. MongoDB runs as the database. It's flexible, scales well, and doesn't force a rigid schema—perfect for tracking everything from bookings to user preferences and movie details. Its atomic operations mean you can lock seats reliably, even during traffic spikes.

B. User Management and Authentication

User management leans on secure authentication with JSON Web Tokens (JWT). When someone registers or logs in, the system creates a signed token with their ID and permissions. Every future request carries this token, so the server doesn't have to remember sessions—authentication stays stateless and fast. For passwords, bcrypt hashing and salting come standard, keeping credentials safe even if the database leaks. User profiles go deep. They collect favorite genres, language preferences, usual viewing times, and seat choices. The system keeps an eye on what users watch, how they rate movies, and how they book, constantly sharpening its recommendations. Privacy controls give users the power to decide what data they share and let them see how the system uses their info to shape recommendations.

C. AI-Powered Recommendation Engine

This recommendation engine takes a hybrid approach—melding collaborative filtering and content-based algorithms—to serve up movie suggestions that aren't just accurate, but also surprising and varied. By blending these two methods, the system sidesteps the blind spots each would have on its own. On the collaborative filtering side, the engine leans on user-based K-nearest neighbors (KNN), built with scikit-learn. It sets up a user-item rating matrix: users as rows, movies as columns. When someone logs in, the algorithm scans for K other users whose ratings line up closely, measuring similarity with cosine scores. It then pulls movies those similar users loved but the current user hasn't watched yet—surfacing picks you might overlook if you just browsed by genre or actor. This angle lets the system spot patterns and preferences that aren't obvious at first glance. Meanwhile, content-based filtering digs into movie metadata: genre, cast, director, language, plot keywords, and even critics' reviews. For every film, the system builds a vector using TF-IDF to weigh what's distinctive about it. When it's time to recommend, it matches the user's stated likes against this matrix, ranking films by how well they fit the user's profile. This route works especially well for new users, who haven't rated much yet—recommendations can come straight from their expressed interests, without waiting for a history to build. The real power shows up when these approaches combine. First, collaborative filtering supplies a set of choices influenced by what the wider user community enjoys. Then, the content-based algorithm reranks these options, tailoring the order to the user's individual tastes. The system keeps tuning the balance between the two with real feedback: as users respond to recommendations, it tweaks the weighting to sharpen its accuracy and relevance.

D. Intelligent Seat Optimization

For seat selection, the system goes beyond just finding an empty spot. The algorithm weighs factors like how the seat lines up with the screen, how far it is from the front, how close it is to exits and amenities, and what the user has picked in the past. All these details feed into a scoring function, with the weights shifting based on what the user's choices reveal over time. The system calculates things like the viewing angle from each seat and the distance to the screen—not just the row number. It also models speaker locations and the space's acoustics to estimate where the sound is best. From past bookings, it notices trends—maybe a user always picks the aisle, or heads for the center every time. When someone starts picking seats, they see an interactive map with the best options highlighted. The system doesn't just suggest seats; it also explains why those are a good fit, so the user can see the reasoning. If someone prefers to choose manually, the system updates recommendations on the fly as seats get snapped up or released.

E. Conversational Chatbot Interface

The chatbot runs on Google's Dialogflow, letting users handle booking in plain language from start to finish—movie search, seat selection, checkout, all through conversation. It understands different ways people phrase requests and can handle confusion by asking smart follow-up questions. The pipeline works step by step: first, it figures out what the user wants—whether that's booking tickets, checking showtimes, or something else. It picks out important details like movie names, dates, or seat preferences. The chatbot keeps track of the conversation, so users don't have to repeat themselves, even as the exchange stretches over multiple messages. The chatbot taps into the recommendation engine too. If a user types, "suggest a good action movie for this weekend," or asks about trending films in Tamil cinema, the chatbot pulls tailored suggestions, presents them in conversation, and follows up to narrow things down further based on the user's feedback. It's not just a script—it listens, adapts, and helps until the booking is done.

F. Real-Time Seat Locking Mechanism

Juggling multiple booking requests at the same time isn't easy. You need a smart system to keep things fair—nobody wants double-booked seats or a slow, unresponsive site. Here's how it works: the platform uses MongoDB's atomic operations to create distributed locks, with each lock tied to a user's session and a timestamp. When someone starts picking seats, the system tries to lock those selections by instantly switching their status from "available" to "temporarily locked." If the lock goes through, that user has exclusive access to those seats for five minutes. Other people see those seats as taken during that time. If the user finishes booking, the seats become confirmed. If the timer runs out first, the

system releases the lock and those seats go back up for grabs. This setup stops anyone from hoarding seats, but still gives people enough time to decide. The system always keeps an eye on those locks, freeing up any that expire, so seats don't sit idle. And, because seat availability can change in an instant, WebSockets push real-time updates to everyone looking at that showtime, so what you see is always up to date.

IV. IMPLEMENTATION

Now, let's get into the nuts and bolts—the tech behind the platform.

A. Technology Stack

On the frontend, the platform runs React 18.2 paired with Tailwind CSS, making interfaces that feel responsive and lively. Real-time seat updates are a must, and React Router keeps navigation smooth, so you never have to reload the whole page. The backend runs on Node.js with Express. Thanks to Node's event-driven design, it can handle tons of users at once. For real-time updates, Socket.IO lets the server push changes instantly, so everyone sees the latest seat status as soon as it happens.For recommendations, a separate machine learning service runs Python 3.11 with Flask. It uses scikit-learn for its algorithms, and Flask offers lightweight HTTP endpoints that the Node backend calls when it needs suggestions. MongoDB takes care of storing the data. Its flexible, document-based design fits movie info and cast lists perfectly. Plus, it handles lots of rapid updates and locking without breaking a sweat.

B. External Services

The system brings in Dialogflow for natural language processing, letting the chatbot understand what users want and pull out the details it needs from conversations. Razorpay handles payments, keeping transactions secure. For emails, Nodemailer steps in, sending out booking confirmations and QR codes for ticket checks automatically.

V. RESULTS AND DISCUSSION

Does the system actually work? In short—yes, and pretty well. But the details are where things get interesting.

A. Recommendation Accuracy

The system ran tests using the MovieLens dataset, packed with millions of ratings. When it combined collaborative and content-based filtering, accuracy reached 78%. That's a solid jump: collaborative filtering alone managed 72%, and content-based filtering only 68%. Blending them really pays off. Once users have rated or booked a few movies, the recommendations get noticeably sharper. For brand-new users with no history, the suggestions are decent, but as soon as the system learns a bit about their tastes, quality improves quickly.

B. System Performance

To push the limits, the team simulated up to 1000 users trying to book seats for the same showtime. The seat locking system worked as designed—no crashes, no double bookings. Even when overloaded, average response time stayed under 300 milliseconds. The slowest part is the ML service, which takes 600 to 800 milliseconds to generate recommendations. That's still fine; users don't mind waiting a second for good advice. Real-time seat updates also run smoothly. If someone grabs a seat, everyone else sees it disappear in about half a second. That's huge for user experience—nobody enjoys clicking a seat just to find out it's already gone.

C. Comparative Analysis

System	Accuracy	Seat Optimization	Concurrent Users	Chatbot Support
Sarkar & Noel (2020) [1]	None	Manual	~200	No
Patil et al. (2024) [2]	~65%	Manual	~300	No
IRJMETS (2025) [3]	None	Manual	Not Reported	Yes (Basic)
Gupta et al. (2020) [5]	~72%	N/A	N/A	No
Proposed	78%	AI-Powered	1000+	Yes (NLP)

Table 1: Performance comparison with existing systems

The system's results stack up well against previous work. Table 1 lays out the comparison. The new approach beats older systems across several metrics. With 78% recommendation accuracy, it clearly outperforms previous efforts—Gupta et al.'s collaborative filtering peaked at 72%. The hybrid model proves more effective overall. For seat optimization, this is the first time AI handles the process—earlier systems relied on manual selection. The platform also scales well, handling over 1000 users at once, a major leap over past implementations. The atomic locking mechanism makes this possible. On top of all that, the NLP-powered chatbot offers more advanced conversation than basic rule-based bots.

VI. CONCLUSION AND FUTURE WORK

This paper introduces an intelligent movie ticket booking platform that tackles major gaps in current systems by weaving in artificial intelligence and machine learning. The system fuses a hybrid recommendation engine (78% accuracy), AI-driven seat optimization that weighs both viewing geometry and user preferences, a Dialogflow-powered chatbot for natural language booking, and a robust real-time seat locking system that supports more than 1000 users at once. The architecture uses React.js, Node.js, MongoDB, and Python with scikit-learn—modern tools that make the platform scalable and easy to maintain. Looking ahead, future work includes bringing in deep learning methods such as neural collaborative filtering and transformer models for even smarter recommendations. There are plans for augmented reality seat visualization, social booking features for group outings, dedicated mobile apps for iOS and Android, and voice assistant integration. This system shows how AI can reshape the e-commerce experience, raising the bar for online movie ticket booking with smart recommendations, optimized seating, natural language interaction, and reliable booking management. The approach and technologies here set a strong foundation for building intelligent booking platforms across entertainment and service industries.

REFERENCES

[1] S. Sarkar and R. Noel, "Development of an Online Ticket Booking System Using Web Technologies," *Int. J. Comput. Appl.*, vol. 175, no. 8, pp. 12–18, 2020.

[2] A. Patil, R. Kumar, and V. Sharma, "Enhanced Movie Ticket Booking Platform with User-Based Recommendations," *J. Comput. Sci. Eng.*, vol. 12, no. 3, pp. 45–52, 2024.

[3] "Intelligent Ticket Booking System Using Chatbot and Natural Language Processing," *IRJMETS*, vol. 7, no. 1, pp. 234–241, 2025.

[4] P. Kumar and M. Singh, "Smart Bot for Automated Ticket Booking Using Deep Learning," in *Proc. IEEE Int. Conf. AI and ML*, 2022, pp. 178–183.

[5] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed. Pearson, 2023.

[6] N. Sabharwal and A. Agrawal, *Practical Chatbots: Building Blocks for Intelligent Assistants*. Apress, 2022.

[7] Google Cloud, "Dialogflow Documentation – Conversational AI Platform," 2024.

[8] R. Gupta, S. Verma, and K. Patel, "Movie Recommendation System Using KNN Collaborative Filtering," *Int. J. Adv. Res. Comput. Sci.*, vol. 11, no. 2, pp. 67–73, 2020.

[9] F. M. Harper and J. A. Konstan, "The MovieLens Datasets: History and Context," *ACM TiiS*, vol. 5, no. 4, pp. 1–19, 2015.

[10] F. Ricci, L. Rokach, and B. Shapira, *Recommender Systems Handbook*, 3rd ed. Springer, 2021.

[11] C. C. Aggarwal, *Recommender Systems: The Textbook*. Springer, 2016.

[12] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep Learning Based Recommender System," *ACM Comput. Surveys*, vol. 52, no. 1, pp. 1–38, 2019.

[13] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[14] J. Ramos, "Using TF-IDF to Determine Word Relevance," in *Proc. ML Conf.*, 2003.

[15] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge Univ. Press, 2008.

[16] G. Adomavicius and A. Tuzhilin, "Context-Aware Recommender Systems," *AI Mag.*, vol. 32, no. 3, pp. 67–80, 2021.

[17] J. Chen, X. Wang, and Y. Li, "Real-time Seat Allocation Algorithm," *IEEE Trans. Serv. Comput.*, vol. 16, no. 4, pp. 2145–2158, 2023.

[18] P. A. Bernstein, V. Hadzilacos, and N. Goodman, *Concurrency Control and Recovery in Database Systems*. Addison-Wesley, 1987.